ConSol Software GmbH

ConSol CM Administratorhandbuch

Version 6.11.1.7

Inhalt

Inhalt 2
A - Einleitung 12
A.1 ConSol CM für Business Process Management 14
A.2 Liste der Handbücher 16
A.3 TecDoc-Server 17
A.4 Die Struktur des Handbuches 18
A.5 Erklärungen zum Layout 20
A.6 Rechtlicher Hinweis 21
A.7 Gender-Disclaimer 21
A.8 Copyright 21
A.9 Grundprinzipien von ConSol CM 22
A.9.1 Systemkomponenten aus der Sicht der Anwender, Administratoren und Kunden 22
A.10 Grundlegende technische Prinzipien und Objekte von ConSol CM 24
A.10.1 Einleitung 25
A.10.2 Das ConSol CM-Dogma 25
A.10.3 Bearbeiter 26
A.10.4 Kunden 26
A.10.5 Tickets 26
A.10.6 Ressourcen 27
A.10.7 Queues 27
A.10.8 Workflows 28
A.10.9 Das CM Action Framework 30
A.10.10 Das Task Execution Framework 30
A.10.11 Zugriff auf Objekte in ConSol CM 30
A.10.12 ConSol CM aus der Perspektive eines Systemadministrators 32
A.11 Starten des Admin Tools 33
A.11.1 Login 33

A.11.2 Troubleshooting: Das Admin Tool startet nicht 36
A.12 Die grafische Benutzeroberfläche (GUI) des Admin Tools 41
A.12.1 Einleitung 41
A.12.2 Grundprinzip 41
A.12.3 Inline-Validierung von Eingabewerten 45
A.12.4 Symbole und andere GUI-Elemente 48
A.12.5 Lokalisierung der im Web Client angezeigten Begriffe 51
B - Abschnitt Zugriff und Rollen 59
B.1 Bearbeiterverwaltung 60
B.1.1 Einführung in die Bearbeiterverwaltung 60
B.1.2 Bearbeiterverwaltung mit dem Admin Tool 61
B.2 Rollenverwaltung 68
B.2.1 Einführung in die Rollenverwaltung 68
B.2.2 Rollenverwaltung mit dem Admin Tool 69
B.2.3 Definieren eines Administrators nur für die Verwaltung der Rollen und Bearbeiter 86
B.3 Sichtenverwaltung 89
B.3.1 Einführung in die Sichtenverwaltung 89
B.3.2 Sichtenverwaltung mit dem Admin Tool 90
B.4 Bearbeiterfunktionen 98
B.4.1 Einleitung 99
B.4.2 Erstellen und Editieren einer Bearbeiterfunktion 101
B.4.3 Löschen einer Bearbeiterfunktion 102
B.4.4 Deaktivieren und Aktivieren einer Bearbeiterfunktion 102
B.4.5 Bearbeiterberechtigungen für Bearbeiterfunktionen 102
C - Abschnitt Ticketdatenmodell und GUI-Design 104
C.1 Verwaltung von Ticketfeldern (Einrichten des Ticketdatenmodells) 106
C.1.1 Einleitung 107
C.1.2 Verwaltung von Ticketfeldern mit dem Admin Tool 109

C.1.3 Tab Ticketdaten 110
C.1.4 Tab Aktivitätsformulare 125
C.1.5 Häufig verwendete Annotationen 131
C.2 Verwaltung von Sortierten Listen 138
C.2.1 Einleitung 138
C.2.2 Verwaltung von Sortierten Listen mit dem Admin Tool 140
C.3 MLA-Verwaltung 149
C.3.1 Einleitung 149
C.3.2 MLA-Verwaltung mit dem Admin Tool 151
C.4 Ticketprotokoll 157
C.4.1 Einleitung 157
C.4.2 Anzeigemodi des Ticketprotokolls im Web Client 158
C.4.3 Allgemeine Informationen über die Sichtbarkeit von Ticket-protokolleinträgen im Web Client 165
C.4.4 Speicherung des Ticketprotokolls und Übertragung an das Data Ware-house (DWH) 168
C.5 Konfiguration der Ticketliste 169
C.5.1 Einleitung 169
C.5.2 Konfiguration von Gruppierung und Sortierung der Ticketliste 171
C.5.3 Konfiguration der Parameter, die für ein Ticket angezeigt werden 176
C.5.4 Die Definition von Kunden-Templates 177
C.5.5 Die Annotation von Ticketfeldern 177
C.5.6 Die Seitenanpassung für die Attribute der Ticketliste 178
C.6 Konfiguration des Web Client Dashboards 187
C.6.1 Einleitung 187
C.6.2 Beispiele 188
C.7 Konfiguration der Graphendarstellung von Relationen 193
C.7.1 Konfiguration der Graphenanzeige im Standardmodus 193
C.7.2 Konfiguration weiterer Relationsansichten - Expertenmodus 195

C.8 Seitenanpassung 196
C.8.1 Allgemeine Einführung in die Seitenanpassung 197
C.8.2 Seitenanpassung im Web Client 198
C.8.3 Reihenfolge und Prioritäten bei der Seitenanpassung 208
C.8.4 Seitenanpassung mit Attributen 209
C.8.5 Seitenanpassungsattribute für Typen, Bereiche und Unterbereiche in alphabetischer Reihenfolge 219
C.8.6 Seitenanpassung für das Web Client Dashboard 281
C.9 Seitenanpassung für die grafische Darstellung von Relationen 323
C.9.1 Seitenanpassung für die Visualisierung von Relationen, Standard 324
C.9.2 Seitenanpassung zur Visualisierung von Relationen, Experten 332
C.9.3 Konfiguration der Graphenansicht von Relationen 337
C.10 Design und Konfiguration von REST-basierten ConSol CM-Client-GUIs 344
C.10.1 Einleitung 344
C.10.2 Konfigurationsprinzip 347
C.10.3 Konfiguration von speziellen Seiten in CM/Track 349
C.10.4 Erstellen von neuen Konfigurationsseiten 366
C.10.5 Verhalten bei Systeminstallationen und -aktualisierungen 366
D - Abschnitt Kundendatenmodell 367
D.1 Einführung in FlexCDM 368
D.1.1 FlexCDM auf einen Blick 369
D.1.2 Einführung in FlexCDM-Objekte 372
D.1.3 Verwaltung von FlexCDM-Objekten mit dem Admin Tool 374
D.2 Kurze Einführung in die FlexCDM-spezifischen Funktionen des Web Clients 376
D.2.1 Einleitung 376
D.2.2 Arbeiten mit FlexCDM im ConSol CM Web Client 377
D.3 Einrichten des Kundendatenmodells 394
D.3.1 Einführung in die Einrichtung des Kundendatenmodells auf Basis von FlexCDM 394
D.3.2 Verwaltung von Kontakten und Firmen mit dem Admin Tool 395

D.4 Verwaltung von Kundenfeldern und GUI-Design für Kundendaten 407
D.4.1 Einleitung 408
D.4.2 Definieren von Kundenfeldern für Kundendaten mit dem Admin Tool 409
D.4.3 Verwendung von Objekten aus FlexCDM in Skripten 423
D.4.4 Verwenden von skriptbasierter Feldvisualisierung für Kundenfelder 427
D.5 Templates für Kundendaten 428
D.5.1 Einführung in die Verwendung von Templates für die Anzeige von Kundendaten 428
D.5.2 Coding von Templates 430
D.5.3 Lokalisieren von Werten von Sortierten Listen in Kunden-Templates 431
D.5.4 Abkürzen von Werten in Kunden-Templates 432
D.5.5 Template-Typen 434
D.6 Verwaltung von Kundengruppen 445
D.6.1 Grundprinzip von Kundendatenmodellen und Kundengruppen 445
D.6.2 Verwaltung von Kundengruppen mit dem Admin Tool 446
D.6.3 Zuweisen von Zugangsberechtigungen für Kundengruppen 451
D.7 Kundenrollen 454
D.7.1 Einleitung 454
D.7.2 Definieren von Kundenrollen mit dem Admin Tool 455
D.8 Kundenrelationen 458
D.8.1 Einleitung 459
D.8.2 Verwaltung von Kundenrelationen mit dem Admin Tool 461
D.8.3 Erstellen von Kundenrelationen im Web Client 464
D.8.4 Verwendung von Relationen in Skripten 465
D.8.5 Kundenrelationen zu Ressourcen 465
D.9 Action Framework - Kundenaktionen 466
D.9.1 Einleitung 467
D.9.2 Verwaltung von Kundenaktionen mit dem Admin Tool 469
D.9.3 Verwendung von Kundenaktionen als Bearbeiter (Benutzer) 475

D.9.4 Beispiele für Kundenaktionsskripte 475
D.9.5 Skripte für das Action Framework: Programmierung von Kundenaktionen 480
D.10 Adress-Vervollständigung 498
D.10.1 Einleitung 499
D.10.2 Einschalten der Adress-Vervollständigung mit dem Admin Tool 502
D.10.3 Importieren von PLZ-/Ort-/Adressdaten in die CM-Datenbank 504
D.10.4 Definieren der Adress-Vervollständigungskonfiguration mit dem Admin Tool 505
D.10.5 Editieren einer Adress-Vervollständigungskonfiguration 509
D.10.6 Löschen einer Adress-Vervollständigungskonfiguration oder von Adress-Vervollständigungsfeldern. 509
E - Abschnitt Globale Konfiguration 510
E.1 Sprachen 511
E.1.1 Sprachen 511
E.1.2 Verwendung von Sprachumgebungen 513
E.2 Bezeichnungen 514
E.2.1 Einleitung 514
E.2.2 Konfigurieren der Bezeichnungen mit dem Admin Tool 514
E.3 Queue-Verwaltung 523
E.3.1 Einleitung 523
E.3.2 Queue-Verwaltung mit dem Admin Tool 524
E.4 Projekte 533
E.4.1 Einleitung 533
E.4.2 Verwalten von Projekten mit dem Admin Tool 533
E.5 Arbeiten mit Kalendern 535
E.5.1 Arbeitszeitkalender 536
E.5.2 Integration von Microsoft Exchange-Kalendern 543
E.6 Textklassen 556
E.6.1 Einleitung 556

E.6.2 Verwaltung von Textklassen mit dem Admin Tool 558
F - Abschnitt Experten 565
F.1 Ticketverwaltung 567
F.1.1 Einführung in die Ticketverwaltung 567
F.1.2 Ticketverwaltung mit dem Admin Tool 567
F.2 Verwaltung des Data Warehouse (DWH) 573
F.2.1 Einleitung 574
F.2.2 DWH-Verwaltung mit dem Admin Tool 576
F.2.3 System-Properties für das DWH 590
F.2.4 Übertragungsmodus 590
F.2.5 DWH-Informationen für Experten 592
F.3 CM-Services 594
F.4 Suche in ConSol CM 596
F.4.1 Konfiguration der Suche 597
F.4.2 ConSol CM-Indexer 609
F.4.3 Action Framework - Suche-Aktionen 620
F.4.4 CSV-Export von Suchergebnissen 637
F.5 Das Task Execution Framework (TEF) 640
F.5.1 Einleitung 640
F.5.2 Admin-Tool-Skripte des Typs Task 643
F.5.3 Programmierung mit Tasks 646
F.5.4 Für das TEF relevante System-Properties 652
F.6 Das ConSol CM Action Framework 653
F.6.1 Aktionen 653
F.6.2 Zusätzliche Komponenten 653
F.6.3 Skripte für das Action Framework 654
F.6.4 Aktionsformulare 670
F.7 E-Mail-Konfiguration 679
F.7.1 E-Mail 680

F.7.2 E-Mail-Backups 691
F.7.3 E-Mail-Verschlüsselung 694
F.8 Verwaltung von Admin-Tool-Skripten und -Templates 700
F.8.1 Admin-Tool-Skripte 701
F.8.2 Admin-Tool-Templates 757
F.9 Deployment (Import/Export) 770
F.9.1 Einleitung 770
F.9.2 Szenarios 771
F.9.3 Deployment (Import/Export) mit dem Admin Tool 772
F.10 Lizenzverwaltung 782
F.10.1 Allgemeine Informationen über Lizenzen in ConSol CM 782
F.10.2 Abschnitte einer Lizenzdatei 783
F.10.3 Verwaltung der ConSol CM-Lizenz mit dem Admin Tool 784
F.10.4 Experteninformationen über den Zugriff auf den Inhalt der CM-Lizenzen 786
F.11 System-Properties 788
F.11.1 Einleitung 788
F.11.2 Übersicht über die System-Properties 789
F.11.3 Setzen der System-Properties 792
F.11.4 Programmieren mit System-Properties 793
F.12 Arbeiten mit Textvorlagen 794
F.12.1 Der ConSol CM Textvorlagen-Manager 795
F.12.2 CM/Doc 838
F.13 Zeitbuchung mit ConSol CM 859
F.13.1 Allgemeine Einführung in die Zeitbuchung mit ConSol CM 860
F.13.2 Manuelle Zeitbuchungen 860
F.13.3 Automatische Zeitbuchung (verfügbar in CM-Versionen 6.9.4.2 und höher) 866
F.13.4 DWH-Reports 869
F.13.5 Seitenanpassung für Zeitbuchungen 869

F.13.6 Verwenden von Zeitbuchungsdaten in Skripten 869
F.14 Authentifizierungsmethoden in ConSol CM 872
F.14.1 Authentifizierungsmethoden für Bearbeiter im Web Client 873
F.14.2 Authentifizierungsmethoden für Kunden in CM/Track 895
F.15 Externe Schnittstellen in ConSol CM 907
F.15.1 Webhooks 908
F.16 Systemarchitektur 920
F.16.1 Architektur eines CM-Systems 921
G - Abschnitt Add-Ons 943
G.1 CM/Resource Pool 944
G.1.1 Einführung in CM/Resource Pool 945
G.1.2 CM/Resource Pool - Admin-Tool-Elemente 950
G.1.3 Kurze Einführung in die Funktionen von CM/Resource Pool im Web Client 953
G.1.4 CM/Resource Pool - Einrichten des Ressourcenmodells 969
G.1.5 CM/Resource Pool - Templates für Ressourcendaten 995
G.1.6 CM/Resource Pool - Ressourcenrelationen 1005
G.1.7 CM/Resource Pool - Ressourcenaktionen 1014
G.1.8 CM/Resource Pool - Zuweisen von Berechtigungen für Ressourcen 1034
G.1.9 CM/Resource Pool - Das Ressourcenpool-Dashboard 1035
G.2 CM/Doc 1045
G.3 Das Kundenportal: CM/Track 1046
G.3.1 Einleitung 1046
G.3.2 CM/Track-Versionen 1047
G.3.3 Authentifizierungsmethoden für Kunden in CM/Track 1048
G.3.4 CM/Track V1 1060
G.3.5 CM/Track V2 1075
G.4 CM/Phone: CTI mit ConSol CM 1101
G.4.1 Einführung in CM/Phone 1101

G.4.2 Setup von CM/Phone 1103
G.4.3 Konfigurieren von CM/Phone im Admin Tool 1104
H - Appendix 1112
H.1 Annotationen 1113
H.1.1 Liste der Feldannotationen 1114
H.1.2 Liste der Gruppenannotationen 1130
H.2 System-Properties 1135
H.2.1 Alphabetische Liste der System-Properties 1136
H.2.2 Liste der System-Properties nach Bereich 1251
H.2.3 Liste der System-Properties nach Modul 1297
H.3 Administrator-E-Mail-Adressen 1401
H.3.1 Einleitung 1402
H.3.2 Standardkonfiguration 1402
H.3.3 Spezifische Benachrichtigungs-E-Mail-Adressen für einzelne Unter-systeme 1404
H.4 Standardimports von Java-Klassen 1408
H.5 Liste der Code-Beispiele 1412
H.6 Marken 1416
Glossar 1418
Index 1425

A - Einleitung

Dieser Abschnitt enthält allgemeine Informationen über den Inhalt und die Struktur dieses Handbuchs, sowie eine Einführung in ConSol CM.

In diesem Kapitel werden folgende Themen behandelt:

A.1 ConSol CM für Business Process Management 14
A.2 Liste der Handbücher 16
A.3 TecDoc-Server 17
A.4 Die Struktur des Handbuches 18
A.5 Erklärungen zum Layout 20
A.6 Rechtlicher Hinweis 21
A.7 Gender-Disclaimer 21
A.8 Copyright 21
A.9 Grundprinzipien von ConSol CM 22
A.9.1 Systemkomponenten aus der Sicht der Anwender, Administratoren und Kunden 22
A.10 Grundlegende technische Prinzipien und Objekte von ConSol CM 24
A.10.1 Einleitung 25
A.10.2 Das ConSol CM-Dogma 25
A.10.3 Bearbeiter 26
A.10.4 Kunden 26
A.10.5 Tickets 26
A.10.6 Ressourcen 27
A.10.7 Queues 27
A.10.8 Workflows 28
A.10.9 Das CM Action Framework 30
A.10.10 Das Task Execution Framework 30
A.10.11 Zugriff auf Objekte in ConSol CM 30
A.10.12 ConSol CM aus der Perspektive eines Systemadministrators 32
A.11 Starten des Admin Tools 33
A.11.1 Login 33
A.11.2 Troubleshooting: Das Admin Tool startet nicht 36

A.12 Die grafische Benutzeroberfläche (GUI) des Admin Tools 41
A.12.1 Einleitung 41
A.12.2 Grundprinzip 41
A.12.3 Inline-Validierung von Eingabewerten 45
A.12.4 Symbole und andere GUI-Elemente 48
A.12.5 Lokalisierung der im Web Client angezeigten Begriffe 51

A.1 ConSol CM für Business Process Management

ConSol CM ist eine Low Code Platform, die besonders gut für den Einsatz als Customer Service-Software geeignet ist.

Mit ConSol CM können Sie Ihre Geschäftsprozesse kontrollieren und lenken. Der Fokus liegt dabei auf der zwischenmenschlichen Kommunikation und Zusammenarbeit, die in allen Bereichen des Customer Service Management erforderlich ist. Bekannte Beispiele von großen ConSol CM-Systemen umfassen Customer Service, RMA-Prozesse, After-Sales, Call-Center und Support-Center, sowie Umgebungen in den Bereichen Reklamations- und Beschwerdemanagement. Mit ConSol CM können Sie auch Kundenportale mit FAQ-Bereichen aufsetzen. Im Prinzip kann jeder in einem Unternehmen eingeführte Geschäftsprozess mit ConSol CM modelliert und zum Leben erweckt werden.

Ab Version 6.11 bietet ConSol CM zusätzlich die für Adaptive Case Management benötigten Funktionalitäten. Auf diese Weise können Sie entscheiden, ob Sie einen streng kontrollierten Geschäftsprozess entwerfen und leben möchten, oder ob ein höheres Maß an Flexibilität erforderlich ist. Je nach Team oder Abteilung können Sie auch beide Konzepte verbinden.

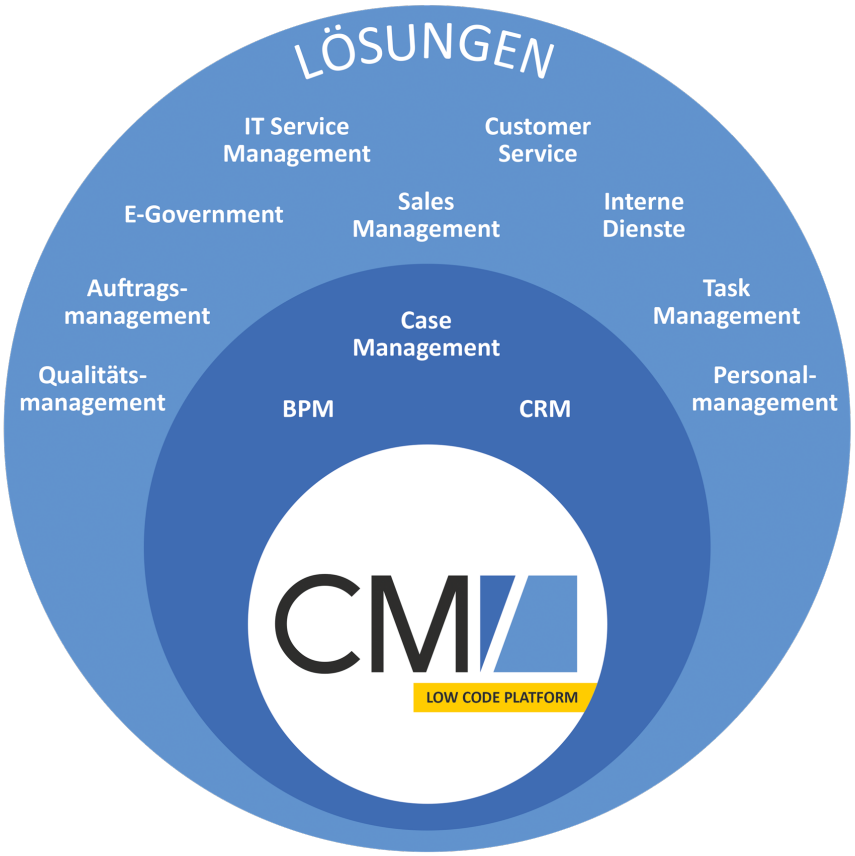


Abbildung 1: *Übersicht über mögliche Einsatzgebiete der Low Code Platform ConSol CM*

Mit ConSol CM können Sie alle für die Geschäftsprozesse relevanten Komponenten verwalten, um die Prozesse der Firma optimal darzustellen und zu lenken. ConSol CM wird in unterschiedlichen Industrien und Branchen verwendet, von Banken und Versicherungen über Modeunternehmen bis zu Herstellern von Automaten oder Waschanlagen. Der flexible Mechanismus für das Prozess-Design und die Workflow-Engine bieten die perfekte Grundlage für die Modellierung und Steuerung von Geschäftsprozessen aller Art, insbesondere im Bereich Customer Service.

A.2 Liste der Handbücher

ConSol CM enthält Dokumentation für mehrere Benutzergruppen. Folgende Dokumente sind verfügbar:

- **Administratorhandbuch**
  Ein detailliertes Handbuch für CM-Administratoren über die ConSol CM-Konfiguration mit dem Admin Tool.
- **Benutzerhandbuch**
  Ein Einführung in den ConSol CM Web Client für Endbenutzer.
- **Betriebshandbuch**
  Eine Beschreibung der ConSol CM-Infrastruktur, der Serverintegration in IT-Umgebungen und des Betriebs des CM-Systems für IT-Administratoren und -Betreiber.
- **DWH-Handbuch**
  Eine detaillierte Beschreibung des Konzepts des Data Warehouse (DWH) von ConSol CM, einschließlich des Datenbankschemas und einer Liste aller Tabellenstrukturen.
- **Process Designer-Handbuch**
  Ein Leitfaden für Workflow-Entwickler über die grafische Benutzeroberfläche des Process Designers und die Programmierung von Workflow-Skripten.
- **Setup-Handbuch**
  Eine technische Beschreibung des ConSol CM-Setups in verschiedenen IT-Umgebungen. Für Experten der CM-Administration.
- **Systemanforderungen**
  Eine Liste aller Voraussetzungen, die für die Installation von ConSol CM erfüllt sein müssen, für IT-Administratoren und CM-Administratoren. Mit jeder ConSol CM-Version veröffentlicht.
- **Technische Release Notes**
  Technische Informationen über die neuen ConSol CM-Funktionen. Für CM-Administratoren und Key-User. Mit jeder ConSol CM-Version veröffentlicht.

A.3 TecDoc-Server

Detaillierte Informationen über alle Aspekte von ConSol CM finden Sie auch auf unserem **TecDoc-Server** unter https://tecdoc.consol.de. Dort finden Sie:

- Alle **ConSol CM-Handbücher**
  Die ConSol CM-Handbücher für unterschiedliche Zielgruppen auf Englisch und Deutsch.
- Die **Release Notes** für jede ConSol CM-Version
  Detaillierte Release Notes-Dokumente, in denen alle neuen Funktionen jeder ConSol CM-Version erklärt sind.
- Die **Systemanforderungen**
  Eine Übersicht über die erforderliche Hardware, Middleware und Software, die für die Installation von ConSol CM Voraussetzung ist.
- Die **Präsentationen über neue Funktionen für Kunden**
  Nicht zu technische Präsentationen, die eine Übersicht über die neuen Funktionen von neuen ConSol CM-Versionen bieten.

A.4 Die Struktur des Handbuches

Zuerst werden die Grundprinzipien von ConSol CM erklärt. Damit erhalten Sie den theoretischen Hintergrund, den Sie benötigen, um CM-Administrator zu werden.

- In der **Einleitung** ist erklärt, wie ConSol CM zur Verwaltung von Geschäftsprozessen verwendet wird. Außerdem lernen Sie einige Grundprinzipien der Applikation kennen. Zudem erfahren Sie, wie das Admin Tool, die wichtigste Administrationsapplikation von ConSol CM, gestartet wird.

In den folgenden Abschnitten werden die Features und Funktionalitäten vom Admin Tool beschrieben.

- **Zugriff und Rollen**
  In diesem Abschnitt sind die grundlegenden Prinzipien und die Konfiguration der Zugangsberechtigungen erklärt. Dort lernen Sie zum Beispiel, wie Sie Rollen definieren und den Bearbeitern (den Benutzern des CM Web Clients) zuweisen und wie Sie Sichten (die To-do-Listen des Systems) konfigurieren.
- **Ticketdatenmodell und GUI-Design**
  Hier sind die Einrichtung des Ticketdatenmodells und die Anordnung der Datenfelder im Web Client (GUI) beschrieben. Dort lernen Sie zum Beispiel, wie Sie die Datenfelder definieren, die für einen bestimmten Prozess benötigt werden, und wie Sie verschiedene Arten von Listen anlegen.
- **Kundendatenmodell**
  In diesem Abschnitt sind die Einrichtung der Datenmodelle für verschiedene Kundengruppen und das dazugehörige GUI-Design beschrieben. Die Abbildung der Kundengruppen in CM basiert auf ***FlexCDM***, dem flexiblen Kundendatenmodell. Dort lernen Sie zum Beispiel, wie Sie einen Satz mit Datenfeldern für die Kundengruppe *Händler* definieren können und ein anderes Datenmodell für die Kundengruppe *Direktkunden*. Außerdem werden dort Kundenrelationen und Kundenaktionen erklärt, zwei Komponenten, die Sie bei der Verwendung von CM als CRM-System unterstützen.
- **Globale Konfiguration**
  In diesem Abschnitt werden einige allgemeine Einstellungen erklärt. Sie lernen dort, wie Queues, zentrale Komponenten von ConSol CM, verwaltet werden. Zudem ist dort beschrieben, wie Sie mit Arbeitszeitkalendern und Projekten arbeiten.
- **Experten**
  Die Zielgruppe dieses Teils des Handbuches sind die CM-Administratoren, die für die erweiterte Konfiguration des CM-Systems zuständig sind. Änderungen an den Systemeinstellungen nehmen Sie am besten zusammen mit einem CM-Consultant vor. In diesem Abschnitt sind folgende Themen beschrieben:
  - Vorbereitung des CM-Systems für Reporting
  - ConSol CM-Dienste
  - Konfiguration des Suchmoduls
  - Task Execution Framework (TEF)
  - E-Mail-Konfiguration

  - Import und Export von Konfigurationen
  - Templateverwaltung
  - Arbeit mit System-Properties
  - Arbeit mit Admin-Tool-Skripten
  - Lizenzverwaltung
  - Installation von Szenarien
- **Add-Ons**
  In diesem Abschnitt sind die drei CM-Add-ons beschrieben. Ein CM-Add-on ist ein CM-Modul, für das eine eigene Lizenz erworben werden muss. Es sind folgende Add-ons verfügbar:
  - **CM/Resource Pool**
    In diesem Abschnitt ist das Datenmodell für den Ressourcenpool erklärt. CM/Resource Pool ist ein eigenes CM-Modul, das separat lizenziert werden muss. Wenn Sie dieses Modul erworben haben, können Sie diesem Abschnitt entnehmen, wie Sie unterschiedliche Objekte, wie IT-Assets, Produkte, SLAs oder andere Objekte, als CM-Ressourcen abbilden können. Neben dem Einrichten des Datenmodells sind auch Ressourcenrelationen und Ressourcenaktionen beschrieben.
  - **CM/Track**
    In diesem Abschnitt ist das Kundenportal, CM/Track, erklärt. Für CM/Track sind die Versionen V1 und V2 verfügbar. Diese werden in eigenen Kapiteln behandelt.
  - **CM/Phone**
    In diesem Abschnitt ist die CTI-Lösung von ConSol CM erklärt.
- **Anhang**
  Hier finden Sie Listen mit allen wichtigen Begriffen, die in diesem Handbuch verwendet werden (Glossar), Listen mit allen Annotationen (wichtig für das GUI-Design), und Listen mit System-Properties (wichtig für die Administration des CM-Systems). Beachten Sie auch die Seite mit den Handelsmarken.

A.5 Erklärungen zum Layout

Es werden folgende Symbole und Farben verwendet, um Informationen hervorzuheben bzw. zu markieren.

Dies ist eine zusätzliche Information.

Dies ist ein wichtiger Hinweis. An dieser Stelle müssen Sie besonders aufpassen!

Dies ist eine Warnung!

Dies ist eine Empfehlung aus der praktischen Erfahrung unserer Consultants.

A.6 Rechtlicher Hinweis

Da wir ein Handbuch zur Verfügung stellen möchten, das Sie bei der Verwaltung Ihres CM-Systems unterstützt und Ihnen gleichzeitig zusätzliche Informationen über verwandte Themen bietet, enthält dieses Handbuch externe Links. Auf diese Weise können Sie bei Bedarf Hintergrundinformationen über ein Thema erhalten. Dies kann Ihnen dabei helfen, die benötigte CM-Konfiguration besser zu verstehen. Trotz sorgfältiger Prüfung übernehmen wir keine Haftung für den Inhalt dieser externen Links. Für ihren Inhalt sind ausschließlich die Betreiber der verlinkten Seiten verantwortlich.

A.7 Gender-Disclaimer

Soweit möglich sind ConSol CM-Handbücher geschlechtsneutral geschrieben und sprechen Sie als Leser oft mit "Sie" an. Wenn Formulierungen wie "Der Benutzer ..." verwendet werden, bezieht diese gewählte männliche Form immer gleichermaßen weibliche Personen ein. Auf konsequente Doppelbezeichnung wurde aufgrund der besseren Lesbarkeit verzichtet.

A.8 Copyright

© 2018 ConSol Consulting & Solutions Software GmbH - Alle Rechte vorbehalten.

A.9 Grundprinzipien von ConSol CM

A.9.1 Systemkomponenten aus der Sicht der Anwender, Administratoren und Kunden

ConSol CM umfasst verschiedene Client-Applikationen. Je nach Ihrer Rolle und Ihren Aufgaben im Unternehmen werden Sie eine oder mehrere dieser Applikationen verwenden.

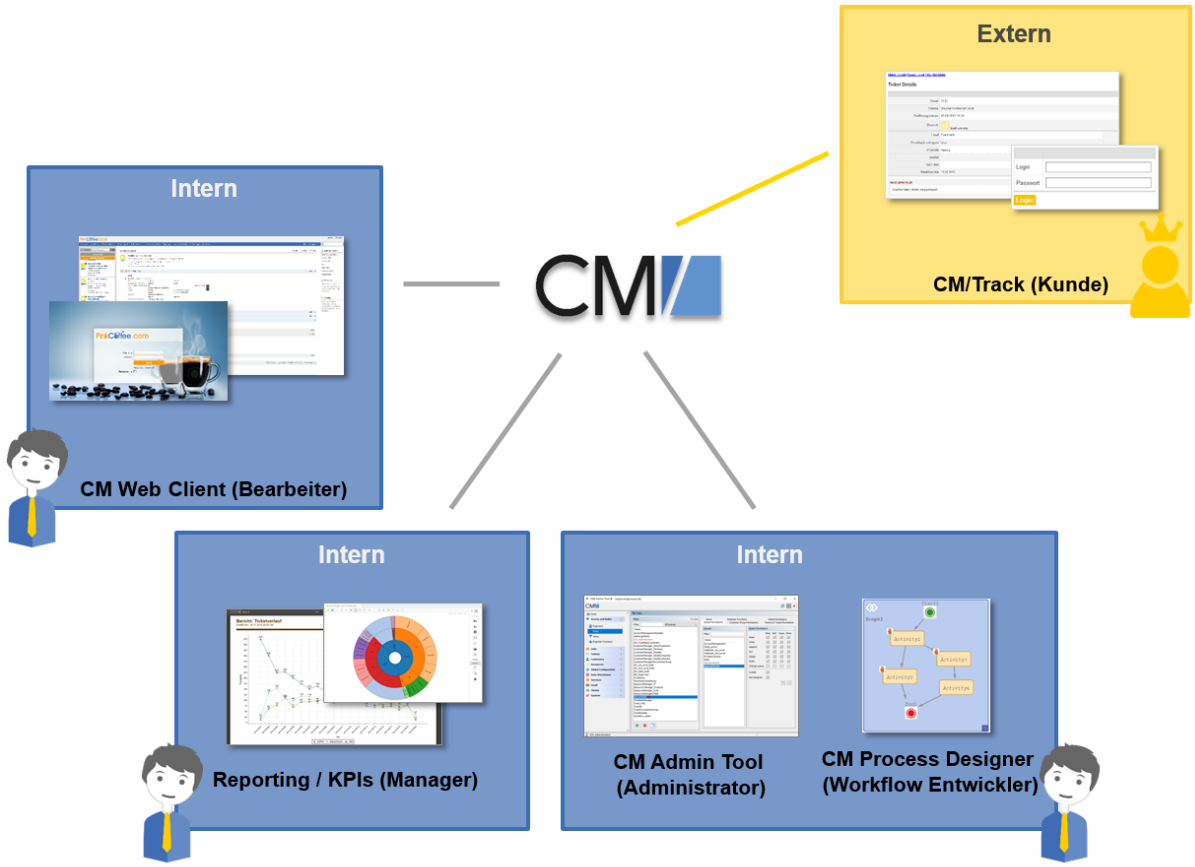


Abbildung 2: *ConSol CM-Systemkomponenten*

- **Web Client**
  Der Hauptzugang zum System für Bearbeiter, die Standardbenutzer des Systems. Bearbeiter arbeiten mit Tickets, Kundendaten und Ressourcen.
- **Portal**
  CM/Track, der Hauptzugang zum System für (interne oder externe) Kunden. CM/Track ist ein eigenes CM-Modul, das separat lizenziert werden muss. Mit diesem Modul können Sie Ihren Kunden über das Portal Zugang zu den Tickets anbieten. Außerdem können Sie FAQs im Internet zur Verfügung stellen.

- **Admin Tool**
  Für alle Aufgaben im Zusammenhang mit der Systemkonfiguration. Als Administrator werden Sie hauptsächlich mit diesem Tool arbeiten. Es wird zur Einrichtung des Systems verwendet. Alle Einstellungen (mit Ausnahme der Workflows) werden im Admin Tool konfiguriert und der Zugang ist auf Benutzer mit Administratorrechten beschränkt.
- **Process Designer**
  Für das Design und die Implementierung der Workflows. Als Workflow-Entwickler werden Sie hauptsächlich mit dem Process Designer arbeiten. In diesem Tool werden alle Workflows graphisch und mit Groovy-Code umgesetzt.

Im Standardlieferumfang ist außerdem ein Data Warehouse (DWH) enthalten, das Reporting über alle Daten Ihrer Tickets ermöglicht.

ConSol CM ist keine isolierte Applikation, sondern kann einfach in die IT-Infrastruktur Ihres Unternehmens integriert werden, zum Beispiel über Web Services oder einen Enterprise Service Bus (ESB) oder die ConSol CM-Webhook-Schnittstelle.

Eine detaillierte Beschreibung der Systemkomponenten aus technischer Sicht finden Sie im Abschnitt Architektur eines CM-Systems für Administratoren.

A.10 Grundlegende technische Prinzipien und Objekte von ConSol CM

In diesem Kapitel werden folgende Themen behandelt:

A.10.1 Einleitung 25
A.10.2 Das ConSol CM-Dogma 25
A.10.3 Bearbeiter 26
A.10.4 Kunden 26
A.10.5 Tickets 26
A.10.6 Ressourcen 27
A.10.7 Queues 27
A.10.8 Workflows 28
A.10.9 Das CM Action Framework 30
A.10.10 Das Task Execution Framework 30
A.10.11 Zugriff auf Objekte in ConSol CM 30
A.10.12 ConSol CM aus der Perspektive eines Systemadministrators 32

A.10.1 Einleitung

Um effizient und korrekt mit ConSol CM arbeiten zu können, müssen Sie ein fundiertes Wissen aller Komponenten eines CM-Systems haben. Der folgende Abschnitt enthält eine erste Einleitung über die grundlegenden CM-Komponente.

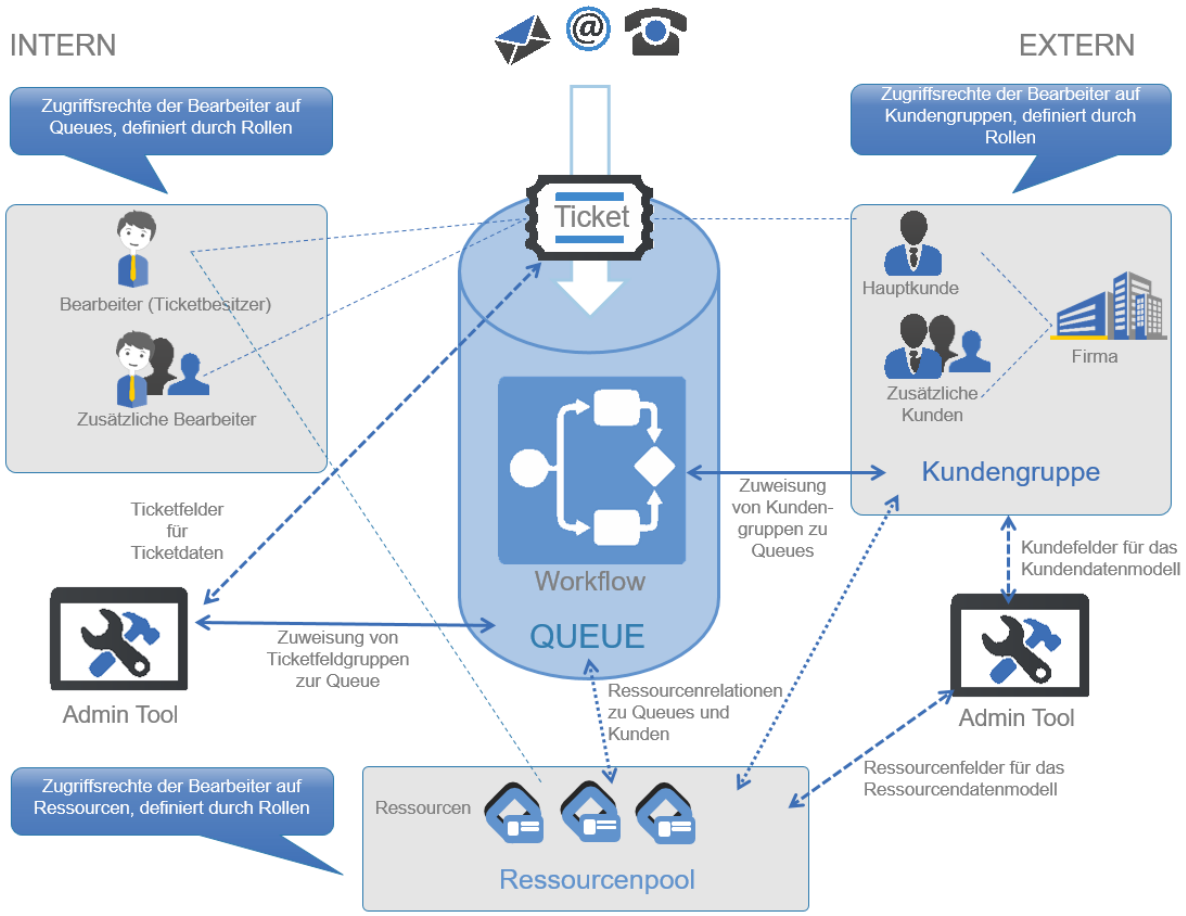


Abbildung 3: *ConSol CM - Grundprinzip (mit CM/Resource Pool)*

A.10.2 Das ConSol CM-Dogma

In ConSol CM gibt es ein Dogma:

> **ConSol CM DOGMA**
>
> **Externe** Seite: Ein Ticket hat immer einen **Hauptkunden**. Das kann ein Kontakt oder eine Firma sein.
>
> **Interne** Seite: Ein Ticket kann keinen oder einen **Hauptbearbeiter** haben, der an dem Ticket arbeiten muss.

A.10.3 Bearbeiter

Die Bearbeiter stellen die **interne** Seite des CM-Systems dar. Alle Benutzer des Web Clients heißen Bearbeiter, unabhängig von ihrer Funktion innerhalb des Unternehmens. Die Bearbeiter arbeiten an den Tickets, um die im Geschäftsprozess definierten Aufgaben auszuführen. Jedes Ticket kann nur einem Bearbeiter zugewiesen sein, der aktuell für das Ticket verantwortlich ist. Ein Ticket kann aber eine beliebige Anzahl an zusätzlichen Bearbeitern haben, die alle eine Bearbeiterfunktion haben, die eine bestimmte Aufgabe im Prozess darstellt.

Alle Bearbeiter haben ein Benutzerkonto, das aus einem Benutzernamen und einem Passwort besteht. Mit diesem Konto melden sie sich im Web Client an. Die Zugangsberechtigungen der Bearbeiter werden mithilfe von Rollen verwaltet. Die Rollen, die die Zugangsberechtigungen für Queues, Kunden und Ressourcen enthalten, werden im Admin Tool definiert und den Bearbeitern zugewiesen.

Details dazu finden Sie im Abschnitt *Bearbeiterverwaltung*.



Es hängt von der Konfiguration Ihres ConSol CM-Systems ab, ob die *Bearbeiter* im System *Bearbeiter* genannt werden. Bearbeiter können im Web Client auch *Agent*, *Mitarbeiter* oder ähnlich heißen. In diesem Handbuch werden Bearbeiter der Einfachheit halber immer Bearbeiter genannt.

A.10.4 Kunden

Mit FlexCDM, dem flexiblen Kundendatenmodell, bietet ConSol CM ein Datenmodell, mit dem Kontakt- und Firmendaten in verschiedenen Konstellationen definiert werden können. Auf diese Weise können Sie sehr einfache einstufige Datenmodelle, die nur Kontaktdaten (z. B. Name, Telefonnummer, E-Mail-Adresse, Adresse) enthalten, und komplexe zweistufige Modelle, die Kontaktdaten (z. B. Name, Telefonnummer, E-Mail-Adresse) und Firmendaten (z. B. Adresse, Postleitzahl, Firmengröße) enthalten, definieren. Sie können unterschiedliche Modelle innerhalb eines Systems definieren, Relationen zwischen Kunden konfigurieren und Aktivitäten zu Kontakten und Firmen hinzufügen. Details zu allen Komponenten von FlexCDM finden Sie im Abschnitt *Kundendatenmodell*, der mit dem Kapitel *Das CM-Kundendatenmodell - FlexCDM* beginnt.



Jedes CM-System verwendet angepasste Kundengruppen und -datenmodelle. Deshalb hängen die verfügbaren Kundengruppen, Hierarchiestufen für Kundenobjekte, Datenfelder, Relationen und Aktivitäten von der individuellen Konfiguration Ihres CM-Systems ab.

A.10.5 Tickets

Das Ticket ist die Kundenanfrage, an der die Bearbeiter arbeiten. Das kann ein Vorfall, ein Servicefall oder eine andere Art von Anfrage sein. Für jede Anfrage wird ein Ticket erstellt. Die Bearbeiter arbeiten an dem Ticket, das heißt, sie führen die notwendigen, im Geschäftsprozess definierten Schritte aus. Der Fortschritt, einschließlich der internen und externen Kommunikation, wird im Ticket dokumentiert. Am Geschäftsprozess können mehrere Bearbeiter und unterschiedliche Teams beteiligt sein. Wenn eine Anfrage gelöst ist, wird das Ticket geschlossen. Geschlossene Tickets gehen nicht verloren, sondern bilden ein umfassendes Archiv und Wissenspool.

In ConSol CM gelten folgende Regeln für Tickets:

- Ein Ticket muss einen Hauptkunden haben. Ein Ticket kann nur einen Hauptkunden haben. Es muss keine Zusatzkunden haben, aber es kann eine beliebige Anzahl an Zusatzkunden haben. Der Kunde stellt die **externe** Seite eines Tickets dar.
- Ein Ticket muss nicht einem Bearbeiter zugewiesen sein; aber wenn es zugewiesen ist, kann es nur einem Bearbeiter gleichzeitig zugewiesen sein. Ein Ticket muss keine zusätzlichen Bearbeiter haben; es kann eine beliebige Anzahl zusätzlicher Bearbeiter haben. Die Zuweisung eines Tickets zu einem Bearbeiter kann manuell oder automatisch erfolgen. Die Bearbeiter stellen die **interne** Seite eines Tickets dar.
- Ein Ticket hat immer einen Namen, auch *Ticketnummer* genannt, einen Betreff und ein Ticket-Icon. Das Ticket-Icon zeigt den Bereich, in dem sich das Ticket aktuell befindet, und kann eine Farbe haben, die den Wert eines gegebenen Datenfeldes angibt. Jedes Ticket hat eine ID, die intern verwendet wird und für den Benutzer nicht sichtbar ist.
- Im Kopfbereich des Tickets stehen die aktuelle Queue, der aktuelle Bereich, der zugewiesene Bearbeiter und das Erstellungsdatum des Tickets.



Es hängt von der Konfiguration Ihres ConSol CM-Systems ab, ob ein Ticket in Ihrem System *Ticket* heißt. Tickets können in Ihrem Web Client *Ticket*, *Vorfall*, *Call*, *Aufgabe* oder ähnlich genannt werden. Jede Queue kann ihren eigenen Begriff für Tickets haben. In diesem Handbuch werden Tickets der Einfachheit halber immer Ticket genannt.

Jedes CM-System verwendet angepasste Ticketdaten. Deshalb hängen die verfügbaren Felder, Relationen und Aktivitäten von der individuellen Konfiguration Ihres CM-Systems ab.

A.10.6 Ressourcen

Ressourcen können verwendet werden, um Objekte zu verwalten, die mit dem Geschäftsprozess verbunden sind. Mögliche Anwendungsfälle sind IT-Assets, SLAs, Produkte oder Newsletter. Alle Ressourcen werden in ***CM/Resource Pool***, einem separaten CM-Modul, gespeichert. Der Administrator definiert das Ressourcenmodell, d. h. die Ressourcentypen, Ressourcendatenfelder, Hierarchie der Ressourcen und die möglichen Relationen zu Tickets, Kunden und anderen Ressourcen.

Als Bearbeiter mit den erforderlichen Berechtigungen können Sie Ressourcen erstellen und mit vorhandenen Tickets, Kunden und anderen Ressourcen verknüpfen. Sie können zum Beispiel einen Computer mit einem Ticket, das wegen einem Problem mit diesem Computer geöffnet wurde, oder mit dem Kunden, der den Computer verwendet, verknüpfen.

Details zu diesem Thema finden Sie im Abschnitt über CM/Resource Pool unter dem Abschnitt Add-ons.

A.10.7 Queues

Die Queue ist die **Kernkomponente** der ConSol CM-Administration. Sie enthält thematisch ähnliche Tickets, die gleich behandelt werden sollen und dem gleichen Geschäftsprozess folgen. Jede Queue hat genau einen **Workflow**, in dem der gewünschte Prozess umgesetzt ist. Die für den Prozess benötigten Ticketdatenfelder werden der Queue zugewiesen. Außerdem basieren die Zugangsberechtigungen, die den Bearbeitern über Rollen erteilt werden, auf Queues.

Die Queue bestimmt daher:

- wie die Tickets aussehen (Ticketdatenfelder)
- wessen Tickets bearbeitet werden (Kundengruppen)
- wie die Tickets bearbeitet werden (Workflow)
- wer an den Tickets arbeiten kann (Berechtigungen)

Queues bilden häufig die Organisationsstruktur des Unternehmens ab. Zum Beispiel kann es eine Queue für jede Abteilung geben, da jede Abteilung ihre eigenen Prozesse hat. Ein Ticket kann von einer Queue in eine andere Queue übergeben werden. In diesem Fall passt es sich der neuen Queue an, d. h. es erhält die Datenfelder der neuen Queue und nur die Bearbeiter mit Berechtigungen für die neue Queue können es bearbeiten

> Es hängt von der Konfiguration Ihres ConSol CM-Systems ab, ob die Queues im System *Queues* genannt werden. Queues können im Web Client auch *Prozesse*, *Teams* oder ähnlich heißen. In diesem Handbuch werden Queues der Einfachheit halber immer Queues genannt.

A.10.8 Workflows

Ein Workflow wird von einem CM-Workflow-Entwickler im ConSol CM Process Designer entworfen und erstellt. Ein Workflow bildet den Geschäftsprozess ab, der im Web Client ausgeführt wird. Der Workflow besteht aus mehreren Schritten, den sogenannten **Aktivitäten**. Es gibt manuelle Aktivitäten, die von den Bearbeitern ausgeführt werden, und automatische Aktivitäten, die vom System ausgeführt werden. Die Aktivitäten sind in *Bereichen* angeordnet, um den Status eines Tickets zu veranschaulichen. Die **Intelligenz** des Prozesses, wie Bedingungen, Entscheidungen, Eskalationen, Erinnerungen, automatisch versendete E-Mails oder andere Aktionen, wird ebenfalls im Workflow definiert. Sie können Prozessketten oder hierarchische Prozessstrukturen implementieren, indem Sie mehrere Workflows verknüpfen.

Als Bearbeiter arbeiten Sie nicht mit dem Workflow selber, sondern sehen die aktuelle Queue (1) und den Bereich des Tickets (2). Der Bereich ist am Symbol des Ticket-Icons und am Bereichsnamen in den Standard-Ticketdaten erkennbar. Außerdem sehen Sie die Workflow-Aktivitäten, die für das Ticket an seiner aktuellen Position verfügbar sind (3). Auf diese Art und Weise haben Sie einen guten Überblick über den aktuellen Status des Tickets.

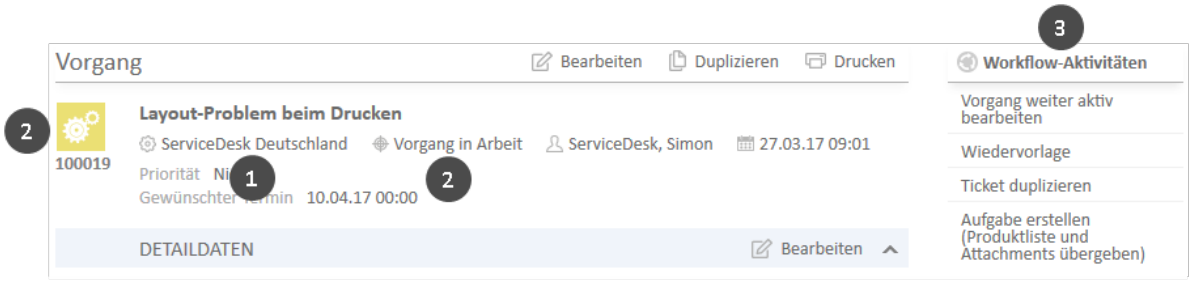


Abbildung 4: *Queue, Bereich und Workflow im Ticket*

Als Administrator können Sie mit dem ConSol CM Process Designer den Geschäftsprozess Ihres Unternehmens modellieren. Ein Prozess kann durch einen oder mehrere Workflows abgebildet werden.

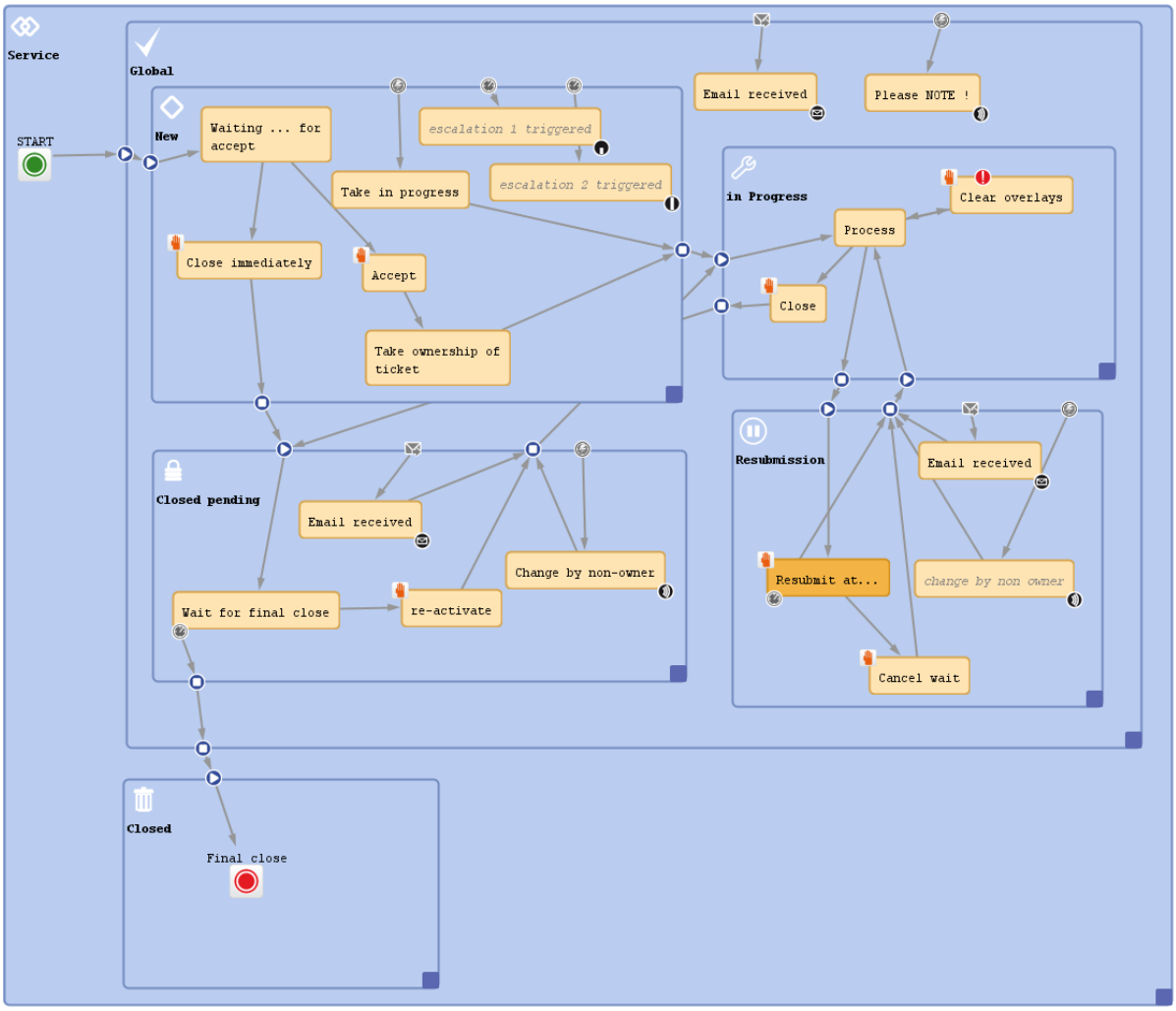


Abbildung 5: *Process Designer: Beispiel für einen Service-Workflow*

Da wir es häufig mit Prozessketten statt mit einzelnen Prozessen zu tun haben, können solche Prozessketten in ConSol CM entworfen werden, indem eine bestimmte Reihenfolge für die Prozesse festgelegt wird. Sie können mit einfachen Prozessketten oder mit einer hierarchischen Struktur arbeiten.

Ein Ticket startet zum Beispiel in einem Eingangspool, wird dann an das 1st Level-Team weitergeleitet, das es an das 2nd Level-Netzwerkteam übergibt. Oder ein Sales-Ticket beginnt als Kundenanfrage, wird zum Lead, der weiterverfolgt wird, sodass eine Opportunity entsteht. Sobald der Kunde den Vertrag unterschrieben hat, wird ein Auftragsticket erstellt, das sogenannte Child-Tickets für die internen Aufgaben bis zur Rechnungsstellung erzeugt. Wenn alle Child-Tickets geschlossen sind, kann das Parent-Ticket ebenfalls geschlossen werden.

Die Intelligenz des Prozesses, wie Eskalationen, Erinnerungen, automatisch erzeugte E-Mails oder andere Aktionen während des Prozesses werden im Workflow mittels Groovy-Skripten definiert.

Eine detaillierte Einführung in das Design und die Modellierung von Prozessen mithilfe des Process Designers finden Sie im ConSol CM Process Designer Handbuch.

A.10.9 Das CM Action Framework

Neben Workflow-Aktivitäten, die bei einem bestimmten Schritt eines Geschäftsprozesses ausgeführt werden, können Aktivitäten, hier Aktionen genannt, auch von anderen Objekten ausgelöst werden:

Kundenaktionen sind Aktionen, die für ein Kundenobjekt ausgeführt werden, d. h. für einen Kontakt oder für eine Firma. Auf diese Weise können Sie zum Beispiel eine Firmenaktion implementieren, die jede Nacht Ihre firmenspezifischen Verkaufszahlen aktualisiert.

Ressourcenaktionen sind Aktionen, die für eine Ressource ausgeführt werden, d. h. für ein Objekt aus CM/Resource Pool. Auf diese Weise können Sie im Web Client eine Aktion anzeigen, die dem Bearbeiter eine Liste aller Kunden für einen bestimmten Newsletter bereitstellt.

Suche-Aktionen sind Aktionen, die für die Ergebnisse einer Detailsuche ausgeführt werden. Auf diese Weise können Sie beispielsweise eine E-Mail an alle Kunden einer Liste an Tickets, die Sie über die Suche abgerufen haben, senden.

Alle Aktionen können entweder manuell oder automatisch ausgeführt werden. Manuelle Aktionen werden im Web Client wie Workflow-Aktivitäten angezeigt. Automatische Aktionen werden ohne Beteiligung eines Bearbeiters im Hintergrund ausgeführt.

A.10.10 Das Task Execution Framework

Mit dem Task Execution Framework (TEF) werden langlaufende Tasks, die mit keiner bestimmten Aktivität verbunden sind, gespeichert und ausgeführt. TEF-Tasks können zum Beispiel für Importskripte hilfreich sein.

A.10.11 Zugriff auf Objekte in ConSol CM

In ConSol CM bilden die unterschiedlichen Objekte (Tickets, Kunden und Ressourcen) ein Netzwerk. Die Objekte sind miteinander verbunden, z. B. ist ein Ticket immer mit einem oder mehreren Kunden verknüpft. Die Verbindungen, die zwischen dem aktuellen Objekt und anderen Objekten in ConSol CM bestehen, werden auf der Seite des Objekts angezeigt, wo Sie auch direkt auf die verknüpften Objekte zugreifen können.

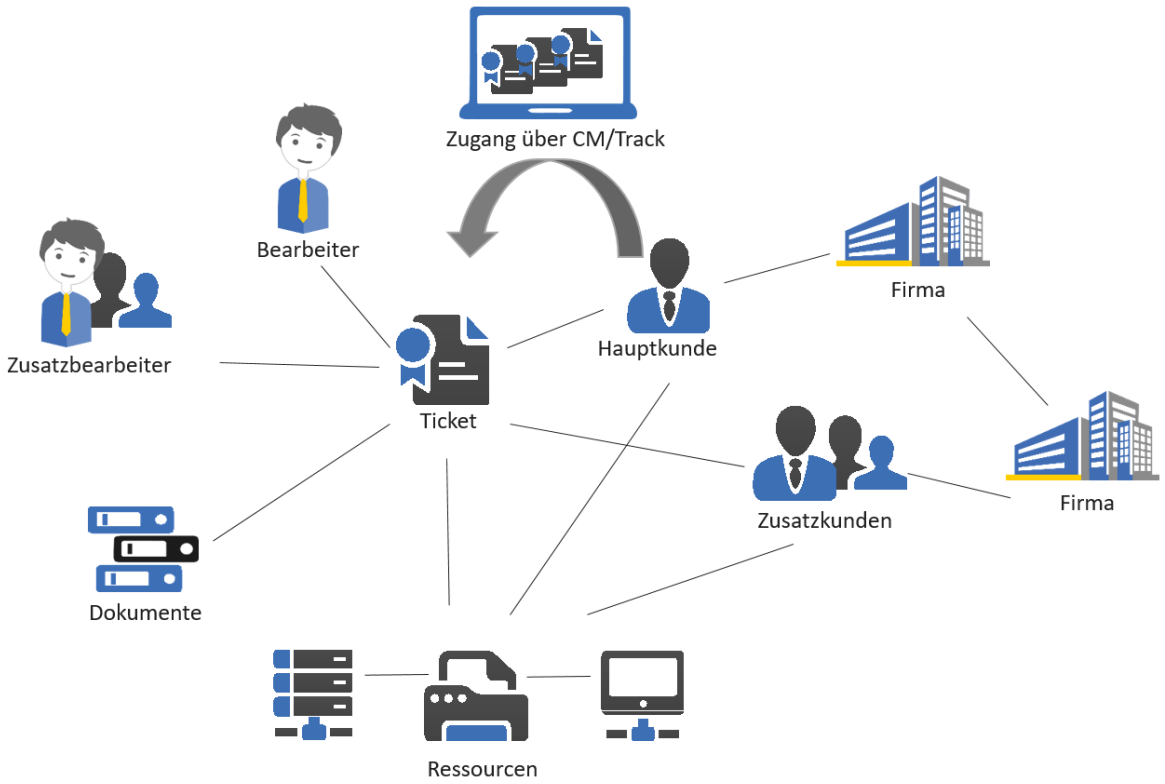


Abbildung 6: *Netzwerk der ConSol CM-Objekte*

Da ConSol CM immer Kontextmenüs oder Links bereitstellt, um die Objekte zu öffnen, die mit dem aktuell geöffneten Objekt in Beziehung stehen, können Sie sich als CM-Bearbeiter einfach in dem Netzwerk bewegen und somit effektiv mit Kunden-, Ticket- und Ressourcendaten arbeiten. Wenn Sie ein Objekt (z. B. ein Ticket) geöffnet haben, können Sie von einem verknüpften Objekt (z. B. dem Hauptkunden des Tickets) zum nächsten verknüpften Objekt (z. B. die Firma des Hauptkunden oder eine Ressource, die mit Hauptkunden oder dem Ticket verknüpft ist) navigieren. Es wird keine weitere Suche benötigt.

Ein Beispiel-Anwendungsfall könnte folgendermaßen aussehen:

1. Ein Kunde ruft bei Ihnen an und fragt nach einem bestimmten Vorgang, aber er kann sich nicht an die Ticketnummer erinnern. Er möchte wissen, welches SLA für den Drucker greift, der das Problem verursacht hat, das im Ticket behandelt wird. Kann der Vertrag eventuell geändert werden?
2. Sie starten die Schnellsuche nach dem Kundennamen und dem Stichwort "Drucker". Sie finden Ticket Nummer 0815 und öffnen es.
3. Sie sehen sich die mit dem Ticket verknüpften Ressourcen an und finden den Drucker Nummer 4711.
4. Sie prüfen, welches SLA als Ressource mit dem Drucker verknüpft ist, und öffnen es. Als Bemerkung in der SLA-Ressource finden Sie "... kann innerhalb von zwei Wochen geändert werden". Nun müssen Sie wissen, welche SLAs prinzipiell für die entsprechende Firma möglich sind.

5. Sie prüfen die Relation der Ressource (Drucker 4711) zu der Firma und öffnen die entsprechende Firmenseite.
6. Die Firma hat drei weitere SLA-Ressourcen für Drucker (d. h. Ressourcenrelationen auf der Firmenseite). Sie besprechen mit dem Kunden, welche in diesem Fall am besten für ihn in Frage kommt. Der Kunde wünscht einen der SLAs für den Drucker.
7. Da diese Änderung zunächst von beiden Seiten genehmigt werden muss, erstellen Sie ein SLA-Änderungsticket direkt von der Firmenseite aus ...

Dieses kurze Beispiel zeigt, wie einfach ein Bearbeiter alle für einen Vorgang oder eine Anforderung relevanten Daten erreichen kann. Wenn Kundenbeziehungen ebenfalls in ConSol CM gepflegt werden, ist die CRM-Komponente (Customer Relationship Management) noch stärker ausgeprägt.

A.10.12 ConSol CM aus der Perspektive eines Systemadministrators

ConSol CM ist eine Java EE-Applikation, die auf einem Standard-Application Server ausgeführt wird. Die Daten werden in einer relationalen Datenbank gespeichert. ConSol CM stellt eine Verbindung zu einem E-Mail-Server her, um die eingehenden E-Mails abzurufen und über einen SMTP-Server E-Mails zu senden. Eine detaillierte Erklärung aller Aspekte des Betriebs von ConSol CM in einer IT-Umgebung finden Sie im *ConSol CM Operations Manual*. Eine erste Einführung finden Sie im Abschnitt *Systemübersicht* in diesem Handbuch.



Eine detaillierte Liste der unterstützten Betriebssysteme, Application Server, Datenbanksysteme und anderen Systeme, sowie der Anforderungen an CPU und Speicherplatz ist in den aktuellen *Systemanforderungen* enthalten.

A.11 Starten des Admin Tools

In diesem Kapitel werden folgende Themen behandelt:

A.11.1 Login 33

A.11.2 Troubleshooting: Das Admin Tool startet nicht 36

A.11.1 Login

Der Großteil der Administration des ConSol CM-Systems wird mit der Java Web Start-Applikation ***Admin Tool*** vorgenommen, die über die Startseite des Application Servers des CM-Systems aufgerufen werden kann. Um das Admin Tool zu starten, können Sie entweder auf den Link auf der Seite klicken oder die jnlp-Datei lokal speichern und von dort aus starten. Java Web Start ist Teil der Standard-JRE.

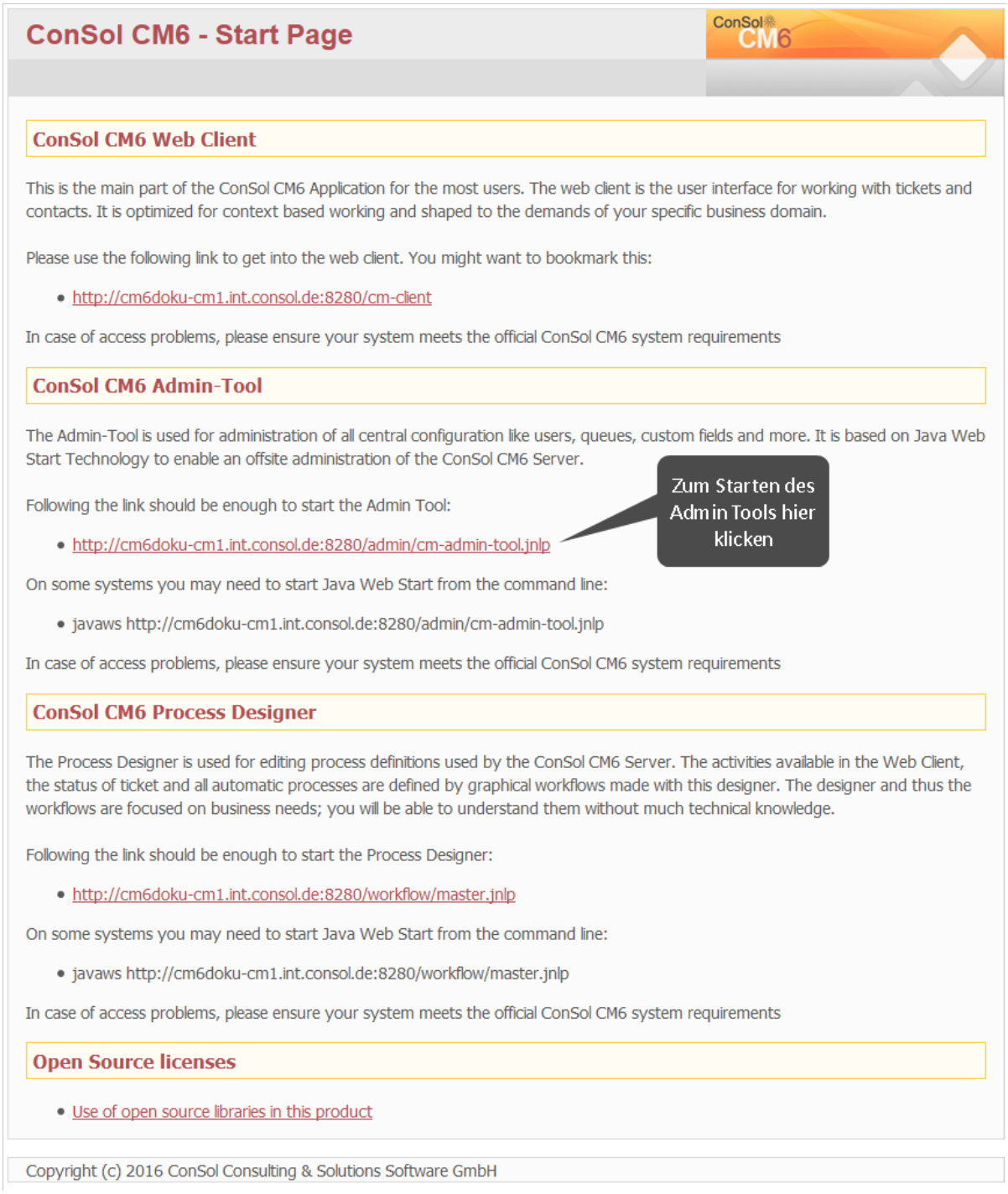


Abbildung 7: *ConSol CM-Startseite*

Nachdem Sie auf `Admin Tool.jnlp` geklickt haben, wird die jnlp-Datei heruntergeladen, das Admin Tool wird gestartet und das Login-Fenster angezeigt (weitere Details finden Sie im Abschnitt Troubleshooting: Das Admin Tool startet nicht).

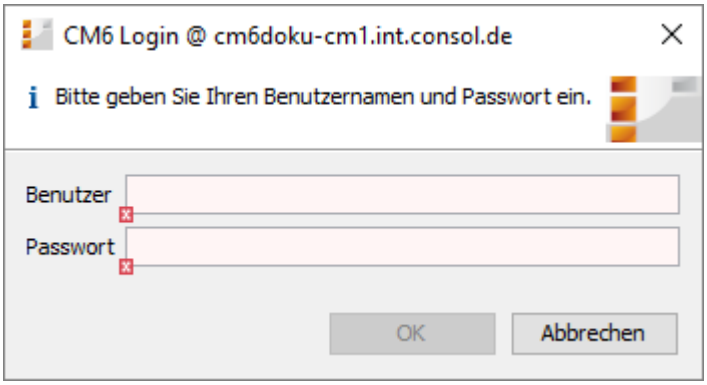


Abbildung 8: *ConSol CM Admin Tool - Login-Fenster*

Geben Sie Ihre Anmeldedaten ein, um Zugang zu den Funktionen des Admin Tools zu erhalten. Der erste Benutzername und das Passwort werden bei der Einrichtung des Systems vergeben. Später können mit dem Admin Tool weitere Administratoren konfiguriert werden.

Nach der erfolgreichen Anmeldung wird die Startseite des Admin Tools angezeigt:

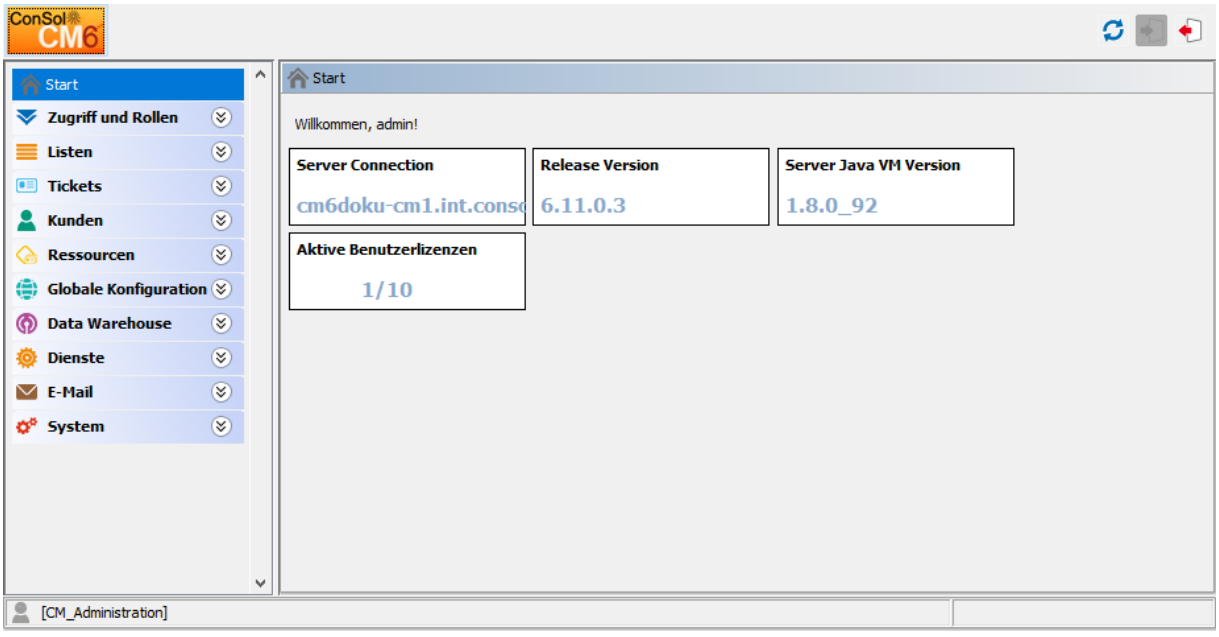


Abbildung 9: *ConSol CM Admin Tool - Startseite*

A.11.2 Troubleshooting: Das Admin Tool startet nicht

A.11.2.1 Richtiger Ablauf

Wenn alles richtig eingerichtet ist, werden nach dem Klick auf den Admin-Tool-Link folgende Schritte durchgeführt:

1. Sie werden in einem Pop-up-Fenster gefragt, ob Sie die jnlp-Datei öffnen möchten - ***Java (TM) Web Start Launcher*** sollte dafür als Standardapplikation angeboten werden - oder ob Sie die jnlp-Datei in Ihr System herunterladen möchten.
   Bestätigen Sie *Öffnen mit Java (TM) Web Start Launcher*.
2. Der Download der jnlp-Datei des Admin Tools wird gestartet. Während des Downloads wird das Logo von ConSol CM angezeigt.
3. Java Web Start startet das Admin Tool. Im Pop-up-Fenster wird die Meldung *Anwendung wird verifiziert* angezeigt.
4. Wenn die Java Web-Konsole aktiviert ist, wird die Konsole geöffnet und Sie können den Fortschritt des Downloads nachverfolgen.
5. Die GUI des Admin Tools wird mit dem Login-Fenster im Vordergrund geöffnet.



Abbildung 10: *ConSol CM Admin Tool - Start: Schritte 1 und 2*



Abbildung 11: *ConSol CM Admin Tool - Start: Schritte 3 und 4*

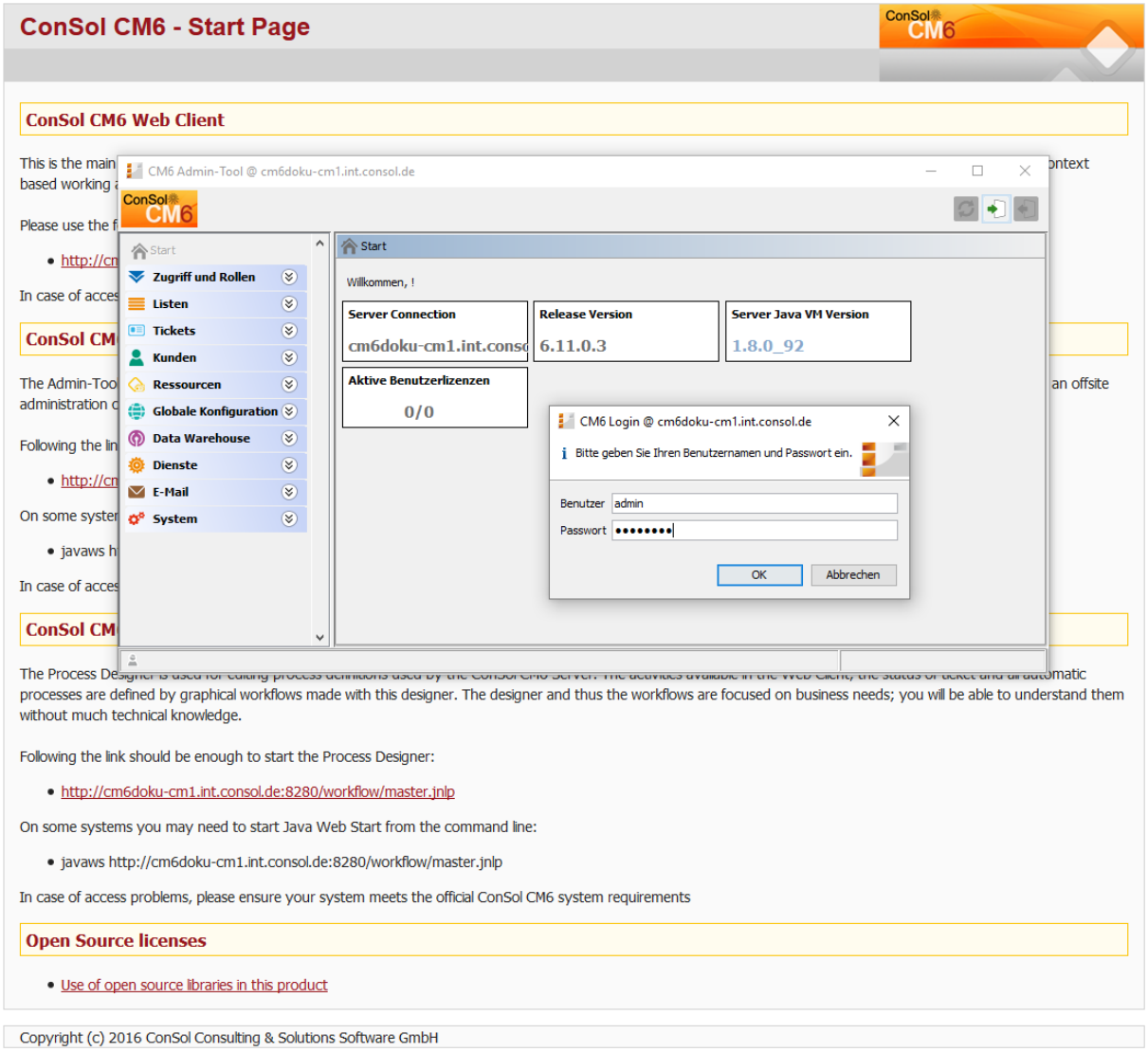


Abbildung 12: *ConSol CM Admin Tool - Start: Schritt 5*

A.11.2.2 Fehlerhafter Ablauf

Prüfen Sie folgende Einstellungen, wenn das Admin Tool nicht gestartet werden kann:

1. *Probleme bei Schritt 1:*
    a. Ist eine unterstützte Version auf Ihrem Computer installiert?
    b. Ist die richtige Java-Version aktiviert?
    Unter Microsoft Windows finden Sie diese in *Systemsteuerung -> Java -> Java -> Ansicht ...*

2. **Probleme bei Schritt 2:**
    a. Kann Ihr Computer als Client über das Netzwerk eine Verbindung zum ConSol CM-Server herstellen? Kann Ihr Webbrowser die jnlp-Datei herunterladen?
    b. Überprüfen Sie die Java-Netzwerkeinstellungen.
    Unter Microsoft Windows finden Sie diese in *Systemsteuerung -> Java -> Allgemein -> Netzwerkeinstellungen.*
3. **Probleme bei Schritt 3:**
    a. Lädt und verifiziert Java Web Start alle Dateien der Admin-Tool-Applikation? Falls nicht, überprüfen Sie die Netzwerkverbindung.
    b. Bei allen anderen Fehlern wird ein Pop-up-Fenster mit einer detaillierten Fehlermeldung angezeigt.
4. **Hinweise zu Schritt 4:**
    a. Aktivieren Sie die Java-Konsole.
    Unter Microsoft Windows finden Sie diese Einstellung in *Systemsteuerung -> Java -> Erweitert -> Konsole anzeigen, Debugging: Tracing aktivieren, Logging aktivieren*.
5. **Probleme bei Schritt 5:**
    a. Geben Sie Ihre Anmeldedaten im Login-Fenster ein. Wenn ein Verbindungsfehler auftritt, prüfen Sie die Proxy-Einstellungen.

Im Abschnitt Die grafische Benutzeroberfläche (GUI) des Admin Tools erfahren Sie, wie Sie mit den GUI-Elementen des Admin Tools arbeiten.

A.12 Die grafische Benutzeroberfläche (GUI) des Admin Tools

In diesem Kapitel werden folgende Themen behandelt:

A.12.1 Einleitung 41

A.12.2 Grundprinzip 41

A.12.3 Inline-Validierung von Eingabewerten 45

A.12.4 Symbole und andere GUI-Elemente 48

A.12.5 Lokalisierung der im Web Client angezeigten Begriffe 51

A.12.1 Einleitung

Der folgende Abschnitt enthält eine Übersicht über die grafische Benutzeroberfläche (GUI) des Admin Tools. Es werden das Grundprinzip und alle Symbole erklärt.

In diesem Handbuch werden die GUI-Elemente so genannt, wie hier beschrieben ist. Um Wiederholungen zu vermeiden, werden die Symbole nicht in jedem Abschnitt beschrieben.

A.12.2 Grundprinzip

Auf der linken Seite sehen Sie den Navigationsbaum (1) mit den Navigationsgruppen (2: ausgeklappte Navigationsgruppe, 4: zugeklappte Navigationsgruppe). Jede Navigationsgruppe enthält mehrere Navigationselemente (3). Klicken Sie auf den Namen der Gruppe, um die Gruppe in der Baumstruktur zu erweitern. Klicken Sie auf ein Navigationselement, um den entsprechenden Tab im Arbeitsbereich auf der rechten Seite zu öffnen.

In der unteren linken Ecke (5) wird der Name des aktuell angemeldeten Admin-Benutzers mit allen ihm aktuell zugewiesenen Rollen angezeigt.

In der oberen rechten Ecke (6) finden Sie die Buttons, mit denen Sie die Daten des Admin Tools neu laden, sich anmelden und sich abmelden können. Der Hauptarbeitsbereich (7) enthält die Daten des aktiven/geöffneten Navigationselements.

In der folgenden Abbildung ist der Start-Tab geöffnet. Dies ist die Startseite, wenn Sie das Admin Tool öffnen.

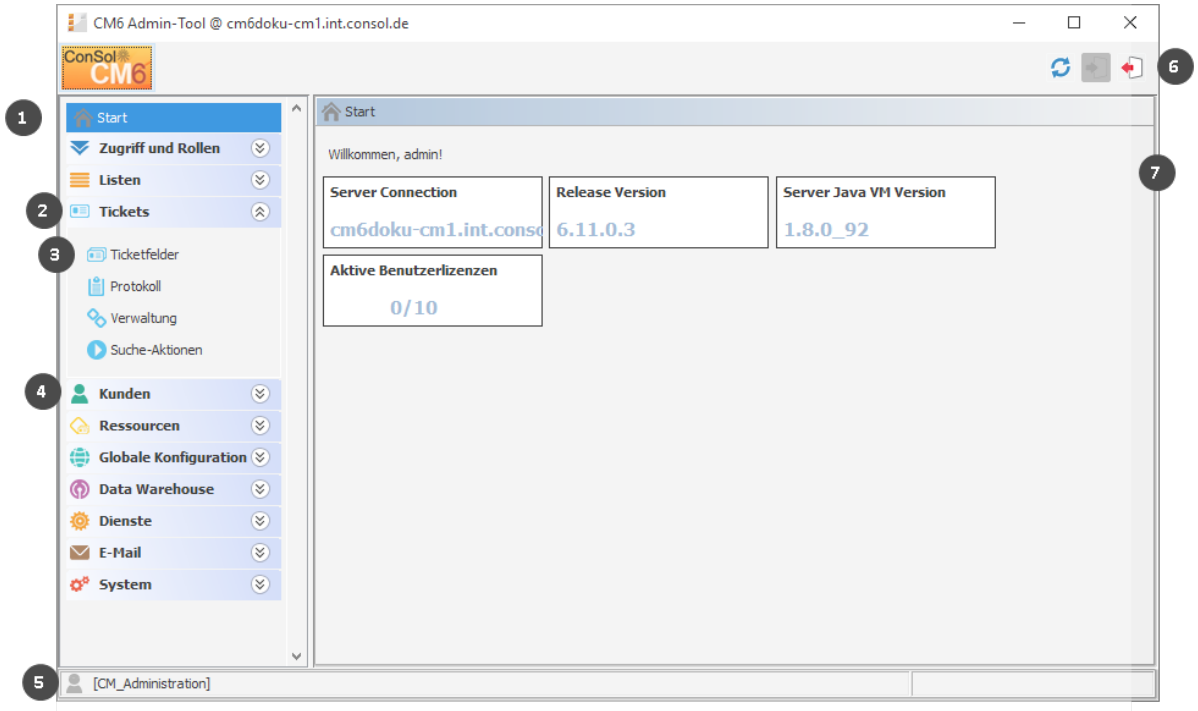


Abbildung 13: *ConSol CM Admin Tool - GUI-Übersicht*

Der Start-Tab enthält folgende Daten;

- **Server Connection**
  Der Server-Name und -Port, auf dem die ConSol CM-Instanz läuft
- **Release Version**
  Die genaue Version von ConSol CM
- **Server Java VM Version**
  Die Version der Java VM (Virtual Machine), die für die aktuelle ConSol CM-Instanz verwendet wird
- **Aktive Benutzerlizenzen**
  Die Anzahl der verwendeten Lizenzen, verglichen mit den verfügbaren Lizenzen.
  Sie können hier einen von zwei Anzeigemodi auswählen, indem Sie den Wert der CM-System-Property cmas-app-admin-tool, admin.tool.consumed.licenses.pool.name ändern. Dieser Wert bezieht sich auf den Abschnitt in der Lizenzdatei:
  - **CONCURRENT_USERS**
    Es wird die Anzahl der gerade im Web Client angemeldeten Bearbeiter angezeigt.
  - **TRACK**
    Nur wenn CM/Track in Betrieb ist. Es wird die Anzahl der gerade in CM/Track (dem Portal von ConSol CM) angemeldeten Kunden angezeigt.

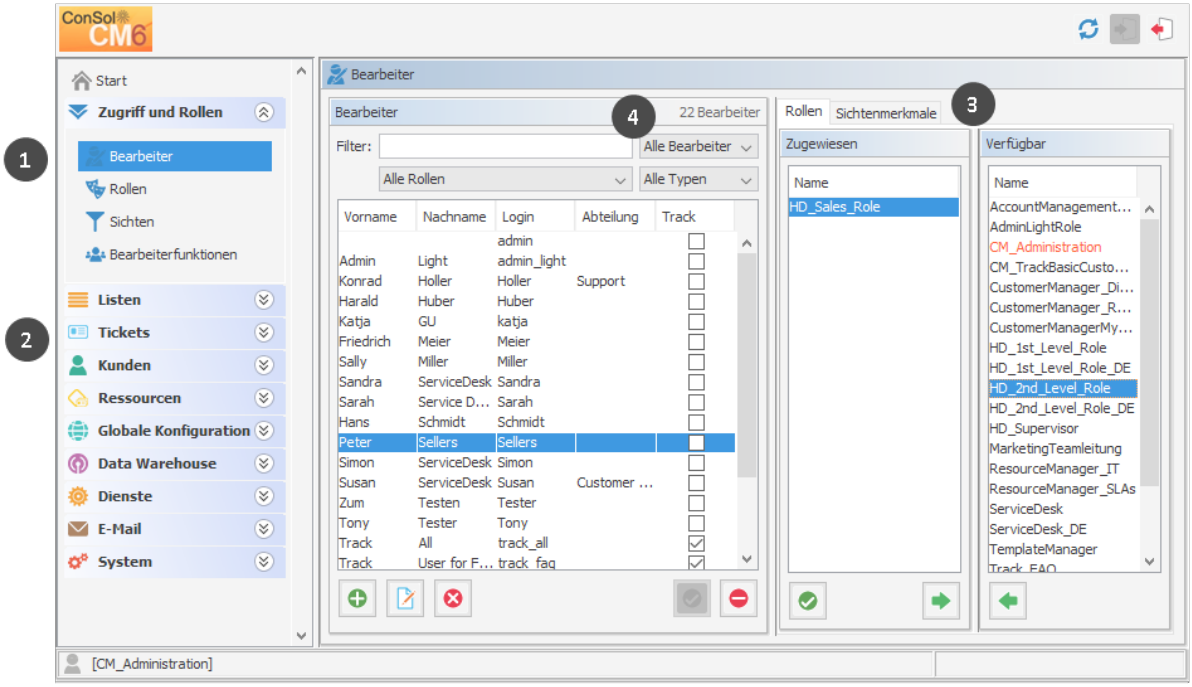


Abbildung 14: *ConSol CM Admin Tool - GUI-Übersicht: Navigationselement offen*

Die Benutzeroberfläche aller anderen Navigationselemente (mit Ausnahme des Start-Tabs) hat die gleiche Struktur. Das folgende Beispiel zeigt den Tab des Navigationselements *Bearbeiter*.

Auf der linken Seite wird der Navigationsbaum (1, 2) angezeigt. Die rechte Seite zeigt die Einstellungen für das aktive Navigationselement (im Beispiel: Bearbeiter, 1).

Im aktiven Tab, im Beispiel *Bearbeiter*, zeigt eine Liste auf der linken Seite die Elemente, die verändert werden können. Elemente können hinzugefügt, bearbeitet, gelöscht, deaktiviert oder aktiviert werden.

Die Attribute eines Elements werden auf der rechten Seite (3)angezeigt. Mit einem Doppelklick oder einem Klick auf das Symbol *Zuweisen* können Sie diese von der Liste der verfügbaren Attribute in die Liste der zugewiesenen Attribute verschieben (Beispiel: verfügbare Rollen und zugewiesene Rollen). Attribute können außerdem mittels Checkboxen und Auswahllisten zugewiesen werden (hier nicht dargestellt).

Es gibt mehrere Möglichkeiten, um Einträge, die bearbeitet werden sollen, schneller zu finden:

- **Filter (4)**
  Mit Filtern können Sie Einträge in Listen (z. B. der Bearbeiterliste) schneller finden. Es gibt zwei Arten von Filtern:
  - **Textfilter**
    Geben Sie die Zeichen des gewünschten Wortes ein (z. B. den Bearbeiternamen). Die Liste wird automatisch aktualisiert, sodass nur die passenden Einträge angezeigt werden.

- **Drop-down-Menüfilter**
  Wählen Sie eine Kategorie (z. B. *Alle Bearbeiter*). Es werden nur die passenden Listeneinträge (z. B. Bearbeiternamen) angezeigt.

- **Sortieren**
  Sie können die Einträge aufsteigend und absteigend sortieren, indem Sie auf eines der Kopfzeilenfelder der Liste klicken. Die kleinen Pfeile nach oben und nach unten zeigen die aktuelle Sortierreihenfolge an.

Normalerweise werden alle im Admin Tool vorgenommenen Änderungen automatisch an das System übermittelt, ohne dass Sie die Daten synchronisieren müssen. Wenn in anderen Modulen Änderungen gemacht wurden und das Admin Tool die neuen Daten verwenden soll, ist eine Synchronisierung der Daten allerdings erforderlich. Klicken Sie dazu in der Symbolleiste auf den Button *Aktualisieren*.

Ein Beispiel dafür ist das Deployment eines neuen Workflows mit dem Process Designer. Bevor Sie den neuen Workflow einer neuen Queue zuweisen können, müssen Sie die Daten synchronisieren, damit das Admin Tool weiß, **dass** es einen neuen Workflow gibt. Das Admin Tool lädt alle Daten aus der Datenbank neu, einschließlich des neuen Workflows. Dieser Workflow kann dann für weitere Operationen verwendet werden und zum Beispiel einer neuen Queue zugewiesen werden.

A.12.3 Inline-Validierung von Eingabewerten

Für alle Datenmodelle (Kundendatenmodell, Ressourcendatenmodell, Ticketdatenmodell) werden die Eingabewerte während der Eingabe validiert, um Inkonsistenzen bei den Namen der Datenbankfelder zu vermeiden. **Der Wert jedes Feldes muss auf der entsprechenden Feldebene eindeutig sein.** Siehe folgende Beispiele:

- Ein neuer Ressourcentyp kann nicht den gleichen Namen haben wie ein vorhandener Ressourcentyp. Er kann aber den gleichen Namen haben wie eine Ressourcenfeldgruppe.
- Eine Ressourcenfeldgruppe kann nicht den gleichen Namen haben wie eine andere Ressourcenfeldgruppe, unabhängig davon, zu welchem Ressourcentyp sie gehört (siehe Beispiel A unten).
- Ein neues Ticketfeld kann nicht den gleichen Namen haben wie ein anderes Feld der gleichen Ticketfeldgruppe. Es kann aber den gleichen Namen haben wie ein anderes Feld in einer anderen Ticketfeldgruppe (siehe Beispiele B und C unten).
- Ein neues Kundendatenmodell kann nicht den gleichen Namen haben wie ein bestehendes Kundendatenmodell.

Ein nicht gültiges Feld ist rot markiert. Es ist nicht möglich, den Feldwert zu speichern.



Wir empfehlen die Verwendung von selbsterklärender Terminologie und Syntax für die Datenfelder. Die Feldnamen sollten so weit wie möglich selbsterklärend und eindeutig sein. Bei einer Implementierung von speziellen Algorithmen können Ausnahmen von dieser Regel sinnvoll sein, im Allgemeinen sollten Sie diese Regel als ConSol CM-Administrator jedoch beachten!

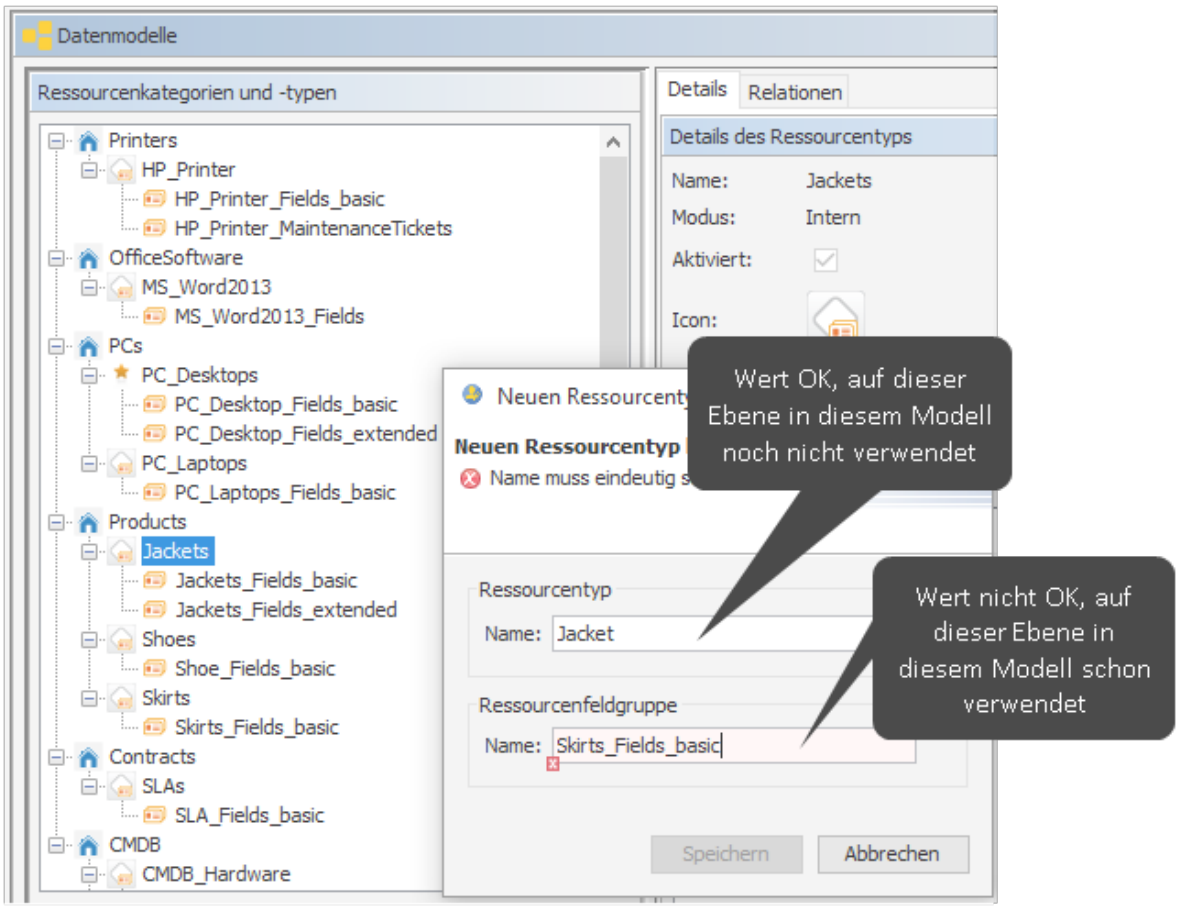


Abbildung 15: *Inline-Validierung von Eingabewerten während der Einrichtung des Ressourcendatenmodells, Beispiel A*

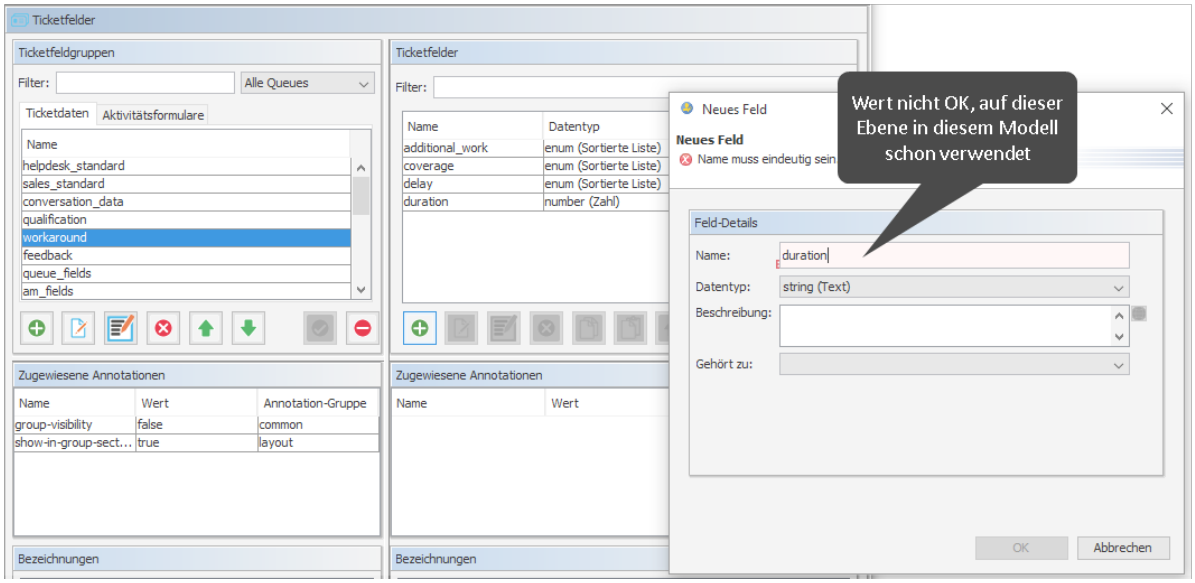


Abbildung 16: *Inline-Validierung von Eingabewerten während der Einrichtung des Ticketdatenmodells, Beispiel B*

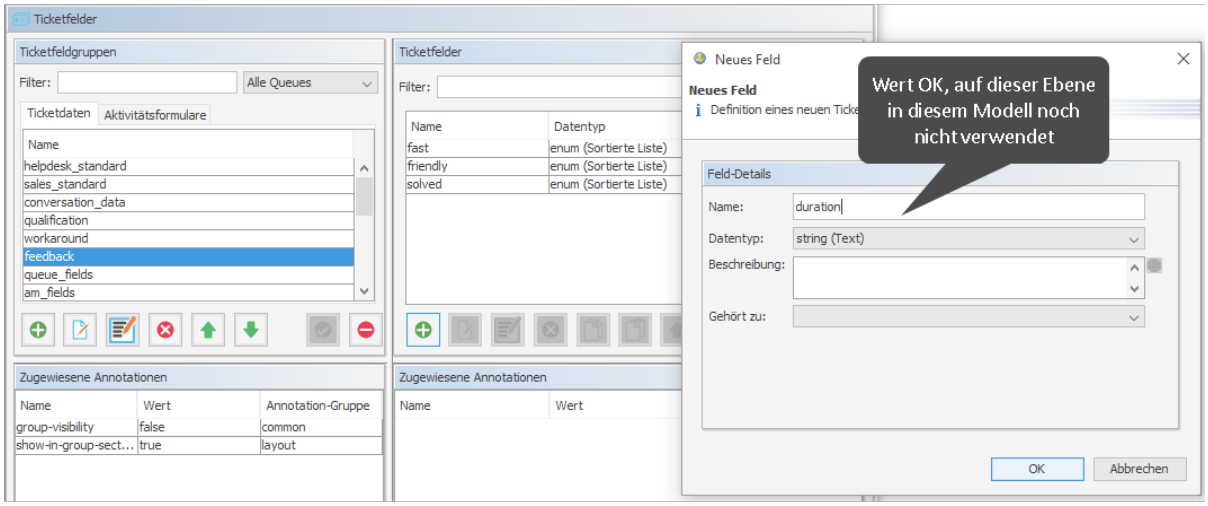


Abbildung 17: *Inline-Validierung von Eingabewerten während der Einrichtung des Ticketdatenmodells, Beispiel C*

A.12.4 Symbole und andere GUI-Elemente

Bei der Verwaltung eines ConSol CM-Systems arbeiten Sie mit folgenden Symbolen. Hier finden Sie die allgemeinen Erklärungen. Die Bedingungen und Auswirkungen der Funktionen in ihrem Kontext sind in den entsprechenden Abschnitten beschrieben.

Symbol	Name	Bedeutung/Funktion	Beispiele
	Hinzufügen / Neu	Hinzufügen (= Erstellen) eines neuen Elements des entsprechenden Typs.	Hinzufügen (Erstellen) eines neuen Bearbeiters, einer neuen Sicht, eines neuen Skripts
	Bearbeiten	Bearbeiten des ausgewählten Elements. Normalerweise wird ein Pop-up-Fenster geöffnet.	Editieren eines Bearbeiters, einer Textklasse, einer Ticketfeldgruppe
	Löschen	Löschen des ausgewählten Elements aus der Datenbank. Es kann danach nicht wiederhergestellt werden.	Löschen eines Bearbeiters, einer Kundenfeldgruppe, einer Ressource
	Kopieren	Kopieren des ausgewählten Elements	Kopieren einer Rolle
	Einfügen	Verfügbar für Ticket-, Kunden- und Ressourcenfelder	Das kopierte Feld in die ausgewählte Zielfeldgruppe einfügen
	Aktivieren	Aktivieren eines Elements, das zuvor deaktiviert war.	Aktivieren eines Bearbeiters, dessen Konto deaktiviert wurde.
	Deaktivieren	Deaktivieren des ausgewählten Elements. Dies kann sicherer sein als das Löschen. Elemente, die schon verwendet werden, können nicht gelöscht werden. Es kann also eine Alternative sein, sie zu deaktivieren.	Deaktivieren eines Bearbeiters (Kontos), z. B. wenn ein Mitarbeiter ein Sabbatjahr nimmt. Deaktivieren einer Kunden, der den Vertrag gekündigt hat.
	Nach oben	Ein Element in einer Liste eine Position nach oben bewegen. Dies kann Auswirkungen auf den Web Client haben.	Eine Sicht in der Liste zwei Positionen nach oben bewegen. Die Sicht wird dann in der Sichtenliste im Web Client in der neuen Position angezeigt.

Symbol	Name	Bedeutung/Funktion	Beispiele
	Nach unten	Ein Element in einer Liste eine Position nach unten bewegen. Dies kann Auswirkungen auf den Web Client haben.	Eine Ticketfeldgruppe in der Liste eine Position nach unten bewegen. Sie wird dann im Bereich mit den Ticketdaten oder im Bereich für Detaildaten an der neuen Position angezeigt.
	Entfernen/Entziehen	Entfernen eines Elements aus dem ausgewählten Element.	Dem ausgewählten Bearbeiter eine Rolle entziehen.
	Zuweisen	Zuweisen eines Elements zum ausgewählten Element.	Zuweisen einer Rolle zu einem Bearbeiter. Zuweisen einer Sicht zu einer Rolle.
	Annotieren (CM-Versionen 6.10.2 und niedriger)	Öffnen des Pop-up-Fensters *Annotation*	Wird verwendet für Ticketfeldgruppen, Ticketfelder, Kundenfeldgruppen, Kundenfelder, Ressourcenfeldgruppen und Ressourcenfelder
	Annotieren (CM-Versionen 6.10.3 und höher)	Öffnen des Pop-up-Fensters *Annotation*	Wird verwendet für Ticketfeldgruppen, Ticketfelder, Kundenfeldgruppen, Kundenfelder, Ressourcenfeldgruppen und Ressourcenfelder
	Lokalisieren / Internationalisieren	Öffnen des Pop-up-Fensters, in dem die lokalisierten Namen der technischen Objekte eingegeben werden können. Es werden die im Admin Tool konfigurierten Sprachen angeboten.	Lokalisieren des Namens eines Ticketfeldes.
	Suche	Öffnen des Suchfensters.	Starten der Ticketsuche.
	Alle auswählen:	Markieren aller Elemente (häufig Checkboxen). Es wird keine Datenbankaktion durchgeführt, nur Hilfestellung auf der GUI.	Markieren aller Berechtigungen einer Rolle für den Queue-Zugriff.

Symbol	Name	Bedeutung/Funktion	Beispiele
	Keine auswählen	Aufheben der Auswahl aller Elemente (häufig Checkboxen). Es wird keine Datenbankaktion durchgeführt, nur Hilfestellung auf der GUI.	Aufheben der Auswahl aller Berechtigungen einer Rolle für den Queue-Zugriff.
	Start	Starten des ausgewählten Elements (normalerweise eines Dienstes)	Starten eines CM-Dienstes
	Stop	Stoppen des ausgewählten Elements (normalerweise eines Dienstes)	Stoppen eines CM-Dienstes
	Hochladen	Öffnen eines Dateibrowsers, um eine Datei in das CM-System hochzuladen	Hochladen eines Skripts
	Herunterladen / Speichern	Speichern einer Datei im Dateisystem	Speichern eines Skripts als Datei im Dateisystem
	Speichern und schließen	Speichern des Elements (normalerweise eines Skripts) und Schließen des Editiermodus im Editor. Wechseln in den Ansichtsmodus.	Speichern eines Skripts im Skriptbereich und Wechseln in den Ansichtsmodus des Skripts
	Verlassen ohne zu speichern	Schließen des Editiermodus im Editor ohne Speichern des Elements (normalerweise eines Skripts). Wechseln in den Ansichtsmodus.	Wechseln in den Ansichtsmodus des Skripts, ohne das editierte Skript im Skriptbereich zu speichern
	Aktualisieren	Aktualisieren der Daten im Admin Tool.	Aktualisieren der Daten im Admin Tool, nachdem im Process Designer Änderungen vorgenommen wurden.

A.12.5 Lokalisierung der im Web Client angezeigten Begriffe

ConSol CM kann für internationale Umgebungen konfiguriert werden. Auf diese Weise kann jeder Bearbeiter im Webbrowser in seiner landesspezifischen Sprache arbeiten. Die folgenden Prinzipien sind hinsichtlich der Lokalisierung/Internationalisierung wichtig:

Der CM-Administrator konfiguriert die Sprachen, die im gesamten System verfügbar sind. Dies geschieht im Admin Tool und ist im Abschnitt Sprachen erklärt.

Die im Admin Tool als verfügbare Sprachen konfigurierten Sprachen können dann für alle im CM-System konfigurierten Begriffe angewendet werden, z. B. für die Begriffe in Ticketfeldern. Mit dem Process Designer können die Bezeichnungen für die Workflow-Aktivitäten ebenfalls lokalisiert werden. Dies ist im ConSol CM Process Designer Handbuch erklärt. Es gibt zwei verschiedene Kategorien von Begriffen im Web Client.

1. **CM-Standardbegriffe / -Bezeichnungen**, die nicht im Admin Tool oder Process Designer geändert werden können. Sie können allerdings im Rahmen der Entwicklung einer kundenspezifischen Software an Ihr System angepasst werden. Begriffe wie die Bezeichnungen der Gruppenüberschriften in der Ticketliste (z. B. *Eigene Tickets*, *Gruppentickets*, *Nicht zugewiesene Tickets*) oder die Bezeichnungen *Favoriten* und *Workspace* gehören in diese Kategorie. Diese Begriffe / Bezeichnungen sind in zwei Standardsprachen verfügbar: Englisch und Deutsch. Englisch ist zusätzlich die globale Standardsprache. Sie können die CM-Standardbegriffe nicht ändern. Sie sind auf Deutsch und Englisch verfügbar und stellen einen festen Satz an Begriffen und Bezeichnungen dar.
2. **Systemspezifische Begriffe, d. h. Begriffe / Bezeichnungen, die mit dem Admin Tool der Process Designer lokalisiert werden können**. Die Lokalisierung mit dem Admin Tool ist im folgenden Abschnitt erklärt. Eine Beschreibung der Lokalisierung mit dem Process Designer finden Sie im *ConSol CM Process Designer Handbuch*.
   Für systemspezifische Felder müssen die lokalisierten Begriffe / Bezeichnungen manuell für jede Sprache eingegeben werden.

In den Web-Clients (CM Web Client und CM/Track) werden die Bezeichnungen in der Sprache angezeigt, die der Bearbeiter bzw. Kunde im Browser eingestellt hat.

A.12.5.1 Die unterschiedlichen Modi zum Lokalisieren von Begriffen und Bezeichnungen im Admin Tool

Grundprinzip der Lokalisierung in CM

Abhängig von der Stelle im Admin Tool gibt es unterschiedliche GUI-Elemente, die zum Festlegen von lokalisierten Werten verwendet werden. Das Grundprinzip gilt allerdings für alle Stellen. Es funktioniert folgendermaßen (am Beispiel eines Ticketfeldes erklärt).

Technischer Name:

Jedes Feld oder Objekt hat einen technischen Namen, z. B. das Ticketfeld *priority*. Dieser Name wird bei der Erstellung des Feldes festlegt und sollte danach nicht mehr verändert werden, da er in Admin-Tool- oder Workflow-Skripten verwendet werden könnte, die fehlschlagen, wenn sich der Name ändert. CM zeigt eine Warnmeldung an und blockiert die Aktion, wenn jemand versucht, einen technischen Feld- oder Objektnamen zu ändern. Der technische Name kann nur geändert werden, wenn das Feld bzw. Objekt neu ist und noch nicht verwendet wird.

Lokalisierte Namen:

Für jedes Feld, Feldgruppe oder Objekt kann ein lokalisierter Name für jede im CM-System konfigurierte Sprache (Navigationsgruppe *Globale Konfiguration*, Navigationselement *Sprachen*) festgelegt werden.

Zum Beispiel kann das Ticketfeld mit dem technischen Namen *priority* auf Englisch *HD Priority* und auf Deutsch *HD Priorität* heißen.

In den Web-Clients (CM Web Client und CM/Track) werden die Bezeichnungen der Felder und Objekte in der Sprache angezeigt, die der Bearbeiter bzw. Kunde im Browser eingestellt hat. Es sind einige Varianten möglich:

1. Variante Nr. 1:
   Die im Browser einstellte Sprache wurde konfiguriert und alle Bezeichnungen und Begriffe werden in dieser Sprache angezeigt. Zum Beispiel wurde die Sprache Französisch im Admin Tool hinzugefügt und alle systemspezifischen Begriffe manuell für Französisch lokalisiert (im Admin Tool und Process Designer). Dann werden alle systemspezifischen Begriffe auf Französisch angezeigt, z. B. die Workflow-Aktivitäten, die Namen der Ticketfelder und die Namen der Sichten.
2. Variante Nr. 2:
   Die im Browser einstellte Sprache wurde konfiguriert, es werden aber nicht alle Bezeichnungen und Begriffe in dieser Sprache angezeigt. Zum Beispiel wurde die Sprache Französisch im Admin Tool hinzugefügt, aber es wurden nur einige systemspezifische Begriffe manuell für Französisch lokalisiert (im Admin Tool und Process Designer). Dann werden die lokalisierten systemspezifischen Begriffe auf Französisch angezeigt, z. B. die Workflow-Aktivitäten, die Namen der Ticketfelder und die Namen der Sichten. Alle nicht lokalisierten Begriffe (d. h. bei denen das Feld für den lokalisierten Begriff leer ist) werden in der Sprache angezeigt, die als Standardsprache in den Spracheinstellungen konfiguriert wurde.
3. Variante Nr. 3:
   Die im Browser eingestellte Sprache wurde nicht konfiguriert. Zum Beispiel wurden die lokalisierten Begriffe und Bezeichnungen für Englisch, Deutsch und Französisch festgelegt, der Bearbeiter hat im Browser aber ES (Spanisch) eingestellt. in diesem Fall werden alle Bezeichnungen und Begriffe in der Sprache angezeigt, die als Standardsprache in den Spracheinstellungenkonfiguriert wurde.

Lokalisierung von Datenfeldern

Die lokalisierten Begriffe für Datenfelder werden direkt in den Abschnitt *Bezeichnungen* im Admin Tool geschrieben. Dies gilt für:

- Ticketfelder (für Ticketdaten)
- Kundenfelder (für Kundendaten)
- Ressourcenfelder (für Ressourcendaten)

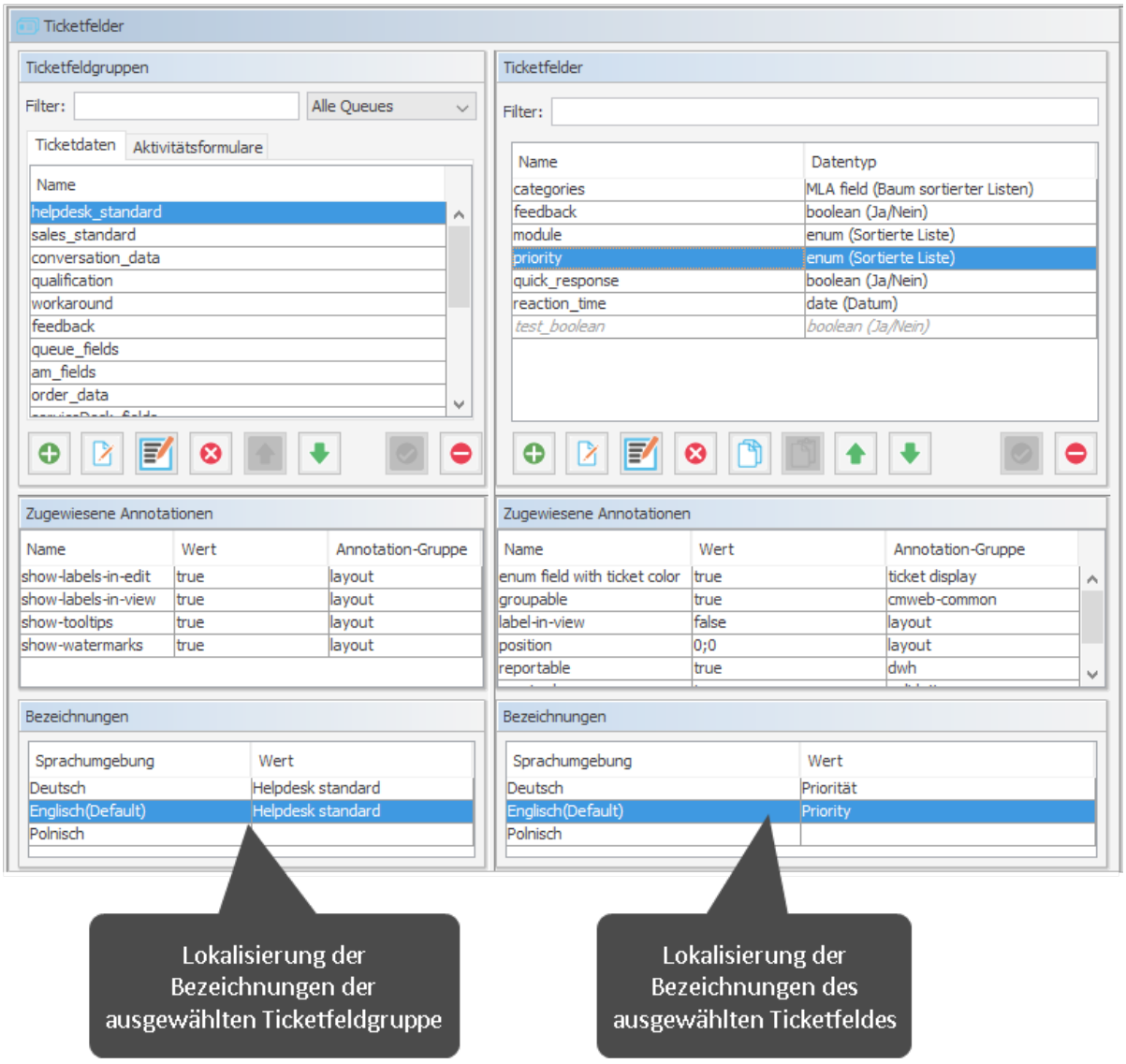


Abbildung 18: *ConSol CM Admin Tool - Lokalisierung von Ticketfeldern*

Um den Wert eines Feldes zu ändern, klicken Sie in das Feld und geben Sie den neuen Begriff ein. Drücken Sie die Eingabetaste, um den neuen Begriff zu speichern.

Im Web Client wird das in der obigen Abbildung gezeigte Ticketfeld wie folgt angezeigt (auf Englisch und auf Deutsch):

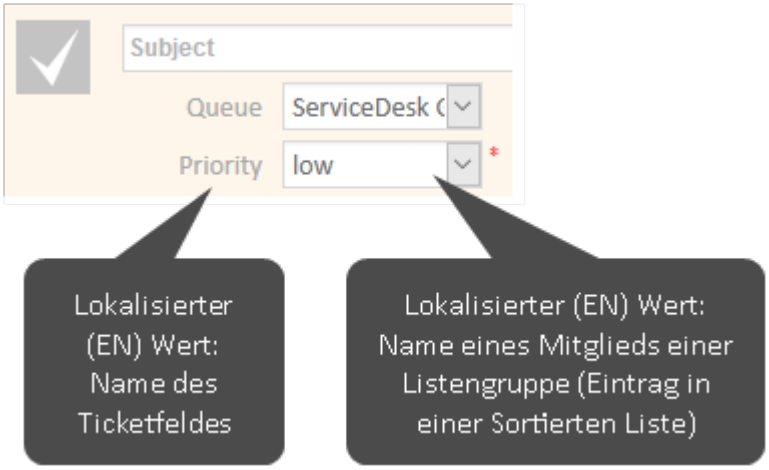


Abbildung 19: *ConSol CM Web Client - Bei Spracheinstellung Englisch im Browser angezeigtes Ticketfeld*

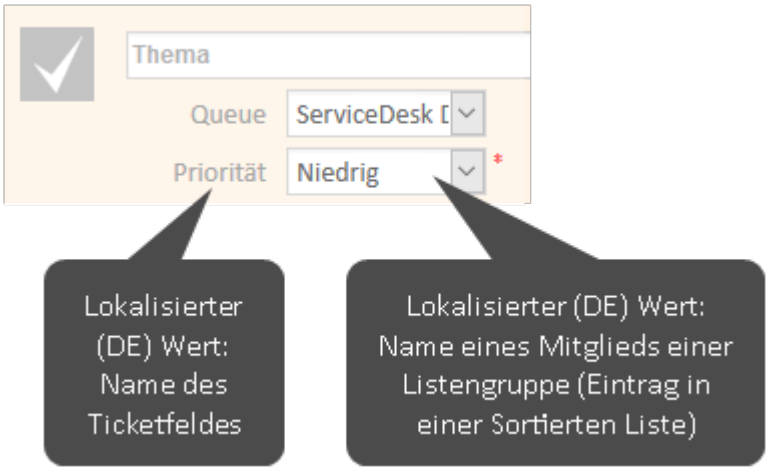


Abbildung 20: *ConSol CM Web Client - Bei Spracheinstellung Deutsch im Browser angezeigtes Ticketfeld*

Lokalisierung von Objekten allgemein, Typ 1

Eine große Anzahl an Objekten kann über den Button *Lokalisieren* (Globussymbol) lokalisiert werden. Es wird ein Pop-up-Fenster mit der Lokalisierungskonfiguration geöffnet.

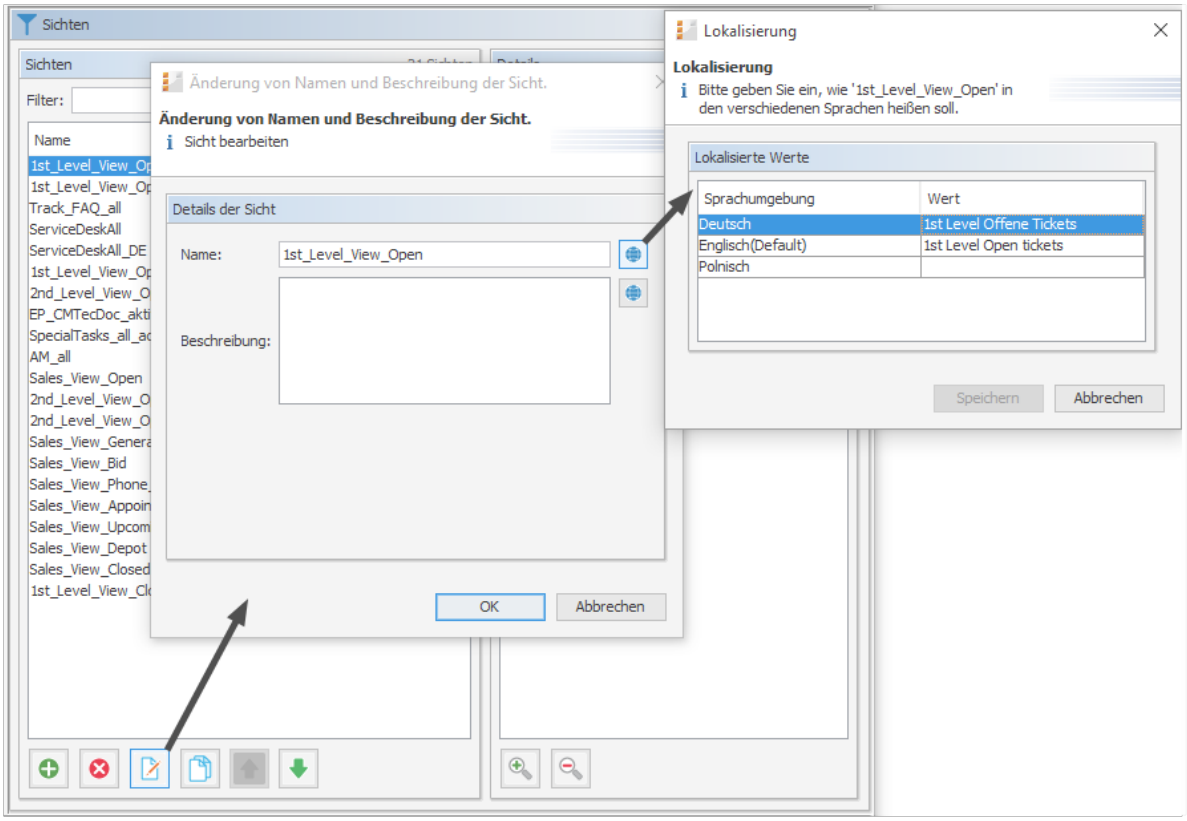


Abbildung 21: *ConSol CM Admin Tool - Lokalisierung von Objekten, Beispiel: Name einer Sicht*

Um den Wert eines Feldes zu ändern, klicken Sie in das Feld und geben Sie den neuen Begriff ein. Drücken Sie die Eingabetaste, um den neuen Wert zu speichern.

Lokalisierung von Objekten allgemein, Typ 2

Einige Objekte werden mithilfe der Lokalisierungstabelle im Pop-up-Fenster, das zum Bearbeiten des Objekts verwendet wird, lokalisiert.

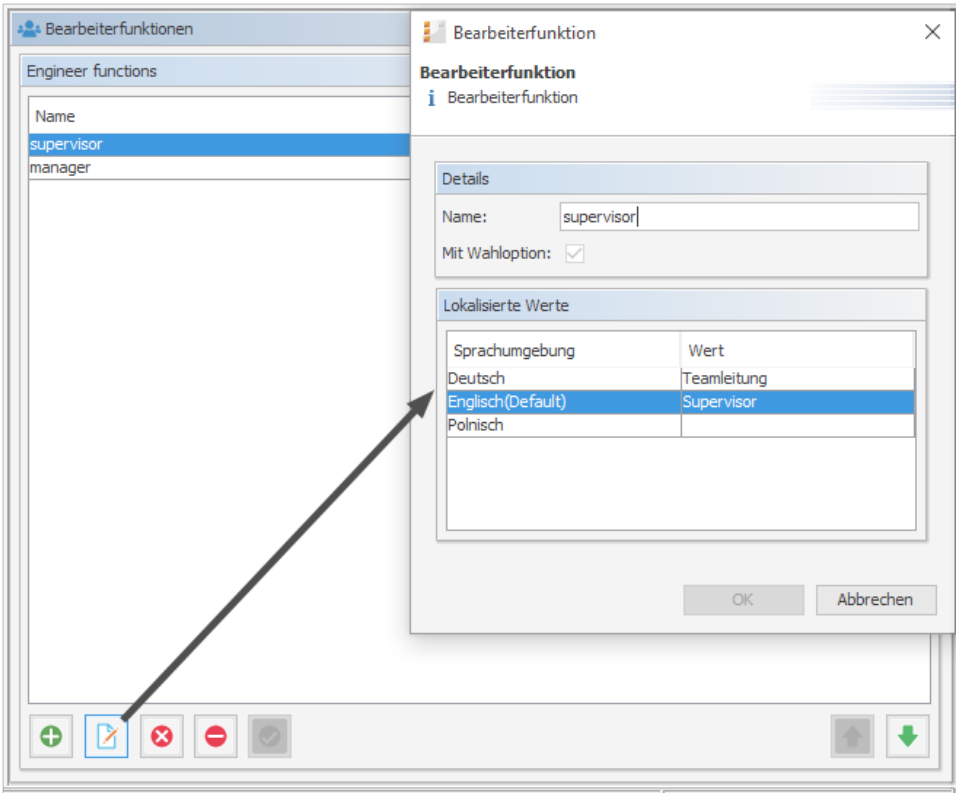


Abbildung 22: *ConSol CM Admin Tool - Lokalisierung von Objekten, Beispiel: Name einer Bearbeiterfunktion*

Verwenden von UTF-8-Symbolen in lokalisierten Feldern

Sie können UTF-8-Symbole verwenden, um grafische Symbole in Textfeldern anzuzeigen. Dies funktioniert für alle lokalisierten Felder, z. B. String-Felder, Queue-Namen oder Sortierte Listen. Kopieren Sie einfach das UTF-8-Symbol aus einer Unicode-Tabelle und fügen Sie es in das Textfeld im Admin Tool ein.

Beispiel:

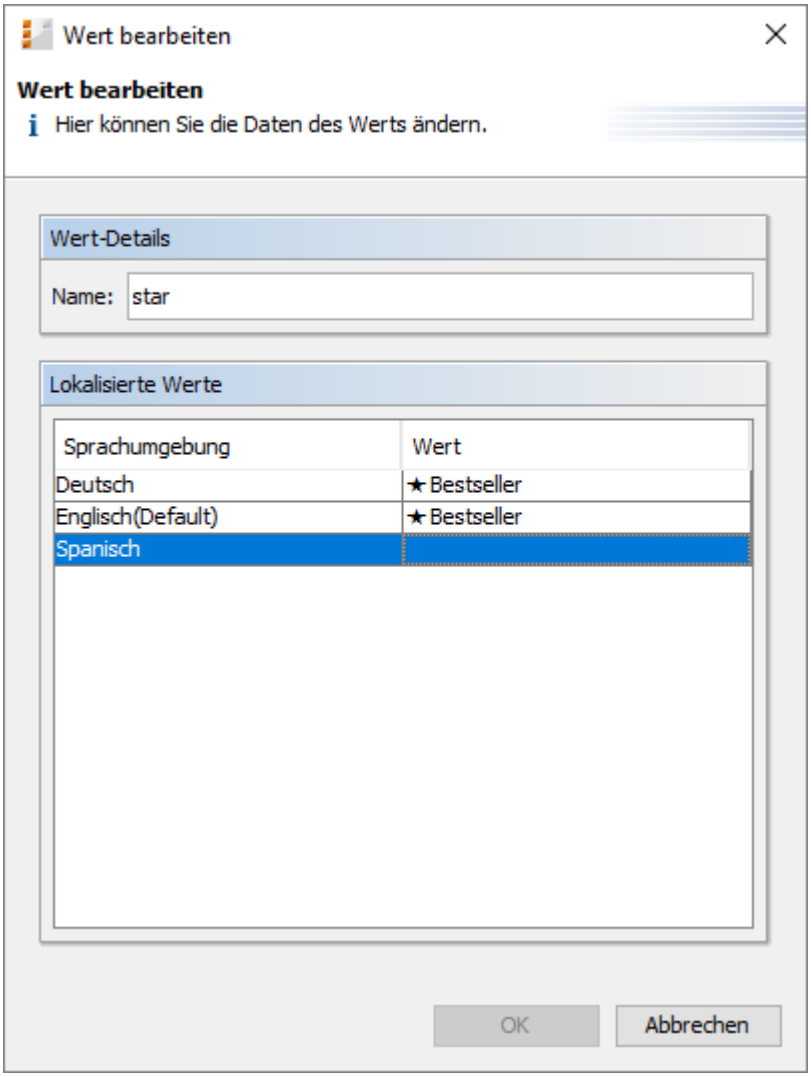


Abbildung 23: *Ein UTF-8-Symbol in einem lokalisierten Feld im Admin Tool*

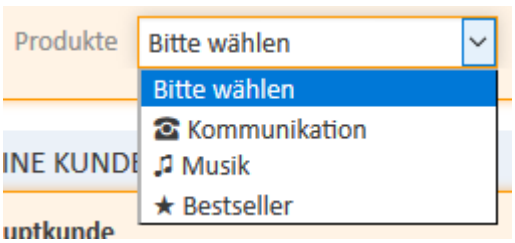


Abbildung 24: *Anzeige von grafischen UTF-8-Symbolen im Web Client, hier: Sortierte Liste*

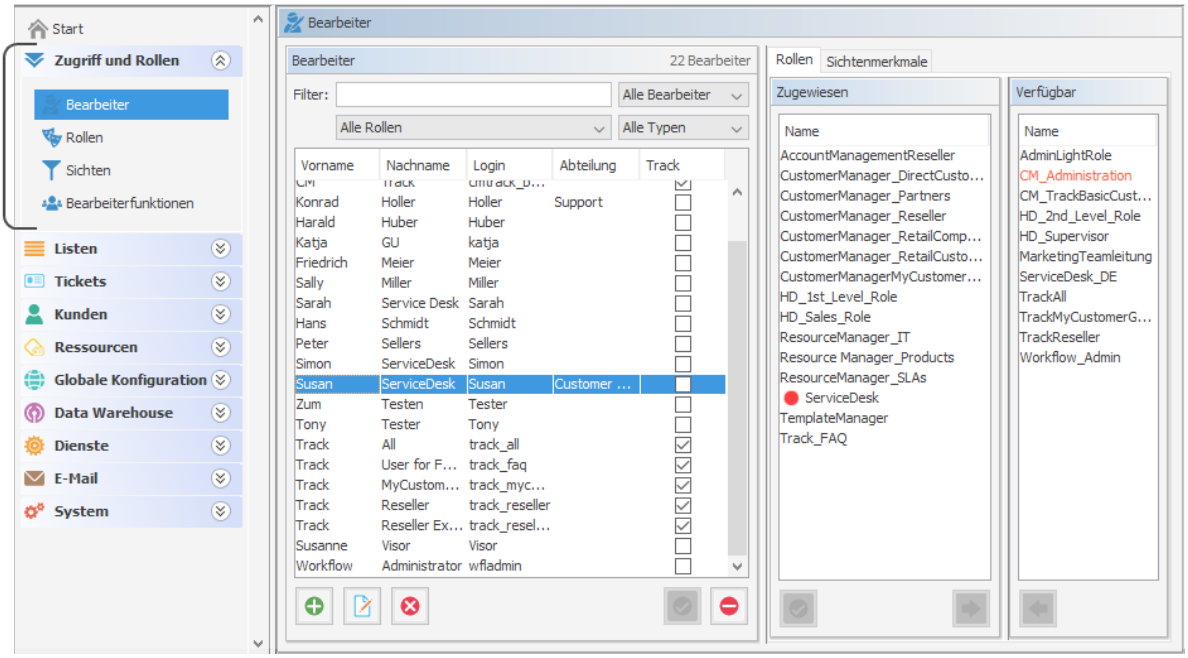
Wenn Sie mit einer MySQL-Datenbank arbeiten, können Sie folgende Einstellungen verwenden, um mit grafischen UTF-8-Symbolen zu arbeiten:

- CHARACTER SET = utf8mb4
- COLLATE = utf8mb4_unicode_ci

Lokalisierung von Bezeichnungen

Mehrere im Web Client verwendete Begriffe können über die Bezeichnungen lokalisiert werden. Dies ist im Abschnitt Bezeichnungen erklärt.

B - Abschnitt Zugriff und Rollen



Dieser Abschnitt enthält Hintergrundinformationen über das Grundprinzip der CM-Zugangsberechtigungen und beschreibt, wie Bearbeiterkonten mit den nötigen Parametern eingerichtet werden.

Lesen Sie die folgenden Abschnitte:

- Bearbeiterverwaltung
- Rollenverwaltung
- Sichtenverwaltung
- Bearbeiterfunktionen

B.1 Bearbeiterverwaltung

In diesem Kapitel werden folgende Themen behandelt:

B.1.1 Einführung in die Bearbeiterverwaltung 60

B.1.2 Bearbeiterverwaltung mit dem Admin Tool 61

B.1.1 Einführung in die Bearbeiterverwaltung

Ein *Bearbeiterkonto* bildet die Grundlage für den Zugang des Bearbeiters oder Administrators zum Web Client, Admin Tool und Process Designer. Beim Einrichten des Systems wird ein Administratorkonto für den ersten Zugang zum Admin Tool erstellt. Mit diesem Konto können Sie weitere Konten einrichten.

Neu erstellte Bearbeiterkonten haben noch keine Berechtigungen. Diese Berechtigungen müssen über eine oder mehrere Rollen im Tab *Rollen* zugewiesen werden. Wenn Sie noch keine Rollen erstellt haben, sehen Sie dort nur die Administratorrolle (siehe Tab Rollen - Zuweisen von Rollen zu einem Bearbeiterkonto).

Sichten definieren, welche Tickets die Bearbeiter in ihrer Ticketliste (To-do-Liste) im Web Client sehen. Diese werden in der Sichtenverwaltung erstellt und über Rollen zugewiesen. Auf der Seite der Bearbeiterverwaltung können Sie dynamische Sichtenmerkmale für bestimmte Bearbeiter voreinstellen (siehe Tab Sichtenmerkmale - Definieren von bearbeiterspezifischen Sichtenmerkmalen).

> Wir empfehlen, dass Sie zuerst mindestens eine Rolle und eine Sicht erstellen, bevor Sie mit dem Erstellen der Bearbeiterkonten beginnen.

B.1.2 Bearbeiterverwaltung mit dem Admin Tool

Sie können die Bearbeiterverwaltung öffnen, indem Sie die Navigationsgruppe *Zugriff und Rollen* im Admin Tool öffnen, und auf das Navigationselement *Bearbeiter* klicken.

B.1.2.1 Liste aller Bearbeiter

Wenn Sie die Bearbeiterverwaltung geöffnet haben, wird eine Liste aller Bearbeiter angezeigt. Die aktuell deaktivierten Bearbeiterkonten werden grau angezeigt. Sie können die Liste der Bearbeiter einfach auf Bearbeiter einer Rolle einschränken (siehe Abschnitt Rollenverwaltung), indem Sie im Drop-down-Menü nach einer Rolle filtern.

B.1.2.2 Erstellen und Editieren eines Bearbeiterkontos

Sie können ein neues Bearbeiterkonto erstellen oder die Einstellungen eines vorhandenen Kontos bearbeiten. Beide Aktionen werden im gleichen Popup-Fenster durchgeführt.

Sie gelangen auf diese Seite, indem Sie unter der Bearbeiterliste in der Navigationsgruppe *Zugriff und Rollen*, Navigationselement *Bearbeiter* auf das Symbol *Hinzufügen* klicken (zum Erstellen eines neuen Kontos) bzw. auf das Symbol *Bearbeiten* klicken (zum Bearbeiten eines vorhandenen Kontos).

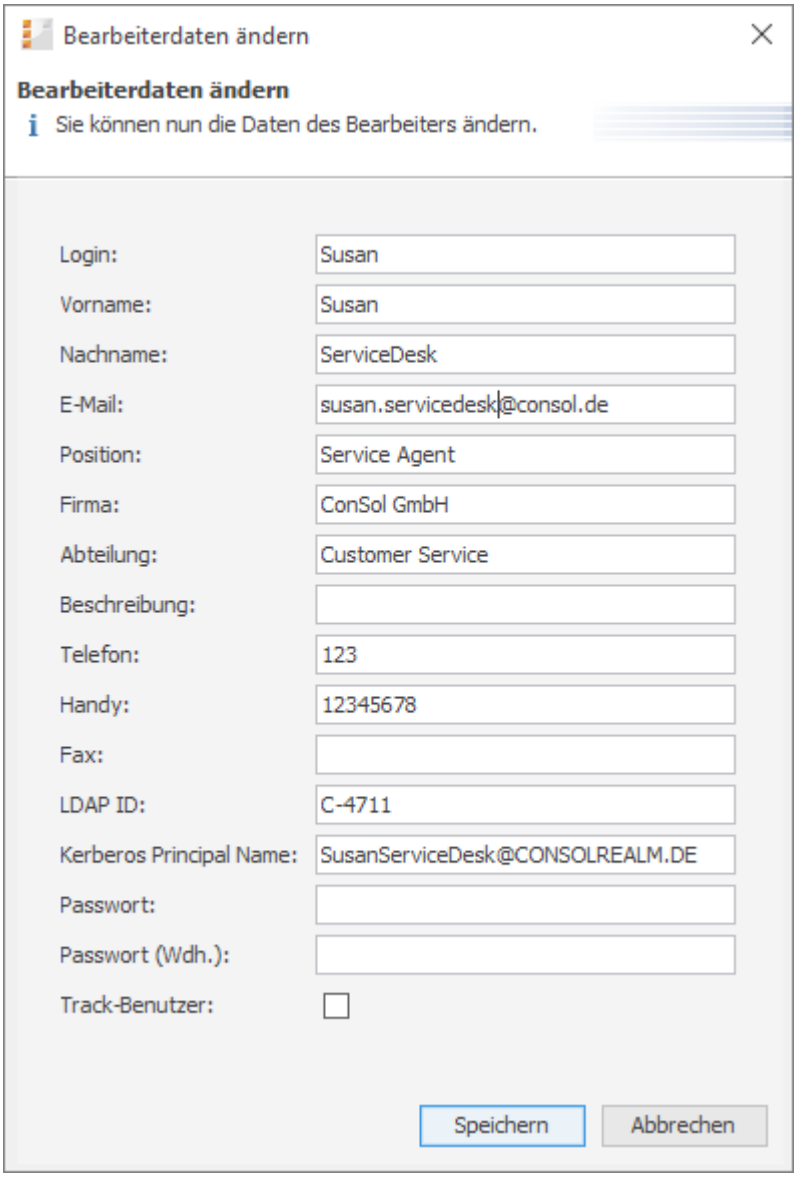


Abbildung 25: *ConSol CM Admin Tool - Zugriff und Rollen, Bearbeiter: Editieren eines Bearbeiterkontos*

Das Fenster enthält die Parameter des Bearbeiterkontos:

- **Login:**
  Pflichtfeld. Dieses Feld enthält den Namen des Kontos, der auf der Login-Seite des Web Clients eingegeben wird. Verwenden Sie hier nur internationale alphanumerische Zeichen, keine Leerzeichen, Satzzeichen oder Sonderzeichen wie Umlaute, Bindestriche oder Ähnliches.
- **Vorname:**
  Optional. Der Vorname des Bearbeiters. Dieses Feld ist optional, wird aber im Web Client für den Bearbeiter angezeigt. Der Eintrag darf Buchstaben, Leerzeichen, Kommata, Punkte und Bindestriche enthalten. Verwenden Sie bitte keine anderen Zeichen.

- **Nachname:**
  Optional. Der Nachname des Bearbeiters. Dieses Feld ist optional, wird aber im Web Client für den Bearbeiter angezeigt. Der Eintrag darf Buchstaben, Leerzeichen, Kommata, Punkte und Bindestriche enthalten. Verwenden Sie bitte keine anderen Zeichen.
- **E-Mail:**
  Pflichtfeld. Die E-Mail-Adresse des Bearbeiters. Verwenden Sie bitte nur internationale alphanumerische Zeichen, Bindestriche, Unterstriche, Punkte und das @-Zeichen. Die Eingabe von mehreren E-Mail-Adressen in einer Zeile ist nicht zulässig.
- **Position:**
  Optional. Die Position oder Funktion des Bearbeiters in der Firma. Das Feld ist optional und hat eine rein beschreibende Funktion. Der Eintrag darf Buchstaben, Leerzeichen, Kommata, Punkte und Bindestriche enthalten. Verwenden Sie bitte keine anderen Zeichen.
- **Firma:**
  Optional. Die Firma des Bearbeiters. Das Feld ist optional und hat momentan eine beschreibende Funktion. Der Eintrag darf Buchstaben, Leerzeichen, Kommata, Punkte und Bindestriche enthalten. Verwenden Sie bitte keine anderen Zeichen.
- **Abteilung:**
  Optional. Die Abteilung in der der Bearbeiter tätig ist. Das Feld ist optional und hat eine beschreibende Funktion. Der Eintrag darf Buchstaben, Leerzeichen, Kommata, Punkte und Bindestriche enthalten. Verwenden Sie bitte keine anderen Zeichen.
- **Beschreibung:**
  Optional. Eine zusätzlicher Beschreibung für das Bearbeiterkonto. Dieses Feld ist optional, und wird im Web Client **nicht** für den Bearbeiter angezeigt. Der Eintrag darf Buchstaben, Leerzeichen, Kommata, Punkte und Bindestriche enthalten. Verwenden Sie bitte keine anderen Zeichen.
- **Telefon:**
  Optional. Die Telefonnummer des Bearbeiters. Das Feld ist optional und hat momentan eine beschreibende Funktion.
- **Handy:**
  Optional. Die Handynummer des Bearbeiters. Das Feld ist optional und hat momentan eine beschreibende Funktion.
- **Fax:**
  Optional. Die Faxnummer des Bearbeiters. Das Feld ist optional und hat momentan eine beschreibende Funktion.

> ⚠ Einige Felder, die Bearbeiterdaten enthalten (wie *Firma*, *Abteilung* oder *Telefon*) sind optionale Felder. Wenn Sie mit Textvorlagen arbeiten, die Felder aus den Bearbeiterdaten enthalten (siehe Abschnitt Der ConSol CM Textvorlagen-Manager), werden die E-Mails oder Kommentare allerdings nicht korrekt erzeugt, wenn die entsprechenden Daten fehlen. Zum Beispiel kann das Feld *Ticket-Bearbeiter, Telefon* in der Vorlage nicht ausgefüllt werden, wenn die entsprechende Angabe für den Bearbeiter in der Bearbeiterverwaltung nicht gemacht wurde! Achten Sie also darauf, alle Daten, die später benötigt werden, direkt richtig einzugeben!

- **LDAP ID:**
  Die LDAP-Benutzer-ID, wenn LDAP zur Authentifizierung verwendet wird. Hier muss kein Passwort angegeben werden.

  > ⓘ Wenn Sie keine LDAP-ID angeben, wird das Login als Parameter für die Authentifizierung mit dem LDAP-Server verwendet (sofern die LDAP-Authentifizierung aktiviert ist)!
  >
  > Detaillierte Informationen über LDAP-Authentifizierung finden Sie im Abschnitt LDAP-Authentifizierung für Bearbeiter im Web Client.

- **Kerberos Principal Name:**
  Der Kerberos Principal Name, wenn Kerberos V5-Protokoll für die Authentifizierung verwendet wird. Bearbeiter können sich mit ihren Windows-Zugangsdaten im Web Client anmelden. Detaillierte Informationen über Single Sign-On mit Kerberos finden Sie im Abschnitt Single Sign-On für ConSol CM mittels Kerberos (in einer Windows-Domäne).
- **Passwort:**
  Pflichtfeld. Das Passwort des Bearbeiters ist obligatorisch. Verwenden Sie hier nur internationale alphanumerische Zeichen und Satzzeichen. Verwenden Sie **keine** Sonderzeichen wie Umlaute. Das eingegebene Passwort wird als Reihe mit Punkten angezeigt. Informationen über eine optionale Passwort-Richtlinie finden Sie in Abschnitt Bearbeiterverwaltung.
- **Passwort (Wdh.):**
  Pflichtfeld. Wiederholen Sie die Eingabe des Passworts hier. Diese Sicherheitsabfrage dient zur Vermeidung falscher Einträge, die ansonsten nicht auffallen würden, weil das Passwort in Punktform dargestellt wird. Informationen über eine optionale Passwort-Richtlinie finden Sie in Abschnitt Bearbeiterverwaltung.

  > ⓘ Dieses Feld wird nur angezeigt, wenn die Authentifizierung der Bearbeiter im Web Client über die CM-Datenbank erfolgt, d. h. es wird nicht angezeigt, wenn LDAP- oder Kerberos-Authentifizierung verwendet wird.

- **Track-Benutzer:**
  Diese Checkbox muss aktiviert werden, wenn Sie einen technischen Bearbeiter (oder *CM/Track-Benutzerprofil)* erstellen möchten, mit dem Zugangsberechtigungen für CM/Track-Benutzer

definiert werden. Die verfügbaren CM/Track-Benutzer (Benutzerprofile) werden im Web Client beim Erstellen oder Ändern eines Kunden angezeigt. Wenn Sie diese Checkbox markieren, wird also kein echter Bearbeiter (Person) mit Zugangsberechtigungen zum System erstellt, sondern ein Benutzerprofil für CM/Track, das einem oder mehreren Kunden zugewiesen wird, die mit diesen Zugangsberechtigungen Zugang zum Portal ***CM/Track*** erhalten. Eine detaillierte Beschreibung darüber, wie Zugang zu CM/Track definiert wird, finden Sie im Abschnitt CM/Track V1: Systemzugang für CM/Track-Benutzer (Kunden) für CM/Track V1 und CM/Track V2: Systemzugang für CM/Track-Benutzer (Kunden) für CM/Track V2.

Klicken Sie zuletzt auf *Speichern*, um Ihren Eintrag zu speichern und das Fenster zu schließen.

B.1.2.3 Löschen eines Bearbeiterkontos

Um ein Bearbeiterkonto zu löschen, wählen Sie das Konto in der Liste aus und klicken Sie auf den Button *Löschen*. Da ein Bearbeiterkonto nur gelöscht werden kann, wenn es nicht für (offene oder geschlossene) Tickets verwendet wird, müssen Sie die Tickets vorher einem anderen Bearbeiter zuweisen. Der Name des gelöschten Bearbeiters wird immer noch in allen von diesem Bearbeiter erzeugten Protokolleinträgen in Tickets und auf Kundenseiten angezeigt.

Wenn Sie die Tickets nicht auf einen anderen Bearbeiter übertragen möchten, können Sie das Bearbeiterkonto deaktivieren. Siehe nächster Abschnitt.

B.1.2.4 Deaktivieren und Aktivieren eines Bearbeiterkontos

Wenn Bearbeiter für einen bestimmten Zeitraum keinen Zugang zum System haben sollen (z. B. weil sie ein Sabbatjahr machen), können ihre Konten deaktiviert werden. Dadurch werden die Tickets dieser Bearbeiter nicht verändert, aber sie können sich nicht mehr anmelden und andere Bearbeiter können ihnen keine Tickets mehr zuweisen.

Um ein Bearbeiterkonto zu deaktivieren, wählen Sie das Konto aus und klicken Sie auf den Button *Deaktivieren*. Der entsprechende Listeneintrag wird danach grau und kursiv angezeigt. Es ist nicht möglich, für dieses Konto neue Tickets zu erstellen oder vorhandene Tickets zu editieren. Klicken Sie zum erneuten Aktivieren des Kontos unten auf der Seite auf den Button *Aktivieren*.

B.1.2.5 Tab *Rollen* - Zuweisen von Rollen zu einem Bearbeiterkonto

In diesem Tab können Sie Rollen zu einem Bearbeiterkonto zuweisen. Wählen Sie das Konto auf der linken Seite aus und wählen Sie dann die gewünschten Rollen aus der Liste der verfügbaren Rollen auf der rechten Seite. Klicken Sie auf den Button *Zuweisen*, um die ausgewählten Rollen in die Liste der zugewiesenen Rollen zu verschieben. Jetzt kann der Bearbeiter, dem dieses Konto gehört, entsprechend den Berechtigungen der Rolle(n) im System arbeiten (siehe auch Rollenverwaltung).

Festlegen von Rollen als Hauptrollen

Aus der Liste der zugewiesenen Rollen können Sie eine Rolle als Hauptrolle des Bearbeiterkontos auswählen. Wählen Sie die gewünschte Rolle aus und klicken Sie unter der Liste auf den Button *Aktivieren*. Die Hauptrolle wird mit einem roten Punkt markiert. Die Sichten der Hauptrolle erscheinen für dieses Bearbeiterkonto jetzt immer oben in der Sichtenliste im Web Client.

Editieren einer der Rollen eines Bearbeiters

Jede Rolle in der Rollenliste eines Bearbeiters hat ein Kontextmenü mit zwei Elementen:

- **Zuweisen/Entfernen**
  Mit diesem Menüelement können Sie dem Bearbeiter eine Rolle zuweisen bzw. die Rolle entfernen. Dies hat den gleichen Effekt wie die Verwendung der Pfeilsymbole unter der Liste.
- **Gehe zu Rolle**
  Über dieses Menüelement können Sie schnell in das Navigationselement *Rollen* dieser Rolle wechseln.

Sie gelangen auf diese Seite, indem Sie in der Navigationsgruppe *Zugriff und Rollen*, Navigationselement *Bearbeiter* in der Liste der zugewiesenen oder verfügbaren Rollen mit der rechten Maustaste auf den Namen einer Rolle klicken.

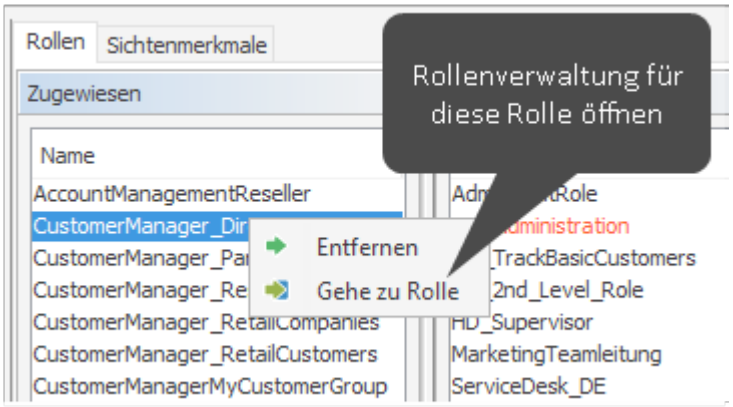


Abbildung 26: *ConSol CM Admin Tool - Zugriff und Rollen, Bearbeiter: Kontextmenü einer Rolle*

B.1.2.6 Tab *Sichtenmerkmale* - Definieren von bearbeiterspezifischen Sichtenmerkmalen

Hier können Sie die dynamischen Sichtenmerkmale für bestimmte Bearbeiter ändern. Dynamische Merkmale werden verwendet, damit der Bearbeiter die Möglichkeit hat, eine Sicht interaktiv im Web Client anzupassen (siehe auch Sichtenverwaltung).

> Sichtenmerkmale werden in diesem Tab nur angezeigt, wenn Sie eine Sicht mit dynamischen Merkmalen erstellt und der Rolle des Bearbeiters zugewiesen haben.

Sie gelangen auf diese Seite, indem Sie im Navigationselement *Bearbeiter*, Navigationsgruppe *Zugriff und Rollen* auf den Tab *Sichtenmerkmale* klicken.

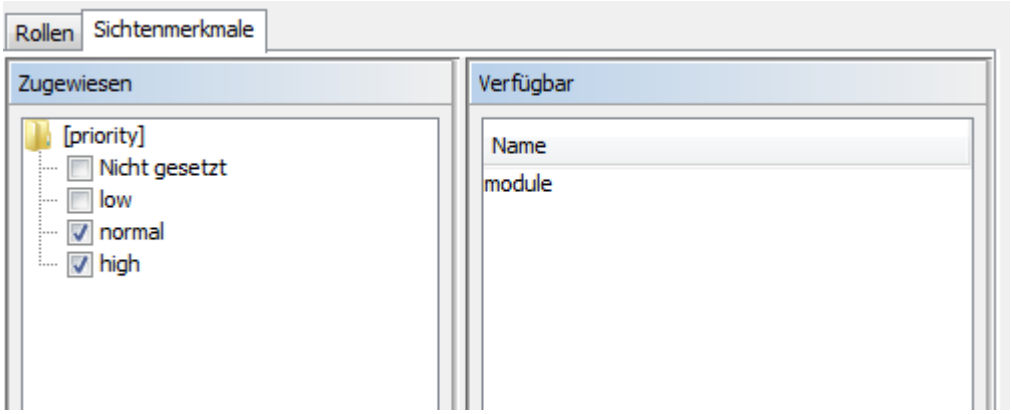


Abbildung 27: *ConSol CM Admin Tool - Zugriff und Rollen, Bearbeiter: Sichtenmerkmale*

Wählen Sie das Bearbeiterkonto auf der linken Seite aus und wählen Sie dann das gewünschte Merkmal aus der Liste der verfügbaren Sichtenmerkmale auf der rechten Seite. Klicken Sie auf den Button *Zuweisen*, um es in die Liste der zugewiesenen Sichtenmerkmale zu verschieben. Die möglichen Werte werden unter dem Merkmal in der Liste angezeigt. Markieren Sie die Checkboxen der Werte, die Sie ändern oder voreinstellen möchten. Der Bearbeiter kann diese Einstellungen im Web Client (Profilseite) ändern. Änderungen, die Sie im Admin Tool machen, sind sofort auf der Profilseite des Bearbeiters sichtbar.

Beispiel:
Sie haben das dynamische Merkmal *Priorität* zugewiesen. In der Liste werden die Werte „Nicht gesetzt“, „Niedrig“, „Normal“ und „Hoch“ angezeigt. Wenn Sie die Werte „Normal“ und „Hoch“ markieren, sieht der Bearbeiter nur Tickets mit normaler oder hoher Priorität, wenn er sich im Web Client anmeldet. Wenn Sie keine Werte markieren, sieht der Bearbeiter mit dieser Sicht keine Tickets. Details dazu finden Sie im Abschnitt Sichtenverwaltung.

⚠ Beachten Sie, dass in einer Sicht mit dynamischen Merkmalen nur die Tickets angezeigt werden, die mit diesem Merkmal übereinstimmen. Wenn Bearbeiter in ihren Bearbeiterprofilen keine Merkmale ausgewählt haben, oder wenn der Administrator die Auswahl im Admin Tool entfernt hat, ist die Sicht für die Bearbeiter leer! Stellen Sie sicher, dass den Benutzern diese Tatsache bewusst ist, und denken Sie auch als Administrator immer daran.

B.2 Rollenverwaltung

In diesem Kapitel werden folgende Themen behandelt:

B.2.1 Einführung in die Rollenverwaltung 68

B.2.2 Rollenverwaltung mit dem Admin Tool 69

B.2.3 Definieren eines Administrators nur für die Verwaltung der Rollen und Bearbeiter 86

B.2.1 Einführung in die Rollenverwaltung

Mit Rollen werden Zugangsberechtigungen erteilt und Sichten zugewiesen. Sie definieren, was ein Bearbeiter tun und sehen darf. Ohne Rolle kann sich ein Bearbeiter zwar im System anmelden, darf aber keine Aktionen ausführen. Nur durch die Zuweisung einer oder mehrerer Rollen erhält der Bearbeiter Berechtigungen im System. Ein Unternehmen sollte für jede Aufgabe, die die Benutzung der Software erfordert, eine Rolle anlegen, in der die entsprechenden Berechtigungen definiert werden. Den Bearbeitern, die mit dieser Aufgabe betraut sind, sollte diese Rolle zugewiesen werden.



Wenn sich die Bearbeiter im System anmelden, haben sie die Berechtigungen aller ihnen zugewiesenen Rollen. Es werden also alle Berechtigungen addiert! Es ist nicht möglich, den Zugang zu Objekten in ConSol CM explizit zu verhindern, es wird immer Zugang gewährt! Die Summe aller erteilten Berechtigungen ergibt die finalen Berechtigungen des Bearbeiters.

Rollen definieren:

- **Zugangsberechtigungen zu einer oder mehreren Queues**
  Es werden zum Beispiel Berechtigungen zum Lesen, Schreiben und Hinzufügen erteilt. Die Berechtigungen gelten für alle Tickets in der jeweiligen Queue.
- **Allgemeine Berechtigungen**
  Hier werden mehrere systemweite Berechtigungen verwaltet, wie zum Beispiel Berechtigungen zur Vorlagenverwaltung, Workflow-Design und Systemadministration. Über die Option *Zugriff und Rollen administrieren* ist es möglich, einen Administrator "light" zu definieren, der CM-Bearbeiter mit ihren Berechtigungen für den Systemzugang verwalten kann, ohne die technischen systemweiten Einstellungen ändern zu können. Dies ist im Abschnitt Definieren eines Administrators nur für die Verwaltung der Rollen und Bearbeiter beschrieben.
- **Zugangsberechtigungen zu Kundendaten**
  Berechtigungen zum Lesen, Schreiben, Ändern und Löschen für unterschiedliche Kundengruppen.
- **Zugangsberechtigungen zu Ressourcendaten**
  Berechtigungen zum Lesen, Schreiben, Ändern und Erstellen, die für einzelne Ressourcentypen zugewiesen werden.
- **Sichten**
  To-do-Listen mit Tickets, die in der Ticketliste im Web Client angezeigt werden.

- **Bearbeiterfunktionen**
  Zusätzliche Bearbeiterfunktionen, die Bearbeitern mit dieser Rolle zugewiesen werden können, z. B. *Genehmiger*.

B.2.2 Rollenverwaltung mit dem Admin Tool

Sie gelangen auf diese Seite, indem Sie das Navigationselement *Rollen* in der Navigationsgruppe *Zugriff und Rollen* öffnen.

Zudem sehen Sie diese Seite, wenn Sie in der Bearbeiterverwaltung in der Rollenliste eines Bearbeiters für eine Rolle auf den Link *Gehe zu Rolle* geklickt haben.

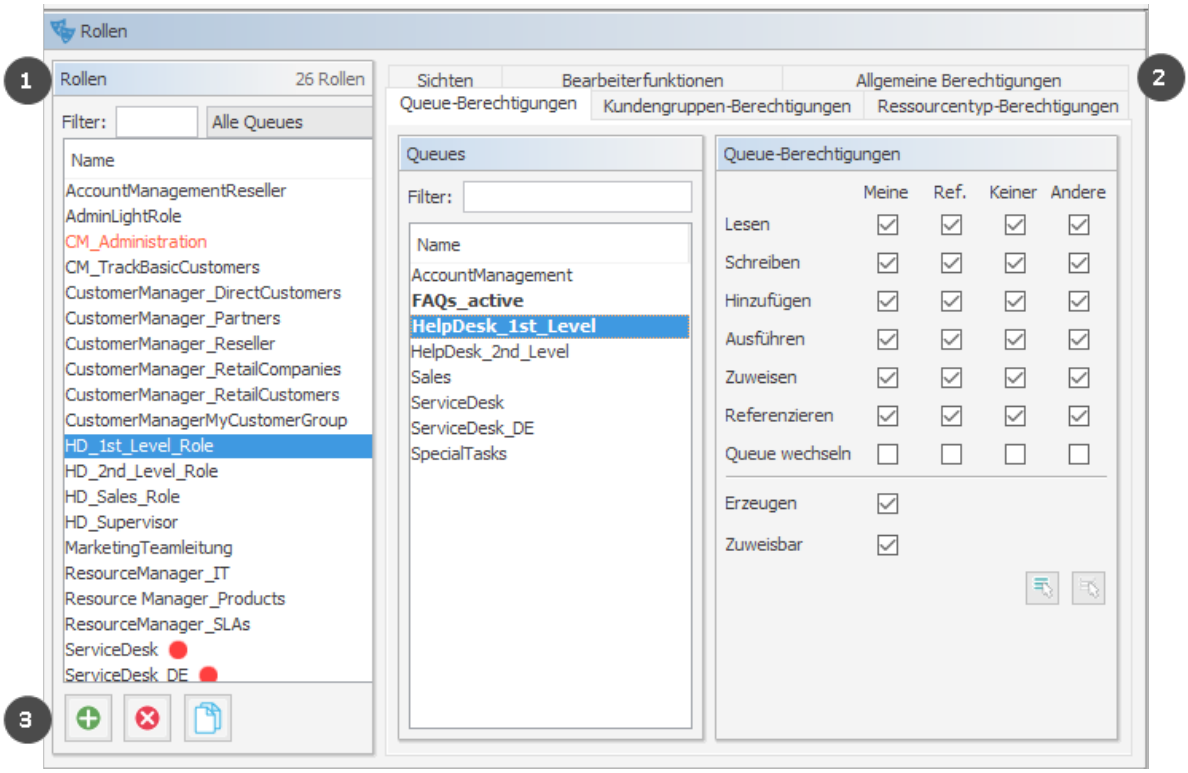


Abbildung 28: *ConSol CM Admin Tool - Zugriff und Rollen, Rollen: Queue-Berechtigungen*

Sie sehen eine Liste aller verfügbaren Rollen auf der linken Seite (1) und die Berechtigungen, die vergeben werden können, auf der rechten Seite (2). In der Liste der Rollen sind alle Rollen, die bei mindestens einem Bearbeiter als **Hauptrolle** festgelegt wurden, mit einem roten Punkt markiert. Sie arbeiten immer mit den Zugangsberechtigungen für die Rolle, die in der Rollenliste ausgewählt wurde. Es kann immer nur eine Rolle gleichzeitig ausgewählt werden. Mit den Buttons unter der Liste der Rollen (3) können Sie eine Rolle hinzufügen, löschen oder kopieren.

Auf der rechten Seite stehen die verfügbaren Tabs:

- Tab Queue-Berechtigungen
- Tab Allgemeine Berechtigungen
- Tab Kundengruppen-Berechtigungen

- Tab Ressourcentyp-Berechtigungen
- Tab Sichten
- Tab Bearbeiterfunktionen



Alle Änderungen in den Tabs der Rollenverwaltung werden sofort, bzw. nach Klick auf den Button *OK* wirksam. Es ist nicht erforderlich, in der Symbolleiste auf den Button *Synchronisieren* zu klicken.

Die Bearbeiter müssen sich im Web Client neu anmelden, um ihre neuen Rollen benutzen zu können. Sichten werden nach dem Drücken von F5 (Seite im Browser aktualisieren) wirksam.

Wenn Sie in der Rollenverwaltung arbeiten, können Sie immer eine Rolle markieren und die Parameter dieser Rolle anzeigen und ändern. Es ist allerdings nicht möglich, eine Liste aller Bearbeiter anzuzeigen, denen diese Rolle zugewiesen ist. Um eine solche Liste zu erhalten, wechseln Sie in die Bearbeiterverwaltung (Navigationselement *Bearbeiter*) und filtern Sie die Bearbeiterliste nach einer bestimmten Rolle.

Bitte beachten Sie, dass Sie den Namen einer Rolle ändern können, indem sie in der Liste der Rollen auf den Namen klicken und ihn wie gewünscht modifizieren.

B.2.2.1 Erstellen einer Rolle

Klicken Sie auf den Button *Hinzufügen* unter der Rollenliste, um eine neue Rolle zu erstellen. Es erscheint ein Pop-up-Fenster, in dem Sie den Namen der Rolle eingeben können. Da der Rollenname nur zu Administrationszwecken verwendet wird und im Web Client nicht angezeigt wird, ist hier keine Lokalisierung erforderlich. Als nächstes müssen die Berechtigungen für diese Rolle in den Tabs auf der rechten Seite definiert werden (siehe auch Bild oben).

Tab *Queue-Berechtigungen*

Die in diesem Tab festgelegten Berechtigungen gelten für die ausgewählte Rolle (linker Teil der Seite) und die ausgewählte Queue (mittlerer Teil der Seite). Wenn Sie hier nichts auswählen, kann ein Bearbeiter mit dieser Rolle keine Tickets sehen und keine Aktionen im System ausführen.

Sie gelangen auf diese Seite, indem Sie im Navigationselement *Rollen*, Navigationsgruppe *Zugriff und Rollen* auf den Tab *Queue-Berechtigungen* klicken.

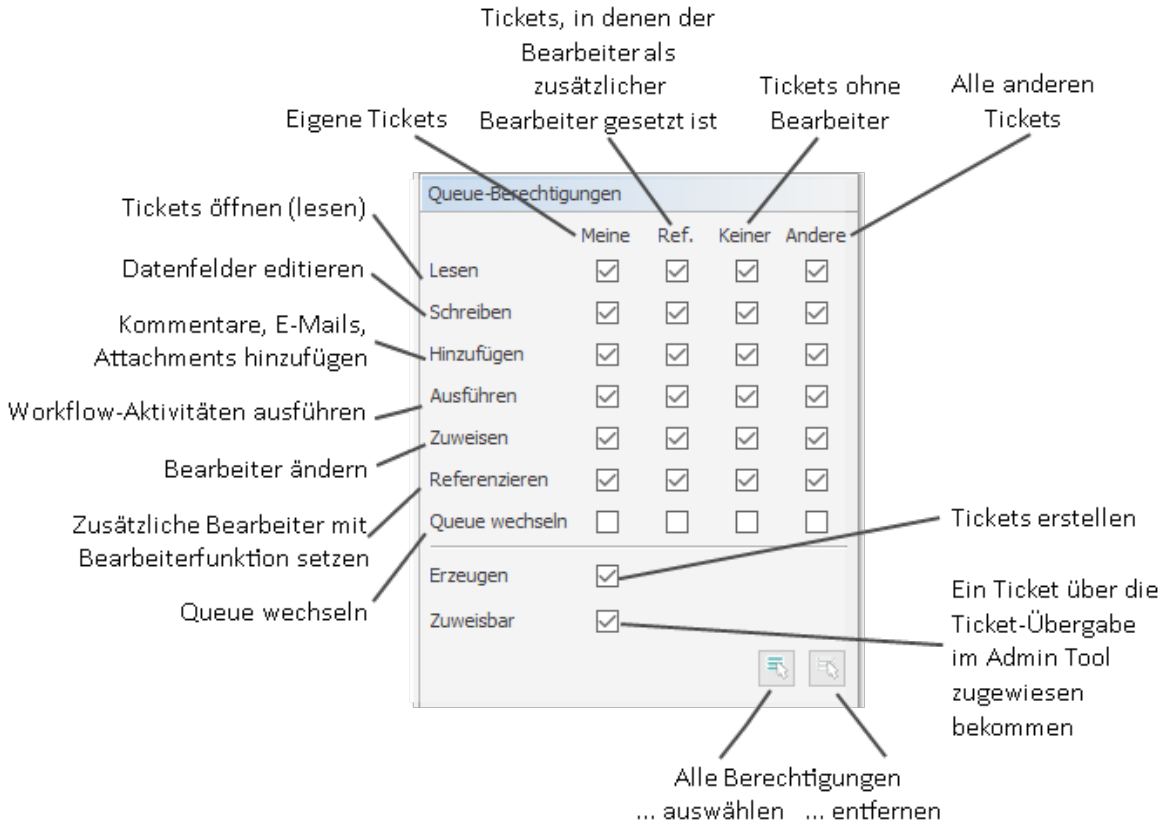


Abbildung 29: *ConSol CM Admin Tool - Zugriff und Rollen, Rollen: Erteilen von Queue-Berechtigungen*

Es können folgende Berechtigungen vergeben werden:

- **Lesen**
  Tickets lesen.
- **Schreiben**
  Datenfelder eines Tickets editieren (Standardfelder, Ticketfelder usw.). Diese Felder können sich entweder im Kopfbereich oder im Bereich für Detaildaten des Tickets befinden.
- **Hinzufügen**
  Informationen zu einem Ticket hinzufügen (Kommentare, E-Mails, Attachments, Zeitbuchungen), d. h. Inhalt zum Ticketprotokoll hinzufügen.
- **Ausführen**
  Workflow-Aktivitäten ausführen, d. h. das Ticket im Workflow vorwärts bewegen.
- **Zuweisen**
  Tickets einem anderen Bearbeiter zuweisen. Die Berechtigung, sich selbst Tickets zuzuweisen und Tickets anzunehmen, ist in diesem Kontext nicht relevant.
  Der Bearbeiter, der das Ticket erhalten soll, muss mindestens eine Rolle mit der Berechtigung *Zuweisbar* haben!

- **Referenzieren**
  Einem Ticket einen zusätzlichen Bearbeiter (mit Bearbeiterfunktion, siehe Tab Bearbeiterfunktionen) zuweisen.
- **Queue wechseln**
  Ein Ticket aus der aktuellen Queue in eine andere Queue verschieben.
  Wenn der aktuelle Bearbeiter die Berechtigung *Queue wechseln* für die entsprechende Ticketmenge (eigene Tickets, referenzierte Tickets usw., sieh unten) hat, wird das Drop-down-Menü zum Ändern der Queue angezeigt. Alle Queues, für die der aktuelle Bearbeiter die Berechtigung *Queue wechseln* hat, sind aufgeführt. Das bedeutet, dass der Bearbeiter die Berechtigung *Queue wechseln* sowohl für die Ausgangs-Queue als auch für die Ziel-Queue benötigt. Auch kann der Bearbeiter den gesamten Vorgang nur durchführen, wenn er die Berechtigung für die richtige Ticketmenge der Ziel-Queue hat. Wenn ein Bearbeiter zum Beispiel ein ihm zugewiesenes Ticket in eine andere Queue verschieben möchte, braucht er dafür mindestens die Berechtigung *Queue wechseln* für die Ticketmenge *Eigene* der Ziel-Queue. Wenn eine andere Ticketmenge gesetzt wurde (z. B. *Ref.* oder *Keiner*), gelingt der Vorgang nicht, da der Ticketbearbeiter während des Vorgangs nicht geändert wird. Deshalb muss die richtige Ticketmenge verwendet werden.

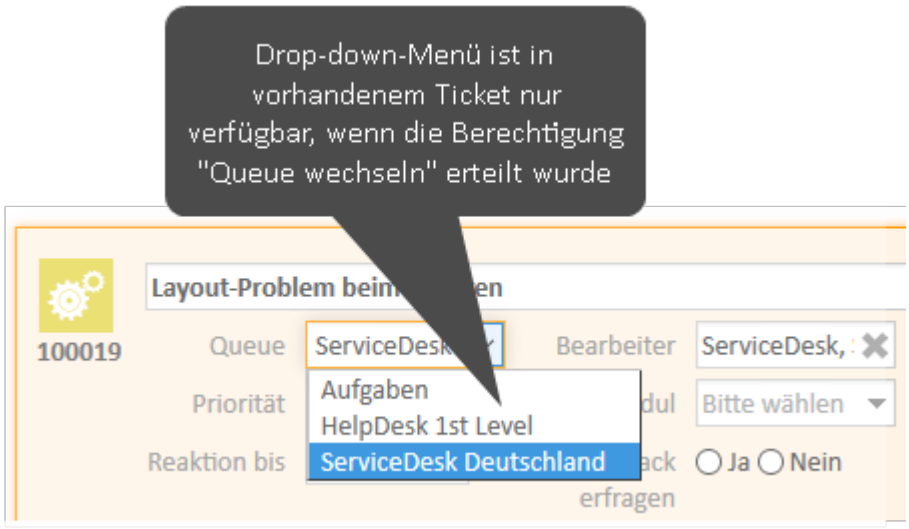


Abbildung 30: *ConSol CM Web Client - Drop-down-Menü zum Wechseln der Queue*

Es hängt vom Workflow der Ziel-Queue ab, wo die Verarbeitung des Tickets fortgesetzt wird:

- Wenn Ausgangs-Queue und die Ziel-Queue den gleichen Workflow haben, beginnt das Ticketverarbeitung in der Ziel-Queue an der ursprünglichen Position des Tickets (d. h. der letzten Position in der Ausgangs-Queue).
- Wenn die Ausgangs-Queue und die Ziel-Queue unterschiedliche Workflows haben, beginnt das Ticket den Prozess in der Ziel-Queue am START-Knoten.

> ⚠ Seien Sie vorsichtig mit der Vergabe der Berechtigung *Queue wechseln*! Normalerweise ist sie nicht erforderlich. Sie kann hingegen Ihre Definition der Prozesskette unterlaufen, wenn Tickets mit Prozess-/Workflow-Komponenten (den Einsprung- und Aussprungknoten) von einem Prozess in einen anderen überführt werden.
>
> Diese Berechtigung sollte nur erteilt werden, wenn es absolut notwendig ist und alle Nebeneffekte gründlich analysiert wurden!

Sie können festlegen, für welche Ticketmenge die Berechtigungen gelten:

- **Meine**
  Eigene Tickets.
- **Ref.**
  Tickets, denen der Bearbeiter als zusätzlicher Bearbeiter (mit Bearbeiterfunktion, siehe Tab Bearbeiterfunktionen) zugewiesen ist.
- **Keiner**
  Tickets, die keinem Bearbeiter zugewiesen sind.
- **Andere**
  Tickets, die anderen Bearbeitern zugewiesen sind.

Klicken Sie auf die jeweilige Checkbox, um eine oder mehrere Berechtigungen für die gewünschte Ticketmenge zu erteilen.

Es gibt zwei allgemeine Berechtigungen, die vergeben werden können:

- **Erzeugen**
  Der Bearbeiter darf Tickets in dieser Queue erstellen.
- **Zuweisbar**
  Andere Bearbeiter können Ticket einem Bearbeiter zuweisen, der eine Rolle mit dieser Berechtigung hat (sofern die anderen Bearbeiter die Berechtigung *Zuweisen* haben!)
  Dem Bearbeiter können Tickets über die Ticketübergabe mit dem Admin Tool zugewiesen werden.

Wenn Sie alle Berechtigungen gleichzeitig auswählen möchten, klicken Sie einfach unter der Liste auf den Button *Alle Berechtigungen*. Ein Klick auf *Keine Berechtigungen* entfernt die gesamte Auswahl.

Tab *Allgemeine Berechtigungen*

Allgemeine Berechtigungen sind globale, von der Queue unabhängige Berechtigungen für eine Rolle. Die Festlegung dieser Berechtigungen ist optional.

Sie gelangen auf diese Seite, indem Sie im Navigationselement *Rollen*, Navigationsgruppe *Zugriff und Rollen*, auf den Tab *Allgemeine Berechtigungen* klicken.



Abbildung 31: *ConSol CM Admin Tool - Zugriff und Rollen, Rollen: Allgemeine Berechtigungen*

Sie können folgende Berechtigungen vergeben:

- **Administrator-Berechtigungen (1)**
  - **System administrieren (Vollzugriff**
    Mit dieser Berechtigung wird Administratorzugang zum gesamten CM-System gewährt. Dieser gilt für das Admin Tool, den Process Designer und den Administratorzugang zum Web Client. Ein Administrator mit dieser Rolle hat Zugriff auf alle Navigationsgruppen und -elemente im Admin Tool.
  - **Zugriff und Rollen administrieren**
    Mit dieser Berechtigung wird Administratorzugang nur auf die Navigationsgruppe *Zugriff und Rollen* gewährt. Details dazu finden Sie im Abschnitt Definieren eines Administrators nur für die Verwaltung der Rollen und Bearbeiter.
- **Workflow-Berechtigungen (2)**
  Erteilt Berechtigungen für das Design und Management von Workflows. Das sind:
  - **Lesen**
  - **Schreiben** (ändern und speichern)
  - **Installieren** (installieren und in Betrieb nehmen).

- **Textvorlagen-Berechtigungen (3)**
  - **Textvorlage speichern** erteilt die Berechtigung
    - zur Verwendung des Textvorlagen-Managers, der zum Erstellen und Editieren von E-Mail- und Kommentarvorlagen eingesetzt wird. Siehe Abschnitt Der ConSol CM Textvorlagen-Manager.
    - zur Verwendung des Dokumentvorlagen-Managers, der zum Definieren von Vorlagen für CM/Doc eingesetzt wird. Nur verfügbar, wenn CM/Doc im CM-System aktiviert ist.
- **Vertretungs-Berechtigungen (4)**
  - **Vertretung einrichten**
    Wenn diese Berechtigung vergeben ist, können sich die Bearbeiter mit dieser Rolle als Vertretung für andere Bearbeiter eintragen, die z. B. krank sind und keine anderen Bearbeiter als Vertretung angegeben haben bzw. momentan nicht verfügbar sind. Im Web Client wird dem Bearbeiter dann in seinem Bearbeiterprofil eine Liste mit den Bearbeitern angezeigt, die er vertreten kann.

Wichtige Informationen über die Konfigurationen von Vertretungen

Beachten Sie, dass es zwei unterschiedliche Szenarien für das Senden von E-Mails gibt und dass das Verhalten des CM-Systems bezüglich des Sendens von Vertretungs-E-Mails in den beiden Szenarien unterschiedlich sein kann!

- **Ein Bearbeiter schreibt eine E-Mail mit dem Ticket-E-Mail-Editor**
  Es hängt vom Wert der System-Property cmweb-server-adapter, forward.mails.to.representatives ab, ob die Vertretungsregel angewendet wird und der Vertreter eine Kopie der E-Mail erhält. Standardmäßig ist diese Property auf „false“ gesetzt. Dies bedeutet, dass diese E-Mail **nicht** an den Vertreter gesendet wird. Ist die Property auf „true“ gesetzt, werden alle E-Mails, die manuell mit CM gesendet werden, an den ursprünglichen Empfänger **und** an den aktuellen Vertreter gesendet. Das CM-System überprüft, ob für eine bestimmte E-Mail-Adresse eine Vertretungsregel aktiv ist! Bedenken Sie dies, wenn Sie die Vertretungsberechtigungen im Admin Tool konfigurieren und informieren Sie Ihre CM-Benutzer (Bearbeiter) über dieses Verhalten. Es kann unerwünschte Auswirkungen haben, insbesondere, wenn Personen als Bearbeiter und als Kontakte im ConSol CM-System erfasst sind (z. B. für einen internen Helpdesk).
- **Eine E-Mail wird automatisch aus dem CM-System gesendet**
  Es hängt von der spezifischen Konfiguration des CM-Systems ab, welche Bearbeiter eine Kopie der E-Mail erhalten. Die E-Mail wird nicht automatisch an die Vertreter gesendet!
  Es kann implementiert werden, dass der Vertreter eine Kopie erhält, dies ist aber nicht obligatorisch. Die ursprüngliche E-Mail kann aus einem Workflow-Skript oder einem Admin-Tool-Skript gesendet werden (das ebenfalls aus dem Workflow aufgerufen werden kann). Es hängt von der Implementierung dieses Skripts ab, wer eine Kopie der E-Mail erhält. Details dazu finden Sie im *ConSol CM Process Designer Handbuch*.

- **Track-Benutzer-Berechtigungen (5)**
  - **Auf Tickets der eigenen Firma zugreifen**
    Benutzer mit dieser Berechtigung dürfen in CM/Track nicht nur ihre eigenen Tickets aufrufen, sondern auch alle Tickets der Firma, zu der sie gehören. Diese Berechtigung ist nur für Rollen sinnvoll, die Zugangsberechtigungen für CM/Track-Benutzer/Benutzerprofile definieren, nicht für einzelne Benutzer.

Tab *Kundengruppen-Berechtigungen*

Damit die Bearbeiter mit den Kundendaten von einer oder mehreren Kundengruppen arbeiten können, z. B. um Händlerdatensätze zu editieren oder einen neuen Kontakt für die Kundengruppe anzulegen, müssen Sie einer oder mehreren Rollen Zugangsberechtigungen für die Kundengruppen erteilen.

Sie gelangen auf diese Seite, indem Sie im Navigationselement *Rollen*, Navigationsgruppe *Zugriff und Rollen*, auf den Tab *Kundengruppen-Berechtigungen* klicken.

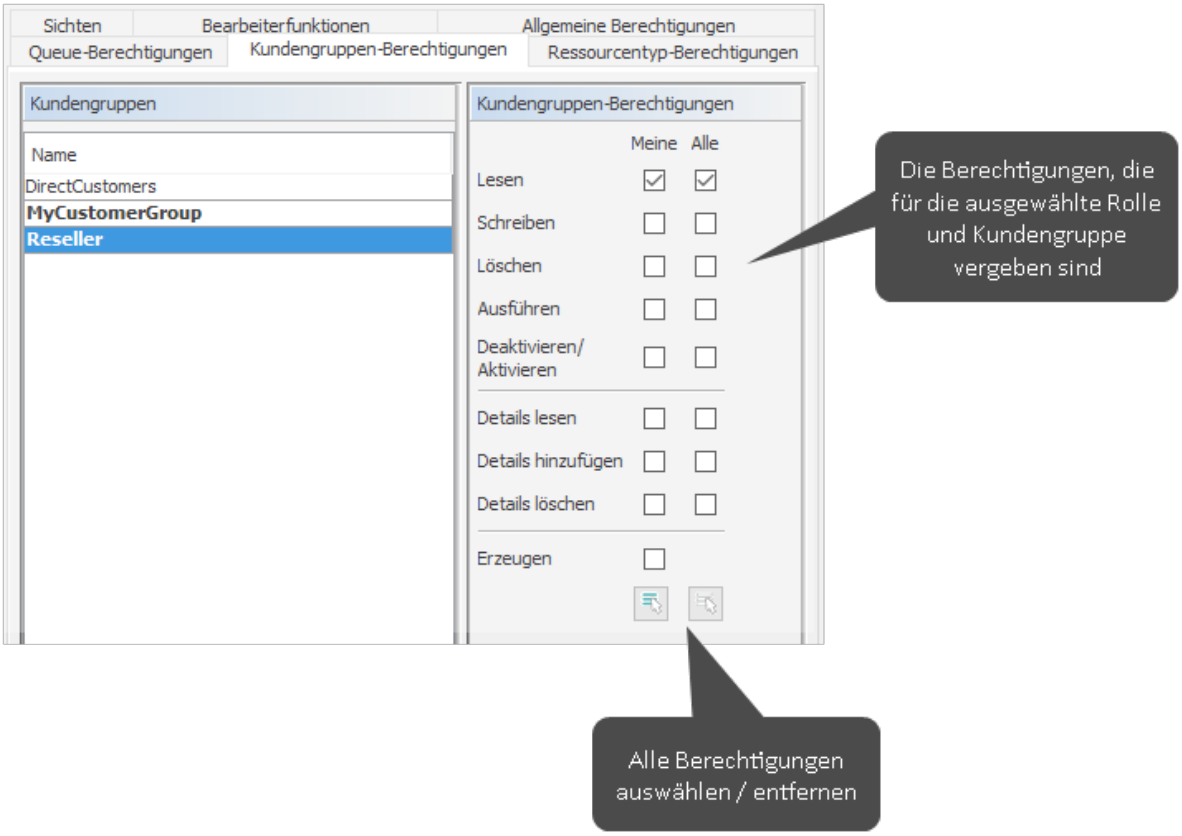


Abbildung 32: *ConSol CM Admin Tool - Zugriff und Rollen, Rollen: Zuweisen von Berechtigungen für Kundengruppen zu einer Rolle*

In verschiedenen Kundenumgebungen hat es sich als praktisch erwiesen, spezielle Rollen nur für die Verwaltung von Kundendaten zu erstellen. Zum Beispiel eine Rolle *CustomerManager_CustomerGroup1* und eine andere Rolle *CustomerManager_CustomerGroup2*. Sie können sogar zwischen CustomerManager_CustomerGroup1_full und CustomerManager_CustomerGroup1 _light unterscheiden. Auf diese Weise können Sie die Berechtigungen zur Kundenverwaltung über die Kundenmanager-Rollen ein- und ausschalten. Außerdem sind die Berechtigungen zur Kundenverwaltung sauber von den Queue-Berechtigungen getrennt. Dies kann insbesondere in gemischten Teams sehr praktisch sein, wenn nicht jeder in der Lage sein soll, alle Kundendaten zu editieren.

Denken Sie dabei daran, allen Bearbeitern der jeweiligen Queues Lese-Berechtigungen für die Kundendaten aller benötigten Kundengruppen zu erteilen. Ansonsten können sie ihre Tickets nicht öffnen!

Die Zugangsberechtigungen, die erteilt werden können, wurden in CM Version 6.9.1 erweitert, verglichen mit vorherigen Versionen. Es wurden Berechtigungen für einen neuen Bereich der Kundenseite hinzugefügt. Sowohl die Kontaktseite als auch die Firmenseite haben im Web Client einen neuen Bereich, den Bereich für Kommentare und Attachments (2).

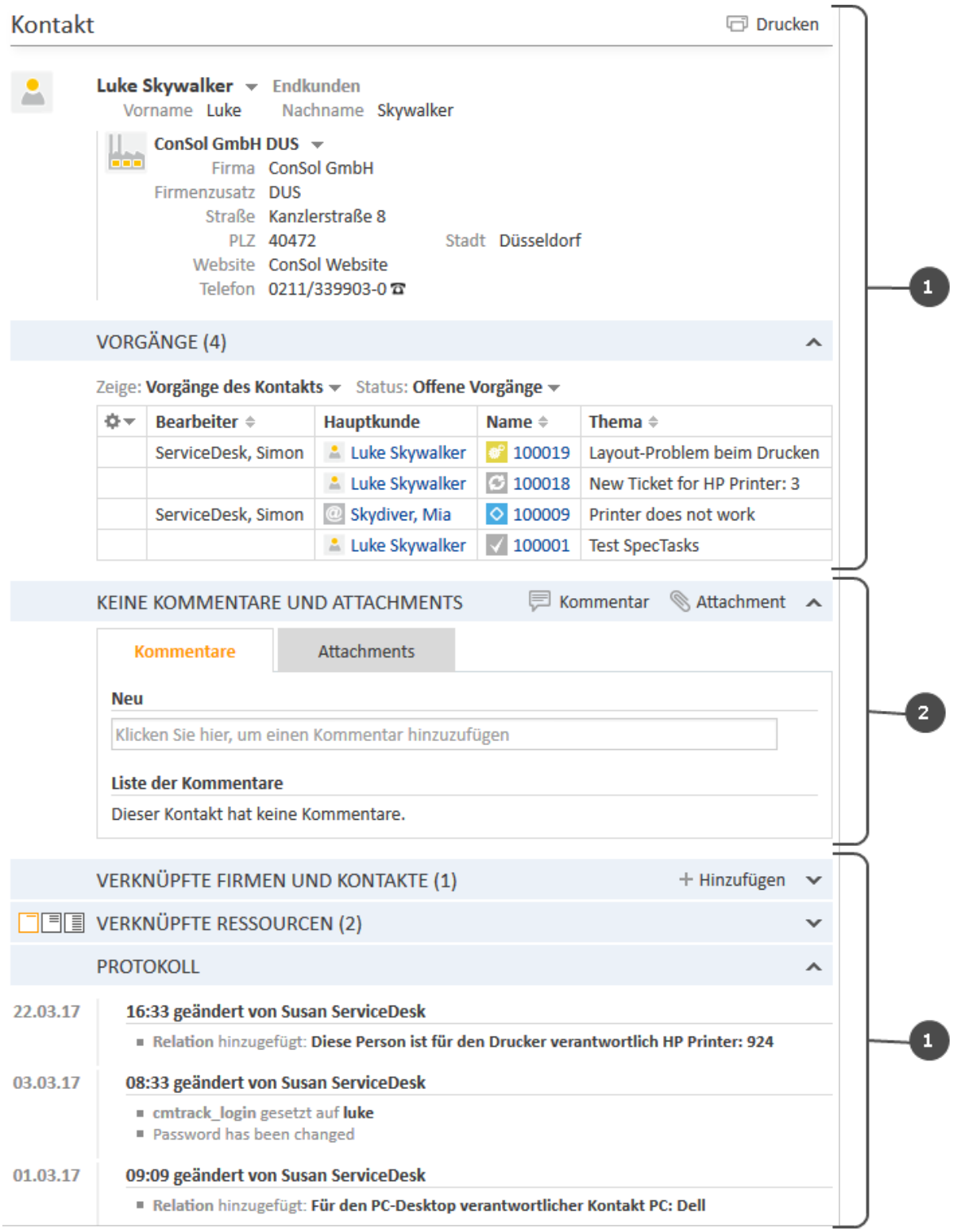


Abbildung 33: *ConSol CM Web Client - Bereiche einer Kontaktseite*

Die folgenden Zugangsberechtigungen können erteilt werden:

- **Art des Kunden**
  Bezieht sich auf die Tickets des Kunden.
  - **Meine**
    Alle Kunden (Hauptkunden und Zusatzkunden) der Tickets, die dem Bearbeiter aktuell zugewiesen sind oder bei denen der Bearbeiter als zusätzlicher Bearbeiter angegeben ist.
  - **Alle**
    Alle Kunden.
- **Allgemeine Bereiche (1)**
  - **Lesen**
    Kundendaten lesen.
  - **Schreiben**
    Kundendaten schreiben/ändern, und die Firma eines Kontakts auf der Kontaktseite über den Link *Wechseln* ändern.
  - **Löschen**
    - Kundendatensatz löschen. Dies gilt für Firmen und Kontakte. Für Kontakte sind zwei Arten von Löschung verfügbar (mit oder ohne verknüpfte Daten).
    - Alle mit einem Kunden aus dieser Kundengruppe verknüpften Tickets an einen anderen Kunden übertragen.
    - Einen Kontakt anonymisieren.
  - **Ausführen**
    Aktionen für diesen Kunden ausführen (Details über Kundenaktionen finden Sie im Abschnitt Action Framework - Kundenaktionen.
  - **Deaktivieren/Aktivieren**
    - Den Kontakt oder die Firma deaktivieren und wieder aktivieren. Für einen deaktivierten Kunden können keine Tickets erstellt werden.
    - Alle mit einem Kunden aus dieser Kundengruppe verknüpften Tickets an einen anderen Kunden übertragen.

> **Information zur Berechtigung Tickets und Ressourcen übertragen**
>
> Bitte beachten Sie Folgendes:
>
> - In CM-Versionen vor 6.9.4.1 war die Berechtigung *Tickets übertragen* nicht beschränkt.
> - In CM-Versionen 6.9.4.1 bis 6.10.4.3 war die Berechtigung *Tickets übertragen* an die Berechtigung *Löschen* (von Kundendaten) gekoppelt.
> - Seit CM-Version 6.10.4.4 wird die Berechtigung *Tickets übertragen* automatisch mit der Berechtigung *Löschen* (von Kundendaten) sowie *Deaktivieren/Aktivieren* (von Kundendaten) vergeben, d. h. ein Bearbeiter kann Daten übertragen, wenn er eine dieser beiden Berechtigungen hat.

- **Bereich für Kommentare und Attachments (2)**
  - **Details lesen**
    Kundendaten im Bereich für Kommentare und Attachments lesen.
  - **Details hinzufügen**
    Kundendaten im Bereich für Kommentare und Attachments hinzufügen/ändern.
  - **Details löschen**
    Kundendaten im Bereich für Kommentare und Attachments löschen.
- **Allgemein**
  - **Erzeugen**
    Kundendatensatz erstellen. Bei einem zweistufigen Kundendatenmodell gilt diese Berechtigung sowohl für Kontaktdatensätze als auch für Firmendatensätze.

> ⚠ Beachten Sie, dass ein Bearbeiter mindestens Lese-Berechtigungen für eine Kundengruppe haben muss, um Tickets für Kunden aus dieser Gruppe öffnen bzw. erstellen zu können!

Tab *Ressourcentyp-Berechtigungen*

Ressourcentyp-Berechtigungen steuern den Zugang der Bearbeiter zu Ressourcen, d. h. Objekten, die im Ressourcenpool gespeichert sind.

Sie gelangen auf diese Seite, indem Sie im Navigationselement *Rollen*, Navigationsgruppe *Zugriff und Rollen*, auf den Tab *Ressourcentyp-Berechtigungen* klicken.

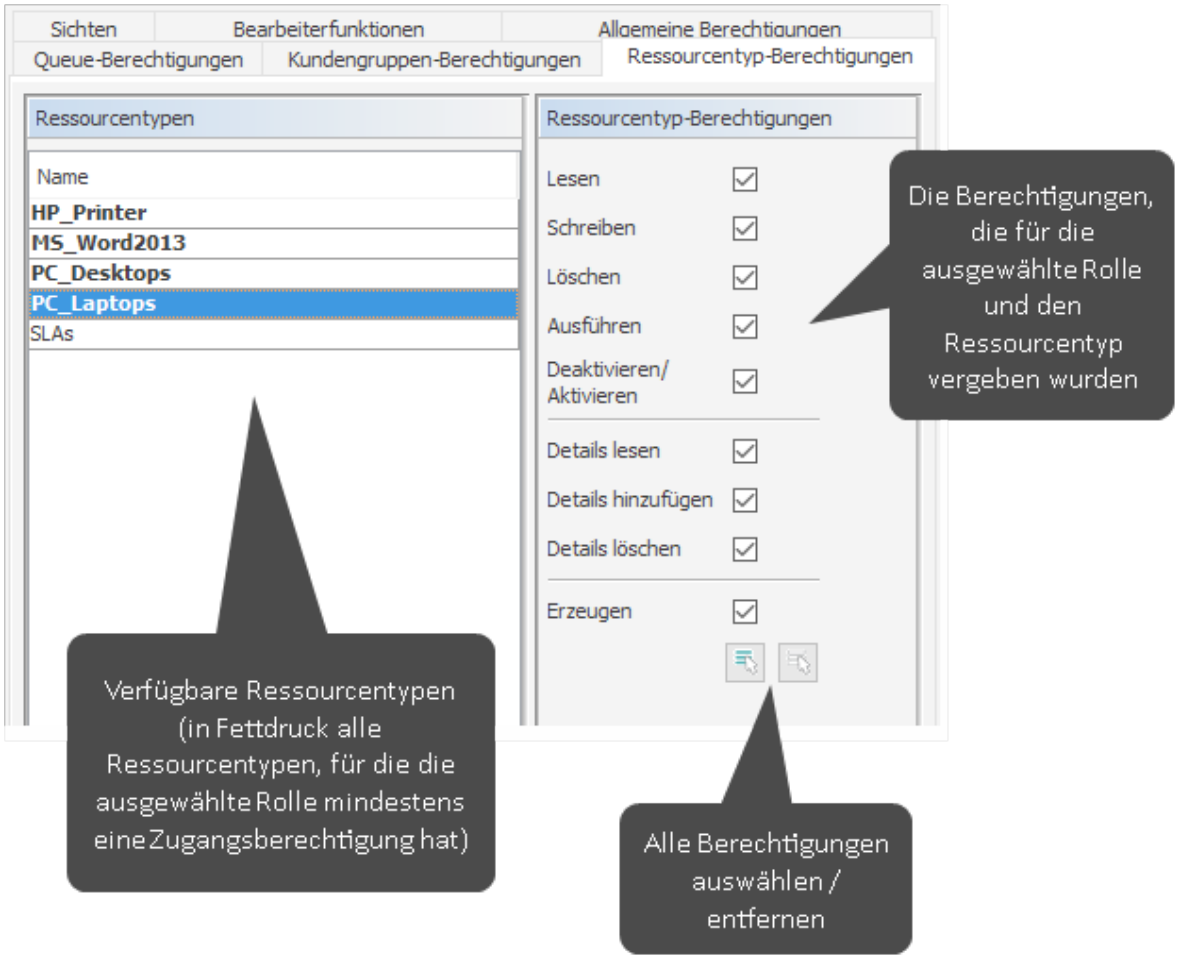


Abbildung 34: *ConSol CM Admin Tool - Zugriff und Rollen, Rollen: Ressourcentyp-Berechtigungen*

Die folgenden Berechtigungen können erteilt werden:

- **Lesen**
  Ressourcen des ausgewählten Typs im Web Client laden und anzeigen.
- **Schreiben**
  Inhalt von Datenfeldern dieses Ressourcentyps ändern.
- **Löschen**
  Ressourcen des entsprechenden Typs aus CM löschen.
- **Ausführen**
  Ressourcenaktionen für diesen Ressourcentyp ausführen.
- **Deaktivieren/Aktivieren**
  Ressourcen des ausgewählten Typs (de)aktivieren.
- **Details lesen**
  Kommentare/Attachments für Ressourcen dieses Typs laden und anzeigen.

- **Details hinzufügen**
  Kommentare/Attachments für Ressourcen dieses Typs hinzufügen und ändern.
- **Details löschen**
  Kommentare und Attachments für Ressourcen dieses Typs löschen.
- **Erzeugen**
  Neue Ressource dieses Typs erstellen.

Tab *Sichten*

Sichten definieren, welche Tickets die Bearbeiter in ihrer Ticketliste im Web Client sehen. In diesem Tab werden die zugewiesenen Sichten links und die verfügbaren Sichten rechts angezeigt (siehe auch Sichtenverwaltung). Die angezeigten Sichten können nach Name oder Queue gefiltert werden. Das Zuweisen von Sichten ist optional.

> Wir empfehlen, einer Rolle mindestens eine Sicht zuzuweisen. Andernfalls sehen die Bearbeiter mit dieser Rolle keine Tickets in ihrer Ticketliste im Web Client.

Sie gelangen auf diese Seite, indem Sie im Navigationselement *Rollen*, Navigationsgruppe *Zugriff und Rollen*, auf den Tab *Sichten* klicken.

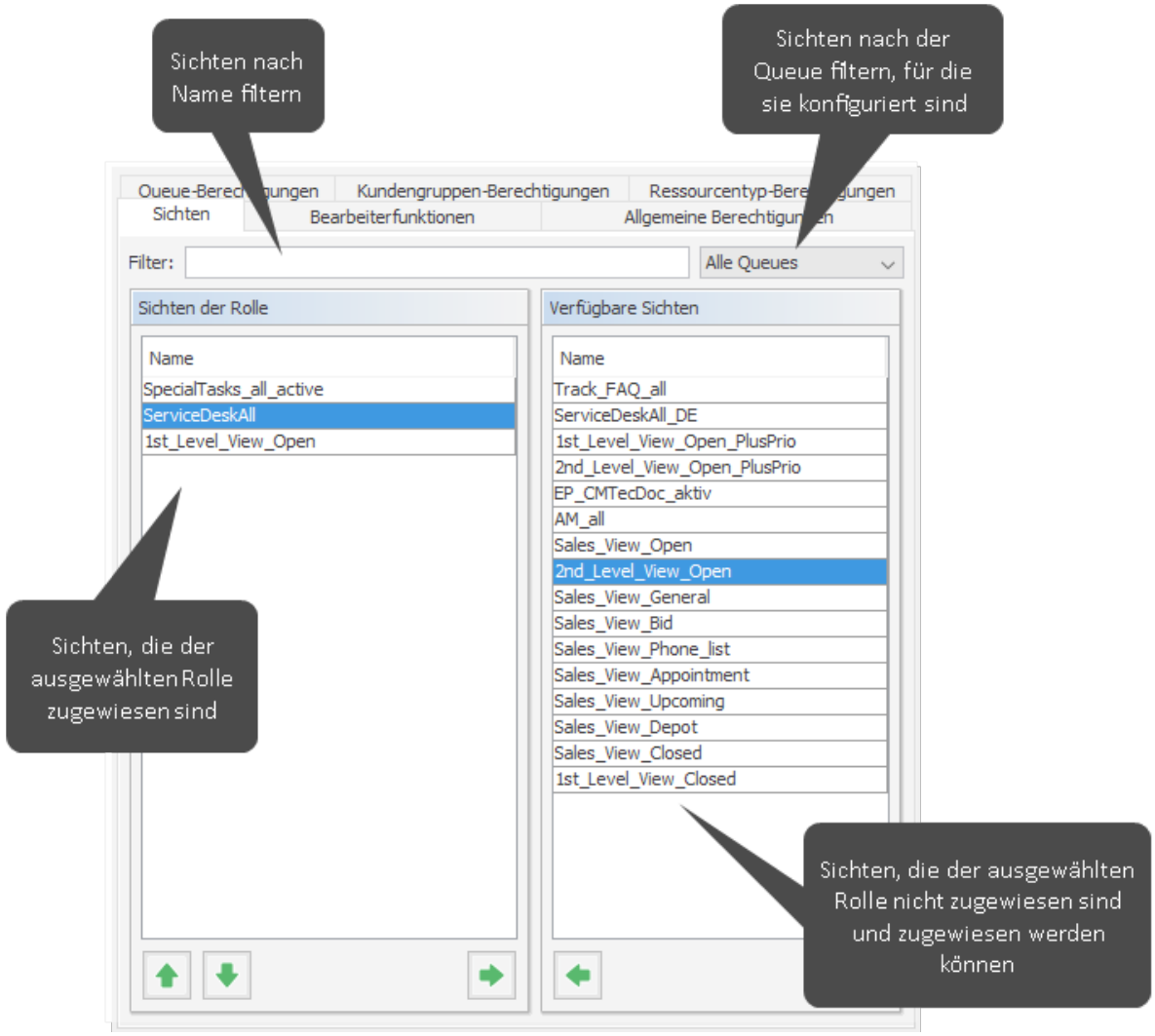


Abbildung 35: *ConSol CM Admin Tool - Zugriff und Rollen, Rollen: Sichten*

Wählen Sie zuerst auf der linken Seite eine Rolle und dann die gewünschten Sichten aus der Liste der *Verfügbaren Sichten*. Klicken Sie auf den Button *Zuweisen*, um die ausgewählten Sichten in die Liste *Sichten der Rolle* zu verschieben. Wenn Sie Sichten aus der Liste entfernen möchten, wählen Sie die jeweiligen Sichten und klicken Sie auf *Entfernen*.

Für normale Rollen können Sie die Reihenfolge der Sichten hier nicht definieren. Im Drop-down-Menü des Web Clients werden die Sichten immer in der Reihenfolge angezeigt, die sie in der Liste der Sichten in der Sichtenverwaltung haben. Siehe auch Abschnitt Sichtenverwaltung. Wenn eine Rolle für mindestens einen Bearbeiter als Hauptrolle markiert ist (und damit einen roten Punkt hat), können die Sichten mit den Pfeiltasten *Nach oben* und *Nach unten* sortiert werden. Die Sortierung ändert die Reihenfolge, in der die Sichten im Web Client angezeigt werden.

Tab *Bearbeiterfunktionen*

In diesem Tab können Sie einer Rolle **Bearbeiterfunktionen** zuweisen. Bearbeiterfunktionen werden verwendet, wenn Sie für ein Ticket einen zusätzlichen Bearbeiter brauchen, z. B. einen Teamleiter, der eine Entscheidung treffen muss, bevor das Ticket im Workflow vorwärts bewegt werden kann. Deswegen müssen Sie dem Teamleiter eine Rolle mit der entsprechenden Bearbeiterfunktion zuweisen. Im Web Client werden die Bearbeiterfunktionen und die damit verknüpften Bearbeiter angezeigt, wenn ein zusätzlicher Bearbeiter gesetzt wird.

Sie gelangen auf diese Seite, indem Sie im Navigationselement *Rollen*, Navigationsgruppe *Zugriff und Rollen*, auf den Tab *Bearbeiterfunktionen* klicken.

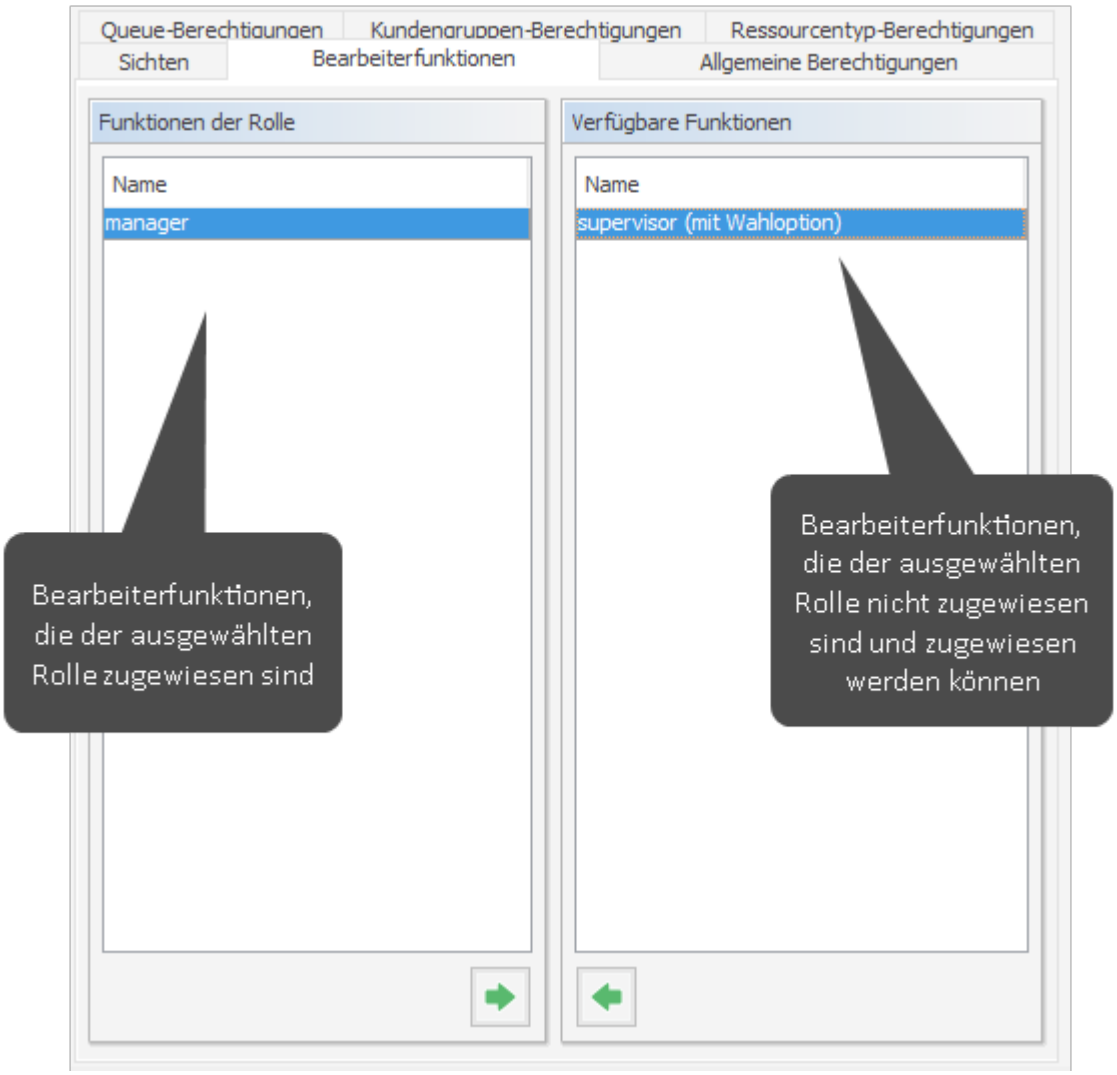


Abbildung 36: *ConSol CM Admin Tool - Zugriff und Rollen, Rollen: Bearbeiterfunktionen*

Wählen Sie auf der linken Seite eine Rolle aus und wählen Sie dann die gewünschten Bearbeiterfunktionen aus der Liste der *Verfügbaren Funktionen*. Klicken Sie auf den Button *Zuweisen*, um

die ausgewählten Funktionen in die Liste *Funktionen der Rolle* zu verschieben. Wenn Sie Funktionen aus der Liste löschen möchten, wählen Sie die entsprechenden Funktionen aus und klicken Sie auf den Button *Entfernen*.

Nachdem Sie eine neue Rolle mit ihren Berechtigungen, Sichten und Bearbeiterfunktionen definiert haben, können Sie diese Rolle den gewünschten Bearbeiterkonten zuweisen. Die Bearbeiter erhalten die Berechtigungen der Rolle sofort nach der Zuweisung (ohne ein zusätzliches Update des Systems).

B.2.2.2 Löschen einer Rolle

Wählen Sie die Rolle aus, die Sie löschen möchten, und klicken Sie unter der Liste der Rollen auf den Button *Löschen*. Wenn Sie im daraufhin angezeigten Bestätigungsfenster *Ja* klicken, wird die Rolle aus der Liste und aus dem System gelöscht.

> ⚠ Denken Sie beim Löschen einer Rolle daran, dass Bearbeiter, die nur diese Rolle haben, sofort alle Berechtigungen im System verlieren.
>
> Wenn Tickets, die z. B. zu einer bestimmten Queue gehören, nicht in anderen Rollenberechtigungen erfasst werden, kann bei Bearbeitern und Administratoren der Eindruck entstehen, dass Tickets fehlen.

B.2.2.3 Kopieren einer Rolle

Wenn Sie eine neue Rolle erstellen möchten und dazu eine vorhandene Rolle als Vorlage verwenden möchten, können Sie diese kopieren. Wählen Sie die vorhandene Rolle aus und klicken Sie unter der Rollenliste auf den Button *Kopieren*. Es wird ein Pop-up-Fenster geöffnet, in dem Sie der Kopie einen Namen geben können. Danach können Sie die Kopie Ihren Wünschen entsprechend ändern.

B.2.2.4 Editieren einer Rolle

Wählen Sie die Rolle, die Sie editieren möchten, in der Liste aus und ändern Sie die Berechtigungen in den jeweiligen Tabs nach Ihren Wünschen. Die Änderungen werden für die Bearbeiter mit dieser Rolle sofort wirksam, nachdem sie sich neu angemeldet haben.

B.2.3 Definieren eines Administrators nur für die Verwaltung der Rollen und Bearbeiter

Manchmal kann es notwendig sein, einen Administrator zu definieren, der keinen vollständigen Systemzugriff hat, sondern nur Bearbeiter und Rollen verwalten darf. Diese Rolle kann zum Beispiel für einen Teamleiter verwendet werden, der neue CM-Bearbeiter in seinem Team erstellen darf, oder für einen Key-User in einem Team, der für eine bestimmte Abteilung den CM-Bearbeitern Berechtigungen erteilen bzw. entziehen darf.

Um diesen Administrator "light" zu definieren, erstellen Sie eine neue Rolle mit der allgemeinen Berechtigung *Zugriff und Rollen administrieren*. Erstellen Sie einen neuen Bearbeiter (in diesem Beispiel *admin_light*) und weisen Sie ihm diese Rolle zu.

Sie gelangen auf diese Seite, indem Sie im Navigationselement *Rollen*, Navigationsgruppe *Zugriff und Rollen*, auf den Tab *Allgemeine Berechtigungen* klicken.

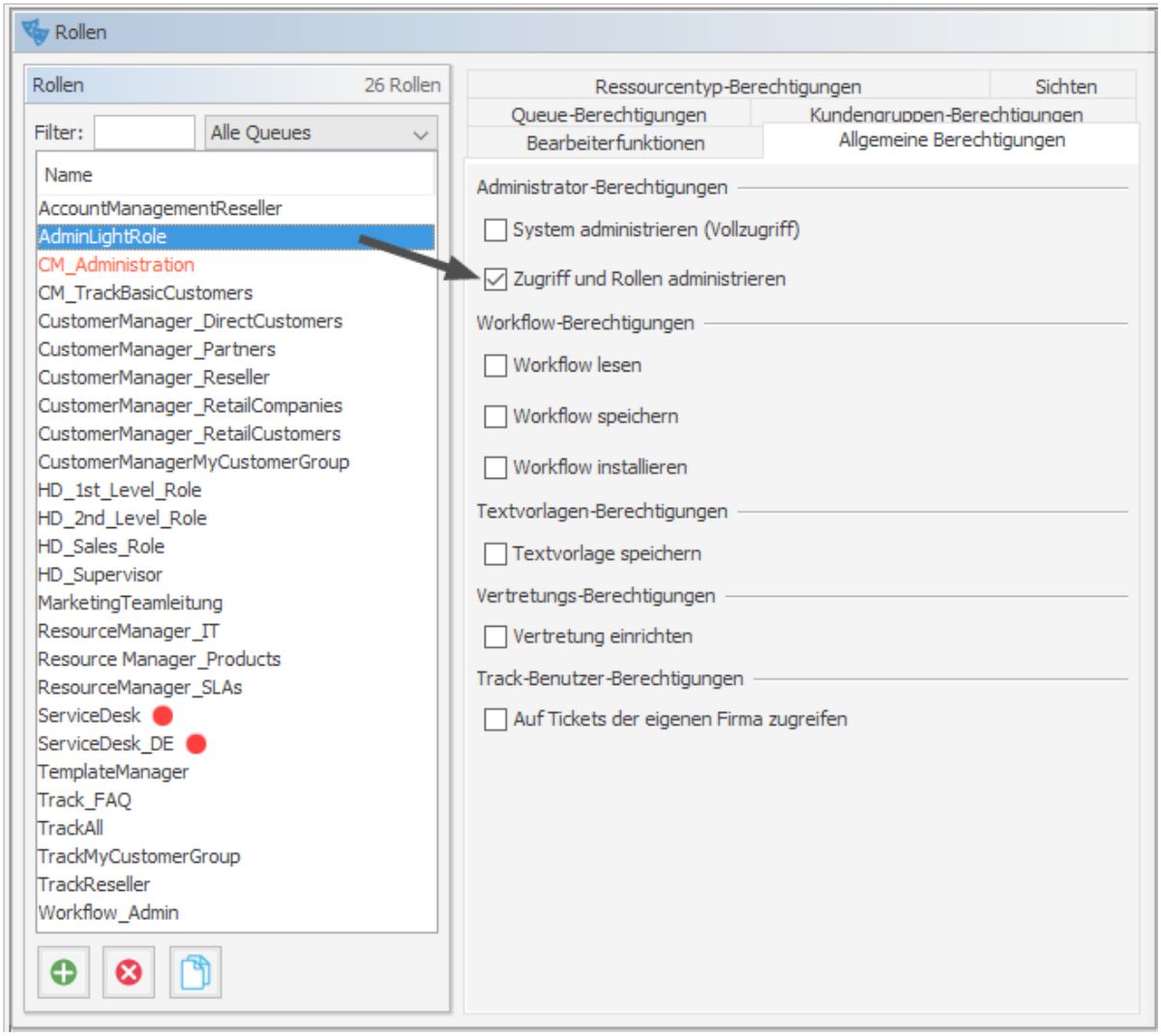


Abbildung 37: *ConSol CM Admin Tool - Zugriff und Rollen, Rollen: Definition einer Administratorrolle für Zugriff und Rollen*

Dem Administrator "light" steht das Navigationselement *Zugriff und Rollen* zur Verfügung. Der Tab *Allgemeine Berechtigungen* ist allerdings nicht verfügbar (d. h. der *admin_light* kann seine eigenen Berechtigungen nicht erweitern).

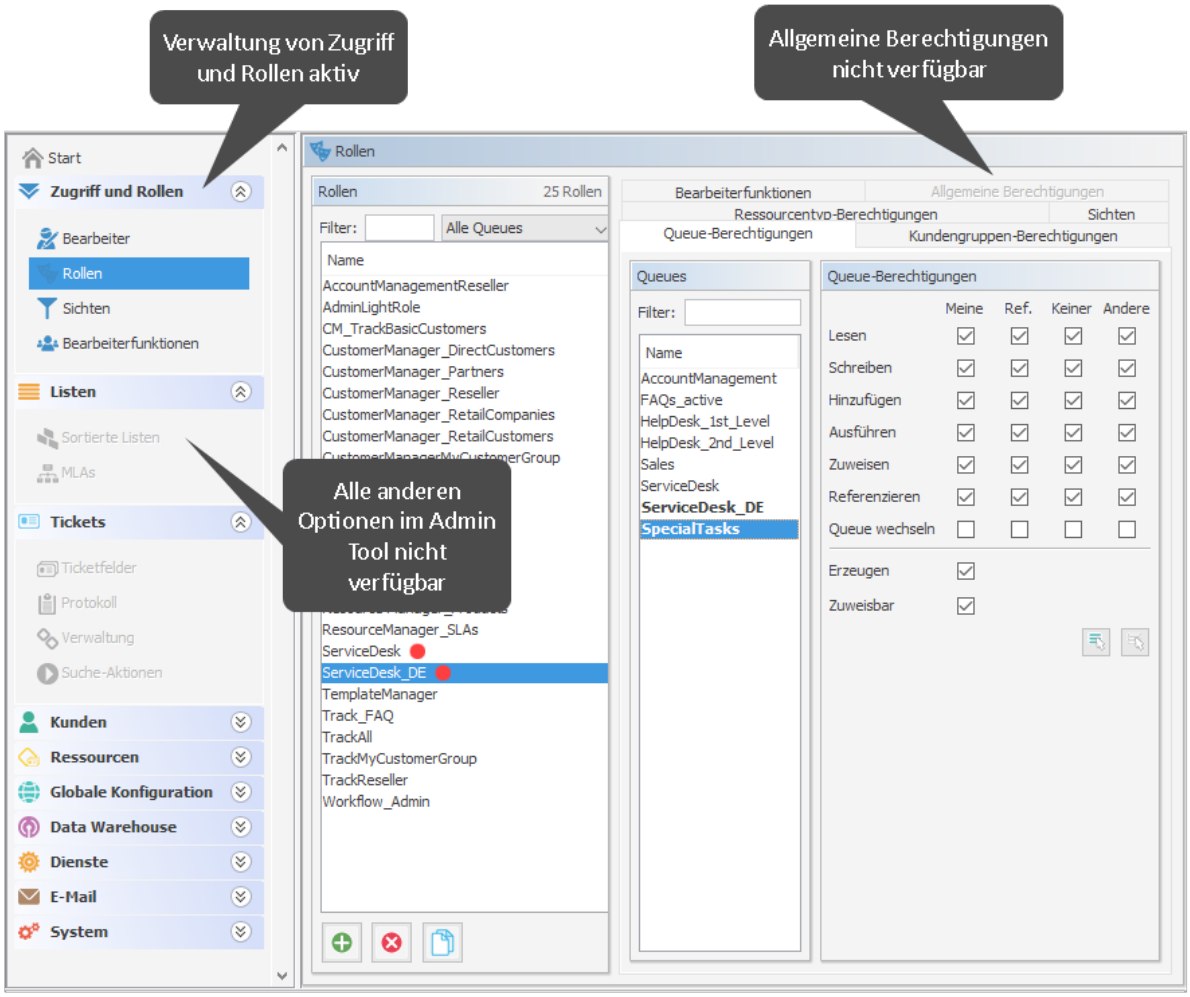


Abbildung 38: *ConSol CM Admin Tool - Ansicht für einen Administrator "light"*

B.3 Sichtenverwaltung

In diesem Kapitel werden folgende Themen behandelt:

B.3.1 Einführung in die Sichtenverwaltung 89

B.3.2 Sichtenverwaltung mit dem Admin Tool 90

B.3.1 Einführung in die Sichtenverwaltung

Sichten dienen dazu, Tickets nach bestimmten Kriterien zu filtern (z. B. alle aktiven Tickets in der Queue *Helpdesk*) und die resultierenden Tickets in der Ticketliste im Web Client anzuzeigen. Da Sichten mit Rollen verknüpft sind, erhalten die Bearbeiter ihre Sichten über die Rollen, die ihnen zugewiesen sind. Die Bearbeiter können im Web Client zwischen ihren Sichten wechseln.

Bearbeiter brauchen die entsprechenden Berechtigungen, um alle Tickets aus einer Sicht sehen zu können. Diese Berechtigungen werden nicht automatisch mit der Zuweisung einer Sicht erteilt, sondern müssen (als Queue- und Kundengruppen-Berechtigungen) in den zugewiesenen Rollen definiert werden. Dieselbe Sicht kann daher für Bearbeiter mit unterschiedlichen Rollen auch unterschiedliche Untermengen von Tickets und Informationen einschließen.

Die Erstellung von Sichten ist optional. Wir empfehlen allerdings, Sichten zu definieren, um zentrale Funktionen des Web Clients nutzen zu können. Ohne eine Sicht, sehen die Bearbeiter in ihrer Ticketliste nämlich keine Tickets. Sie können Tickets dann nur über die Suchfunktion aufrufen.

B.3.2 Sichtenverwaltung mit dem Admin Tool

Sie können Sichten erstellen, editieren oder löschen, indem Sie die Navigationsgruppe *Zugriff und Rollen* im Admin Tool öffnen auf das Navigationselement *Sichten* klicken.

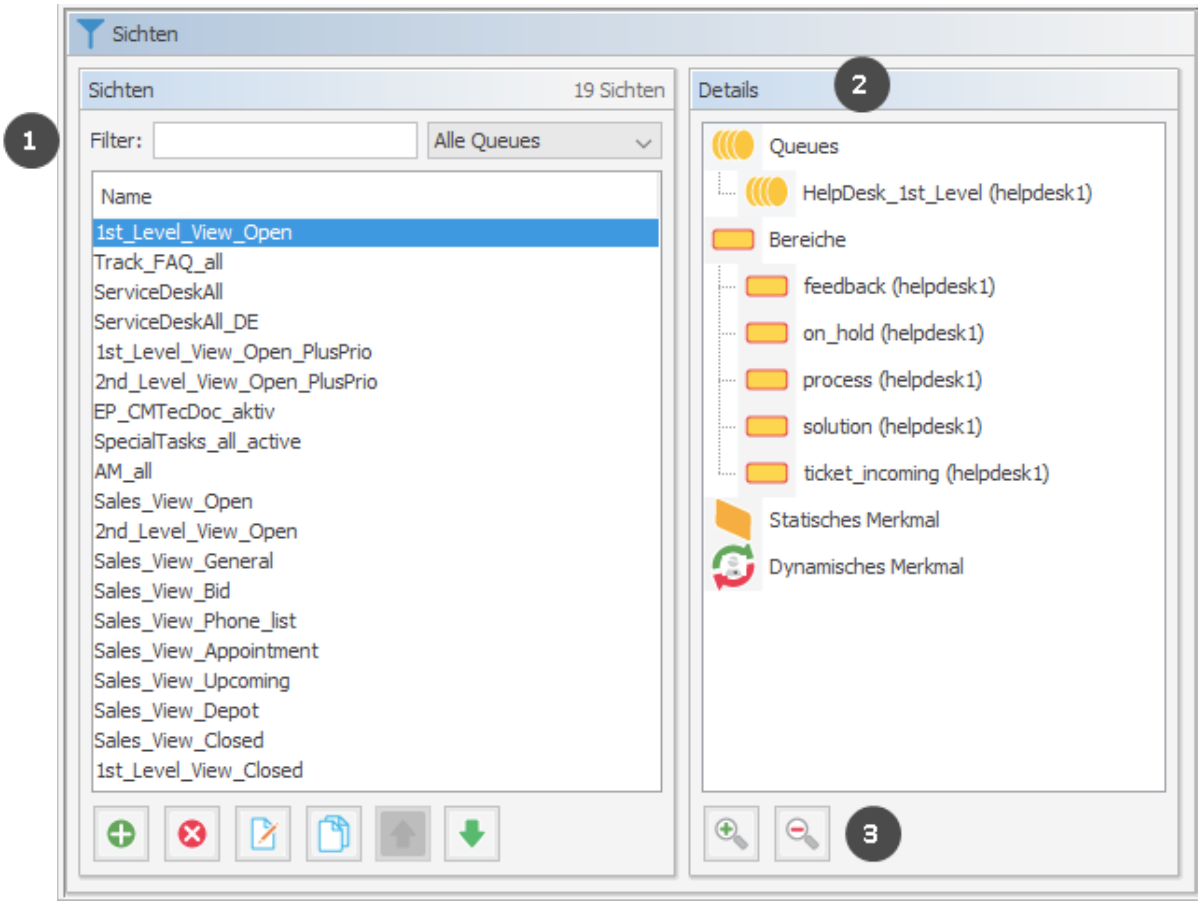


Abbildung 39: *ConSol CM Admin Tool - Zugriff und Rollen, Sichten: Sichtenverwaltung*

Auf der linken Seite befindet sich eine Liste aller verfügbarer Sichten. Sie können die Liste filtern, indem Sie den Namen einer Sicht eingeben oder eine Queue auswählen (1). Die Komponenten der ausgewählten Sicht werden auf der rechten Seite angezeigt (2). Sie können die Unterbäume mithilfe der Buttons unter den Komponenten (3) ein- bzw. ausklappen.

B.3.2.1 Erstellen einer Sicht

Klicken Sie unter der Sichtenliste auf den Button *Hinzufügen* und geben Sie der neuen Sicht im Pop-up-Fenster *Sichtenassistent* einen Namen. Sie können dort zudem eine Beschreibung eingeben.

Klicken Sie auf den Button *Lokalisieren*, um den Namen und die Beschreibung der Sicht zu lokalisieren. Im Pop-up-Fenster *Lokalisierung* werden auf der linken Seite die verfügbaren Sprachumgebungen angezeigt. Geben Sie im Feld *Wert* jeder zusätzlichen Sprache auf der rechten Seiten den entsprechenden Namen bzw. die Beschreibung für die Sicht ein. Nachdem Sie auf *Speichern* geklickt haben, wird der Name bzw. die Beschreibung in der Sprache angezeigt, die der Bearbeiter im Browser eingestellt hat.

Klicken Sie auf *Weiter >*, um mit der Definition der Sichtenmerkmale fortzufahren:

- Queue-Filter
- Bereichsfilter
- Statisches Merkmal
- Dynamisches Merkmal

Queue-Filter

Als Erstes wählen Sie die Queues für die neue Sicht aus. Wählen Sie die gewünschten Queues in der Liste *Nicht zugewiesen* und verschieben Sie sie in die Liste *Zugewiesen*, indem Sie auf den Button *Zuweisen* klicken. Wenn Sie eine zugewiesene Queue entfernen möchten, wählen Sie sie aus und klicken Sie auf den Button *Entfernen* Klicken Sie auf den Button *Weiter >*, um Bereichsfilter zu definieren.

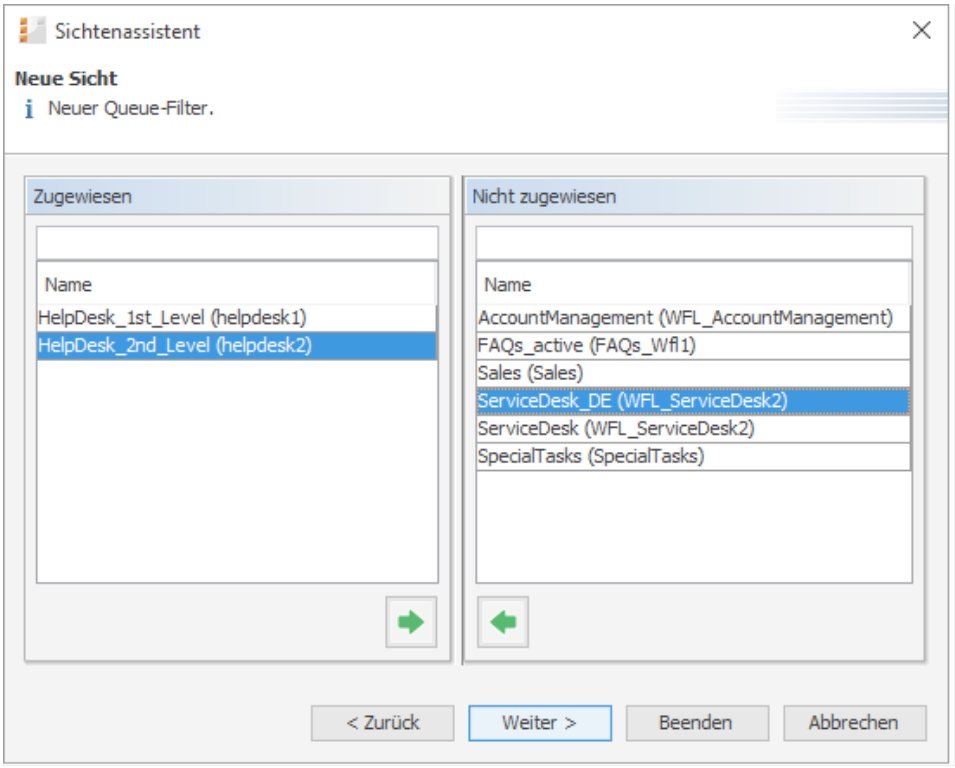


Abbildung 40: *ConSol CM Admin Tool - Zugriff und Rollen, Sichten: Auswahl der Queues*

Bereichsfilter

Als nächstes können Sie die Sicht auf bestimmte Workflow-Bereiche der ausgewählten Queues beschränken. Ein Bereich (*Scope*) umfasst eine Reihe von Workflow-Aktivitäten, die ein gemeinsames Thema haben, z. B. Tickets mit einem Termin.

Wählen Sie die gewünschten Bereiche in der Liste *Nicht zugewiesen* und verschieben Sie sie in die Liste *Zugewiesen*, indem Sie auf den Button *Zuweisen* klicken. Wenn Sie einen zugewiesenen Bereich ent-

fernen möchten, wählen Sie ihn aus und klicken Sie auf den Button *Entfernen* Klicken Sie auf den Button *Weiter >*, wenn Sie weitere Merkmale definieren möchten. Ansonsten können Sie auch auf *Beenden* klicken, um die Sicht zu erstellen.

Wenn Sie der Sicht im Sichtenassistenten keine Bereiche zuweisen, wird zwar eine Sicht mit diesem Namen erstellt, die Sicht enthält im Web Client aber keine Tickets.

Beim Einrichten der Sicht können Sie nur Bereiche auswählen, die zuvor bei der Workflow-Entwicklung definiert wurden. Stellen Sie sicher, dass die Workflows alle benötigten Bereiche enthalten. Wenn Sie zum Beispiel eine Sicht mit **aktiven** und **inaktiven** Tickets haben möchten, brauchen Sie zwei unterschiedliche Bereiche für diese Tickets im Workflow. Ansonsten ist es nicht möglich, eine Sicht mit **aktiven** und eine andere Sicht mit **inaktiven/wartenden** Tickets zu definieren!

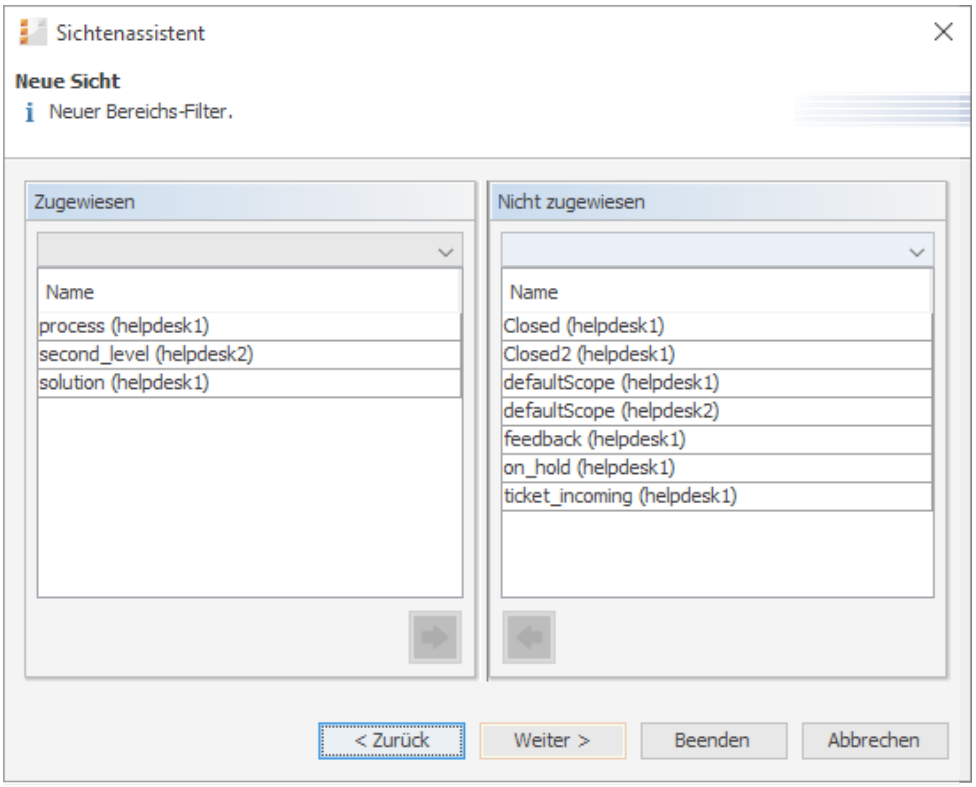


Abbildung 41: *ConSol CM Admin Tool - Zugriff und Rollen, Sichten: Auswählen der Bereiche*

Statisches Merkmal

Sie können die Sicht durch ein statisches Merkmal einschränken, sodass nur Tickets mit einem bestimmten Wert in einem definierten Datenfeld angezeigt werden, z. B. Tickets zu einem bestimmten Produkt oder Tickets mit einer hohen Priorität. Das Merkmal ist statisch, weil es vom Bearbeiter nicht

im Web Client geändert werden kann. Eine detaillierte Beschreibung der Arbeit mit Sichten finden Sie im *ConSol CM Benutzerhandbuch*. Nur Datenfelder des Typs ENUM können als statisches Merkmal verwendet werden.

Wählen Sie ein Datenfeld in der Liste *Feld* (z. B. *Produkt*) und wählen Sie den gewünschten Wert in der darunter liegenden Liste *Wert* (z. B. *crm*). Klicken Sie auf den Button *Weiter >*, wenn Sie zusätzlich ein dynamisches Merkmal definieren möchten. Ansonsten können Sie auch auf *Beenden* klicken, um die Sicht zu erstellen.

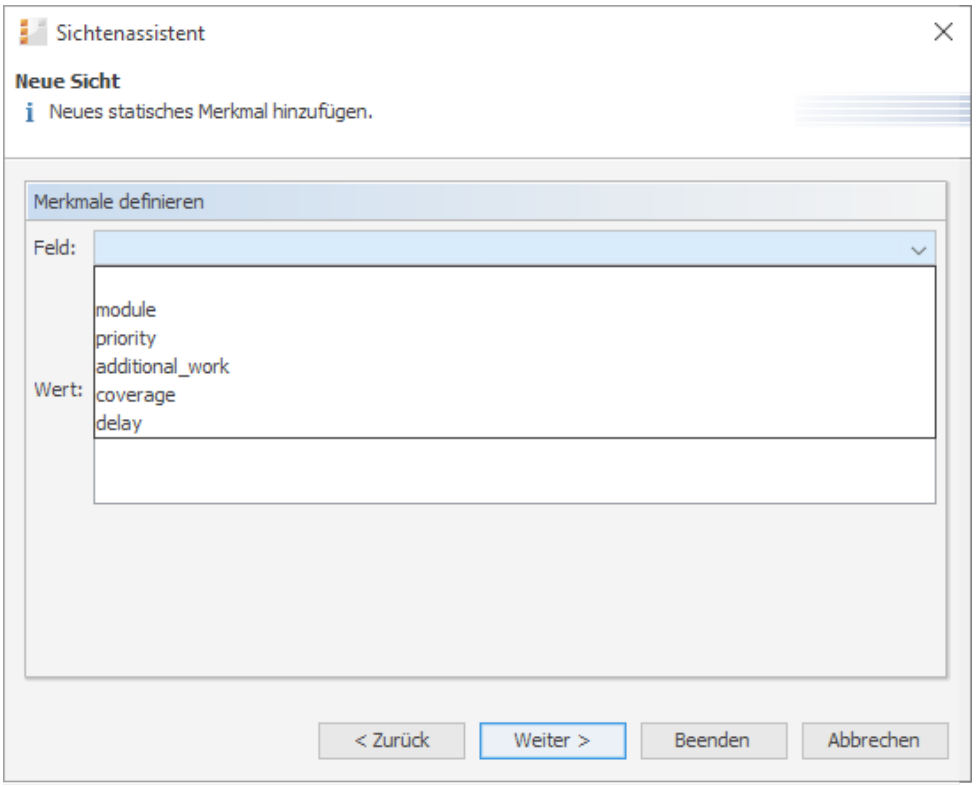


Abbildung 42: *ConSol CM Admin Tool - Zugriff und Rollen, Sichten: Auswählen des statischen Merkmals*

Dynamisches Merkmal

Wie ein statisches Merkmal wird auch ein dynamisches Merkmal dazu benutzt, nur Tickets mit bestimmten Werten in definierten Datenfeldern anzuzeigen. Im Gegensatz zu einem statischen Merkmal können die Bearbeiter die Werte für den Parameter bei einem dynamischen Merkmal allerdings selber auswählen. Das geschieht im Web Client auf der Seite *Bearbeiterprofil*. Zudem kann der Administrator den Wert im Tab *Sichtenmerkmale* der Bearbeiterverwaltung (siehe Abschnitt Bearbeiterverwaltung) für jeden Bearbeiter einzeln anpassen. Eine detaillierte Beschreibung der Arbeit mit Sichten finden Sie im *ConSol CM Benutzerhandbuch*. Nur Datenfelder des Typs ENUM können als dynamisches Merkmal verwendet werden.

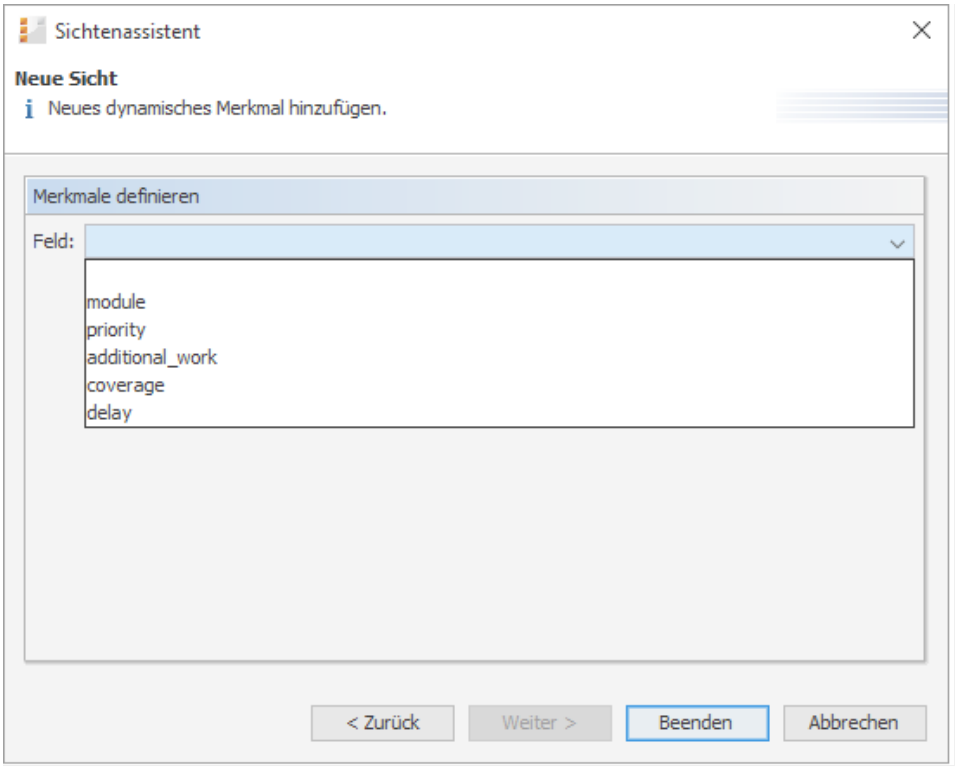


Abbildung 43: *ConSol CM Admin Tool - Zugriff und Rollen, Sichten: Auswählen des dynamischen Merkmals*

Klicken Sie auf *Beenden*, um die Sicht zu erstellen. Sie können das Fenster jederzeit verlassen, ohne zu speichern, indem Sie auf *Abbrechen* klicken. Klicken Sie auf *< Zurück*, um zum vorherigen Schritt der Sichtdefinition zurückzukehren

Jetzt können Sie die neue Sicht in der Sichtenliste auf der linken Seite sehen. Die zugewiesenen Merkmale werden im Bereich *Details* auf der rechten Seite angezeigt.

⚠ Beachten Sie, dass in einer Sicht mit dynamischen Merkmalen nur die Tickets angezeigt werden, die mit diesem Merkmal übereinstimmen. Wenn ein Bearbeiter in seinem Bearbeiterprofil keine Merkmale ausgewählt hat, oder wenn der Administrator die Auswahl im Admin Tool entfernt hat (*Sichtenmerkmale* in der Bearbeiterverwaltung), ist die Sicht für den Bearbeiter leer! Stellen Sie sicher, dass den Benutzern diese Tatsache bewusst ist, und denken Sie auch als Administrator immer daran.

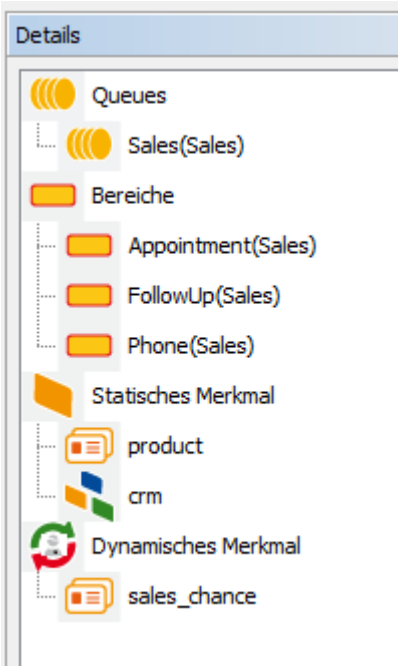


Abbildung 44: *ConSol CM Admin Tool - Zugriff und Rollen, Sichten: Anzeigen der Sichtendetails*

Sie können die Details aus- oder einklappen, indem Sie unter der Liste auf das Plus- bzw. das Minus-Symbol klicken.

⚠ Wir empfehlen Ihnen dringend, keine Sichten zu definieren, die geschlossene Tickets enthalten!

Die Anzahl der geschlossenen Tickets wird bei der Arbeit mit der Software stark zunehmen. Deswegen würden Sichten mit geschlossenen Tickets immer die (per System-Property definierte) maximal zulässige Anzahl an Tickets pro Sicht erreichen. Das kann negative Auswirkungen auf die Leistung des Web Clients haben und in den meisten Fällen werden die gesuchten Tickets noch nicht einmal zu den ersten 50 oder 100 Tickets gehören.

Schlussfolgerung: Eine Sicht mit geschlossenen Tickets hilft den Bearbeitern nicht weiter und kann die Systemgeschwindigkeit senken. Eine Sicht mit geschlossenen Tickets kann daher höchstens für Testumgebungen sinnvoll sein.

B.3.2.2 Editieren einer Sicht

Wählen Sie die Sicht, die Sie editieren möchten, in der Sichtenliste aus. Die Details der Sicht werden auf der rechten Seite angezeigt. Klicken Sie zum Editieren der ausgewählten Sicht einfach mit der rechten Maustaste auf ein Filtermerkmal.

Das folgende Drop-down-Menü wird geöffnet:

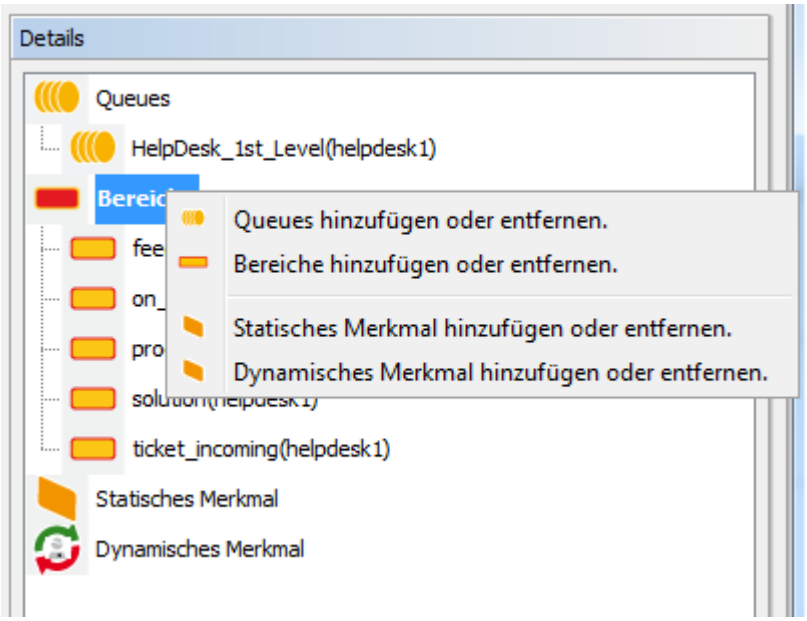


Abbildung 45: *ConSol CM Admin Tool - Zugriff und Rollen, Sichten: Editieren einer Sicht*

Das Menü enthält die folgenden Optionen:

- Queues hinzufügen oder entfernen
- Bereiche hinzufügen oder entfernen
- Statisches Merkmal hinzufügen oder entfernen
- Dynamisches Merkmal hinzufügen oder entfernen

Klicken Sie einfach auf den gewünschten Menüeintrag. Das entsprechende Fenster des *Sichtenassistenten* wird angezeigt. Dort können Sie die Filtermerkmale, wie in Erstellen einer Sicht beschrieben, hinzufügen oder löschen. Der Sichtenassistent kann auch durch einen Doppelklick auf ein Filtermerkmal geöffnet werden.

> Sie können die Sichtenmerkmale nicht editieren, indem Sie auf *Bearbeiten* klicken. Hier können Sie nur den Namen und die Beschreibung der Sicht ändern.

B.3.2.3 Löschen einer Sicht

Klicken Sie unter der Sichtenliste auf den Button *Löschen*, um die ausgewählte Sicht zu löschen. Es wird ein Pop-up-Fenster angezeigt, in dem Sie bestätigen müssen, dass Sie die Sicht wirklich löschen möchten. Wenn Sie auf *Ja* klicken, steht die Sicht den Bearbeitern nicht mehr zur Verfügung. Die Berechtigungen der Bearbeiter sind davon nicht betroffen.

B.3.2.4 Kopieren einer Sicht

Mit dem Button *Kopieren* können Sie beim Erstellen einer Sicht Zeit sparen. Die ausgewählte Sicht wird als Ganzes kopiert und Sie können die Kopie danach editieren. Die neue Sicht hat zunächst den gleichen Namen wie die kopierte Sicht. Sie können den Namen ändern, indem Sie einen Doppelklick auf den Namen machen oder auf den Button *Bearbeiten* klicken.

B.4 Bearbeiterfunktionen

In diesem Kapitel werden folgende Themen behandelt:

B.4.1 Einleitung 99
B.4.2 Erstellen und Editieren einer Bearbeiterfunktion 101
B.4.3 Löschen einer Bearbeiterfunktion 102
B.4.4 Deaktivieren und Aktivieren einer Bearbeiterfunktion 102
B.4.5 Bearbeiterberechtigungen für Bearbeiterfunktionen 102

B.4.1 Einleitung

Bearbeiterfunktionen werden verwendet, wenn Sie für ein Ticket einen zusätzlichen Bearbeiter brauchen, z. B. einen Teamleiter, der eine Entscheidung treffen muss, bevor das Ticket im Workflow vorwärts bewegt werden kann.

Sie können Bearbeiterfunktionen im Admin Tool mit dem Navigationselement *Bearbeiterfunktionen* der Navigationsgruppe *Zugriff und Rollen* verwalten.

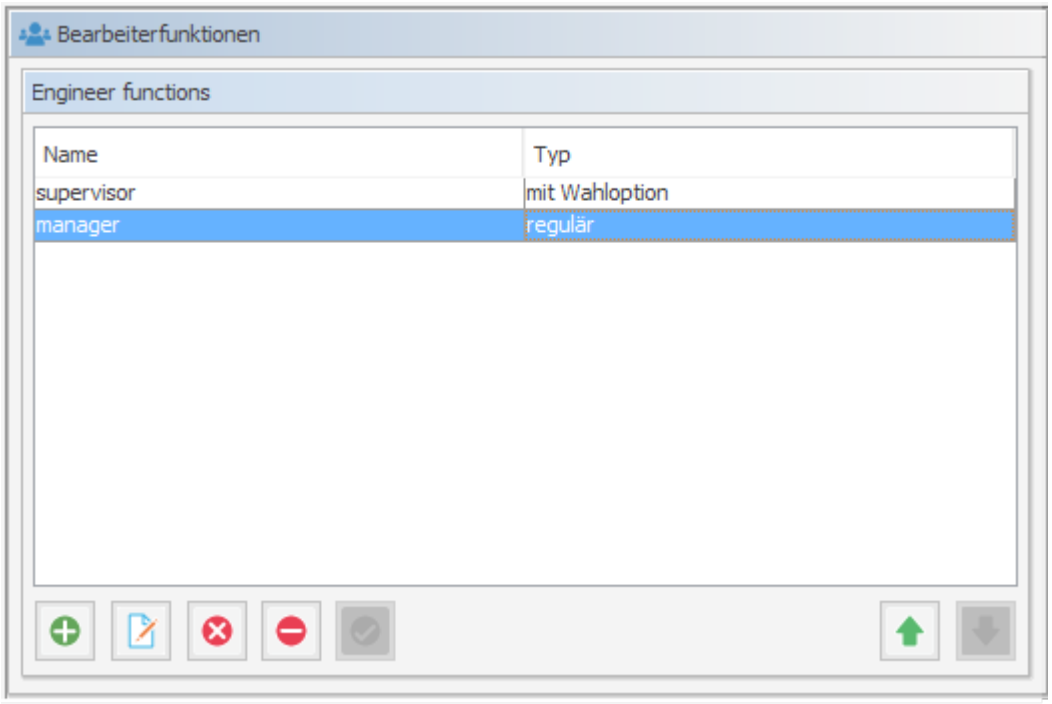


Abbildung 46: *ConSol CM Admin Tool - Zugriff und Rollen, Bearbeiterfunktionen*

Die zugehörigen Aktivitäten für so einen Prozess müssen im Workflow erstellt werden. Bearbeiterfunktionen werden Bearbeiterrollen zugewiesen, die wiederum den entsprechenden Bearbeitern zugewiesen werden. Im Web Client können beim Hinzufügen eines Bearbeiters zu einem Ticket eine Funktion und ein Bearbeiter ausgewählt werden.

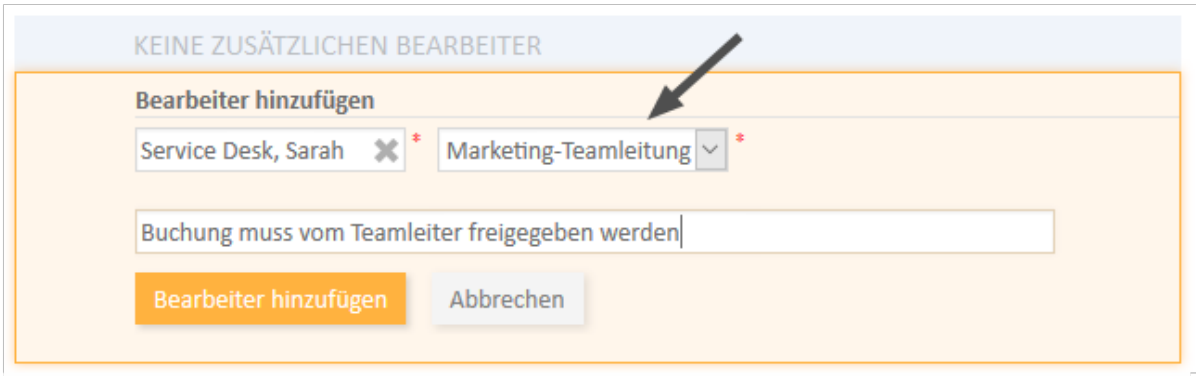


Abbildung 47: *ConSol CM Web Client - Zuweisen eines zusätzlichen Bearbeiters mit einer Bearbeiterfunktion*

B.4.2 Erstellen und Editieren einer Bearbeiterfunktion

Eine Bearbeiterfunktion wird mit ihrem Namen definiert. Wenn Sie auf den Button *Hinzufügen* klicken, öffnet sich ein Pop-up-Fenster, in dem Sie den Namen eingeben können. Dasselbe Fenster erscheint, wenn Sie auf *Bearbeiten* klicken, um eine Bearbeiterfunktion zu editieren. Die Checkbox *Mit Wahloption* muss markiert werden, wenn zusätzliche Bearbeiter die Berechtigung zum Ausführen einer bestimmten Aktivität haben sollen, z. B. eine Genehmigung erteilen, damit das Ticket vorwärts bewegt werden kann. Der Genehmigungsstatus wird dann im Ticket angezeigt.

> ⚠ Nach der Erstellung einer Bearbeiterfunktion kann die Checkbox *Mit Wahloption* nicht mehr deaktiviert werden.

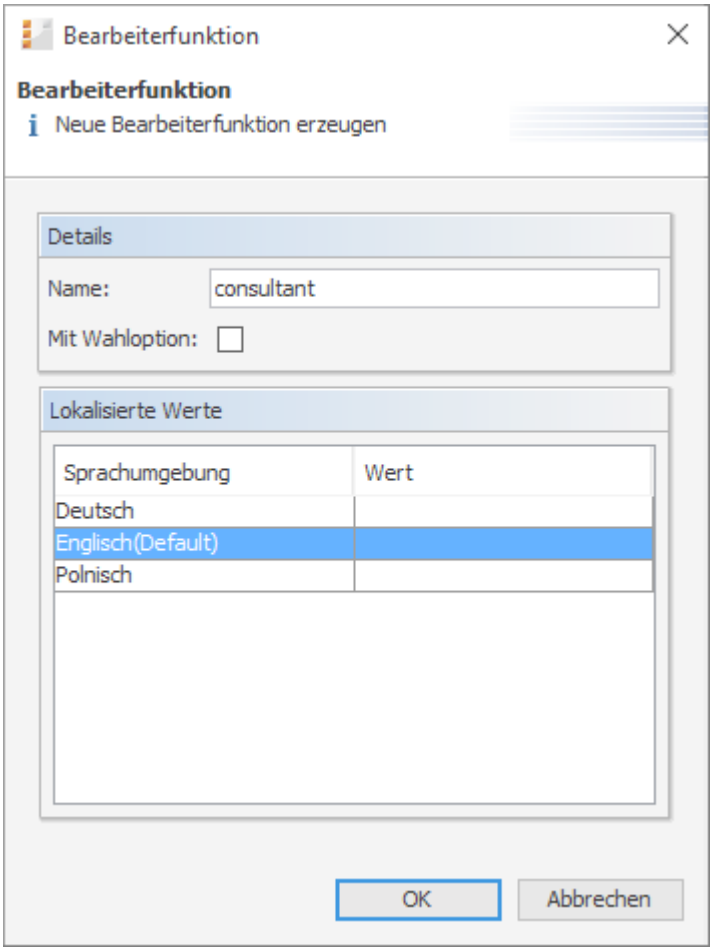


Abbildung 48: *ConSol CM Admin Tool - Zugriff und Rollen, Bearbeiterfunktionen: Erstellen oder Editieren einer Bearbeiterfunktion*

Sie können den Namen der Bearbeiterfunktion in diesem Fenster auch lokalisieren. Die verfügbaren Sprachumgebungen werden auf der linken Seite der Tabelle angezeigt. Geben Sie im Feld *Wert* auf der

rechten Seite in jeder zusätzlichen Sprache den Namen der Bearbeiterfunktion ein. Klicken Sie auf *OK*. Die Bearbeiterfunktion wird jetzt erstellt und der Name wird in der Sprache angezeigt, die der Bearbeiter im Browser eingestellt hat.

B.4.3 Löschen einer Bearbeiterfunktion

Eine Bearbeiterfunktion kann nur gelöscht werden, wenn sie keiner Rolle zugewiesen ist. Andernfalls wird eine Warnung angezeigt, und Sie können diese Bearbeiterfunktion nur deaktivieren (siehe unten).

Um eine Bearbeiterfunktion zu löschen, wählen Sie sie in der Liste aus und klicken Sie auf den Button *Löschen*. Wenn Sie im daraufhin angezeigten Bestätigungsfenster *Ja* klicken, wird die Bearbeiterfunktion aus der Liste und aus dem System gelöscht.

B.4.4 Deaktivieren und Aktivieren einer Bearbeiterfunktion

Wenn eine Bearbeiterfunktion noch einer Rolle zugewiesen ist, aber nicht mehr benötigt wird, können Sie sie deaktivieren. Wählen Sie dazu die Bearbeiterfunktion aus und klicken Sie auf den Button *Deaktivieren*. Der entsprechende Listeneintrag wird danach kursiv angezeigt. Eine Bearbeiterfunktion kann nicht mehr zugewiesen werden. Klicken Sie unten auf der Seite einfach auf *Aktivieren*, um die Funktion wieder zu aktivieren.

B.4.5 Bearbeiterberechtigungen für Bearbeiterfunktionen

Bearbeitern, bzw. deren Rollen, können spezielle Berechtigungen erteilt werden, die sich nur auf die Tickets beziehen, denen der Bearbeiter mit dieser Rolle als Bearbeiter mit einer bestimmten Funktion zugewiesen ist. Diese Berechtigungen werden als Queue-Berechtigungen erteilt (siehe auch Abschnitt Rollenverwaltung). Dieses Prinzip stellt eine gute Grundlage für ein komplexes Prozessmanagement dar, bei dem Personen mit unterschiedlichen Rollen und Verantwortlichkeiten gleichzeitig an einem Vorfall arbeiten.

In der Rollenverwaltung (Navigationsgruppe *Zugriff und Rollen*, Navigationselement *Rollen*) können Sie folgende Berechtigungen erteilen/entziehen (eine detaillierte Beschreibung der Berechtigungen finden Sie im Abschnitt Rollenverwaltung):

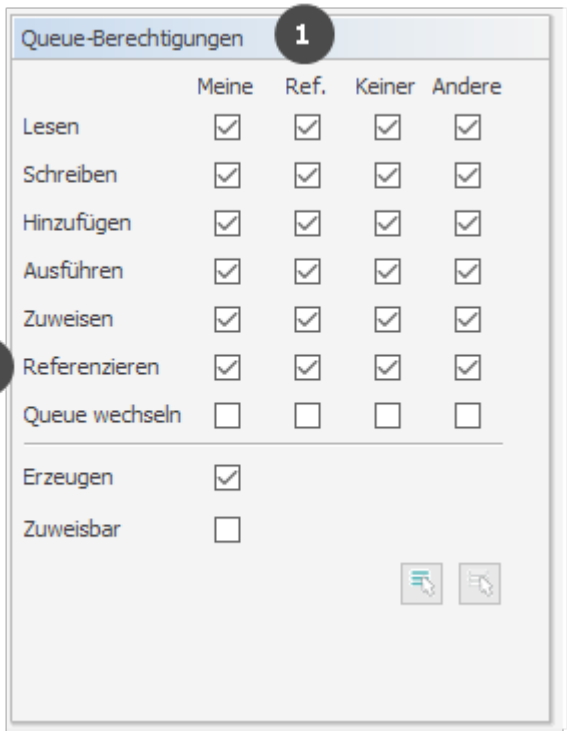
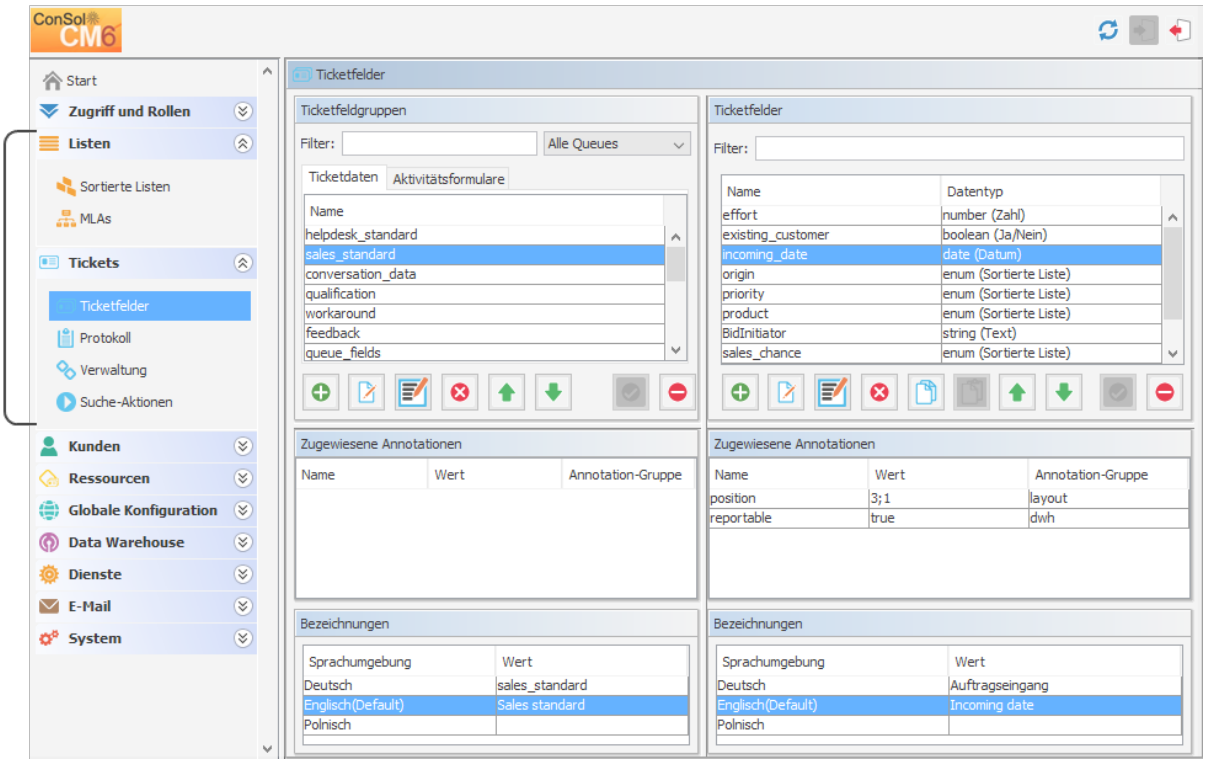


Abbildung 49: *ConSol CM Admin Tool - Zugriff und Rollen, Rollen: Berechtigungen für Bearbeiterfunktionen*

- **Spalte *Ref.* (1)**
  Die Berechtigungen gelten für alle Tickets in dieser Queue, bei denen ein Mitglied dieser Rolle als zusätzlicher Bearbeiter mit einer bestimmten Bearbeiterfunktion gesetzt ist.
- **Zeile *Referenzieren* (2)**
  Für die ausgewählten Tickettypen wird die Berechtigung zum Hinzufügen eines zusätzlichen Bearbeiters erteilt.

C - Abschnitt Ticketdatenmodell und GUI-Design



In diesem Abschnitt lernen Sie, wie das Datenmodell für Ticketdaten definiert wird, d. h. die Definition der Ticketfelder und die Positionierung der Ticketfelder auf der GUI des Web Clients. Einige der Datenstrukturen (Sortierte Listen, MLAs) können allerdings auch für Kundenfelder und Ressourcenfelder verwenden werden, siehe Abschnitte Verwaltung von Verwaltung von Kundenfeldern und GUI-Design für Kundendaten und CM/Resource Pool - Einrichten des Ressourcenmodells. Das Web Client Dashboard, eine in Version 6.9.4 eingeführte neue Funktion, wird ebenso beschrieben wie die Seitenanpassung, ein leistungsfähiger Mechanismus zur Konfiguration des Layouts und der Funktionalitäten des Web Clients.

Außerdem erfahren Sie, wie Sie die Bezeichnungen konfigurieren, die im Web Client angezeigt werden.

In diesem Abschnitt werden folgende Themen behandelt:

- Verwaltung von Ticketfeldern (Einrichten des Ticketdatenmodells)
- Verwaltung von Sortierten Listen
- MLA-Verwaltung
- Ticketprotokoll
- Konfiguration der Ticketliste

- Konfiguration des Web Client Dashboards
- Seitenanpassung

C.1 Verwaltung von Ticketfeldern (Einrichten des Ticket-datenmodells)

In diesem Kapitel werden folgende Themen behandelt:

C.1.1 Einleitung 107

C.1.2 Verwaltung von Ticketfeldern mit dem Admin Tool 109

C.1.3 Tab Ticketdaten 110

C.1.4 Tab Aktivitätsformulare 125

C.1.5 Häufig verwendete Annotationen 131

C.1.1 Einleitung

Ticketfelder werden für Tickets definiert. Standardmäßig enthalten alle Tickets Informationen über den zugewiesenen Bearbeiter, das Erstellungsdatum, die aktuelle Queue und den aktuellen Bereich. Diese Standardfelder (1) werden im oberen Teil des Tickets neben dem Ticket-Icon angezeigt. Zusätzlich zu diesen Feldern kann der CM-Administrator Ticketfelder definieren, die an den mit ConSol CM umgesetzten Anwendungsfall angepasst sind. Beispiele für Ticketfelder sind *Priorität*, *Softwaremodul*, *Reaktionszeit* oder *Verkaufspotential*. Ticketfelder gehören immer zu Ticketfeldgruppen, die dann den Queues zugewiesen werden, damit die Ticketfelder verfügbar sind. Die Ticketfelder können in der Kopfzeile (2) oder dem Bereich für Detaildaten (3) angezeigt werden.

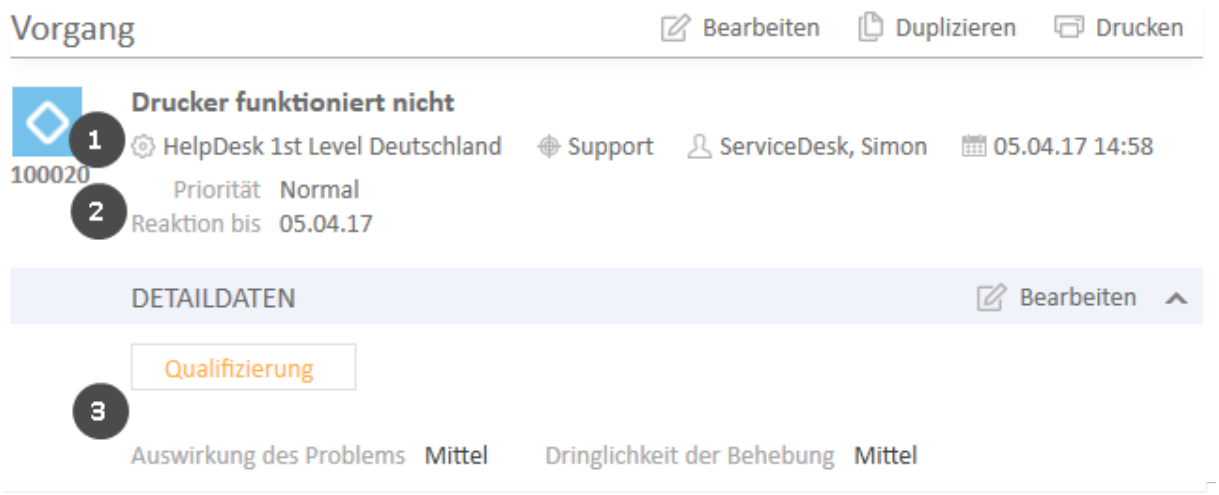


Abbildung 50: *ConSol CM Web Client - Ticketfelder*

Eine **Ticketfeldgruppe**:

- kann der Queue zugewiesen werden. Zum Beispiel kann die Ticketfeldgruppe *helpdesk_fields* der Queue *HelpDesk* zugewiesen werden.
- kann während des Prozesses im Web Client ein- und ausgeblendet werden. Zum Beispiel können Sie die Ticketfeldgruppe *Lösung* erst später im Prozess einblenden, sodass sie angezeigt wird, wenn der Bearbeiter eine Lösung für das vom Kunden gemeldete Problem gefunden hat. Einzelne Ticketfelder können nicht ein- oder ausgeblendet werden.
- kann unter den Standardfeldern oder als Tab im *Bereich für Detaildaten* angezeigt werden. Im letzteren Fall ist der (lokalisierte) Name der Ticketfeldgruppe der Titel und das Mouseover des Tabs. Wenn die Ticketfeldgruppe im Kopfbereich des Tickets angezeigt wird, ist der Gruppenname nicht sichtbar.
- wird mittels Gruppenannotationen konfiguriert. Annotationen werden verwendet, um spezielle Parameter und Eigenschaften einer Ticketfeldgruppe zu definieren, wie zum Beispiel der initiale Anzeigemodus auf der Benutzeroberfläche. Weitere Informationen über die verfügbaren Gruppenannotationen finden Sie in Liste der Gruppenannotationen.
- wird auf der GUI des Web Clients entsprechend der Position in der Liste der Ticketfeldgruppen angeordnet, die die Reihenfolge der Ticketfeldgruppen im Kopfbereich des Tickets bzw. die Reihenfolge der Tabs festlegt.

Ein **Ticketfeld**:

- wird immer innerhalb einer Ticketfeldgruppe definiert.
- wird einer Queue als Teil einer Ticketfeldgruppe zugewiesen.
- kann durch Annotationen unsichtbar gemacht werden, aber als Einzelfeld nicht während des Prozesses ein- oder ausgeblendet werden.
- wird mittels Feldannotationen konfiguriert. Mit Annotationen werden spezielle Parameter und Eigenschaften eines Ticketfeldes definiert, z. B. die Position auf der Benutzeroberfläche. Weitere Informationen über die verfügbaren Feldannotationen finden Sie in Liste der Feldannotationen.
- wird auf der GUI des Web Clients entweder entsprechend dem Wert der Annotation `position` oder, wenn keine `position` angegeben ist, entsprechend der Position in der Liste angeordnet (das erste Ticketfeld in der Liste wird zuerst auf der GUI angezeigt).

Um neue **Ticketfelder** für eine Queue zu definieren, müssen Sie die folgenden Schritte ausführen:

1. Definieren Sie eine Ticketfeldgruppe und setzen Sie die entsprechenden Annotationen.
2. Definieren Sie alle Sortierten Listen, die Sie für die Ticketfelder benötigen werden (siehe Verwaltung von Sortierten Listen). Wenn das Ticketfeld *Priorität* zum Beispiel eine Liste der Prioritäten enthalten soll, müssen Sie diese Liste zuerst definieren.
3. Definieren Sie alle Ticketfelder in der neuen Ticketfeldgruppe.
4. Weisen Sie die neue Ticketfeldgruppe der Queue zu, in der die Ticketfelder benötigt werden.
5. Testen Sie das Ergebnis auf der GUI des Web Clients. Sie müssen sich dazu nicht erneut anmelden, es reicht aus, eine Seite mit einem Ticket in der entsprechenden Queue mit F5 zu aktualisieren.

Diese Schritte werden in den folgenden Abschnitten detailliert erklärt.

Wenn einige Datenfelder in einem oder mehreren Prozessen als Formular angeboten werden sollen, müssen dazu Aktivitätsformulare (ACFs) definiert werden. Dies ist im Abschnitt Tab Aktivitätsformulare erklärt.

C.1.2 Verwaltung von Ticketfeldern mit dem Admin Tool

Im Admin Tool werden Ticketfelder mit dem Navigationselement *Ticketfelder* in der Navigationsgruppe *Tickets* verwaltet.

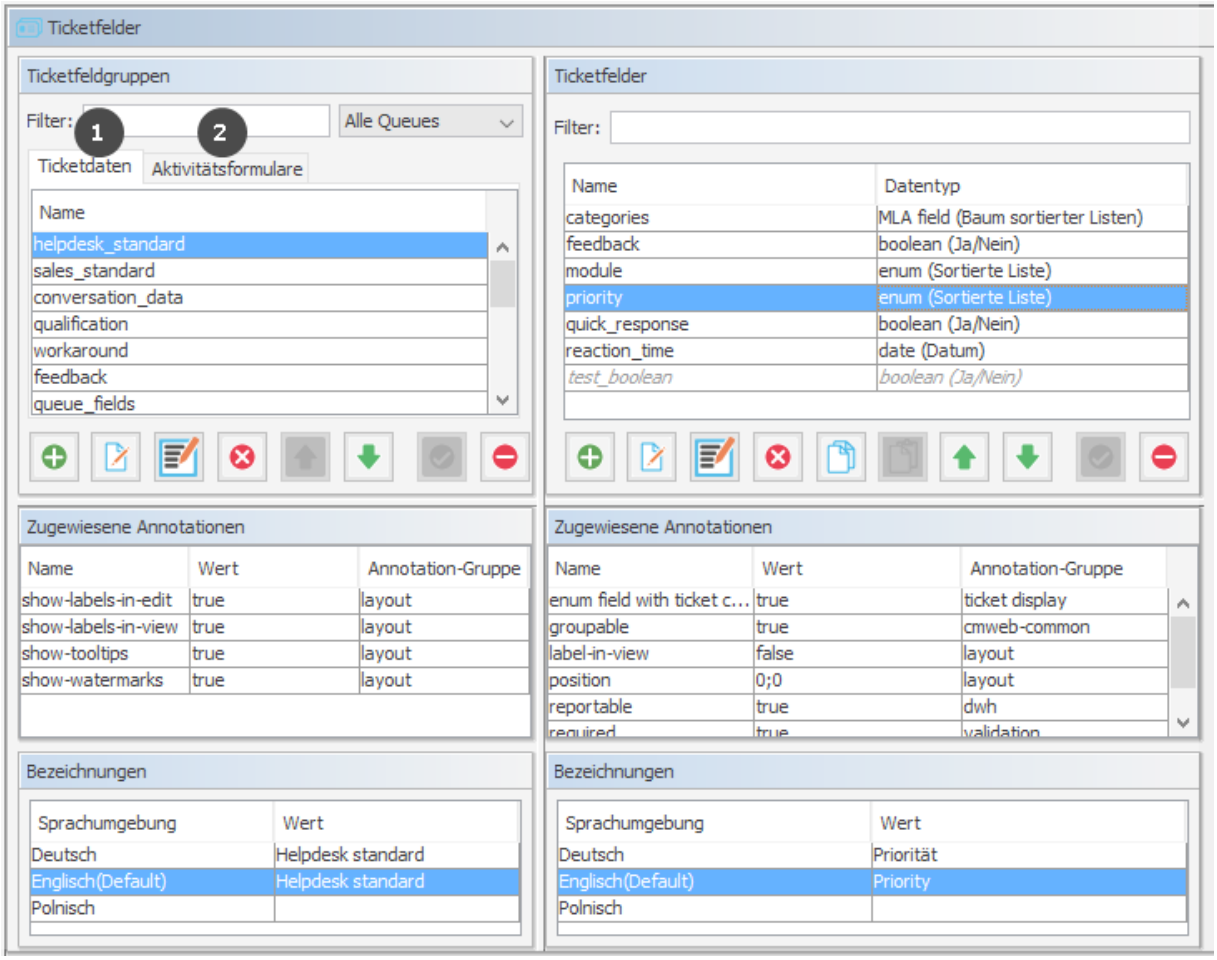


Abbildung 51: *ConSol CM Admin Tool - Tickets, Ticketfelder*

Das Navigationselement *Ticketfelder* hat zwei Tabs:

- Tab Ticketdaten (1): In diesem Tab werden Ticketfeldgruppen und Ticketfelder definiert.
- Tab Aktivitätsformulare (2): In diesem Tab werden Aktivitätsformulare definiert.

Diese beiden Tabs sind in den folgenden Abschnitten detailliert erklärt. Sie können zwischen den Tabs wechseln, indem Sie auf die Kopfzeile des Tabs klicken.

C.1.3 Tab *Ticketdaten*

In diesem Tab können Sie Gruppen und Felder für die Daten des Tickets definieren.

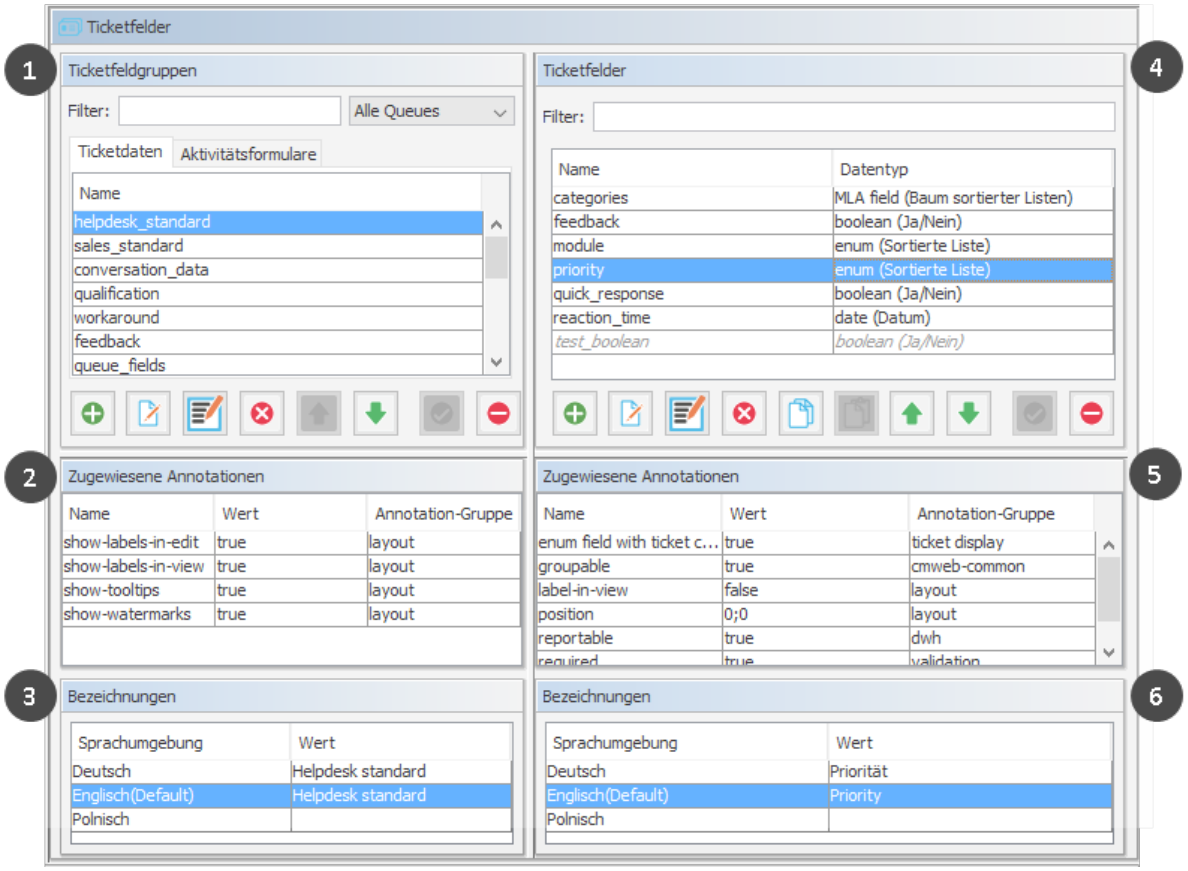


Abbildung 52: *ConSol CM Admin Tool - Tickets, Ticketfelder*

Der Tab *Ticketdaten* hat sechs Bereiche:

- Ticketfeldgruppen (1)
- Annotationen für Ticketfeldgruppen (2)
- Bezeichnungen für Ticketfeldgruppen (3)
- Ticketfelder (4)
- Annotationen für Ticketfelder (5)
- Bezeichnungen für Ticketfelder (6)

C.1.3.1 Erstellen einer Ticketfeldgruppe

Klicken Sie zum Erstellen einer Ticketfeldgruppe einfach unter der Liste auf der linken Seite auf den Button *Hinzufügen*.

Es wird folgendes Pop-up-Fenster geöffnet:

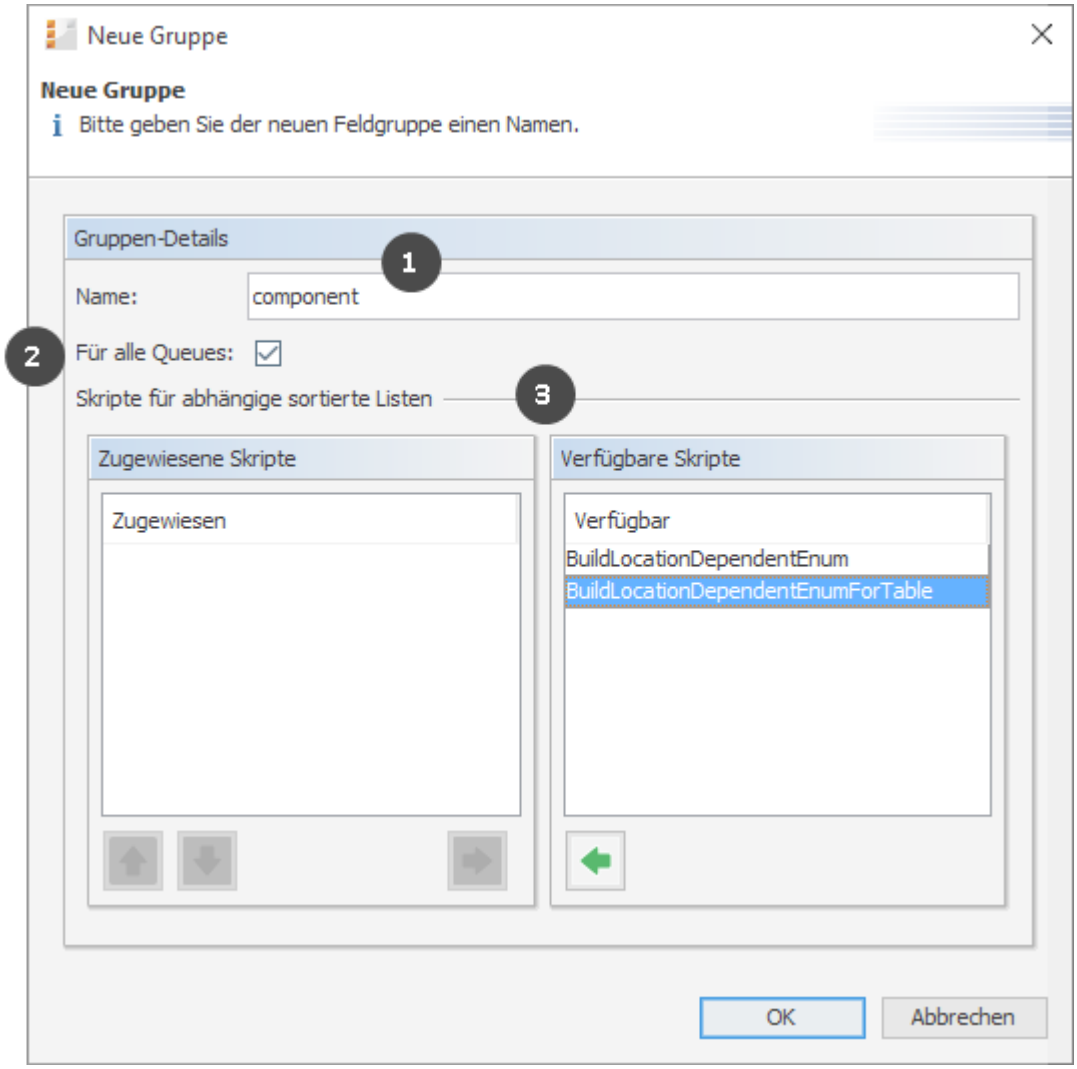


Abbildung 53: *ConSol CM Admin Tool - Tickets, Ticketfelder: Erstellen einer Ticketfeldgruppe*

- **Name (1)**
  Geben Sie einen Namen für die Ticketfeldgruppe ein. Der Name muss eindeutig sein. Die lokalisierten Namen für eine Ticketfeldgruppe werden im Hauptbereich von *Ticketfelder* gesetzt. Details dazu finden Sie im Abschnitt Lokalisierung von Datenfeldern
- **Für alle Queues (2)**
  Wenn diese Checkbox markiert ist, sind die Ticketfelder dieser Gruppe automatisch allen Queues zugewiesen. Eine *für alle Queues* markierte Ticketfeldgruppe wird automatisch auch jeder neuen Queue zugewiesen. Normalerweise sind Ticketfeldgruppen für Ticketdaten nur in bestimmten Queues verfügbar (siehe Queue-Verwaltung).
- **Verfügbare Skripte, d. h. Skripte für abhängige sortierte Listen (3)**
  Skripte für abhängige sortierte Listen definieren die Struktur von abhängigen sortierten Listen (hierarchische Listen mit mehreren Ebenen), die in Ticketfeldern und Feldgruppen verwendet

werden. Mit abhängigen sortierten Listen können Sie die Auswahl in Listen mit mehreren Ebenen einschränken. Auf diese Weise können Sie ein Element aus der Liste auswählen, und es werden in der darunterliegenden Hierarchieebene nur die für diese Auswahl passenden Ergebnisse angezeigt. Die Sortierten Listen (einfache Listen) für Ticketfelder müssen in der Verwaltung von Sortierten Listen erstellt werden, während die Skripte, mit denen diese zu abhängigen sortierten Listen verbunden werden, als Admin-Tool-Skript erstellt werden, siehe auch Abschnitt Admin-Tool-Skripte.
Um Skripte für abhängige sortierte Listen einer Ticketfeldgruppe zuzuweisen, wählen Sie die gewünschten Skripte in der Liste *Verfügbare Skripte* aus und verschieben Sie sie in die Liste *Zugewiesene Skripte*, indem Sie auf den Button *Zuweisen* klicken.

C.1.3.2 Editieren einer Ticketfeldgruppe

Wenn Sie eine Listengruppe editieren möchten, wählen Sie sie in der Liste aus und klicken Sie auf den Button *Bearbeiten*. Es wird dasselbe Fenster geöffnet wie für die Erstellung einer Ticketfeldgruppe. Sie können alle Felder ändern und Ihre Änderungen durch Klicken auf *OK* speichern.

C.1.3.3 Annotieren einer Ticketfeldgruppe

Ticketfeldgruppen werden annotiert, um ihre Eigenschaften zu definieren, z. B. wo eine Gruppe im Web Client angezeigt wird, ob eine Gruppe indexiert ist oder ob sie sichtbar sein soll. Sie können zum Beispiel definieren, ob eine Gruppe im Web Client sichtbar sein soll (Annotation `group-visibility`) oder ob sie im *Bereich für Detaildaten* des Web Clients angezeigt werden soll (Annotation `show-in-group-section`). Wählen Sie zum Zuweisen von Annotationen eine Gruppe aus und klicken Sie auf den Button *Annotieren*.

Es wird folgendes Pop-up-Fenster geöffnet:

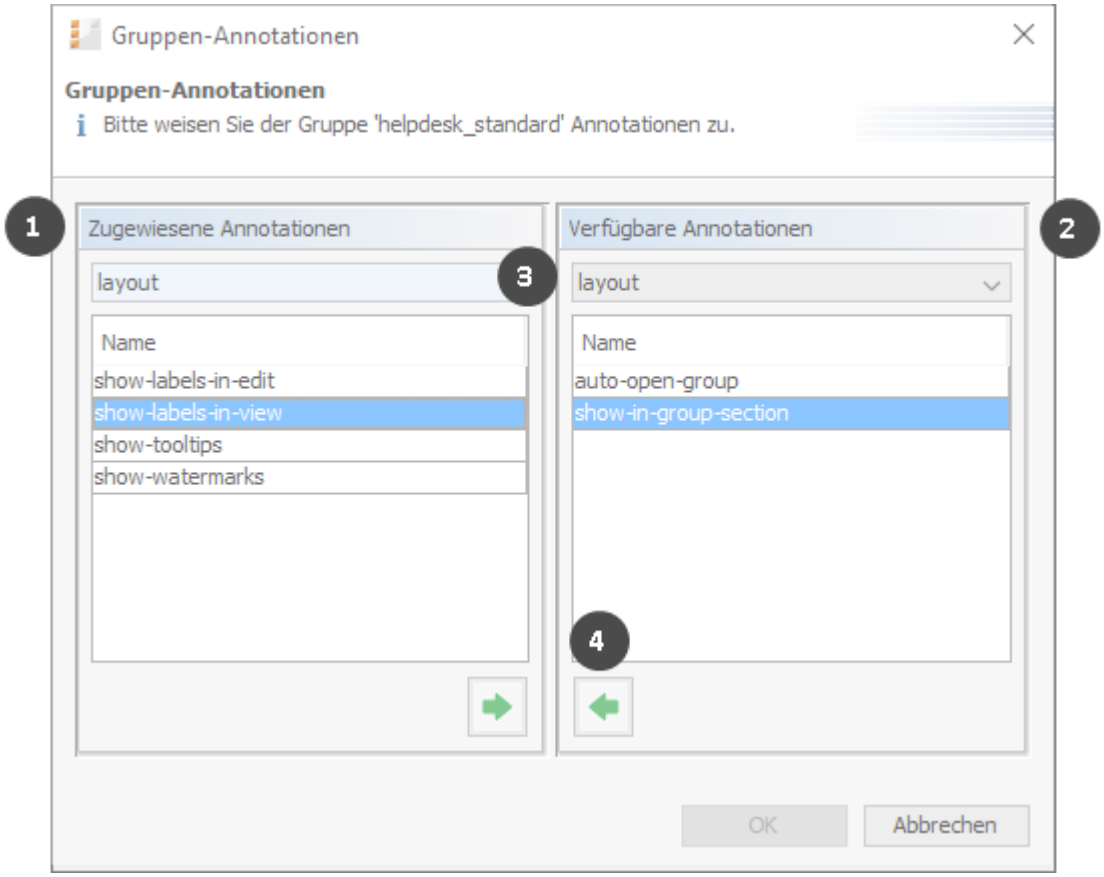


Abbildung 54: *ConSol CM Admin Tool - Tickets, Ticketfelder: Zuweisen von Annotationen zu einer Ticketfeldgruppe*

Im rechten Teil des Fensters werden die verfügbaren Annotationen (2) angezeigt. Im Auswahlfeld über der Liste können Sie die Liste nach Annotationstyp filtern, z. B. *common* oder *layout* (3). Wählen Sie die gewünschten Annotationen aus und verschieben Sie sie in die Liste *Zugewiesene Annotationen* auf der linken Seite (1), indem Sie auf den Button *Zuweisen* (4) klicken. Diese Liste kann ebenfalls nach Annotationstyp gefiltert werden. Klicken Sie auf *OK*, um die Annotationen zur Ticketfeldgruppe zuzuweisen und das Fenster zu schließen. Details dazu finden Sie in Annotationen, Abschnitt *Gruppenannotationen*.

Die Annotationen werden jetzt mit ihrem Standardwert (sofern verfügbar, z. B. „true“ oder „false“) in der unteren linken Ecke der Admin-Tool-Seite angezeigt. Sie können den Wert verändern, indem Sie einen Doppelklick in das entsprechende Feld *Wert* machen und den gewünschten Wert eingeben. Drücken Sie danach die Eingabetaste.

Die Ticketfeldgruppen werden im Web Client in der Reihenfolge angezeigt, in der sie in dieser Liste stehen. Wählen Sie eine Gruppe aus und klicken Sie auf die Symbole *Nach oben* und *Nach unten*, wenn Sie die Position der Gruppe in der Liste ändern möchten.

C.1.3.4 Löschen einer Ticketfeldgruppe

Eine Ticketfeldgruppe kann nur gelöscht werden, wenn sie keiner Queue bzw. keinem Ticket zugewiesen ist. Andernfalls wird eine Warnung angezeigt, dass Sie die Gruppe nur deaktivieren können (siehe unten). Um eine Ticketfeldgruppe zu löschen, wählen Sie sie in der Liste aus und klicken Sie auf den Button *Löschen*. Wenn Sie das daraufhin angezeigte Fenster mit *Ja* bestätigen, wird die Gruppe mit ihren Feldern aus der Liste und aus dem System gelöscht.

C.1.3.5 Aktivieren oder Deaktivieren einer Ticketfeldgruppe

Wenn Sie eine Ticketfeldgruppe nicht löschen können, oder für den Fall, dass sie später noch einmal benötigt wird, nicht löschen möchten, können Sie sie deaktivieren. Wählen Sie dazu die Gruppe aus und klicken Sie auf den Button *Deaktivieren*. Der entsprechende Listeneintrag wird danach kursiv angezeigt. Deaktivierte Ticketfeldgruppen werden im Web Client nicht angezeigt. Klicken Sie unter der Gruppenliste einfach auf den Button *Aktivieren*, um die Gruppe wieder zu aktivieren.

C.1.3.6 Erstellen eines Ticketfeldes

Ticketfelder enthalten Daten für Tickets, z. B. Priorität, Service-Level, Deadline oder Hardwaremodul. Die Felder einer Ticketfeldgruppe werden auf der rechten Seite erstellt. Wählen Sie zuerst die gewünschte Gruppe auf der linken Seite aus und klicken Sie dann rechts unter dem Bereich *Felder* auf den Button *Hinzufügen*. Es wird folgendes Pop-up-Fenster geöffnet:

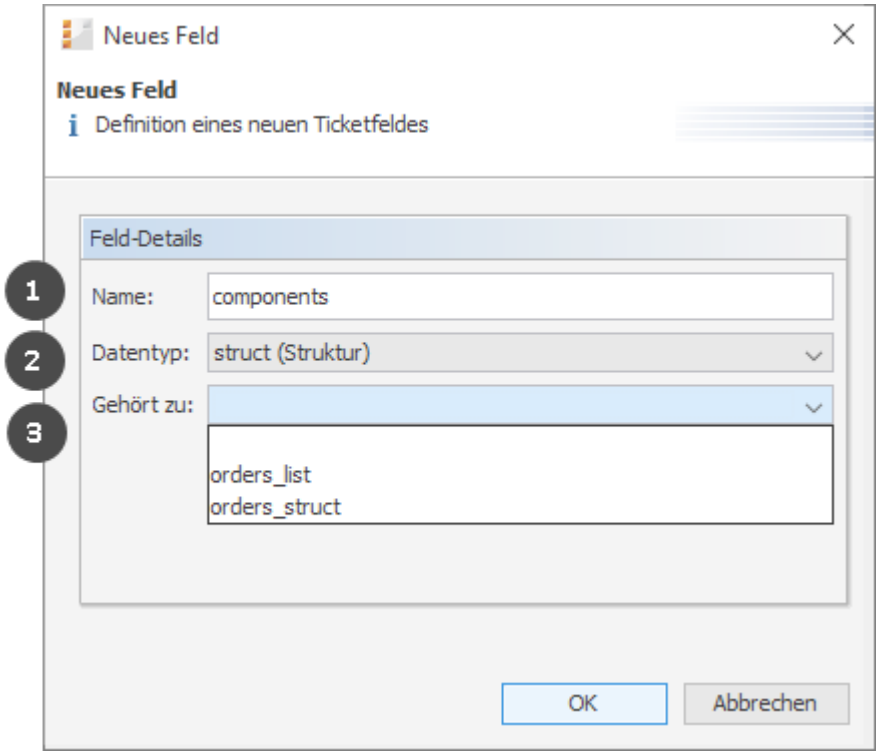


Abbildung 55: *ConSol CM Admin Tool - Tickets, Ticketfelder: Erstellen eines Ticketfeldes*

Füllen Sie folgende Informationen aus:

- **Name (1):**
  Geben Sie einen Namen für das Ticketfeld ein. Der Name muss innerhalb der Ticketfeldgruppe eindeutig sein. Die lokalisierten Namen für ein Ticketfeld werden im Hauptbereich von *Ticket-felder* gesetzt. Details dazu finden Sie im Abschnitt Lokalisierung von Datenfeldern
- **Datentyp (2)**
  Wählen Sie einen der verfügbaren Datentypen für das neue Ticketfeld. Siehe Arten von Daten-feldern.
- **Gehört zu (3):**
  In diesem Feld werden die verfügbaren Ticketfelder für die Datentypen LIST und STRUCT ange-zeigt, mit denen Listen und Tabellen erstellt werden. Wählen Sie im Drop-down-Feld, zu wel-cher LIST oder STRUCT das Ticketfeld gehört (sofern relevant).

Arten von Datenfeldern

Die folgenden Datentypen sind für Ticketfelder, Kundenfelder und Ressourcenfelder verfügbar:

- **autocomplete (Autovervollständigung)**
  Ein Datenfeld, das eine skriptbasierte Autocomplete-Liste enthält. Das ist eine dynamische Liste, die auf einem Skript des Typs *Text-Autovervollständigung* basiert. Eine detaillierte Beschreibung von skriptbasierten Autocomplete-Listen finden Sie im Abschnitt Skriptbasierte Autocomplete-Listen.
- **boolean (Ja/Nein)**
  Werte: true/false. Je nach Einstellung in der Annotation `boolean-type` wird der Wert als Checkbox, Radio-Button oder Drop-down-Liste angezeigt.

Wenn Sie in CM-Workflows oder im Admin Tool mit Skripten arbeiten, sollten Sie wissen, dass das Verhalten von Boolean-Feldern, die als Checkboxen dargestellt werden, d. h. mit der Annotation `boolean-type` = „checkbox “(Standardwert) versehen sind, anders ist als in früheren CM-Versionen!

- **In CM-Versionen vor 6.9.4.0:**
  Wenn ein Boolean-Feld noch nicht angefasst wurde, ist der Wert „false“. Wenn es aktiviert wird, ist der Wert „true“, und wenn es danach wieder deaktiviert wird, ist der Wert wieder „false“.
- **In Version 6.9.4.0 und höher:**
  Wenn ein Boolean-Feld noch nicht angefasst wurde, ist der Wert „NULL“. Wenn es aktiviert wird, ist der Wert „true“, und wenn es danach wieder deaktiviert wird, ist der Wert „false“.

Felder, die in der Datenbank bereits mit Werten gefüllt sind, werden während des Updates von einer Version vor 6.9.4.0 auf Version 6.9.4.0 und höher nicht verändert.

Boolean-Felder, die als Radio-Button (Annotation `boolean-type` = „radio“) oder Drop-down-Liste (Annotation `boolean-type` = „select“) dargestellt sind, hatten schon immer das Verhalten, das für Version 6.9.4.0 oder höher beschrieben ist, d. h. sie haben den Wert „NULL“, wenn sie noch nicht angefasst wurden.

- **date (Datum)**
  Format und Genauigkeit können über Annotationen festgelegt werden.
- **enum (Sortierte Liste)**
  Für Sortierte Listen. Die Bearbeiter können im Web Client zwischen einem der Listenwerte wählen. Die Sortierten Listen und ihre Werte müssen zuvor in der Verwaltung von Sortierten Listen erstellt worden sein. Wählen Sie den gewünschten *Listentyp* und die *Listengruppe* in den Feldern darunter.
- **list (Liste)**
  Ein Feld dieses Datentyps ist der erste Schritt zur Erstellung einer Liste (eine Spalte) oder einer Tabelle (mehrere Spalten) von Eingabefeldern im Web Client.
  - Für eine **Tabelle** wird im nächsten Schritt ein anderes Feld des Typs *struct* erstellt (siehe unten), das später die Eingaben der einzelnen Listenfelder enthält (die die Spalten der Tabelle sind). Wenn Sie also eine Tabelle erstellen möchten, müssen Sie zuerst ein Feld des Typs *struct* erzeugen (siehe unten), bevor Sie die Felder für die Tabellenspalten hinzufügen können.
  - Für eine **einfache Liste** ist der nächste Schritt die Erstellung der Felder, die zu der Liste gehören. In diesem Fall ist kein *struct* erforderlich.

  Für alle Felder, die zu einer Liste oder Tabelle gehören, müssen Sie die Abhängigkeiten im Feld *Gehört zu* angeben (siehe unten). Ein Tabellenfeld (ein normales Datenfeld), gehört zum Beispiel immer zu einem *struct* und ein *struct* gehört immer zu einer *list*.

- **struct (Struktur)**
  Mit einem Datenfeld dieses Typs wird eine Datenstruktur (Zeile einer Tabelle) definiert, die ein oder mehrere Felder umfasst. Das ist der zweite Schritt zur Erstellung einer Tabelle, nachdem Sie zuerst ein Feld des Typs *list* erstellt haben. Im nächsten Schritt fügen Sie die Felder für die Spalten der Tabelle hinzu. Die Abhängigkeiten müssen für jedes Datenfeld im Feld *Gehört zu* (siehe unten) gesetzt werden, d. h. ein *struct* gehört immer zu einer *list*.

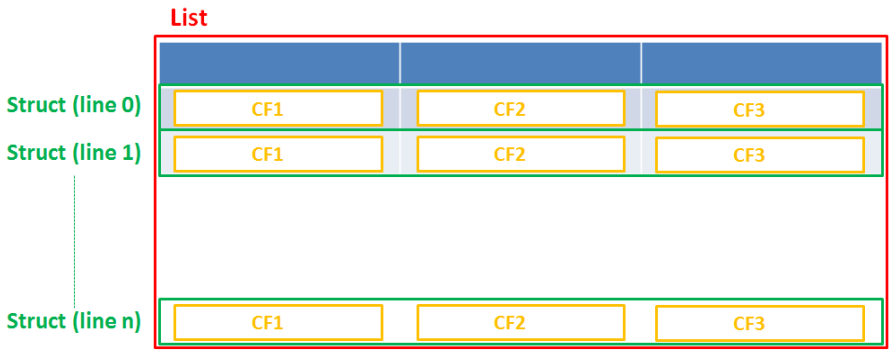


Abbildung 56: *Schema: List of Structs*

Technisch gesehen ist eine Liste ein Array, der in jedem Feld eine Map (= Paare von Schlüssel:Wert) enthält.

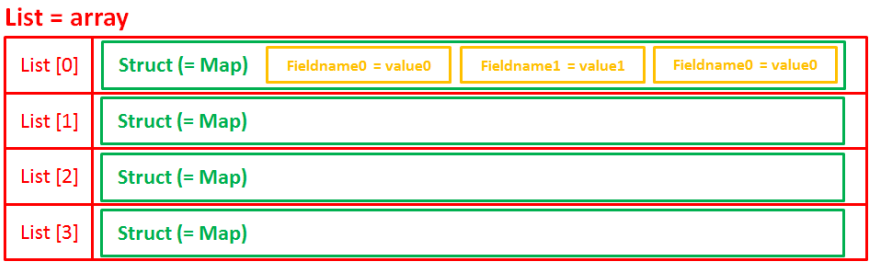


Abbildung 57: *List of structs, technisches Prinzip*

- **number (Zahl)**
  Für Ganzzahlen (Integer).
- **fixed point number (Festkommazahl)**
  Für Festkommazahlen, z. B. Währungen. Sie müssen die Gesamtanzahl der Ziffern (*Genauigkeit*) und die Anzahl der Ziffern nach dem Komma (*Skalierung*) in den entsprechenden Feldern eingeben.
- **string (Text)**
  Für bis zu 4000 alphanumerische Zeichen.

> ⚠ Beschränkung bei der Verwendung einer Oracle-Datenbank: Es können höchstens 4000 Bytes in UTF-Codierung gespeichert werden. Ab Oracle12c.

- **long string (Text)**
  Für große Objekte

  > ⚠ Beim Datentyp *long string (Text)* hängt die Grenze von dem für ConSol CM verwendeten Datenbanksystem ab: MS SQL Server: 2 GByte; MySQL: 4 GByte; Oracle: 16 - 64 GByte (je nach Page-Größe des Tablespace).

- **short string (Text)**
  Für bis zu 255 alphanumerische Zeichen.

  > ⓘ Bei Feldern des Datentyps *string (Text)* können Sie die Felddefinition mithilfe von Annotationen präzisieren. Ein Feld des Datentyps *string* kann zum Beispiel so definiert werden, dass es eine URL enthält und automatisch als Hyperlink angezeigt wird. Lesen Sie dazu den folgenden Abschnitt.

- **contact data reference (Referenz auf ein Kontaktdatenfeld)**
  Besonderer Datentyp, der intern als Referenz auf die mit dem Ticket verknüpften Kontakte verwendet wird. In FlexCDM (d. h. ab CM-Version 6.9.0), wird dieser Datentyp nicht mehr angezeigt, sondern nur noch intern im CM-System verwendet.
- **MLA field (Baum sortierter Listen)**
  Dieser Datentyp wird für Felder verwendet, die hierarchische Listen mit einer Baumstruktur enthalten, das sogenannte *MLA* (Multi Level Attributes). Der Name dieses Feldes ist der Name des neuen MLA, das in der MLA-Verwaltung angelegt werden muss. Die Gruppe des Feldes muss angegeben werden, nachdem das MLA erstellt wurde.

> ⚠ Der Datentyp, den Sie beim Erstellen eines Datenfeldes verwenden, kann danach nicht mehr verändert werden!

Details über String-Felder: Verwenden von Annotationen zum Anpassen von Strings

String-Felder werden häufig für Kunden-, Ticket- und Ressourcendaten verwendet und in Strings kann unterschiedlicher Inhalt gespeichert werden, zum Beispiel ein Textfeld mit einem Kommentar, ein einfaches Eingabefeld mit nur 20 Zeichen, eine URL oder ein Passwort. Das Anpassen von String-Feldern erfolgt mithilfe von speziellen Annotationen, die auf der Seite Annotationen des Anhangs aufgeführt sind. Da die Arbeit mit diesen Annotationen zu den alltäglichen Aufgaben eines CM-Administrators gehört, sind die wichtigsten und am häufigsten verwendeten Annotationen zusätzlich an dieser Stelle erklärt.

Wie kann ich ...

... statt einer einzelnen Zeile ein **Textfeld** einfügen?

Wert für Annotation `text-type`: „textarea“

Die Größe des Textfeldes kann angepasst werden. Es wird je nach Webbrowser als Standardtextfeld angezeigt. Verwenden Sie die Annotation `field-size`, wenn das Textfeld eine bestimmte Größe haben soll.

... die Eingabe des Feldes für **Passwörter** ausblenden?

Wert für Annotation `text-type`: „password“

Es werden nur Punkte angezeigt. Diese Annotation legt **nicht** fest, dass das Feld ein Passwort enthält! Es definiert nur den Anzeigemodus! Verwenden Sie die Annotation `password`, um ein String-Feld zu definieren, das das CM/Track-Passwort enthält.

... einen **Hyperlink** anzeigen, den Namen statt dem Link anzeigen?

Wert für Annotation `text-type`: „url“

Die Eingabe wird im Ansichtsmodus als Hyperlink angezeigt. Der String muss einem bestimmten URL-Muster entsprechen:

- "^((?:mailto\:|(?:(?:ht|f)tps?)\://)1\S+)(?: (?:\|)?(.*))?$"

Der erste Teil des Strings ist der Link (URL) und der zweite Teil ist der Name, der angezeigt werden soll.

Beispiel: "http://consol.de ConSol"

... einen **Dateilink** anzeigen?

Wert für Annotation `text-type`: „file-url“

Die Eingabe wird als Link auf eine Datei im Dateisystem angezeigt. Der Webbrowser muss solche Links zulassen/unterstützen!

Beispiel: Aktivieren von file://-URLs in einem Firefox-Browser

Fügen Sie die folgenden Zeilen entweder zur Konfigurationsdatei `prefs.js` oder zu `user.js` im Benutzerprofil hinzu. Auf einem Windows-System befindet sich diese normalerweise in einem Ordner wie `C:\Benutzer\<BENUTZERNAME>\AppData\Roaming\Mozilla\Firefox\Profiles\uvubg4fj.default`

- user_pref("capability.policy.localfilelinks.checkloaduri.enabled", "allAccess");
- user_pref("capability.policy.localfilelinks.sites", "http://cm-server.domain.com:8080");
- user_pref("capability.policy.policynames", "localfilelinks");

Alternativ kann ein Add-on des Firefox-Browsers wie ***Local Filesystem Links*** installiert werden, um einfacher auf die referenzierten Dateien und Ordner zugreifen zu können.

Dieser Link wird auch als Tooltip angezeigt.

Die URL ist korrekt formatiert, wenn folgende Bedingungen erfüllt sind:

- Sie beginnt mit `file:` gefolgt von normalen Schrägstrichen:
  - drei Schrägstriche "///" für Dateien, die sich auf demselben Computer befinden wie der Browser (alternativ "//localhost/") oder

  - zwei Schrägstriche gefolgt vom Servernamen und einem weiteren Schrägstrich für Dateien auf Dateiservern, die vom Rechner, auf dem der Webbrowser läuft, erreichbar sind.
- Danach folgt der vollständige Dateipfad, der mit dem Dateinamen endet.
- Auf Microsoft Windows-Systemen wird der Pfad ebenfalls mit normalen Schrägstrichen anstelle von umgekehrten Schrägstrichen geschrieben.
- Der Laufwerksbuchstabe eines lokalen Pfads auf Microsoft Windows-Systemen wird wie üblich verwendet, zum Beispiel `C:`.
- Pfade mit Leerzeichen und Sonderzeichen wie "{, }, ^, #, ?" müssen auf Microsoft Windows-Systemen mit Prozentzeichen kodiert werden (z. B. mit "%20" für ein Leerzeichen).

Beispiel-URLs:

- file://file-server/path/to/my/file.ext
- file:///linux/local/file.pdf
- file:///C:/Users/myuser/localfile.doc

Siehe auch die Erklärung von `file-url` im Abschnitt Liste der Feldannotationen

... ein **Label** definieren?

Wert für Annotation `text-type`: „label“

Dies ist ein schreibgeschütztes Feld, das grau angezeigt wird. Verwenden Sie die Annotation `label-group`, um das Label und die dazugehörigen Eingabefelder zu verknüpfen. Berücksichtigen Sie die Annotationen für Label (`show-label-in-edit`, `show-label-in-view`), bevor Sie spezielle Label-Felder implementieren!

... ein Feld für **gültige E-Mail-Adressen** definieren?

Wert für Annotation `email`: „true“

Das Feld darf nur gültige E-Mail-Adressen enthalten. Die Eingabe wird gemäß des Standardformats für E-Mail-Adressen validiert <name>@<domain>.

... eine skriptbasierte Autocomplete-Liste definieren?

Wert für die Annotation `text-type` = „autocomplete“

Optional: Wert für die Annotation `autocomplete-script` = <Name des entsprechenden Skripts>

Eine skriptbasierte Autocomplete-Liste wird verwendet, um ein Drop-down-Menü bereitzustellen, das anhand der Eingabe, die der Bearbeiter bereits vorgenommen hat, dynamisch gefüllt wird. Wenn der Benutzer zum Beispiel "Mei" eingibt, werden die möglichen Werte "Meier", "Meister" und "Meinert" als Liste angezeigt und der Bearbeiter kann den für das Feld erforderliche Wert auswählen. Sie kennen dieses Verhalten von anderen Autocomplete-Feldern, z. B. der Suche nach Bearbeitern für ein Ticket oder die Suche nach Kunden bei der Ticketerstellung. In diesen Fällen erzeugt CM die Liste allerdings automatisch und das Verhalten kann nicht beeinflusst oder angepasst werden. Im Gegensatz dazu können skriptbasierte Autocomplete-Listen vom CM-Administrator implementiert werden. Die Werte basieren auf einem Ergebnissatz, der dynamisch erstellt wird. Der Ergebnissatu kann Strings, Bearbeiter, Kunden (Units) und Ressourcen enthalten.

Eine detaillierte Beschreibung von skriptbasierten Autocomplete-Listen finden Sie im Abschnitt Skript-basierte Autocomplete-Listen.

... ein Feld für persönliche Daten definieren?

Wert für Annotation `personal-data`: „true“

Diese Annotation kann Ticket- und Kontaktfeldern zugewiesen werden. Kontaktfelder mit dieser Annotation werden gelöscht, wenn ein Kontakt anonymisiert wird. Ticketfelder mit dieser Annotation werden gelöscht, wenn der Hauptkunde des Tickets anonymisiert wird. Siehe Beispiel 8: Löschen von Kundendaten für Informationen über das Anonymisieren von Kontakten.

> ⚠ Beachten Sie bei der Definition von Feldern für persönliche Daten, dass das Löschen des Feldes während des Anonymisierungsprozesses wie eine normale Aktualisierung behandelt wird. Das heißt, Business-Event-Trigger, die auf Änderungen an Ticketfeldern reagieren, feuern und das Aktionsskript zur Kontaktaktualisierung wird ausgeführt.
>
> Dies kann zu unerwünschten Nebeneffekten führen.

C.1.3.7 Kopieren eines Ticketfeldes

Es ist nicht nötig, dass Sie alle Datenfelder über das Pop-up-Fenster erstellen. Sie können stattdessen auch ein vorhandenes Datenfeld kopieren. Das kann ein Ticketfeld, ein Kundenfeld oder ein Ressourcenfeld sein. Wählen Sie das entsprechende Datenfeld in der Datenfeldgruppe und klicken Sie auf den Button *Kopieren*. Navigieren Sie dann in die Datenfeldgruppe, in die Sie das kopierte Feld einfügen möchten. Das kann eine Ticketfeldgruppe, eine Kundenfeldgruppe oder eine Ressourcenfeldgruppe sein. Sie können ein Feld von einem Objekttyp (z. B. Ticket) in einen anderen (z. B. Kunde) kopieren. Klicken Sie in die Liste der Datenfelder der Zielfeldgruppe und klicken Sie auf den Button *Einfügen*.

Es kann jeder Feldtyp kopiert werden, von einfachen String-Feldern über Sortierte Listen bis hin zu ganzen Strukturen (STRUCTS). Um ein STRUCT zu kopieren, markieren und kopieren Sie nur die Liste. Die untergeordneten Objekte (z. B. STRUCT mit Feldern) werden automatisch kopiert.

Folgendes wird kopiert:

- fast alle Annotationen mit ihren Werten
- alle lokalisierten Werte

Folgendes wird nicht kopiert:

- die folgenden Annotationen
  - *ldapid*: Diese Annotation kann in einem Datenmodell nur einmal gesetzt werden. Eine Duplizierung würde zu einem ungültigen Modell führen.
  - *username*: Diese Annotation kann in einem Datenmodell nur einmal gesetzt werden. Eine Duplizierung würde zu einem ungültigen Modell führen.
  - *password*: Diese Annotation kann in einem Datenmodell nur einmal gesetzt werden. Eine Duplizierung würde zu einem ungültigen Modell führen.

- *position*: Der Wert dieser Annotation muss eindeutig sein. Die Annotation wird also dupliziert, aber der Wert wird aus der Kopie entfernt.
- *ticket-list-position*: Diese Annotation sollte nur für spezielle Fälle verwendet werden und daher nur für wenige Felder gesetzt werden. Durch die Duplizierung würde sie möglicherweise multipliziert, obwohl sie für die Mehrzahl der Kopien nicht benötigt wird.

Benennung der kopierten Felder:

- Kopie in eine andere Feldgruppe: es wird der Feldname verwendet
- Kopie in die gleiche Feldgruppe: es wird <Feldname>_copy verwendet

Neben den Buttons zum Kopieren und Einfügen können Sie auch die Tastenkombinationen verwenden:

- Kopieren: STRG-C
- Einfügen: STRG-V

C.1.3.8 Editieren eines Ticketfeldes

Wenn Sie ein Ticketfeld editieren möchten, wählen Sie es in der Liste aus und klicken Sie auf den Button *Bearbeiten*. Es wird dasselbe Fenster geöffnet wie für die Erstellung eines Ticketfeldes. Mit Ausnahme von *Datentyp*, *Listentyp* und *Listengruppe* können Sie alle Felder ändern und die Änderungen durch Klicken auf *OK* speichern.

C.1.3.9 Annotieren eines Ticketfeldes

So wie Ticketfeldgruppen werden auch Ticketfelder annotiert, um die Eigenschaften des Feldes zu definieren, z. B. ob es schreibgeschützt ist und an welcher Stelle der Web-Client-GUI es angezeigt werden soll. (Im Abschnitt Annotationen finden Sie eine Liste aller verfügbaren Annotationen.) Wählen Sie ein Feld aus und klicken Sie unter der Liste auf den Button *Annotieren*. Es wird folgendes Pop-up-Fenster geöffnet:

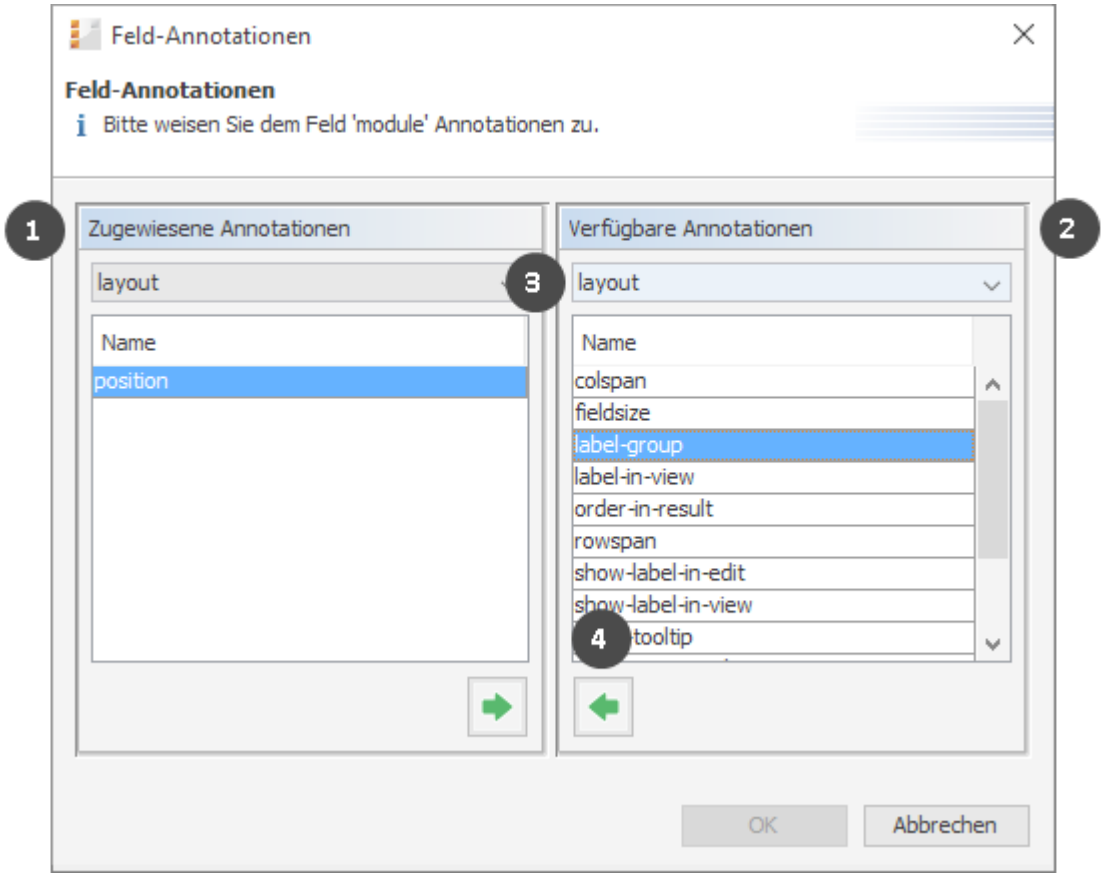


Abbildung 58: *ConSol CM Admin Tool - Tickets, Ticketfelder: Zuweisen von Annotationen zu einem Ticketfeld*

Im rechten Teil des Fensters werden die verfügbaren Annotationen (2) angezeigt. Im Auswahlfeld über der Liste können Sie die Liste nach Annotationstyp filtern (3). Wählen Sie die gewünschten Annotationen aus und verschieben Sie sie in die Liste *Zugewiesene Annotationen* auf der linken Seite (1), indem Sie auf den Button *Zuweisen* (4) klicken. Diese Liste kann ebenfalls nach Annotationstyp gefiltert werden. Klicken Sie auf *OK*, um die Annotationen zum Ticketfeld zuzuweisen und das Fenster zu schließen.

Die Annotationen für das ausgewählte Feld werden jetzt mit ihrem Standardwert (sofern verfügbar, z. B. „true“ oder „false“) in der unteren rechten Ecke der Admin-Tool-Seite angezeigt. Sie können den Wert verändern, indem Sie einen Doppelklick in das entsprechende Feld *Wert* machen und den gewünschten Wert eingeben. Drücken Sie danach die Eingabetaste.

Ticketfelder werden im Web Client in der Reihenfolge angezeigt, in der sie in der Liste stehen, außer Sie haben das Layout mithilfe der Annotation `position` festgelegt. Sie können die Position eines Feldes in der Liste verändern, indem Sie unter der Liste auf die Buttons *Nach oben* und *Nach unten* klicken.

Hinweis zum Layout der Ticketdaten:

Sie können mehrere Spalten mit Datenfeldern pro Zeile definieren, z. B. kann die Annotation `position` mit 0, 1 oder 2 enden.

0;0 0;1 0;2

1;0 1;1 1;2

C.1.3.10 Löschen eines Ticketfeldes

Ein Ticketfeld kann nur gelöscht werden, wenn es keiner Queue bzw. keinem Ticket zugewiesen ist. Andernfalls wird eine Warnung angezeigt, dass Sie das Feld nur deaktivieren können (siehe unten). Um ein Ticketfeld zu löschen, wählen Sie es in der Liste aus und klicken Sie auf den Button *Löschen*. Wenn Sie das daraufhin angezeigte Fenster mit *Ja* bestätigen, wird das Ticketfeld aus der Liste und aus dem System gelöscht.

C.1.3.11 Aktivieren oder Deaktivieren eines Ticketfeldes

Wenn Sie ein Ticketfeld nicht löschen können, oder für den Fall, dass es später noch einmal benötigt wird, nicht löschen möchten, können Sie es deaktivieren. Wählen Sie dazu das Feld aus und klicken Sie auf den Button *Deaktivieren*. Der entsprechende Listeneintrag wird danach kursiv angezeigt. Ein deaktiviertes Ticketfeld wird im Web Client nicht angezeigt. Klicken Sie unter der Liste der Ticketfelder einfach auf *Aktivieren*, um das Feld wieder zu aktivieren.

C.1.3.12 Sichtbarkeit von Ticketdaten in CM/Track

Wenn CM/Track, ein ConSol CM-Add-on, das ein Kundenportal bereitstellt, in Ihrem CM-System aktiv ist, müssen Sie Ticketfelder für CM/Track konfigurieren. Außerdem muss die Sichtbarkeit der Ticket-felder im Kundenportal konfiguriert werden. Dies ist im Abschnitt CM/Track V2: Datenverfügbarkeit für Kunden detailliert beschrieben.

C.1.3.13 Verwenden von skriptbasierter Feldvisualisierung für Ticketfelder

Mit der skriptbasierten Feldvisualisierung können Sie die Anzeige von Daten in Ticketfeldern ver-bessern. Details dazu finden Sie im Abschnitt Skripte des Typs Feld-Visualisierung.

C.1.4 Tab *Aktivitätsformulare*

C.1.4.1 Einleitung

Ein Aktivitätsformular ist ein Webformular, das während des Prozesses angezeigt wird, z. B. wenn der Bearbeiter an dem Ticket arbeitet. Wenn der Bearbeiter auf eine Workflow-Aktivität klickt, wird zuerst das ACF angezeigt. Der Bearbeiter muss bestimmte Datenfelder ausfüllen und die Änderungen speichern, um fortzufahren und die Workflow-Aktivität durchzuführen.

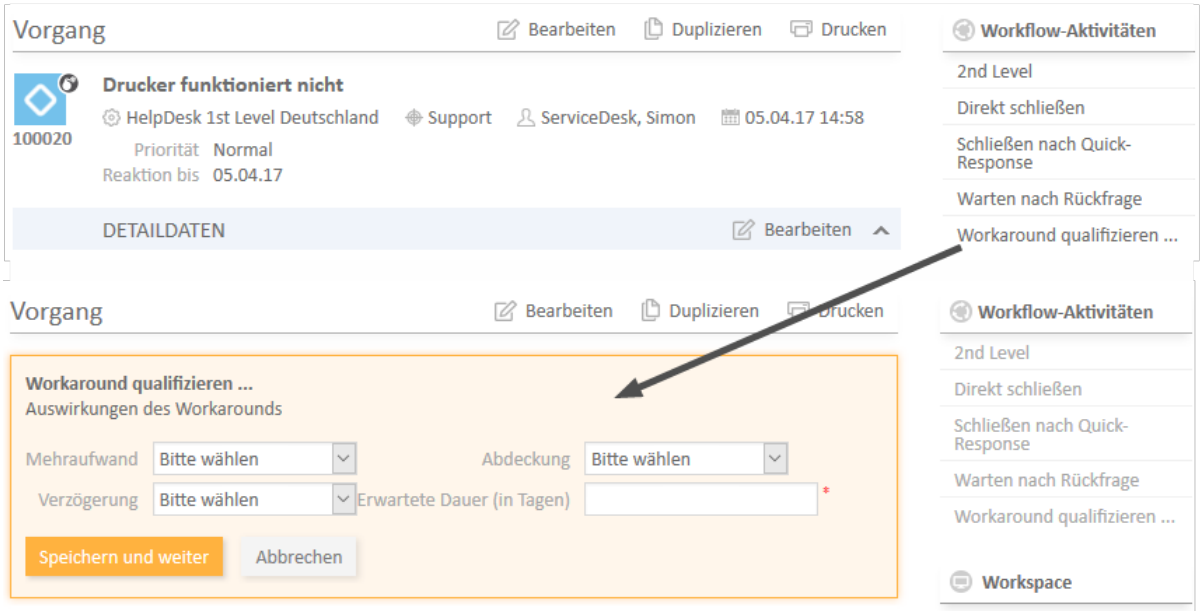


Abbildung 59: *ConSol CM Web Client - Beispiel für ACF*

Die Ticketfelder, die in einem ACF verwendet werden, müssen zuerst im ConSol CM Admin Tool als normale Ticketfelder definiert werden. ACFs können Ticketfelder aus einer oder mehreren Ticketfeldgruppen enthalten. Die ACFs werden mit dem ConSol CM Process Designer in den Prozess (Workflow) integriert. Dort können Sie auch festlegen, welche Ticketfelder Pflichtfelder sind und welche Felder optional sind. Die Anzeige eines ACF kann von einer Bedingung abhängen, z. B. wird das ACF *Grund der Ablehnung der Anfrage* nur angezeigt, wenn der Kunde den Status *Gold* oder *Platin* hat. ACF-Abhängigkeiten werden mithilfe von Skripten konfiguriert. Dies ist detailliert im *ConSol CM Process Designer Handbuch* beschrieben.

C.1.4.2 Definition von Aktivitätsformularen (ACFs)

In diesem Tab können Sie Aktivitätsformulare (ACF) erstellen, die im Process Designer Aktivitäten zugewiesen werden können. Diese werden verwendet, um Eingaben im Web Client zu erfassen, wenn bei einer manuellen Workflow-Aktivität Informationen für den nächsten Schritt vorbereitet werden, z. B. ein Ticket muss analysiert werden, bevor es vorwärts bewegt werden kann, oder es wird Feedback für ein Ticket benötigt. ACFs bestehen aus einem Satz an Ticketfeldern, die zuvor im Tab *Ticketdaten* erstellt worden sind. Ein ACF kann Ticketfelder aus mehr als einer Ticketfeldgruppe enthalten. Alle Ticketfeldgruppen müssen allerdings der Queue zugewiesen sein, der der Workflow zugewiesen ist, in

dem das ACF verwendet wird. Detaillierte Informationen über den Einsatz von ACFs im Workflow und die Funktionen, die mit ACF-Skripten programmiert werden können, finden Sie im *ConSol CM Process Designer Handbuch*.

Die folgende Abbildung zeigt die verfügbaren ACFs. Sie gelangen auf diese Seite, indem Sie das Navigationselement *Ticketfelder* der Navigationsgruppe *Tickets* öffnen und den Tab *Aktivitätsformulare* auswählen.

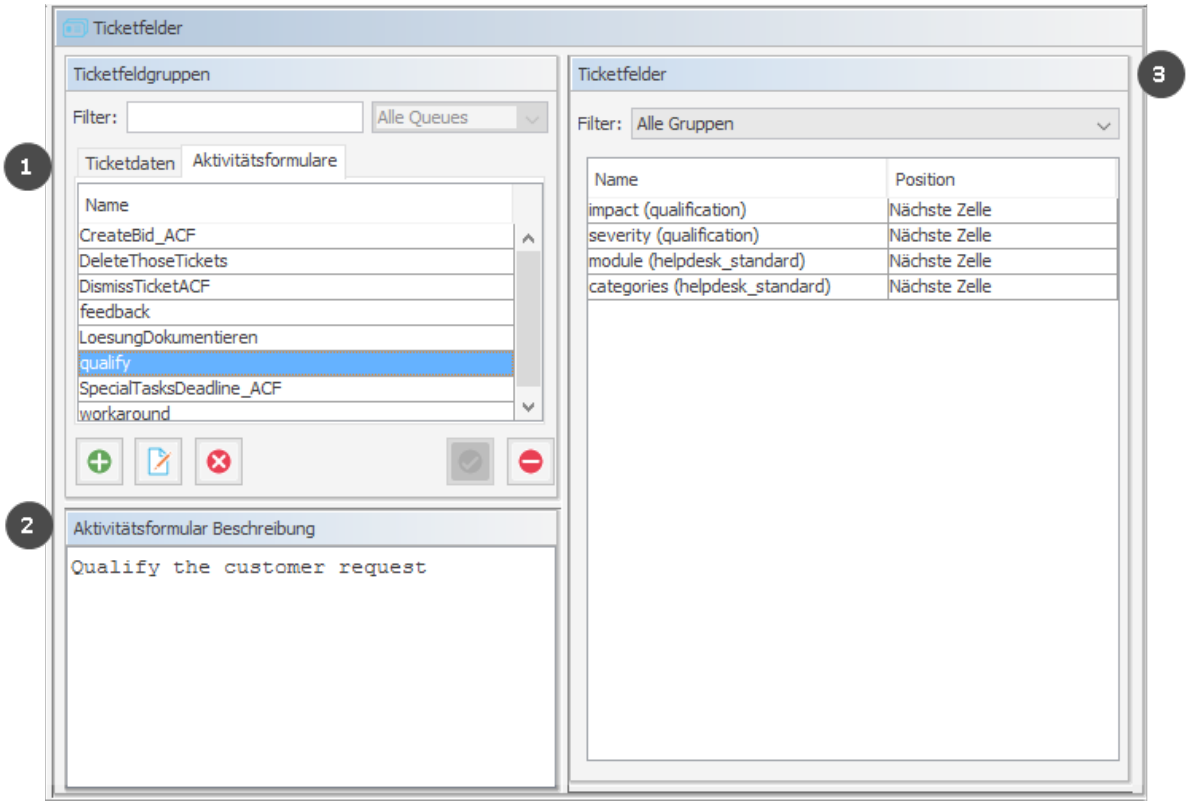


Abbildung 60: *ConSol CM Admin Tool - Tickets, Ticketfelder: Aktivitätsformulare*

Der Tab *Aktivitätsformulare* hat drei Abschnitte:

- **Aktivitätsformulare (1)**
  Enthält die verfügbaren Aktivitätsformulare
- **Aktivitätsformular Beschreibung (2)**
  Enthält die Beschreibung des ausgewählten Aktivitätsformulars
- **Ticketfelder (3)**
  Enthält die Ticketfelder des ausgewählten Aktivitätsformulars

C.1.4.3 Erstellen eines Aktivitätsformulars

Klicken Sie zum Erstellen eines ACFs einfach unter der Liste auf der linken Seite auf den Button *Hinzufügen*. Es wird folgendes Pop-up-Fenster geöffnet (das gleiche Fenster wird zum Erstellen und Editieren eines ACFs angezeigt).

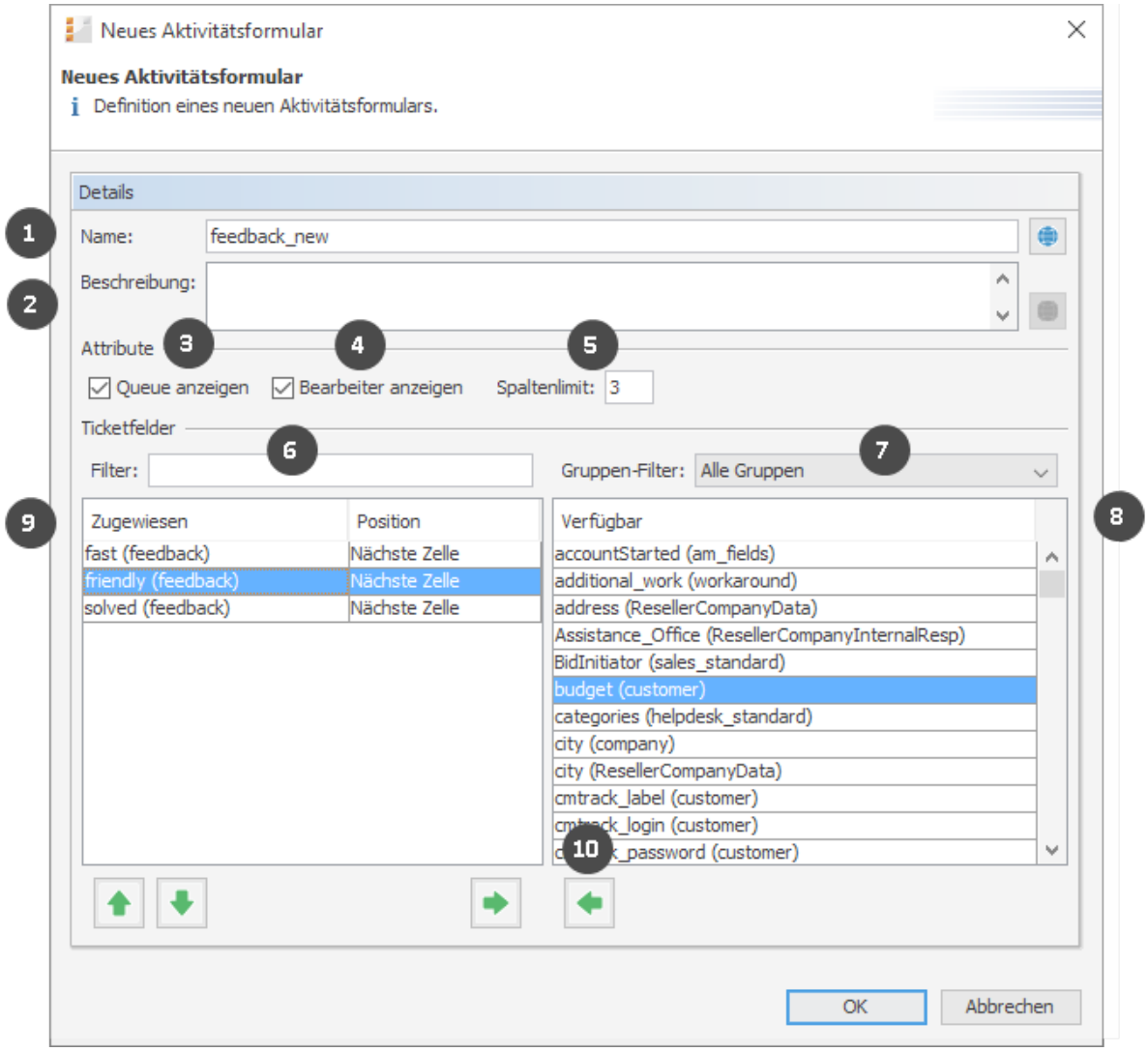


Abbildung 61: *ConSol CM Admin Tool - Tickets, Ticketfelder: Aktivitätsformular erstellen*

Geben Sie folgende Daten ein oder wählen Sie die gewünschten Werte aus:

- **Name (1)**
  Geben Sie den Namen des ACFs in diesem Feld ein. Sie können den Namen lokalisieren, indem Sie auf den Button *Lokalisieren* klicken. Details dazu finden Sie im Abschnitt Lokalisierung von Objekten allgemein, Typ 1.
- **Beschreibung (2)**
  Geben Sie eine Beschreibung des ACFs in diesem Feld ein. Die Beschreibung wird im Web Client als Überschrift des ACFs angezeigt. Sie können die Beschreibung lokalisieren, indem Sie auf den Button *Lokalisieren* klicken. Details dazu finden Sie im Abschnitt Lokalisierung von Objekten allgemein, Typ 1.
- **Queue anzeigen (3)**
  Markieren Sie diese Checkbox, wenn die Queue des Tickets mit dem ACF im Web Client angezeigt werden soll.

- **Bearbeiter anzeigen (4)**
  Markieren Sie diese Checkbox, wenn der Bearbeiter des Tickets mit dem ACF im Web Client angezeigt werden soll.
- **Spaltenlimit (5)**
  Nummernfeld. Definiert die Anzahl der Spalten für die Anzeige der Ticketfelder im ACF. „0“ bedeutet, dass es kein Spaltenlimit gibt, das heißt, alle Ticketfelder stehen in einer einzigen Zeile. Sie können für die einzelnen Ticketfelder auch mit dem Parameter *In neuer Zeile anzeigen* arbeiten (siehe unten).
- **Filter (6)**
  In diesem Feld können Sie eine Zeichenfolge eingeben, um die zugewiesenen Ticketfelder nach Namen zu filtern.
- **Gruppen-Filter (7)**
  Wählen Sie eine Ticketfeldgruppe aus der Liste aus, wenn Sie in der Liste der verfügbaren Ticketfelder nur die Ticketfelder anzeigen möchten, die zu dieser Gruppe gehören.
- **Listen mit Ticketfeldern (8/9)**
  Die Liste auf der rechten Seite (8) enthält die verfügbaren Ticketfelder mit der entsprechenden Ticketfeldgruppe. Sie können die Einträge aufsteigend und absteigend sortieren, indem Sie auf das Kopfzeilenfeld der Liste klicken. Die kleinen Pfeile nach oben und nach unten zeigen die aktuelle Sortierreihenfolge an. Wählen Sie die Ticketfelder für das ACF in dieser Liste aus und verschieben Sie sie in die Liste *Zugewiesen* auf der linken Seite (9), indem Sie auf *Zuweisen* (10) klicken.
  Für jedes zugewiesene Ticketfeld können Sie die Position festlegen. Die folgenden Werte sind möglich:
  - **Nächste Zelle** (Standardwert)
    Das Feld wird neben dem vorherigen Feld angezeigt und nicht in einer neuen Zeile.
  - **Neue Zeile**
    Es wird eine neue Zeile angezeigt, die mit diesem Feld beginnt.
  - **Neue Tabelle**
    Es wird in einer neuen Zeile eine neue Tabelle begonnen. Diese Option sollte für Datenstrukturen verwendet werden, z. B. eine *List of Structs*, die viel Platz benötigt und dadurch die Lesbarkeit des ACFs beeinträchtigt, da viel Platz für eine Spalte verwendet wird. Alle anderen Spalten würden ohne die Option *Neue Tabelle* nach rechts verschoben. Die folgenden vier Abbildungen verdeutlichen diese Funktion.

Die zugewiesenen Ticketfelder werden im Web Client in der Reihenfolge angezeigt, in der sie in dieser Liste stehen. Sie können die Reihenfolge ändern, indem Sie ein Element in der Liste auswählen und auf den Button *Nach oben* oder *Nach unten* klicken. Wenn Sie zugewiesene Ticketfelder entfernen möchten, wählen Sie sie aus und klicken Sie auf den Button *Entfernen*.

Klicken Sie zuletzt auf *OK*, um Ihre Eingaben zu speichern und das Fenster zu schließen.

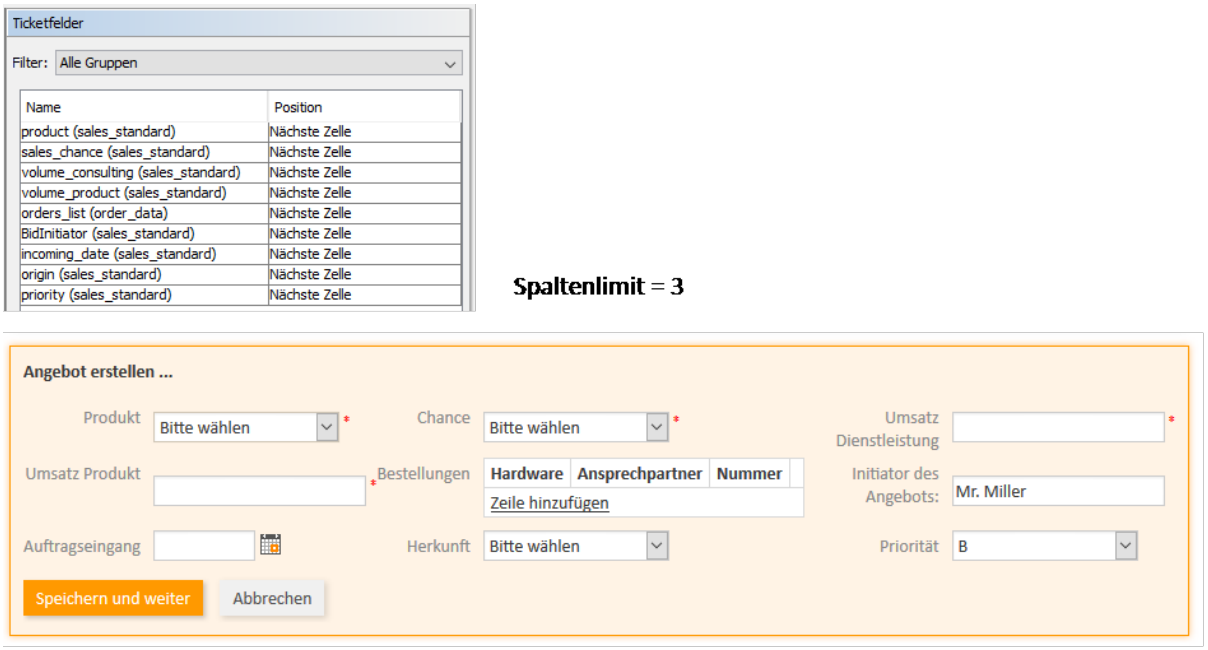


Abbildung 62: *Beispiel für die Optimierung des ACF-Layouts(1), noch nicht optimiert.*

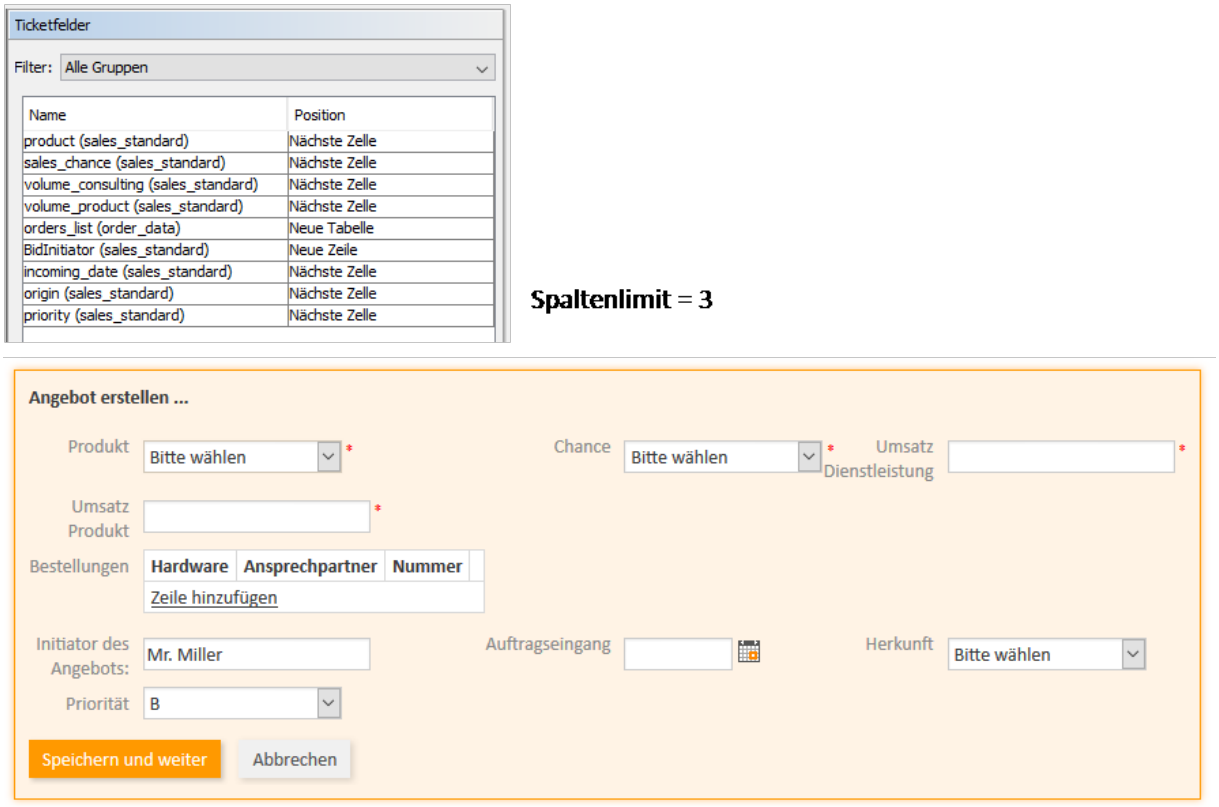


Abbildung 63: *Beispiel für die Optimierung des ACF-Layouts (2), Optimiert, Tabelle in eigener Spalte*

C.1.4.4 Editieren eines Aktivitätsformulars

Wenn Sie ein ACF editieren möchten, wählen Sie es in der Liste aus und klicken Sie auf den Button *Bearbeiten* oder doppelklicken Sie auf den Namen des ACFs. Zum Ändern der Details wird das gleiche Pop-up-Fenster geöffnet wie bei der Erstellung eines ACFs. Speichern Sie die Änderungen durch Klicken auf *OK*.

C.1.4.5 Löschen eines Aktivitätsformulars

Ein ACF kann nur gelöscht werden, wenn es keiner Workflow-Aktivität zugewiesen ist. Andernfalls wird eine Warnung angezeigt, dass Sie das ACF nur deaktivieren können (siehe unten). Um ein ACF zu löschen, wählen Sie es in der Liste aus und klicken Sie auf den Button *Löschen*. Wenn Sie das daraufhin angezeigte Fenster mit *Ja* bestätigen, wird das ACF aus der Liste und aus dem System gelöscht.

C.1.4.6 Aktivieren oder Deaktivieren eines Aktivitätsformulars

Wenn Sie ein ACF nicht löschen können, oder es für den Fall, dass es später noch einmal benötigt wird, nicht löschen möchten, können Sie es deaktivieren. Wählen Sie dazu das ACF aus und klicken Sie auf den Button *Deaktivieren*. Der entsprechende Listeneintrag wird danach kursiv angezeigt. Ein deaktiviertes ACF ist im Process Designer nicht verfügbar. ACFs, die aktuell verwendet werden, können nicht deaktiviert werden. Klicken Sie unter der ACF-Liste einfach auf den Button *Aktivieren*, um das ACF wieder zu aktivieren.

C.1.4.7 Lokalisieren eines Aktivitätsformulars

Klicken Sie im Erstellungs- oder Editierfenster auf den Button *Lokalisieren*, um den Namen und die Beschreibung des ACFs zu lokalisieren. Details dazu finden Sie im Abschnitt Lokalisierung von Objekten allgemein, Typ 1.

C.1.5 Häufig verwendete Annotationen

Hier finden Sie eine Liste mit den häufig verwendeten Annotationen auf Feldebene. Eine komplette Liste von Gruppen- und Feldannotationen finden Sie in Annotationen.

groupable 132
sortable 132
readonly 132
visibility 132
text-type 132
reportable 134
field indexed 134
position 134
colspan 134
rowspan 135
field-group 135
fieldsize 135
enum field with ticket color 136
accuracy 136
format 136
maxLength 136
minLength 136
required 137

groupable

- **Typ**: cmweb-common
- **Beschreibung**: Ermöglicht die Gruppierung nach diesem Feld in der Ticketliste. Eine detaillierte Erklärung dazu finden Sie im Abschnitt Gruppierung.
- **Werte**:
  - *true*: Wird nur für Datenfelder des Typs ENUM verwendet. Entfernen Sie die Annotation, wenn Sie die Gruppierung deaktivieren möchten.

sortable

- **Typ**: cmweb-common
- **Beschreibung**: Ermöglicht die Sortierung nach diesem Feld in der Ticketliste. Eine detaillierte Erklärung dazu finden Sie im Abschnitt Sortierung.
- **Werte**:
  - *true*: Wird für Datenfelder des Typs DATE oder ENUM verwendet. Entfernen Sie die Annotation, wenn Sie die Sortierung deaktivieren möchten.
    Für Felder des Typs ENUM : Funktioniert nur, wenn alle Werte des ENUM-Feldes indiziert werden.

readonly

- **Typ**: common
- **Beschreibung**Zeigt an, dass das Ticketfeld nicht verändert werden kann.
- **Werte**:
  - *true* / *false*: Das Feld ist schreibgeschützt, wenn der Wert auf „true“ gesetzt ist. Wenn kein Wert oder ein anderer Wert als „false“ gesetzt ist, wird „true“ angenommen.

visibility

- **Typ**: common
- **Beschreibung**: Definiert, wann das Feld sichtbar ist.
- **Werte**:
  - *edit*: Das Feld wird im Bearbeitungsmodus angezeigt.
  - *view*: Das Feld wird im Ansichtsmodus angezeigt.
  - *none*: Das Feld ist nicht sichtbar.
  - Wenn ein anderer Wert oder kein Wert gesetzt ist, ist das Feld immer sichtbar.

text-type

- **Typ**: component-type
- **Beschreibung**: Definiert mögliche Arten eines Felds des Datentyps STRING .

- **Werte**:
  - *text* (Standardwert): Einzeiliges Eingabefeld
  - *textarea*: Mehrzeiliges Eingabefeld
  - *password*: Eingabefeld für Passwörter.
    Das Passwort wird im Ansichtsmodus als ******* angezeigt.
  - *label*: Eingabe wird als Label angezeigt, d.h. das Feld wird nur angezeigt, es ist keine Dateneingabe möglich.
  - *autocomplete*: Das Feld wird als Autocomplete-Liste verwendet. Eine detaillierte Erklärung dazu finden Sie im Abschnitt Skriptbasierte Autocomplete-Listen.
  - *url*: Eingabe wird im Anzeige-Modus als Link dargestellt. Der String muss dafür diesem URL-Muster entsprechen:
    "^((?:mailto\:|(?:(?:ht|f)tps?)\://)1\S+)(?: (?:\|)?(.*))?$"
    Beispiel: "http://consol.de ConSol"
  - *file-url*: Die Eingabe wird als Link auf eine Datei im Dateisystem angezeigt. Der Webbrowser muss solche Links zulassen/unterstützen! Informationen dazu finden Sie im Abschnitt Details über String-Felder: Verwenden von Annotationen zum Anpassen von Strings. Dieser Link wird auch als Tooltip angezeigt.

    Die URL ist korrekt formatiert, wenn folgende Bedingungen erfüllt sind:

    Sie beginnt mit „file:“ gefolgt von normalen Schrägstrichen:

    - drei Schrägstriche "///" für Dateien, die sich auf demselben Computer befinden wie der Browser (alternativ "//localhost/") oder
    - zwei Schrägstriche gefolgt vom Servernamen und einem weiteren Schrägstrich für Dateien auf Dateiservern, die vom Rechner, auf dem der Webbrowser läuft, erreichbar sind.

    Danach folgt der vollständige Dateipfad, der mit dem Dateinamen endet. Auf Microsoft Windows-Systemen wird der Pfad ebenfalls mit normalen Schrägstrichen anstelle von umgekehrten Schrägstrichen geschrieben.

    Der Laufwerksbuchstabe eines lokalen Pfads auf Microsoft Windows-Systemen wird wie üblich verwendet, zum Beispiel C:. Pfade mit Leerzeichen und Sonderzeichen wie "{, }, ^, #, ?" müssen auf Microsoft Windows-Systemen mit Prozentzeichen kodiert werden (z. B. mit "%20" für ein Leerzeichen).

    Beispiel-URLs:

    - file://file-server/path/to/my/file.ext
    - file:///linux/local/file.pdf
    - file:///C:/Users/myuser/localfile.doc

    Siehe auch Erklärung im Abschnitt Details über String-Felder: Verwenden von Annotationen zum Anpassen von Strings.

reportable

- **Typ**: dwh
- **Beschreibung**: Gibt an, dass das Feld für Reports verwendet werden kann und an das DWH übertragen werden soll.
- **Werte**:
  - *true* / *false*: Das Feld kann in Reports verwendet werden, wenn der Wert auf „true“ gesetzt ist.

field indexed

- **Typ**: indexing
- **Beschreibung**: Gibt an, dass für das Feld ein Datenbankindex erstellt wird. Wenn es möglich sein soll, Ergebnistabellen (im Web Client) durch Klicken auf die Spaltenüberschrift nach einer Spalte zu sortieren, muss das entsprechende Feld indiziert sein!
- **Werte**:
  - *transitive* (Standardwert): Alle Daten werden angezeigt (Ticketdaten, Kundendaten und Ressourcendaten).
  - *unit*: Für Kundendaten verwendet. Es werden nur die Unit und die Parent Unit (d. h. Firma) als Suchergebnis zurückgegeben, Tickets werden nicht angezeigt.
  - *local*: Für Kundendaten verwendet. Es wird nur die Unit als Suchergebnis zurück-gegeben, Firmen und Tickets werden nicht angezeigt.
  - *<Annotation nicht gesetzt>*: Das Feld ist nicht indiziert.

position

- **Typ**: layout
- **Beschreibung**: Definiert die Position eines Feldes im Rasterlayout oder in einer (LIST OF STRUCTS).
- **Werte**:
  - **<Zahl>;<Zahl>**: Die Werte geben die Zeile und Spalte an (Zeile;Spalte), die Nummerierung beginnt bei 0;0. Wenn kein Wert gesetzt ist, wird das Datenfeld in der ersten freien Zelle des Rasters angezeigt.
  - **0;<Zahl>**: Es wird nur der Wert der Spalte verwendet, der Wert der Zeile wird ignoriert.

colspan

- **Typ**: layout
- **Beschreibung**: Legt fest, wie viele Spalten für das Feld im Layout reserviert werden.
- **Werte**:
  - **<Zahl>**: Anzahl der Spalten.

⚠ Diese Annotation funktioniert nur, wenn auch die Annotation `position` für das Feld gesetzt ist.

rowspan

- **Typ**: layout
- **Beschreibung**: Legt fest, wie viele Zeilen das Feld im Layout belegt.
- **Werte**:
  - **<Zahl>**: Anzahl der Zeilen.

⚠ Diese Annotation funktioniert nur, wenn auch die Annotation `position` für das Feld gesetzt ist.

field-group

- **Typ**: layout
- **Beschreibung**: Ermöglicht die Gruppierung der Felder im Ansichtsmodus. Die Annotation wird im Bearbeitungsmodus ignoriert.
- **Werte**:
  - **<string>**: Um Felder zu gruppieren, muss für jedes Feld dieselbe Zeichenfolge als Annotation gesetzt werden. Zwei oder mehr Datenfelder sind verbunden, wenn sie für diese Annotation den gleichen Wert besitzen. Die Gruppe der verbundenen Datenfelder wird nur angezeigt, wenn in allen von ihnen Werte gesetzt sind.
- Entfernt in ConSol CM-Version 6.11.0.1

fieldsize

- **Typ**: layout
- **Beschreibung**: Die angezeigte Feldgröße im Ticketlayout.
- **Werte**:
  - **<Zeilen>;<Spalten>**: Angezeigte Feldgröße.
    Format für Felder des Datentyps STRING und NUMBER: n gibt die Anzahl der Zeichen an. Für String-Felder ist das die Anzahl an Großbuchstaben M in einer Monospace-Schriftart).
    Format für `textarea`: Zeilen;Spalten (entspricht <textarea Zeilen="" Spalten="">).
    Sortierte Listen werden anstatt als Drop-down-Liste als Auswahlfeld mit n Elementen angezeigt. Format für Sortierte Listen: n.
    Hinweis: Diese Annotation dient nur zur Konfiguration des Layouts. Verwenden Sie für die Validierung `maxlength` vom Typ `group validation`.
  - **<Zahl>**: Für Datenfelder des Typs ENUM. Definiert, wie viele Werte im Listenfeld direkt sichtbar sind. Nur zu Layout-Zwecken.

enum field with ticket color

- **Typ**: ticket display
- **Beschreibung**: Legt die Hintergrundfarbe des Ticket-Icons in der Ticketliste und im Ticket fest.
- **Werte**:
  - *true* / *false*: Das Feld muss in den Sortierten Listen vorhanden sein, wo Listen, Werte und Farben definiert werden.

accuracy

- **Typ**: validation
- **Beschreibung**: Für Ticket-, Kunden und Ressourcenfelder des Typs DATUM, um den Detailgrad der angezeigten Daten zu definieren
- **Werte**:
  - *date* (Standardwert): Datum wird ohne Zeit angezeigt.
  - *date-time*: Datum wird mit Zeit angezeigt.
  - *only-time*: Es wird nur die Zeit nicht aber das Datum angezeigt.

format

- **Typ**: validation
- **Beschreibung**: Verwendet, um das Format von Datumsfeldern zu validieren.
- **Werte**:
  - **<Datumsformat>**: Das Muster für das Datum basiert auf `SimpleDateFormat`, z. B. dd.MM.yyyy.
    Denken Sie daran, den richtigen Wert für `colspan` einzusetzen, wenn das Format Stunden/Minuten beinhaltet. Eine Formatreferenz finden Sie in http://docs.oracle.com/javase/6/docs/api/java/text/SimpleDateFormat.html.

maxLength

- **Typ**: validation
- **Beschreibung**: Definiert die maximale Länge der Eingabe in Datenfelder des Typs STRING.
- **Werte**:
  - **<Zahl>**: Kann für Datenfelder des Typs STRING verwendet werden.

minLength

- **Typ**: validation
- **Beschreibung**: Definiert die minimale Länge der Eingabe in Datenfelder des Typs STRING.
- **Werte**:
  - **<Zahl>**: Kann für Datenfelder des Typs STRING verwendet werden.

required

- **Typ**: validation
- **Beschreibung**: Gibt an, dass es sich um ein Pflichtfeld handelt.
- **Werte**:
  - *true* / *false*: Das Feld ist ein Pflichtfeld, wenn der Wert auf „true“ gesetzt ist. Der Benutzer kann das Ticket nicht speichern, ohne einen Wert in das Pflichtfeld eingegeben zu haben. Im Web Client sind Pflichtfelder mit einem roten Sternchen gekennzeichnet.

C.2 Verwaltung von Sortierten Listen

In diesem Kapitel werden folgende Themen behandelt:

C.2.1 Einleitung 138

C.2.2 Verwaltung von Sortierten Listen mit dem Admin Tool 140

C.2.1 Einleitung

Sortierte Listen, auch Enums genannt, sind Listen mit vordefinierten Listenwerten. Sie definieren eine Sortierte Liste im Navigationselement *Sortierte Listen* der Navigationsgruppe *Tickets*. Sortierte Listen werden einmal definiert und können an mehreren Stellen verwendet werden:

- als Auswahlliste (in Drop-down-Menüs) für Ticketfelder, Kundenfelder oder Ressourcenfelder des Typs ENUM
- als hierarchische Liste für Ticketfelder, Kundenfelder oder Ressourcenfelder des Typs MLA (Multi Level Attributes, siehe Abschnitt MLA-Verwaltung)
- als abhängige sortierte Listen, d. h. Sortierte Listen, die eine Hierarchie bilden, eine Datenstruktur, die mithilfe von Skripte des Typs Abhängige sortierte Liste realisiert wird.

> In der Verwaltung der Sortierten Listen definieren Sie nur die Listen, d. h. die Strukturen mit verschiedenen Listenwerten. Um diese Listen im Web Client anzuzeigen (als Werte von Ticketfeldern, Kundenfeldern oder Ressourcenfeldern), müssen Sie einen der folgenden Schritte ausführen:
>
> - ein Ticketfeld des Typs ENUM erstellen und mit der Sortierten Liste verknüpfen.
> - ein Kundenfeld des Typs ENUM erstellen und mit der Sortierten Liste verknüpfen.
> - ein Ressourcenfeld des Typs ENUM erstellen und mit der Sortierten Liste verknüpfen.
> - ein MLA erstellen, das automatisch als Ticketfeld mit der Ticketfeldgruppe, als Kundenfeld mit der Kundenfeldgruppe oder als Ressourcenfeld mit der Ressourcenfeldgruppe verknüpft wird, die beim Einrichten des MLAs angegeben wurde.
> - ein Skript für abhängige sortierte Listen erstellen und einer Ticketfeldgruppe, Kundenfeldgruppe oder Ressourcenfeldgruppe zuweisen.

Beispiele:

Eine Liste mit Ländernamen (Deutschland, Italien, Frankreich usw.) wird im Kundenfeld *Land* verwendet, das zu einem Datensatz mit Adressen gehört. Dieselbe Liste kann in den Ticketdaten als Ticketfeld *Standort der Maschine* und in weiteren Datenfeldern verwendet werden. Ein anderes typisches Beispiel sind Listen mit Prioritäten (hoch, normal, niedrig usw.).

Je nach Wert der Annotation `enum-type` wird das Listenfeld im Web Client folgendermaßen angezeigt:

- **Drop-down-Menü**
  enum-type nicht gesetzt oder enum-type = select

- **Radio-Buttons**
  enum-type = radio
- **Autovervollständigen-Liste**
  enum-type = autocomplete

Wenn die Annotation `enum-type` nicht gesetzt ist, wird standardmäßig ein Drop-down-Menü angezeigt (siehe Beispiel in der folgenden Abbildung).

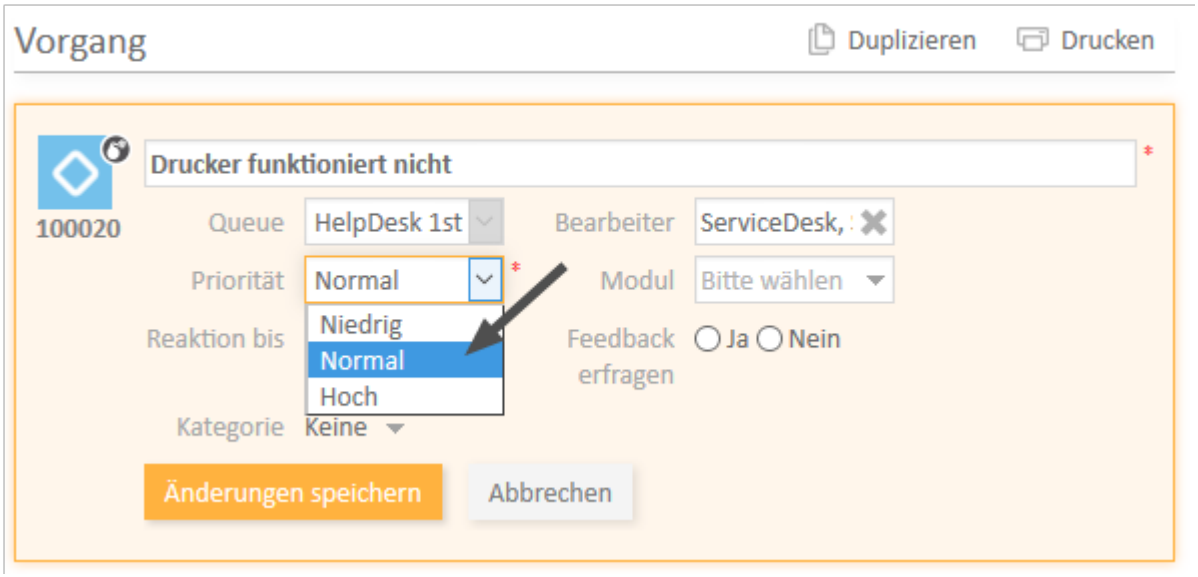


Abbildung 64: *ConSol CM Web Client - Sortierte Liste für die Priorität (die lokalisierten Listenwert werden als Listenelemente angezeigt)*

C.2.2 Verwaltung von Sortierten Listen mit dem Admin Tool

Die folgende Abbildung zeigt die verfügbaren Sortierten Listen. Sie erreichen diese Seite, indem Sie das Navigationselement *Sortierte Listen* der Navigationsgruppe *Listen* öffnen.

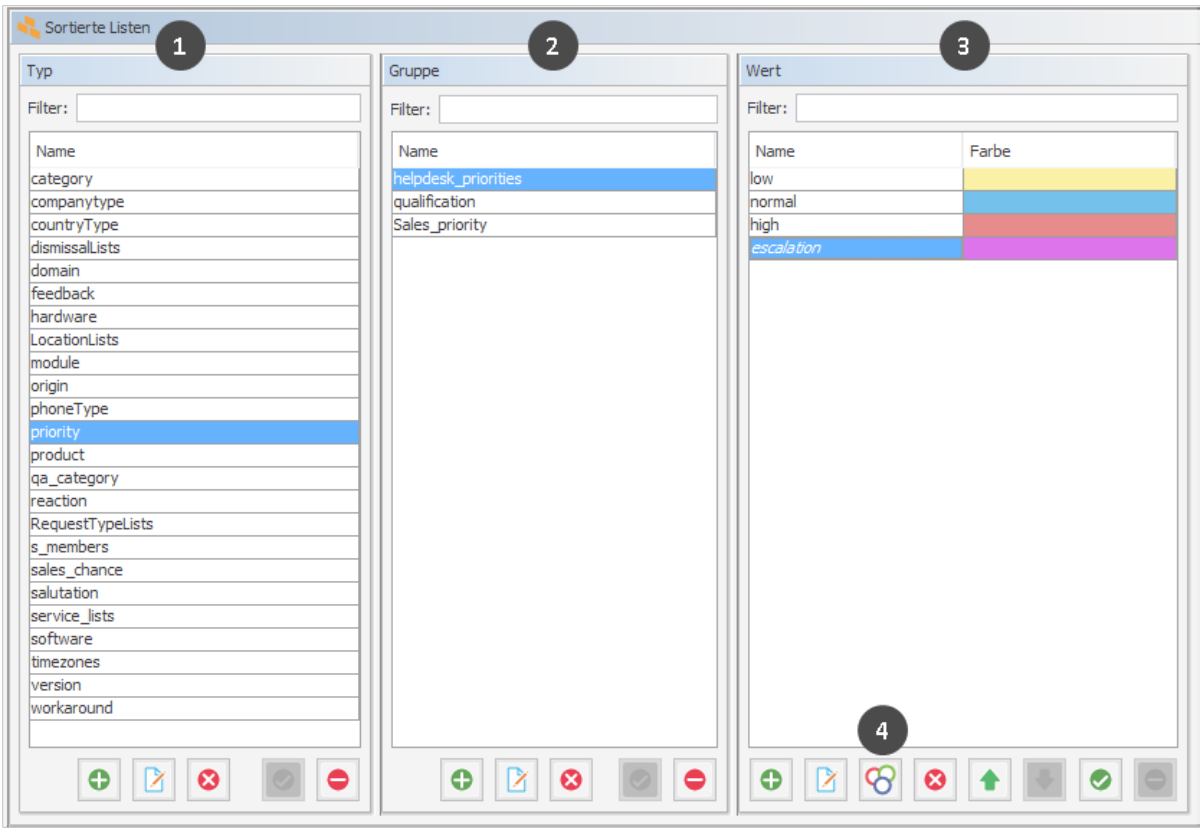


Abbildung 65: *ConSol CM Admin Tool - ConSol CM Admin Tool - Listen, Sortierte Listen*

Sortierte Listen werden auf drei Ebenen organisiert:

- **Typ (1)**
  Der Typ hilft Ihnen bei der Organisation Ihrer Listen im Admin Tool. Sein Name wird nie im Web Client angezeigt und hat keinerlei weitere Auswirkungen.
- **Gruppe (2)**
  Die Gruppe stellt eine Gruppe mit Listenwerten dar, d. h. die eigentliche Liste.
- **Wert (3)**
  Der Wert stellt einen Einzelwert innerhalb der Liste dar. Sie können für jeden Wert eine Farbe aussuchen (4), die in der definierten Queue als Hintergrundfarbe für das Ticket-Icon verwendet wird. Mit den Pfeilen nach oben und nach unten können Sie die Position des Wertes innerhalb der Liste im Web Client festlegen.

C.2.2.1 Listentypen

Erstellen eines Listentyps

Klicken Sie zum Erstellen eines neuen Listentyps einfach unter der Liste im Bereich *Typ* auf der linken Seite des Fensters auf den Button *Hinzufügen* Es wird folgendes Pop-up-Fenster geöffnet.

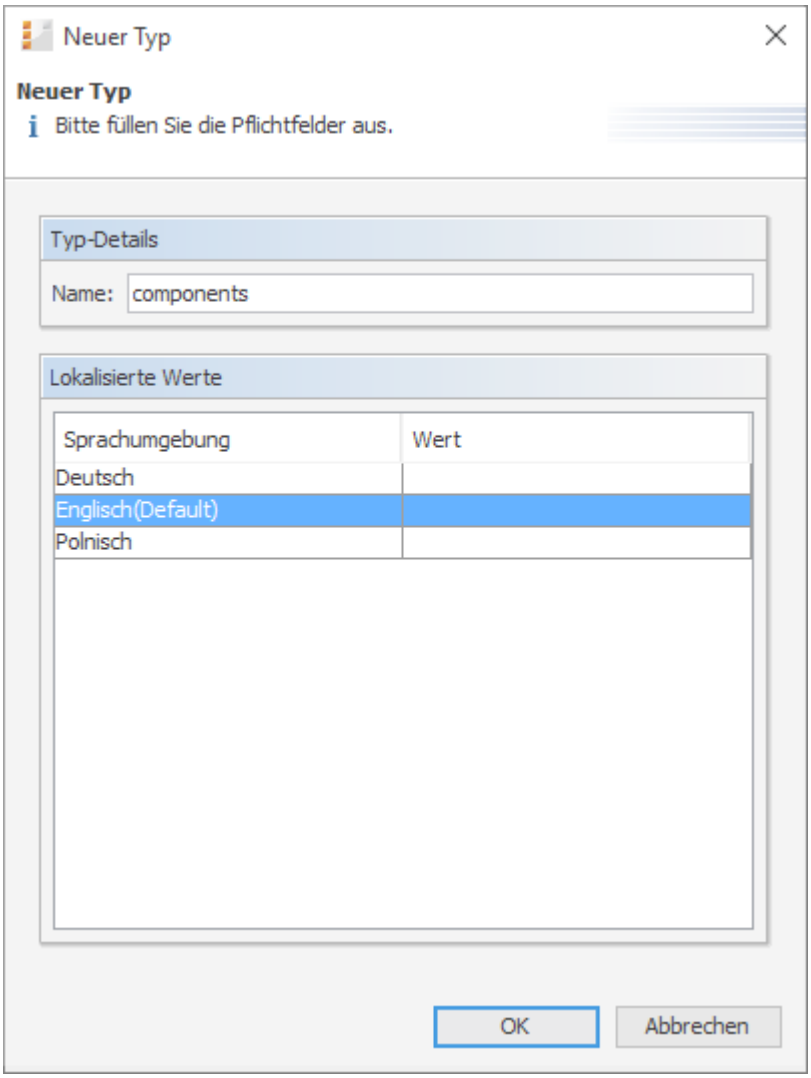


Abbildung 66: *ConSol CM Admin Tool - Listen, Sortierte Listen: Erstellen eines Listentyps*

- **Name:**
  Geben Sie den Namen für den neuen Listentyp ein. Der Name muss eindeutig sein.
- **Lokalisierte Werte:**
  Geben Sie im Feld *Wert* in jeder zusätzlichen Sprache den Namen der Typs ein. Details dazu finden Sie im Abschnitt Lokalisierung von Objekten allgemein, Typ 2.

Klicken Sie auf *OK*, um den Listentyp zu erstellen und das Fenster zu schließen.

Editieren eines Listentyps

Wenn Sie einen Listentyp editieren möchten, wählen Sie ihn in der Liste aus und klicken Sie auf den Button *Bearbeiten*. Es wird dasselbe Fenster geöffnet wie für die Erstellung eines Listentyps. Sie können alle Felder ändern und Ihre Änderungen durch Klicken auf *OK* speichern.

Löschen eines Listentyps

Ein Listentyp kann nur gelöscht werden, wenn es keine Listengruppen für ihn mehr gibt. Sie müssen also alle zu diesem Typ gehörigen Gruppen zuerst entweder löschen oder einem anderen Typ zuweisen. Um einen Listentyp zu löschen, wählen Sie ihn in der Liste aus und klicken Sie auf den Button *Löschen*. Wenn Sie das daraufhin angezeigte Fenster mit *Ja* bestätigen, wird der Typ aus der Liste und aus dem System gelöscht.

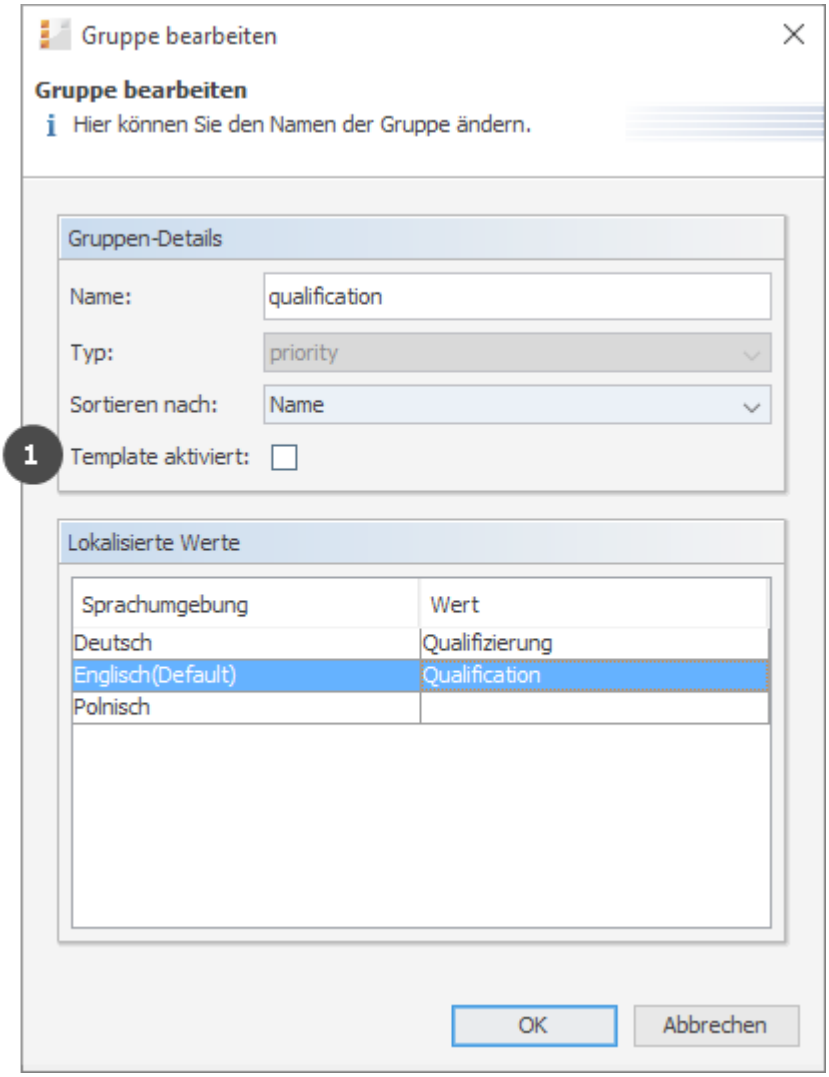
Aktivieren oder Deaktivieren eines Listentyps

Wenn Sie einen Listentyp nicht löschen können, oder ihn für den Fall, dass er später noch einmal benötigt wird, nicht löschen möchten, können Sie ihn deaktivieren. Wählen Sie dazu den Typ aus und klicken Sie auf den Button *Deaktivieren*. Der entsprechende Listeneintrag wird danach kursiv angezeigt. Klicken Sie unter der Typenliste einfach auf den Button *Aktivieren*, um den Typ wieder zu aktivieren.

C.2.2.2 Listengruppen

Erstellen einer Listengruppe

Eine Listengruppe stellt eine Liste dar, d. h. eine Listengruppe ist eine Sammlung von Listenwerten. Alle Gruppen eines Listentyps (d. h. alle Listen, die zu diesem Typ gehören) werden im mittleren Bereich des Fensters *Sortierte Listen* erstellt und gepflegt. Wählen Sie zum Erstellen einer neuen Listengruppe auf der linken Seite den gewünschten Typ aus und klicken Sie dann unter dem Bereich *Gruppe* auf den Button *Hinzufügen*. Es wird folgendes Pop-up-Fenster geöffnet.



ConSol CM Admin Tool - Listen, Sortierte Listen: Erstellen einer Listengruppe

- **Name:**
  Geben Sie den Namen für die neue Listengruppe ein. Der Name muss eindeutig sein.
- **Typ:**
  In diesem Feld wird der für die Gruppe ausgewählte Listentyp angezeigt. Sie können auch einen anderen Typ in der Auswahlliste wählen, z. B. wenn Sie die Gruppe einem anderen Typ zuweisen möchten.
- **Sortieren nach:**
  Hier können Sie festlegen, wie die Werte der Gruppe sortiert werden sollen:
  - **Benutzerdefiniert**
    Sie können die Reihenfolge mit den Pfeilsymbolen unter der Werteliste bestimmen.
  - **Name**
    Die Werte werden bei der Erstellung alphabetisch sortiert.

- **Template aktiviert (1):**
  Wenn der Wert auf „true“ gesetzt ist (Checkbox markiert), wird die Sortierte Liste als *Enum-Parameter* im Textvorlagen-Manager verwendet. Der Wert ist nur für dieses Modul relevant. Wenn Sie nicht sicher sind, welchen Wert die Checkbox haben soll, lassen Sie sie leer. Siehe auch Abschnitt Der ConSol CM Textvorlagen-Manager, String- und Enum-Parameter.
- **Lokalisierte Werte:**
  Geben Sie im Feld *Wert* in jeder zusätzlichen Sprache den Namen der Gruppe ein. Details dazu finden Sie im Abschnitt Lokalisierung von Objekten allgemein, Typ 2.

Klicken Sie auf *OK*, um die Listengruppe zu erstellen und das Fenster zu schließen.

Editieren einer Listengruppe

Wenn Sie eine Listengruppe editieren möchten, wählen Sie sie in der Liste aus und klicken Sie auf den Button *Bearbeiten*. Es wird dasselbe Fenster geöffnet wie für die Erstellung einer Listengruppe Sie können alle Felder ändern und Ihre Änderungen durch Klicken auf *OK* speichern.

Löschen einer Listengruppe

Eine Listengruppe kann nur gelöscht werden, wenn sie nicht in Tickets oder MLAs verwendet wird. Um eine Gruppe zu löschen, wählen Sie sie in der Liste aus und klicken Sie auf den Button *Löschen*. Wenn Sie das daraufhin angezeigte Fenster mit *Ja* bestätigen, wird die Gruppe aus der Liste und aus dem System gelöscht.

Aktivieren oder Deaktivieren einer Listengruppe

Eine Listengruppe kann nicht deaktiviert werden, wenn sie noch in MLAs verwendet wird.

Wenn Sie eine Listengruppe nicht löschen können, oder für den Fall, dass sie später noch einmal benötigt wird, nicht löschen möchten, können Sie sie deaktivieren. Wählen Sie dazu die Gruppe aus und klicken Sie auf den Button *Deaktivieren*. Der entsprechende Listeneintrag wird danach kursiv angezeigt. Klicken Sie unter der Gruppenliste einfach auf den Button *Aktivieren*, um die Gruppe wieder zu aktivieren.

C.2.2.3 Listenwerte

Erstellen eines Listenwerts

Die einzelnen Werte einer Listengruppe (d. h. die Listenwerte) werden auf der rechten Seite des Fensters erstellt. Wählen Sie die gewünschte Gruppe aus und klicken Sie unter dem Bereich *Wert* auf *Hinzufügen*. Es wird folgendes Pop-up-Fenster geöffnet.

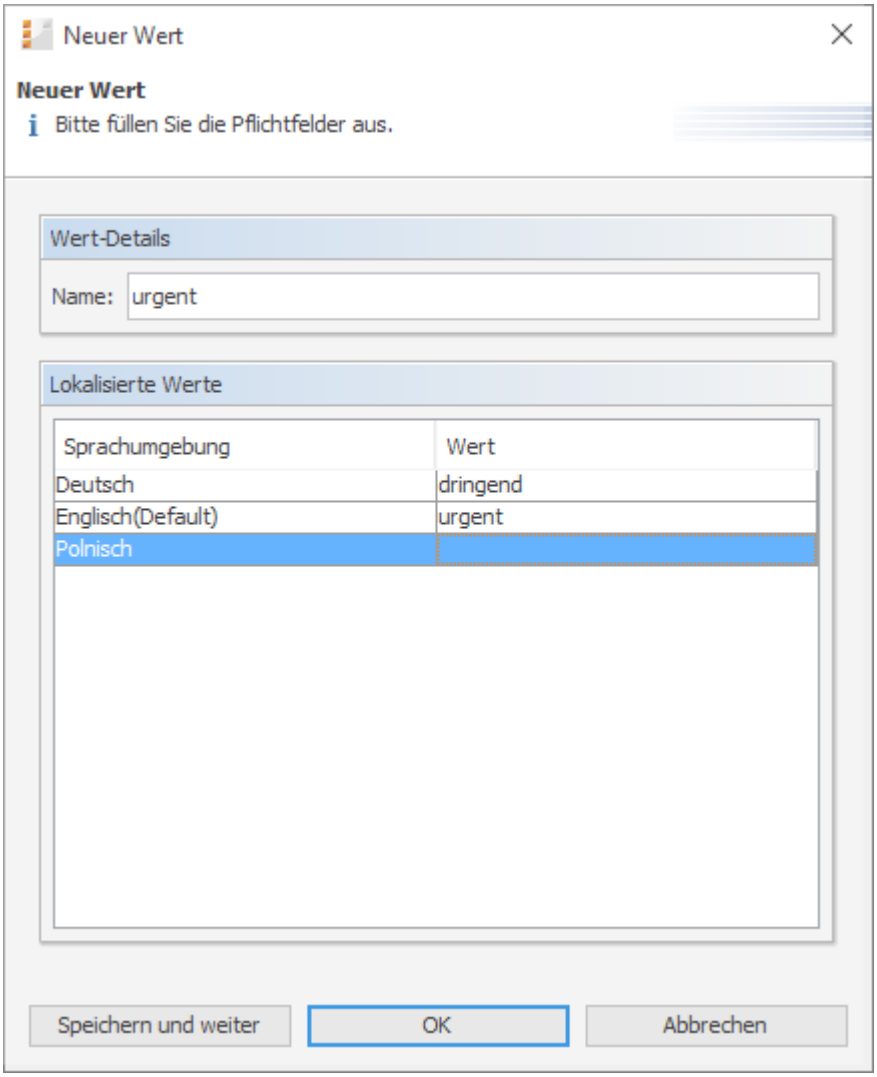


Abbildung 67: *ConSol CM Admin Tool - Listen, Sortierte Listen: Erstellen eines Listenwerts*

- **Name:**
  Geben Sie den Wert ein, der im Web Client in der Sortierten Liste angezeigt werden soll.
- **Lokalisierte Werte:**
  Geben Sie im Feld *Wert* in jeder verfügbaren Sprache den Namen des Werts ein. Details dazu finden Sie im Abschnitt Lokalisierung von Objekten allgemein, Typ 2.

Klicken Sie auf *Speichern und weiter*, wenn Sie weitere Werte für diese Listengruppe erstellen möchten. Um die Erstellung der Liste abzuschließen, klicken Sie auf *OK*.

Editieren eines Listenwerts

Wenn Sie einen Listenwert editieren möchten, wählen Sie ihn in der Liste aus und klicken Sie auf den Button *Bearbeiten*. Es wird ein Pop-up-Fenster geöffnet, in dem Sie den Namen und die lokalisierten Werte ändern können. Klicken Sie auf *OK*, um Ihre Änderungen zu speichern.

Festlegen einer Hintergrundfarbe

Sie können einem Listenwert eine Farbe zuweisen, indem Sie auf den Button *Farbe* klicken. Diese kann zum Beispiel für Prioritäten verwendet werden. Die Priorität eines Tickets ist dann im Web Client direkt aus der Hintergrundfarbe des Ticket-Icons ersichtlich. Damit diese Einstellung wirksam wird, muss die entsprechende Annotation gesetzt werden, siehe folgende Information.

> Beachten Sie, dass die Farbe des Ticket-Icons für eine Queue nur von einer Sortierten Liste bestimmt werden kann. Sie müssen in der Verwaltung von Ticketfeldern (Einrichten des Ticketdatenmodells) die Annotation `enum field with ticket color` dem entsprechenden Ticketfeld des Typs *enum (Sortierte Liste)* zuweisen. Sie können die Farbe des Ticket-Icons zum Beispiel in der Queue *Helpdesk* mit dem Ticketfeld *Priorität* bestimmen und für die Queue *Vertrieb* dazu das Ticketfeld *Wahrscheinlichkeit eines Vertragsabschlusses* verwenden.

Das Pop-up-Fenster enthält eine Reihe von Farben, aus denen Sie die gewünschte Hintergrundfarbe auswählen können. Klicken Sie auf die gewünschte Farbe, um sie für den ausgewählten Wert der Liste als Hintergrundfarbe festzulegen. Sie können die ausgewählte Farbe im *Vorschau*-Bereich prüfen. Klicken Sie auf *OK*, um Ihre Auswahl zu speichern. Klicken Sie auf *Zurücksetzen*, wenn Sie die zuletzt gespeicherte Farbe wiederherstellen möchten.

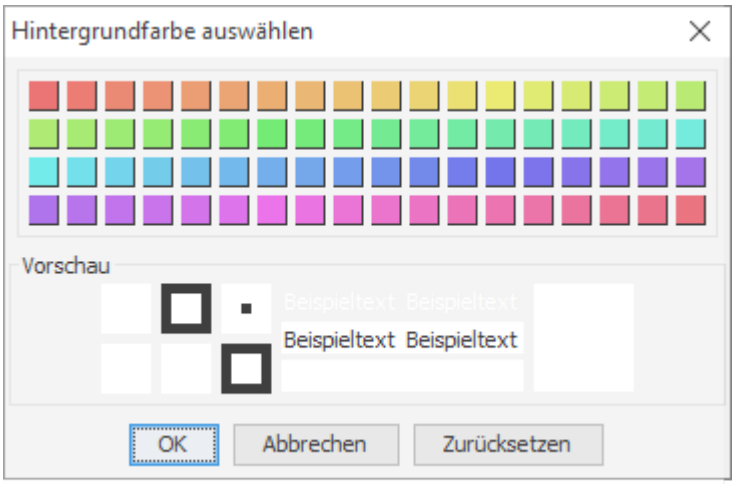


Abbildung 68: *ConSol CM Admin Tool - Listen, Sortierte Listen: Festlegen der Hintergrundfarbe für einen Listenwert*

Löschen eines Listenwerts

Ein Listenwert kann nur gelöscht werden, wenn er nicht in einem MLA verwendet wird. Um einen Wert zu löschen, wählen Sie ihn in der Liste aus und klicken Sie auf den Button *Löschen*. Wenn Sie das daraufhin angezeigte Fenster mit *Ja* bestätigen, wird der Wert aus der Liste und aus dem System gelöscht.

⚠ Vergewissern Sie sich, dass der Listenwert nicht in Workflow-Skripten referenziert wird, bevor Sie ihn löschen! Das wird vom Admin Tool nicht überprüft!

Ändern der Reihenfolge in der Werteliste

Wenn Sie im Feld *Sortieren nach* einer Listengruppe die Option *Benutzerdefiniert* ausgewählt haben, können Sie die Listenwerte mit den Pfeilsymbolen unter der Liste sortieren. Klicken Sie auf den Button *Nach oben*, um den ausgewählten Wert eine Position nach oben zu verschieben, bzw. klicken Sie auf den Button *Nach unten*, um ihn eine Position nach unten zu verschieben. Wenn Sie den Wert von *Sortieren nach* von *Benutzerdefiniert* in *Name* ändern, werden die Listenwerte automatisch in alphabetischer Reihenfolge nach Namen geordnet.

Aktivieren oder Deaktivieren eines Listenwerts

Ein Listenwert kann nicht deaktiviert werden, wenn er noch in MLAs verwendet wird.

Wenn Sie einen Listenwert nicht löschen können, oder ihn für den Fall, dass er später noch einmal benötigt wird, nicht löschen möchten, können Sie ihn deaktivieren. Wählen Sie dazu den Wert aus und klicken Sie auf den Button *Deaktivieren*. Der Listeneintrag wird danach kursiv angezeigt und der Wert ist im Web Client nicht mehr verfügbar. Klicken Sie unter der Werteliste einfach auf den Button *Aktivieren*, um den Wert wieder zu aktivieren.

C.2.2.4 Einfügen einer Sortierten Liste in das Datenmodell

Sortierte Listen können in Ticketfeldern (d. h. für Ticketdaten), Kundenfeldern (d. h. für Kundendaten) und Ressourcenfeldern (d. h. für Ressourcendaten) verwendet werden. Das nachfolgende Beispiel zeigt eine Sortierte Liste, die in den Ticketdaten eingesetzt wird.

Sortierte Listen für Ticketdaten

Um eine Sortierte Liste in ein Ticketdatenmodell einzufügen, damit sie in Queues verfügbar ist und im Web Client angezeigt wird, muss ein Ticketfeld des Typs *enum (Sortierte Liste)* definiert werden. Wählen Sie als Datentyp *enum (Sortierte Liste)* (1) und wählen Sie den Listentyp und de Listengruppe, zu der die Liste gehört (2). Eine detaillierte Beschreibung der Arbeit mit Ticketfeldern finden Sie im Abschnitt Verwaltung von Ticketfeldern (Einrichten des Ticketdatenmodells).

Sortierte Listen für Kundendaten

Um eine Sortierte Liste in ein Kundendatenmodell einzufügen, damit sie im Web Client für Firmen- und Kontaktdaten verfügbar ist, muss ein Kundenfeld des Typs *enum (Sortierte Liste)* definiert werden. Wählen Sie als Datentyp *enum (Sortierte Liste)* (1) und wählen Sie den Listentyp und de Listengruppe, zu der die Liste gehört (2). Eine detaillierte Beschreibung der Arbeit mit Kundenfeldern finden Sie im Abschnitt Verwaltung von Kundenfeldern und GUI-Design für Kundendaten.

Sortierte Listen für Ressourcendaten

Um eine Sortierte Liste in ein Ressourcendatenmodell einzufügen, damit sie in Ressourcen verfügbar ist und im Web Client angezeigt wird, muss ein Ressourcenfeld des Typs *enum (Sortierte Liste)* definiert werden. Wählen Sie als Datentyp *enum (Sortierte Liste)* (1) und wählen Sie den Listentyp und die Listengruppe, zu der die Liste gehört (2). Eine detaillierte Beschreibung der Arbeit mit Ressourcenfeldern finden Sie im Abschnitt CM/Resource Pool - Einrichten des Ressourcenmodells.

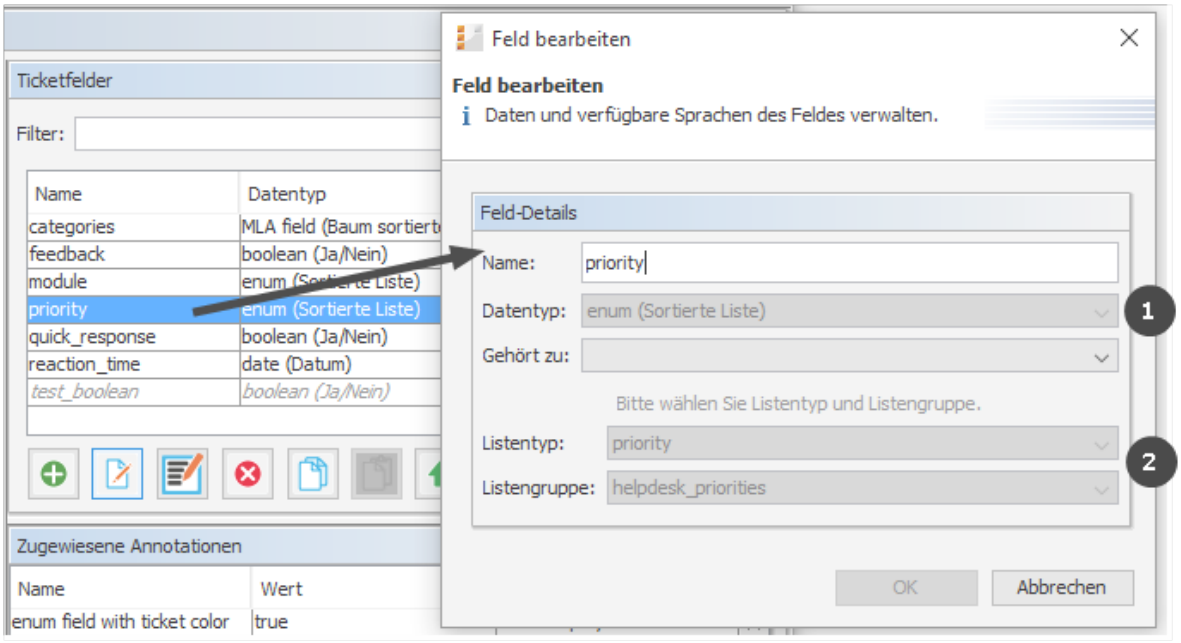


Abbildung 69: *ConSol CM Admin Tool - Tickets, Ticketfelder: Definition eines Ticketfeldes des Typs enum (Sortierte Liste)*

C.3 MLA-Verwaltung

In diesem Kapitel werden folgende Themen behandelt:

C.3.1 Einleitung 149

C.3.2 MLA-Verwaltung mit dem Admin Tool 151

C.3.1 Einleitung

MLA ist die Abkürzung für Multi Level Attribute Mit einem MLA können hierarchische Datensätze abgebildet werden, die aus mehreren Listen bestehen, die eine Baumstruktur bilden. Jedes Element der Liste kann zur nächsten Ebene führen, wobei der Name des Elements der Name der untergeordneten Liste ist. Ein MLA kann für Ticketdaten, Kundendaten oder Ressourcendaten verwendet werden.

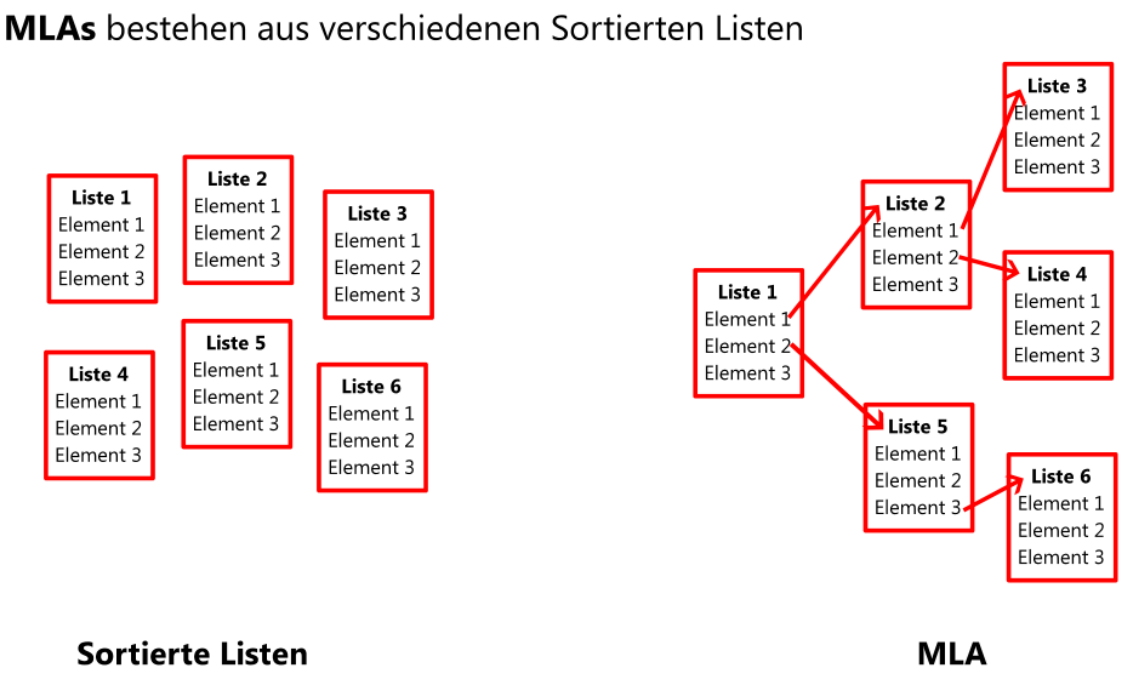


Abbildung 70: *ConSol CM Admin Tool - Prinzip des Aufbaus eines MLA*

Beispiel:

Für das Qualitätsmanagement möchten Sie bestimmte Hardware- oder Softwareprodukte in einem Ticket angeben. Dafür können Sie ein MLA mit dem Namen *QA_MLA* erstellen. Im nächsten Schritt erstellen Sie die erste Ebene mit den Elementen *Hardware* und *Software*. Für jedes Element einer Ebene können Sie weitere Ebenen erstellen, z. B. *Grafikkarte*, *Bildschirm* und *Hauptplatine* für das Element *Hardware* usw. Die folgende Abbildung zeigt so ein MLA im Web Client.

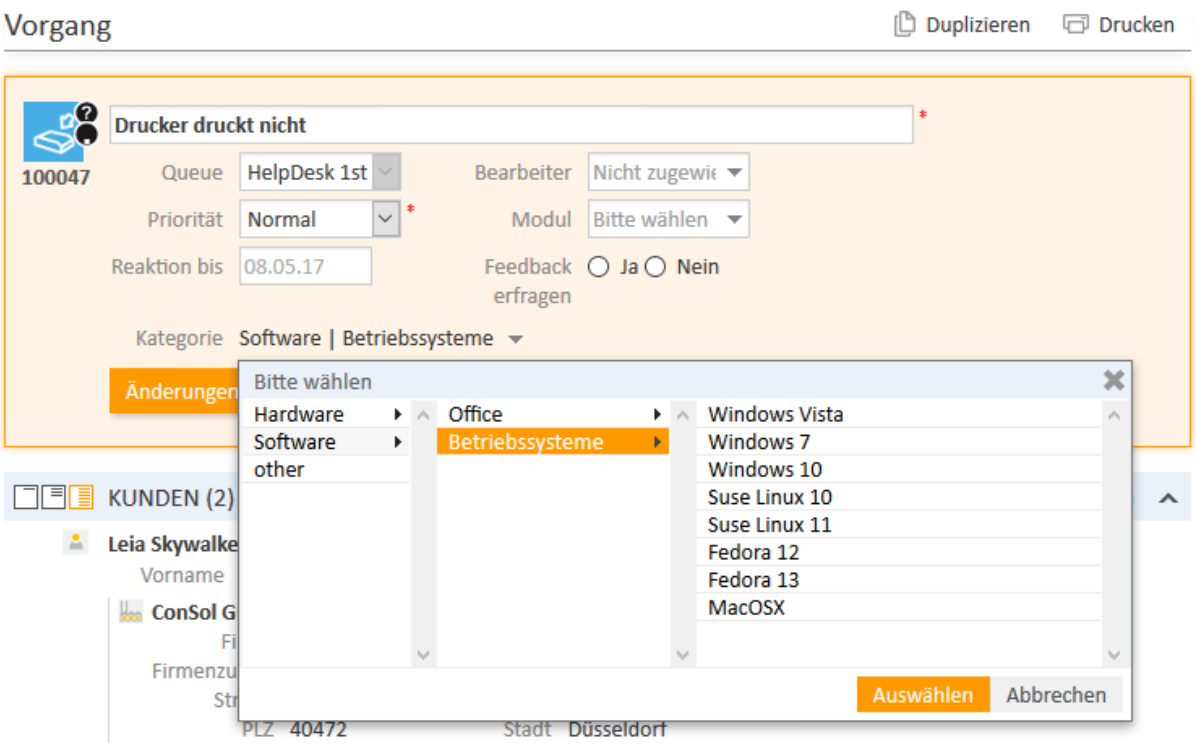


Abbildung 71: *ConSol CM Web Client - MLA für Auswahl von Hardware und Software*

Die Sortierten Listen für die einzelnen MLA-Ebenen ...

- können in der Verwaltung von Sortierten Listen erstellt werden und bei der Definition einer neuen MLA-Ebene lediglich referenziert werden.
- können in der MLA-Verwaltung beim Einrichten eines neuen MLAs neu erstellt werden.

C.3.2 MLA-Verwaltung mit dem Admin Tool

Die folgende Abbildung zeigt die verfügbaren MLAs. Sie gelangen auf diese Seite, indem Sie das Navigationselement *MLAs* in der Navigationsgruppe *Listen* öffnen.

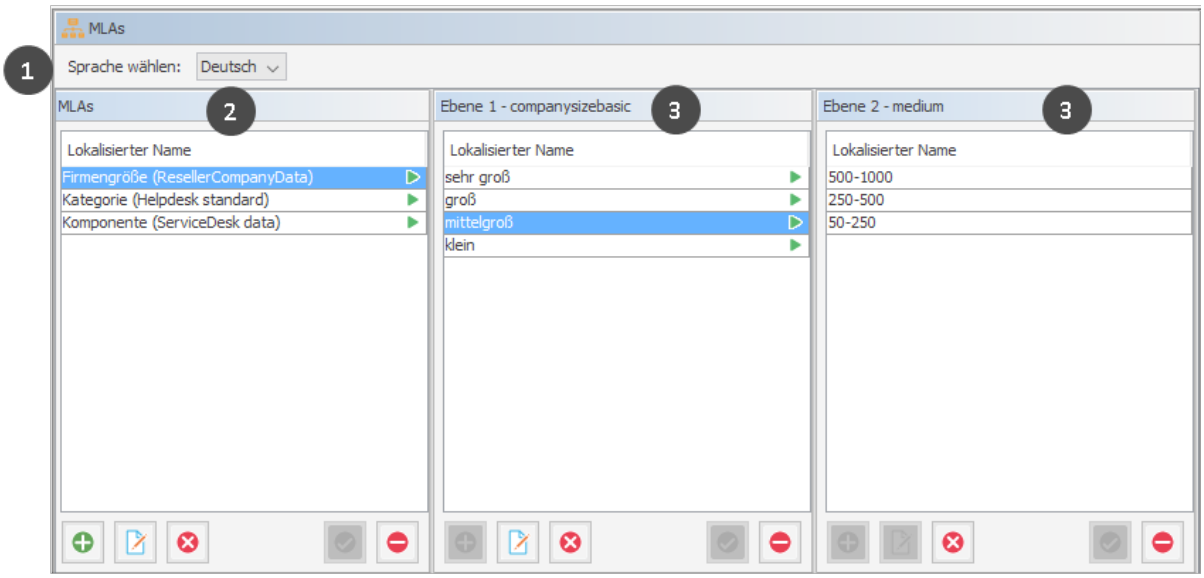


Abbildung 72: *ConSol CM Admin Tool - Listen, MLAs*

Alle Einträge werden in der ausgewählten Sprache mir ihren lokalisierten Namen angezeigt (d. h. so wie sie im Web Client zu sehen sind). Sie können die Anzeigesprache dieser Seite ändern, indem Sie im Feld *Sprache wählen* über der Liste eine andere Sprachumgebung auswählen (1). Die erste Spalte enthält die verfügbaren MLAs (2). Die folgenden Spalten enthalten die Listenwerte für die entsprechenden Ebenen (3).

C.3.2.1 Erstellen eines MLAs

Klicken Sie zum Erstellen eines MLAs unter der MLA-Liste in der linken Ecke der Seite auf den Button *Hinzufügen*. Es wird folgendes Pop-up-Fenster geöffnet.

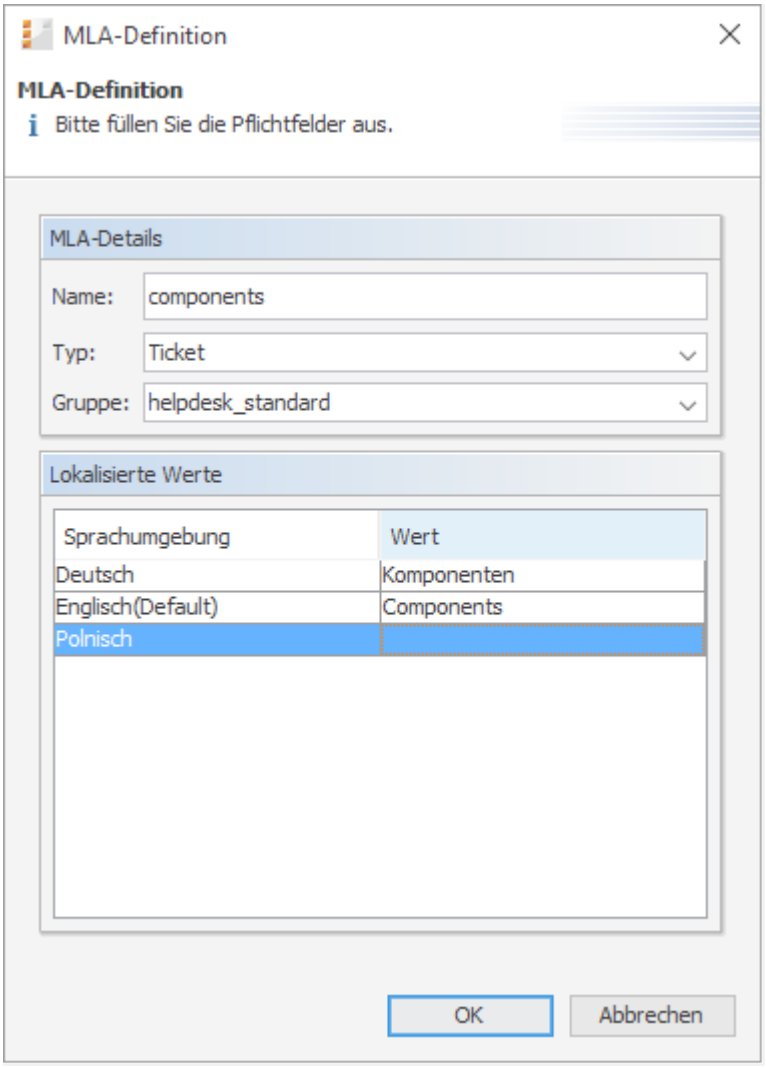


Abbildung 73: *ConSol CM Admin Tool - Listen, MLAs: Erstellen eines MLAs*

- **Name:**
  Geben Sie einen Namen für das neue MLA ein. Der Name muss eindeutig sein.
- **Typ:**
  - **Ticket**
    Das MLA wird in Ticketdaten verwendet, d. h. in einem Ticketfeld.
  - **Kundenobjekt**
    Das MLA wird in Kundendaten verwendet, d. h. in einem Kundenfeld
  - **Ressource**
    Das MLA wird in Ressourcendaten verwendet, d. h. in einem Ressourcenfeld.
- **Gruppe:**
  Wählen Sie die gewünschte Ticketfeldgruppe (Ticketdaten), Kundenfeldgruppe (Kundendaten) oder Ressourcenfeldgruppe (Ressourcendaten) im Drop-down-Feld aus. Für das neue MLA wird

automatisch ein Ticketfeld, Kundenfeld oder Ressourcenfeld vom Typ *MLA field (Baum strukturierter Listen)* in dieser Gruppe erstellt. Das ist notwendig, damit das MLA im Web Client angezeigt wird. Das Ticketfeld, Kundenfeld oder Ressourcenfeld kann, wie in den Abschnitten Verwaltung von Ticketfeldern (Einrichten des Ticketdatenmodells), Verwaltung von Kundenfeldern und GUI-Design für Kundendaten und CM/Resource Pool - Einrichten des Ressourcenmodells, annotiert werden.

- **Lokalisierte Werte:**
  Geben Sie im Feld *Wert* in jeder zusätzlichen Sprache den Namen des MLAs ein. Details dazu finden Sie im Abschnitt Lokalisierung von Objekten allgemein, Typ 2.

Klicken Sie auf *OK*, um die Details des neuen MLA zu speichern.

Sie können auch zuerst das Ticketfeld, Kundenfeld oder Ressourcenfeld für das MLA erstellen. In diesem Fall finden Sie den lokalisierten Namen des Ticketfeldes, Kundenfeldes oder Ressourcenfeldes bereits in der Liste der verfügbaren MLAs.

Beachten Sie, dass ein ENUM innerhalb eines MLAs nur einmal verwendet werden kann. Es kann aber in anderen MLAs verwendet werden.

Erstellen einer MLA-Ebene

Nachdem Sie einen Namen und ein Ticketfeld, Kundenfeld oder Ressourcenfeld für das MLA erstellt haben, können Sie mit der Definition der Ebenen fortfahren. Wählen Sie das MLA in der Liste aus und klicken Sie unter *Ebene 1* auf den Button *Hinzufügen*. Im daraufhin angezeigten Fenster *Sortierte Listen Ebene* können Sie eine Sortierte Liste für diese Ebene angeben.

Dieses Menü ist nur bei der Erstellung eines neuen MLA verfügbar, nicht aber beim Editieren eines vorhandenen MLA.

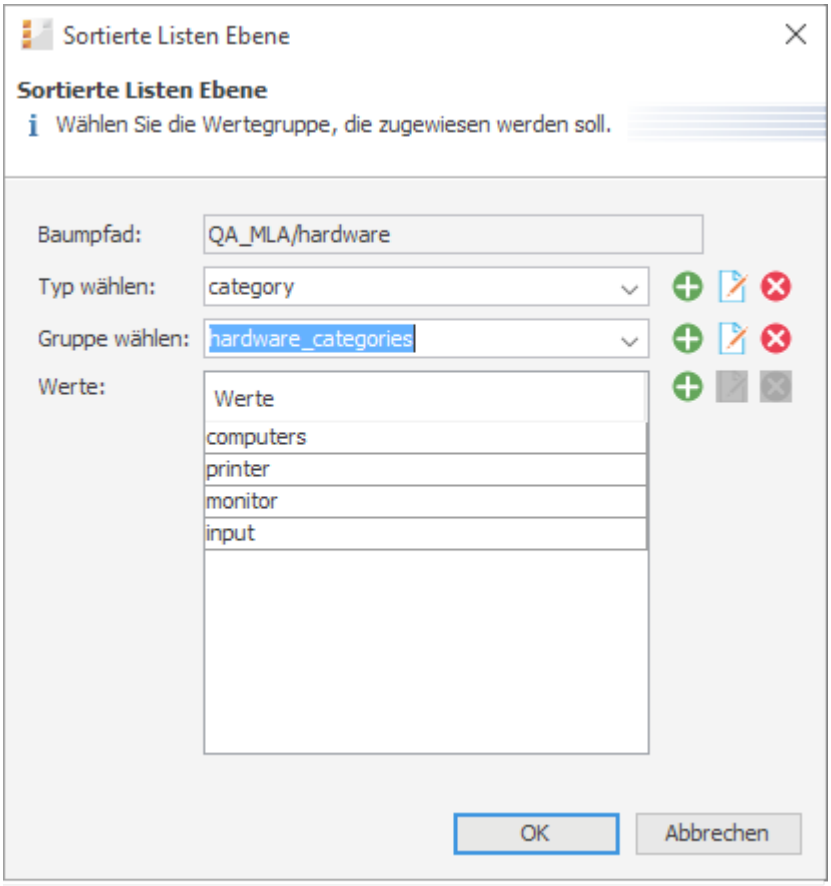


Abbildung 74: *ConSol CM Admin Tool - Listen, MLAs: Erstellen einer MLA-Ebene*

- **Baumpfad:**
  In diesem Feld wird der Baumpfad der neuen MLA-Ebene angezeigt. Damit können Sie immer sehen, an welcher Position sich die Ebene innerhalb des MLAs befindet. Dieses Feld ist schreibgeschützt.
- **Typ wählen:**
  Wählen Sie einen Listentyp aus der Liste, um die entsprechenden Listengruppen zu verwenden, die zuvor in der Verwaltung der Sortierten Listen erstellt wurden. Alternativ können Sie hier in der MLA-Verwaltung direkt einen neuen Listentyp anlegen. Dieser neue Typ wird danach auch in der Verwaltung der Sortierten Listen angezeigt.
- **Gruppe wählen:**
  Wählen Sie aus den Listen, die in der Verwaltung der Sortierten Listen zum ausgewählten Typ gehören, die gewünschte Listengruppe für diese Ebene aus. Wenn Sie im vorherigen Schritt einen neuen Listentyp erstellt haben, müssen Sie hier auch eine neue Listengruppe erstellen. Die neue Listengruppe wird danach ebenfalls in der Verwaltung der Sortierten Listen angezeigt.
- **Listenwerte**
  Das sind die Listenwerte der neuen Ebene, die im Web Client angezeigt werden. Sie können die Liste entweder ohne Änderungen übernehmen, oder Werte hinzufügen bzw. löschen. Die Änderungen sind danach sofort in der Verwaltung der Sortierten Listen sichtbar. Wenn Sie eine neue

Listengruppe erstellt haben, müssen Sie auch einen oder mehrere Listenwerte für die neue Gruppe erstellen. Sie können entweder alle benötigten Sortierten Listen erstellen, bevor Sie mit der Definition des MLA beginnen, oder während der Definition einer Ebene in der MLA-Verwaltung eine Sortierte Liste erstellen, indem Sie neben den entsprechenden Feldern auf den Button *Hinzufügen* klicken. Durch Klicken auf *Bearbeiten* oder *Löschen* können Sie Listentypen, -gruppen und -werte editieren oder löschen. Berücksichtigen Sie dabei, dass sich die Änderungen auch auf andere MLAs auswirken, in denen die gleiche Liste verwendet wird. Sie können eine Sortierte Liste nicht löschen, wenn sie bereits in einem anderen MLA verwendet wird.

Klicken Sie auf *OK*, um die neue MLA-Ebene zu erstellen und das Fenster zu schließen.

Für jeden Wert auf der Ebene können Sie wie oben beschrieben neue Ebenen hinzufügen. Wählen Sie dazu den Wert in der Liste aus und klicken Sie unten im Bereich der nächsten Ebene (rechts) auf den Button *Hinzufügen*.

Wenn Sie mit der Definition des MLAs fertig sind und feststellen, dass Sie einen zusätzlichen Wert für eine oder mehrere Ebenen benötigen, müssen Sie diesen Wert in der Verwaltung von Sortierten Listen in der entsprechenden Listengruppe hinzufügen.

Editieren eines Ebenenwerts

Wenn Sie einen Wert einer Ebene ändern möchten, wählen Sie ihn in der Liste aus und klicken Sie auf den Button *Bearbeiten*. Sie können den Objektnamen und die lokalisierten Werte ändern, sollten aber berücksichtigen, dass sich dies auf andere MLAs auswirkt, die dieselbe Liste verwenden.

Löschen einer Ebene

Eine Ebene kann nur gelöscht werden, wenn Sie nicht in einem Ticket verwendet wird. Klicken Sie zum Löschen unter der entsprechenden Ebene auf den Button *Löschen*. Wenn Sie das daraufhin angezeigte Fenster mit *Ja* bestätigen, werden die Ebene und alle von hier abhängigen Ebenen aus der Liste und aus dem System gelöscht.

Aktivieren oder Deaktivieren einer Ebene

Wenn Sie eine Ebene nicht löschen können, oder sie für den Fall, dass sie später noch einmal benötigt wird, nicht löschen möchten, können Sie sie deaktivieren. Klicken Sie dazu unter der entsprechenden Ebene auf *Deaktivieren*. Die Werte der Ebene (einschließlich der Werte der abhängigen Ebenen) werden danach kursiv angezeigt. Klicken Sie auf *Aktivieren*, um die Ebene wieder zu aktivieren.

C.3.2.2 Editieren eines MLA

Wenn Sie ein MLA editieren möchten, wählen Sie es in der Liste aus und klicken Sie auf den Button *Bearbeiten*. Es wird dasselbe Fenster geöffnet wie für die Erstellung eines MLA. Sie können alle Felder außer der Ticketfeldgruppe, der Kundenfeldgruppe bzw. der Ressourcenfeldgruppe ändern. Klicken Sie auf *OK*, um Ihre Änderungen zu speichern.

C.3.2.3 Löschen eines MLA

Sie können ein MLA nur löschen, wenn es noch nicht verwendet wurde. Wenn es schon verwendet wurde, wird eine Warnung angezeigt, dass Sie dieses MLA nur deaktivieren können (siehe unten). Um ein MLA zu löschen, wählen Sie es in der Liste aus und klicken Sie auf den Button *Löschen*. Wenn Sie

das daraufhin angezeigte Fenster mit *Ja* bestätigen, wird das MLA (und das Ticketfeld in der Verwaltung der Ticketfelder, das Kundenfeld im Kundendatenmodell oder das Ressourcenfeld im Ressourcenmodell) aus der Liste und aus dem System gelöscht.

C.3.2.4 Aktivieren oder Deaktivieren eines MLA

Wenn Sie ein MLA nicht löschen können, oder es für den Fall, dass es später noch einmal benötigt wird, nicht löschen möchten, können Sie es deaktivieren. Wählen Sie dazu das MLA aus und klicken Sie auf den Button *Deaktivieren*. Der entsprechende Listeneintrag wird danach kursiv angezeigt. Ein deaktiviertes MLA wird im Web Client nicht angezeigt. Klicken Sie unter der MLA-Liste einfach auf den Button *Aktivieren*, um das MLA wieder zu aktivieren.

C.4 Ticketprotokoll

In diesem Kapitel werden folgende Themen behandelt:

C.4.1 Einleitung 157
C.4.2 Anzeigemodi des Ticketprotokolls im Web Client 158
C.4.3 Allgemeine Informationen über die Sichtbarkeit von Ticketprotokolleinträgen im Web Client 165
C.4.4 Speicherung des Ticketprotokolls und Übertragung an das Data Warehouse (DWH) 168

C.4.1 Einleitung

Für jedes Ticket wird das Ticketprotokoll in der ConSol CM-Datenbank gespeichert, d. h. der komplette Lebenszyklus eines Tickets kann nachverfolgt werden. Dies beinhaltet Aktionen wie Workflow-Aktivitäten oder eingehende E-Mails, sowie automatische Aktionen und Systemaktionen wie die Änderung des zuständigen Bearbeiters oder die Prüfung, ob ein Zeit-Trigger feuern muss. Diese Funktion gab es in einer ähnlichen Form schon in ConSol CM Version 5.

Das Ticketprotokoll wird im Protokollbereich der Ticketseite angezeigt.

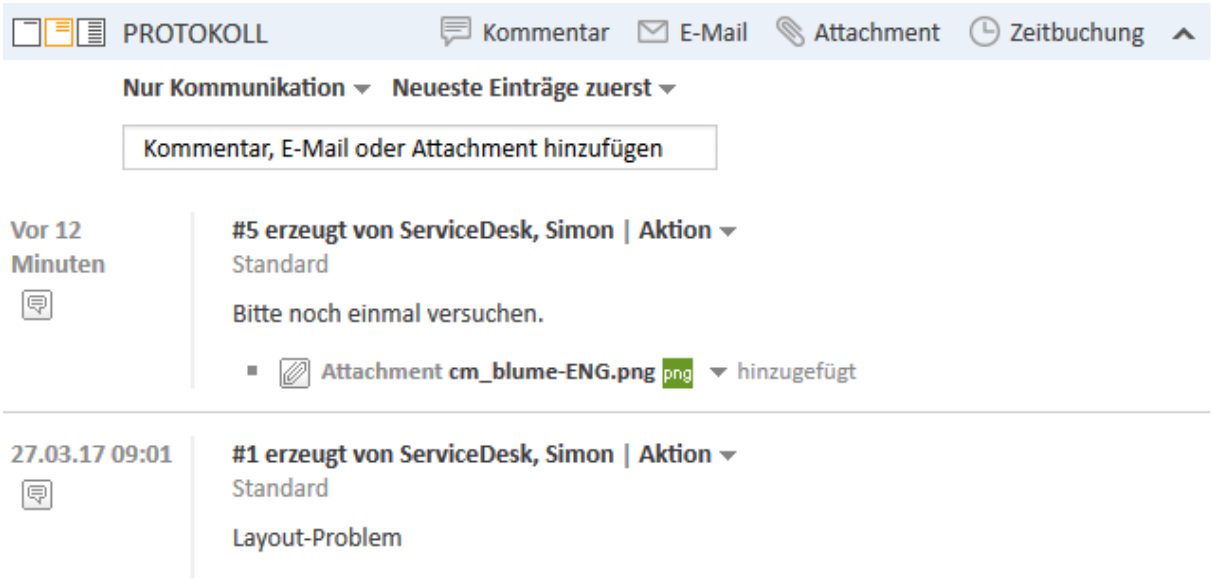


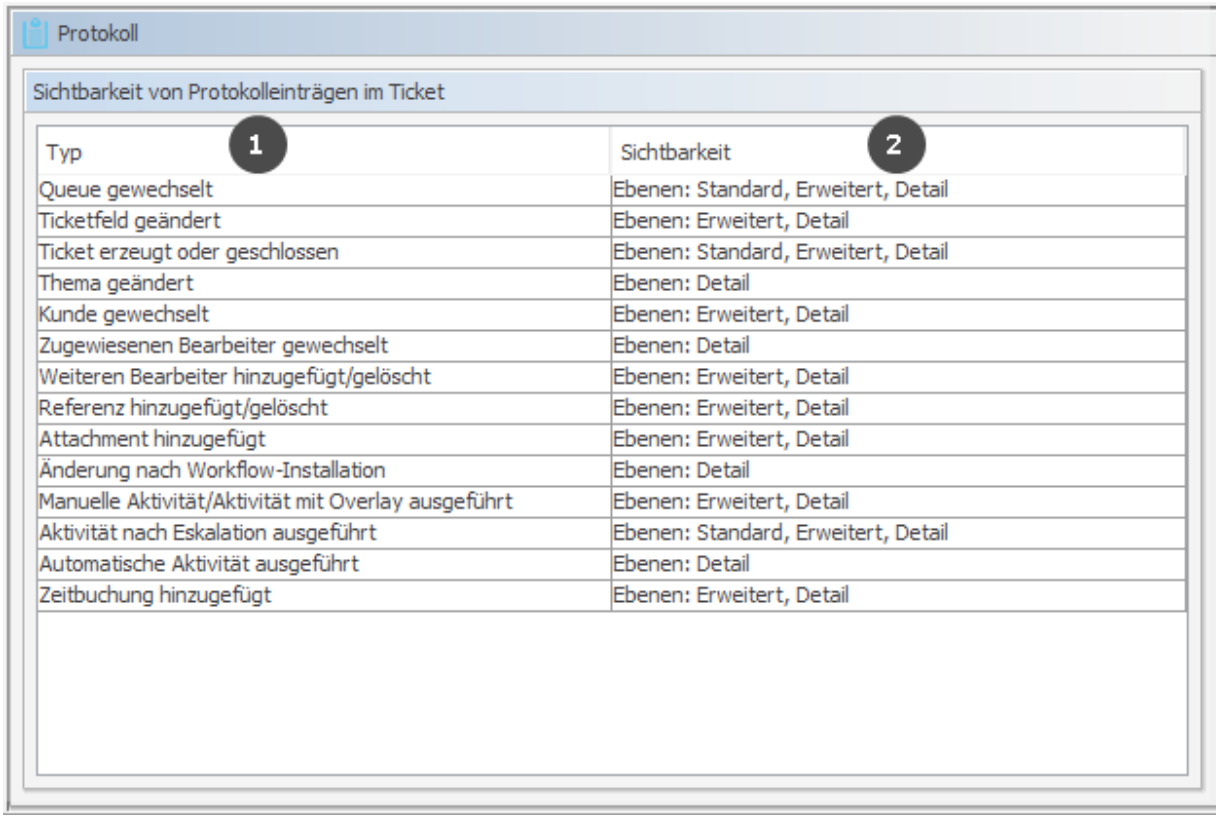
Abbildung 75: *ConSol CM Web Client - Protokollbereich*

Wenn das DWH in Betrieb ist (siehe Abschnitt Verwaltung des Data Warehouse (DWH)) können Sie einstellen, ob die Protokolldaten an das DWH übertragen werden sollen (siehe Abschnitt Speicherung des Ticketprotokolls und Übertragung an das Data Warehouse (DWH)).

C.4.2 Anzeigemodi des Ticketprotokolls im Web Client

Der Anzeigemodus für das Ticketprotokoll kann im Navigationselement *Protokoll* der Navigationsgruppe *Tickets* konfiguriert werden. In diesem Element können Sie das Sichtbarkeitslevel für die am Ticket ausgeführten Aktionen bzw. eingetretenen Events konfigurieren. Die Einträge der angegebenen Typen werden im Ticketprotokoll angezeigt, wenn der Benutzer das entsprechende Sichtbarkeitslevel ausgewählt hat. Dies ist wichtig, wenn der Anzeigemodus *Anzeigen alle Einträge* verwendet wird.

> Die Sichtbarkeit von E-Mails, Kommentaren und Attachments (d. h. ihr Sichtbarkeitslevel und ob Texteinträge vollständig oder verkürzt angezeigt werden) kann über Textklassen gesteuert werden. Die Sichtbarkeit von Einträgen, denen keine Textklasse zugewiesen ist, wird durch die Standardtextklasse gesteuert (in der Standardinstallation von ConSol CM heißt die Standardklasse für E-Mails und Kommentare *default_class* und die Standardklasse für Attachments *default_attachment_class*).



Typ	Sichtbarkeit
Queue gewechselt	Ebenen: Standard, Erweitert, Detail
Ticketfeld geändert	Ebenen: Erweitert, Detail
Ticket erzeugt oder geschlossen	Ebenen: Standard, Erweitert, Detail
Thema geändert	Ebenen: Detail
Kunde gewechselt	Ebenen: Erweitert, Detail
Zugewiesenen Bearbeiter gewechselt	Ebenen: Detail
Weiteren Bearbeiter hinzugefügt/gelöscht	Ebenen: Erweitert, Detail
Referenz hinzugefügt/gelöscht	Ebenen: Erweitert, Detail
Attachment hinzugefügt	Ebenen: Erweitert, Detail
Änderung nach Workflow-Installation	Ebenen: Detail
Manuelle Aktivität/Aktivität mit Overlay ausgeführt	Ebenen: Erweitert, Detail
Aktivität nach Eskalation ausgeführt	Ebenen: Standard, Erweitert, Detail
Automatische Aktivität ausgeführt	Ebenen: Detail
Zeitbuchung hinzugefügt	Ebenen: Erweitert, Detail

Abbildung 76: *ConSol CM Admin Tool - Tickets, Protokoll*

Im Bereich zum Editieren des Ticketprotokolls wird eine Liste aller konfigurierten Werte mit folgenden Angaben angezeigt:

- **Typ (1)**
  Der Typ der ausgeführten Aktion.

- **Sichtbarkeit (2)**
  Das Sichtbarkeitslevel im Web Client. Es gibt drei Level:
  - Standard (1. Level)
  - Erweitert (2. Level)
  - Detail (3. Level)

Die folgende Abbildung zeigt den Aktionstyp *Zeitbuchung hinzugefügt*, der für die Sichtbarkeitslevel *Erweitert* und *Detail* konfiguriert ist.

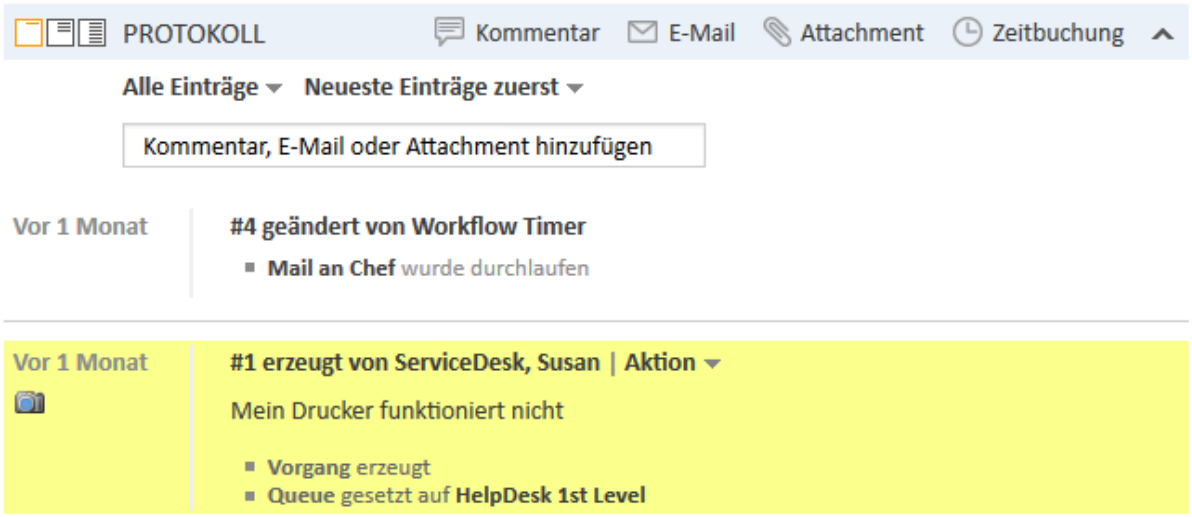


Abbildung 77: *ConSol CM Web Client - Zeitbuchungseintrag auf dem Sichtbarkeitslevel Standard nicht sichtbar*

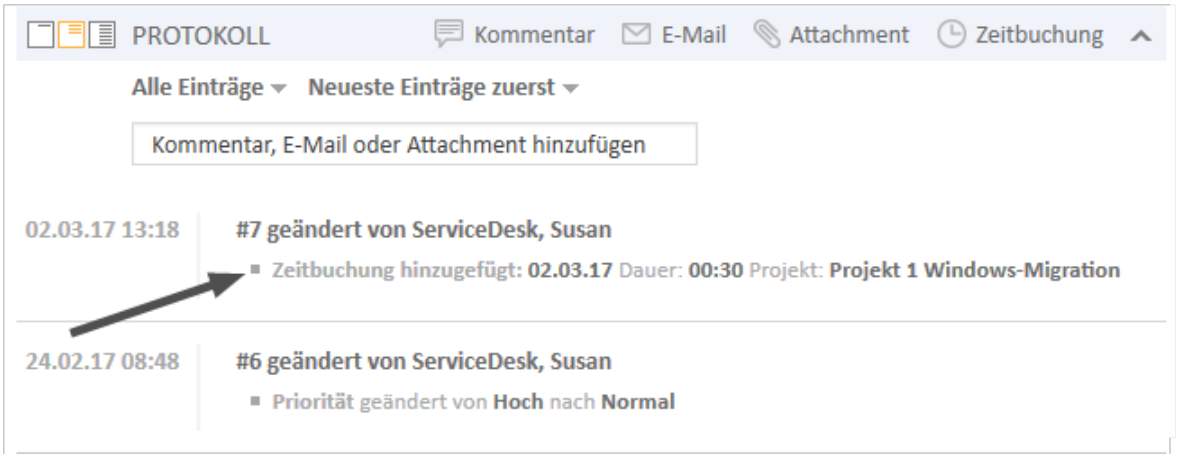


Abbildung 78: *ConSol CM Web Client - Zeitbuchungseintrag auf dem Sichtbarkeitslevel Erweitert sichtbar*

Es ist nicht möglich, neue Aktionstypen zur Liste hinzuzufügen. Um die Sichtbarkeit eines vorhandenen Eintrags zu editieren, doppelklicken Sie auf den Sichtbarkeitswert, den Sie ändern möchten, und wählen Sie im Drop-down-Menü die gewünschte Option.

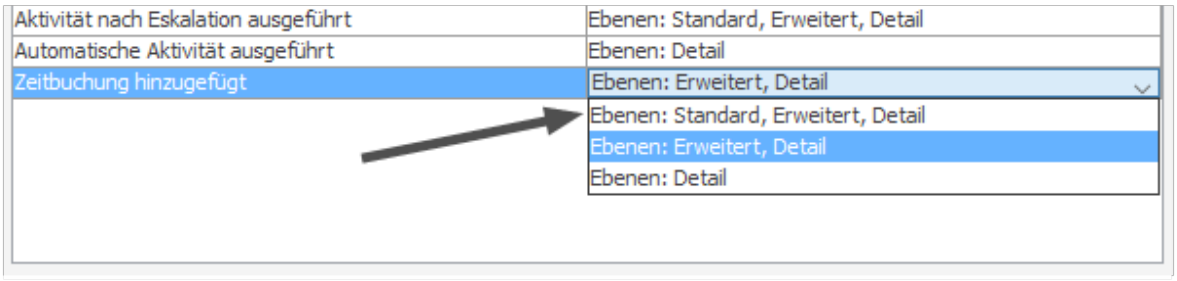


Abbildung 79: *ConSol CM Admin Tool - Tickets, Protokoll: Ändern des Sichtbarkeitslevels für einen Aktionstyp*

Die nächste Abbildung zeigt die Sichtbarkeit des Aktionstyps *Zeitbuchung hinzugefügt*, nachdem die Einstellung im Navigationselement *Protokoll* der Navigationsgruppe *Tickets* verändert wurde:

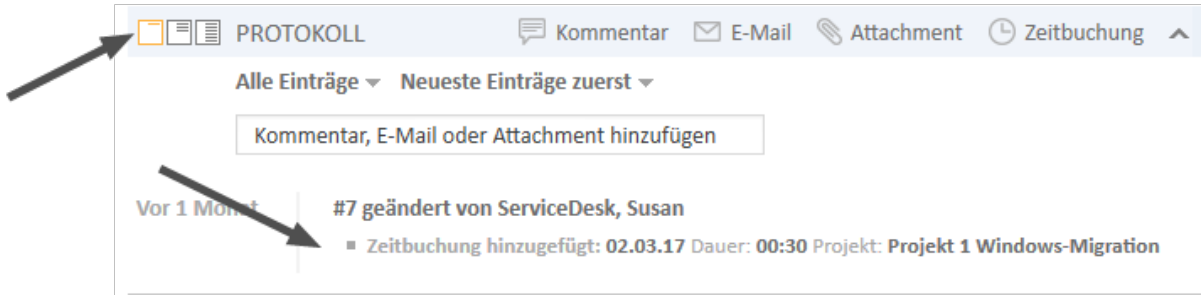


Abbildung 80: *ConSol CM Web Client - Zeitbuchungseintrag auf dem Sichtbarkeitslevel Standard sichtbar*

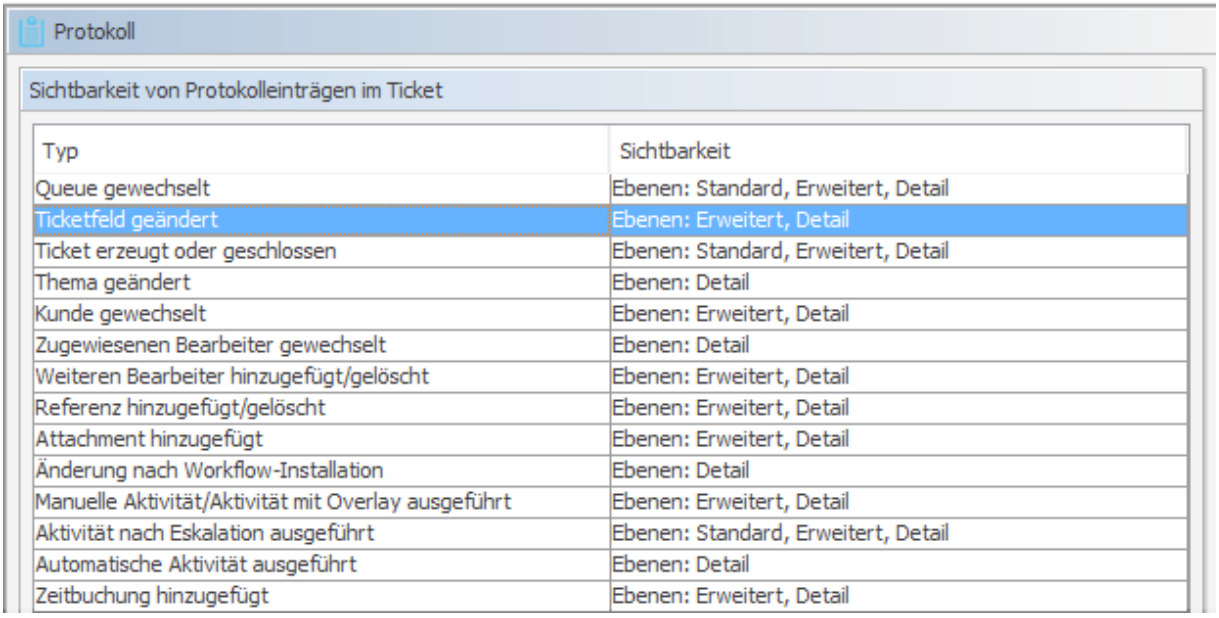
C.4.2.1 Einstellen des Anzeigemodus für Änderungen an Ticketfeldern im Ticketprotokoll

Das Sichtbarkeitslevel im Ticketprotokoll kann an zwei Stellen gesetzt werden:

- Global mit dem Parameter *Ticketfeld geändert*
- Speziell für ein oder mehrere Ticketfelder mit der Annotation `visibility configuration`

> Denken Sie daran, dass die Sichtbarkeit von Attachments nur hier konfiguriert wird (*Attachment hinzugefügt*). Textklassen (siehe Abschnitt Textklassen) werden für Attachments nicht berücksichtigt.

Im folgenden Beispiel wird die Änderung eines Ticketfeldes (d. h. das Setzen eines neuen Wertes) in den Levels *Erweitert* und *Detail* des Ticketprotokolls angezeigt. Dies ist eine systemweite Einstellung, die im Navigationselement *Protokoll* der Navigationsgruppe *Ticket* vorgenommen wird.



Typ	Sichtbarkeit
Queue gewechselt	Ebenen: Standard, Erweitert, Detail
Ticketfeld geändert	Ebenen: Erweitert, Detail
Ticket erzeugt oder geschlossen	Ebenen: Standard, Erweitert, Detail
Thema geändert	Ebenen: Detail
Kunde gewechselt	Ebenen: Erweitert, Detail
Zugewiesenen Bearbeiter gewechselt	Ebenen: Detail
Weiteren Bearbeiter hinzugefügt/gelöscht	Ebenen: Erweitert, Detail
Referenz hinzugefügt/gelöscht	Ebenen: Erweitert, Detail
Attachment hinzugefügt	Ebenen: Erweitert, Detail
Änderung nach Workflow-Installation	Ebenen: Detail
Manuelle Aktivität/Aktivität mit Overlay ausgeführt	Ebenen: Erweitert, Detail
Aktivität nach Eskalation ausgeführt	Ebenen: Standard, Erweitert, Detail
Automatische Aktivität ausgeführt	Ebenen: Detail
Zeitbuchung hinzugefügt	Ebenen: Erweitert, Detail

Abbildung 81: *ConSol CM Admin Tool - Tickets, Protokoll: Sichtbarkeitskonfiguration für Ticketfeld geändert*

Für das Ticketfeld *volume_product* wird die Sichtbarkeitskonfiguration auf *on every level* gesetzt. Auf diese Weise sind Änderungen an diesem Ticketfeld immer im Ticketprotokoll sichtbar, unabhängig von dem Sichtbarkeitslevel, das der Bearbeiter ausgewählt hat (*Standard*, *Erweitert* oder *Detail*). Diese Einstellung wird im Navigationselement *Ticketfelder* der Navigationsgruppe *Tickets* vorgenommen.

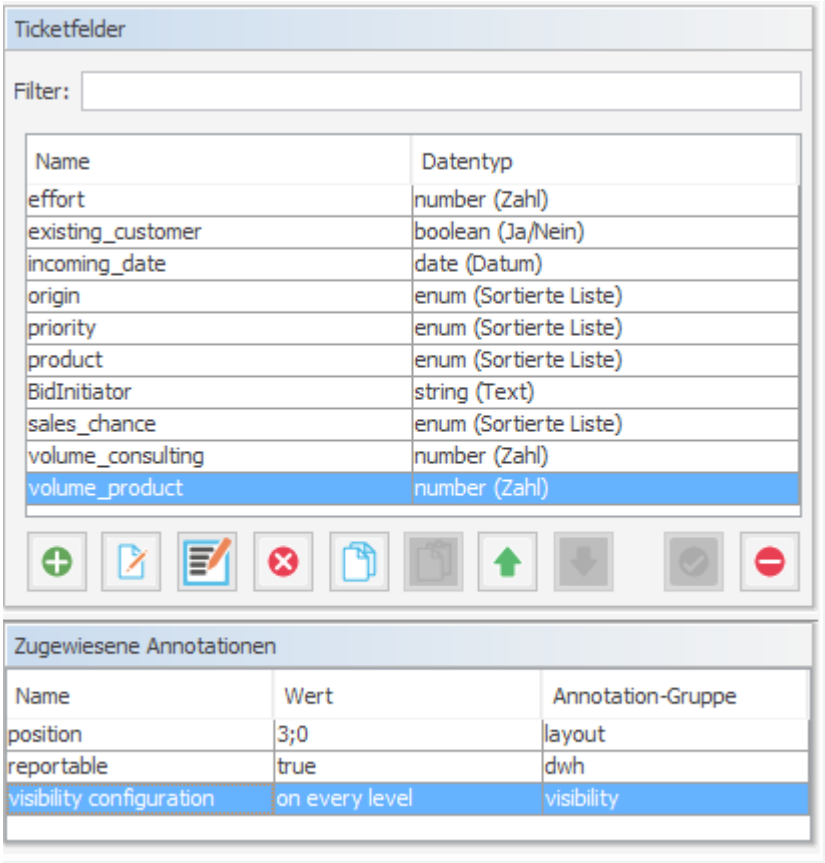


Abbildung 82: *ConSol CM Admin Tool - Tickets, Ticketfelder: Einstellen der Sichtbarkeitskonfiguration für ein bestimmtes Ticketfeld*

C.4.2.2 Ändern des Anzeigeverhaltens für die Sichtbarkeitslevel *Standard* und *Erweitert* zum Optimieren der Anzeige des Ticketprotokolls

Tickets mit einer großen Anzahl an Protokolleinträgen, z. B. Kommentaren, können so groß sein, dass ein schneller Überblick über alle Einträge nicht mehr möglich ist. Sie können die Funktion **lazy loading** verwenden (siehe Attribute der Seitenanpassung *headHistoryElementsCount* und *tailHistoryElementsCount*), um den vom Ticket benötigten Platz zu verringern.

Außerdem können Sie die Anzahl der Zeilen ändern, die für Kommentare im Sichtbarkeitslevel *Standard* und *Erweitert* angezeigt werden. Dies kann für Einträge im Ticketprotokoll geschehen, die mit einer Textklasse markiert sind, die für die beiden ersten "kurzen" Level gilt, d. h. für Text mit der Sichtbarkeit *Standard (gekürzt)* und *Erweitert (gekürzt)*.

Die Konfiguration erfolgt über die Attribute der Seitenanpassung des Typs *acimSection*, Bereich *ticketEditPage/acimSection*, siehe acimSection (Typ oder Unterbereich).

- **basicViewCharactersLimit** (vorher **standardViewCharactersLimit**)
  Für Standard (gekürzt). Definiert die Anzahl der angezeigten Zeichen, Standardwert: „150“.
- **extendedViewCharactersLimit**
  Für Erweitert (gekürzt). Definiert die Anzahl der angezeigten Zeichen, Standardwert: „350“.

Im folgenden Beispiel sind für beide Level Grenzwerte gesetzt. Diese gelten aber nur, wenn das Level in Kurzform angezeigt wird (entsprechend der Textklasse, hier: *default_class*).

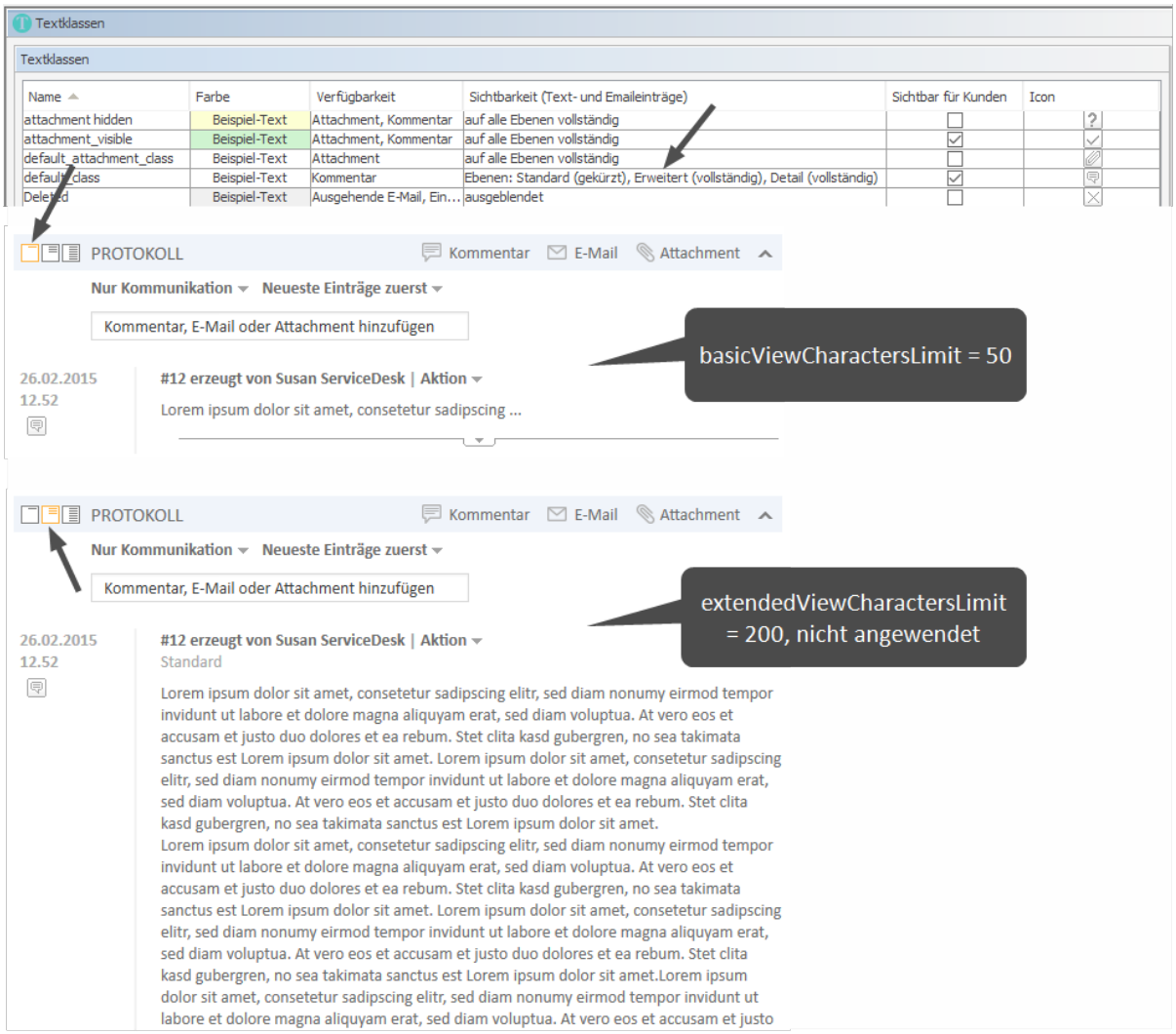


Abbildung 83: *Konfiguration der Kommentaransicht mit einer Textklasse plus Seitenpassung (ViewCharactersLimit), 1*

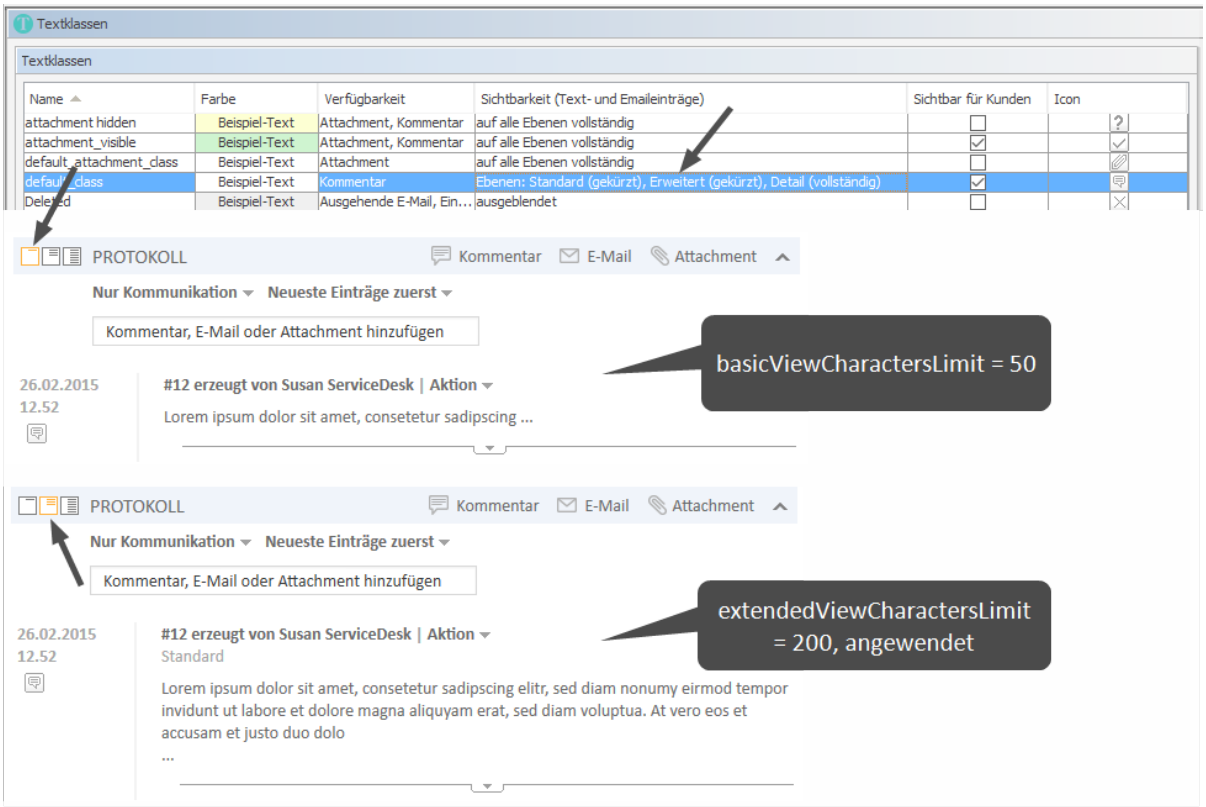


Abbildung 84: *Konfiguration der Kommentaransicht mit einer Textklasse plus Seitenpassung (ViewCharactersLimit), 2*

C.4.3 Allgemeine Informationen über die Sichtbarkeit von Ticket-protokolleinträgen im Web Client

Es gibt **zwei Sichtbarkeitseinstellungen**, mit denen die Sichtbarkeit von Ticketprotokolleinträgen konfiguriert wird:

- Der ausgewählte Anzeigemodus
- Das ausgewählte Sichtbarkeitslevel (einschließlich der zusätzlichen Angabe, ob die Einträge im ausgewählten Sichtbarkeitslevel *vollständig* oder *gekürzt* angezeigt werden sollen).

Es gibt **zwei Arten von Ticketprotokolleinträgen**, für die Sie die Sichtbarkeitseinstellungen definieren können:

- **Kommentare/Kommunikation**
  Bezieht sich auf Kommentare, E-Mails und Attachments. Ihr Sichtbarkeitslevel und ob die Texteinträge vollständig oder verkürzt angezeigt werden, kann mittels Textklassen gesteuert werden. Die Sichtbarkeit von Einträgen, denen keine Textklasse zugewiesen ist, wird durch die Standardtextklasse gesteuert (in der Standardinstallation von ConSol CM heißt die Standardklasse für E-Mails und Kommentare *default_class* und die Standardklasse für Attachments *default_attachment_class*).
- **Andere Einträge**
  Zum Beispiel Einträge darüber, dass das Ticket die Queue oder den Bereich gewechselt hat, Bearbeiter zugewiesen oder entfernt wurden, oder automatische Aktivitäten ausgeführt wurden. Eine vollständige Liste aller möglichen Typen anderer Einträge finden Sie im Admin Tool im Navigationselement *Ticketprotokoll*, in dem Sie das Sichtbarkeitslevel für diese Einträge festlegen können.

C.4.3.1 Der Anzeigemodus im Web Client

Der Anzeigemodus bestimmt, welche Arten von Ticketprotokolleinträgen generell im Ticketprotokoll angezeigt werden. Für die zwei Arten von Ticketprotokolleinträgen (Kommentare/Kommunikation und andere Einträge) gibt es auch zwei Anzeigemodi im Ticketprotokoll:

- Anzeigemodus *Anzeigen Kommunikation* (es werden nur Kommunikationseinträge angezeigt)
- Anzeigemodus *Anzeigen alle Einträge* (es werden Kommunikationseinträge und alle anderen Einträge angezeigt)

Anzeigemodus *Anzeigen Kommunikation*

Es werden nur Kommentare, E-Mails und Attachments angezeigt (bei Attachments wird nicht das ganze Attachment angezeigt, sondern ein Eintrag, dass ein Attachment hinzugefügt wurde, und ein Link, mit dem das Attachment geöffnet werden kann). Ob ein Kommunikationseintrag angezeigt wird oder nicht und in welcher Länge er angezeigt wird, hängt vom ausgewählten Sichtbarkeitslevel ab (einschließlich der zusätzlichen Angabe, ob die Einträge im ausgewählten Sichtbarkeitslevel vollständig oder gekürzt angezeigt werden sollen).

Anzeigemodus *Anzeigen alle Einträge*

Es werden neben den Kommunikationseinträgen auch alle anderen Einträge angezeigt (z. B. Einträge darüber, dass ein Ticket die Queue oder den Bereich gewechselt hat). Eine vollständige Liste aller möglichen Typen anderer Einträge finden Sie im Admin Tool im Navigationselement *Ticketprotokoll*, in dem Sie das Sichtbarkeitslevel für diese Einträge festlegen können.
Diese anderen Einträge sind nur sichtbar, wenn der Anzeigemodus *Anzeigen alle Einträge* ausgewählt wird. Wenn diese Bedingung erfüllt ist, bestimmt das ausgewählte Sichtbarkeitslevel, ob die verschiedenen Arten von Einträgen angezeigt werden. Im Gegensatz zu Kommunikationseinträgen wird für diese anderen Einträge kein Unterschied zwischen vollständiger und gekürzter Sichtbarkeit gemacht.

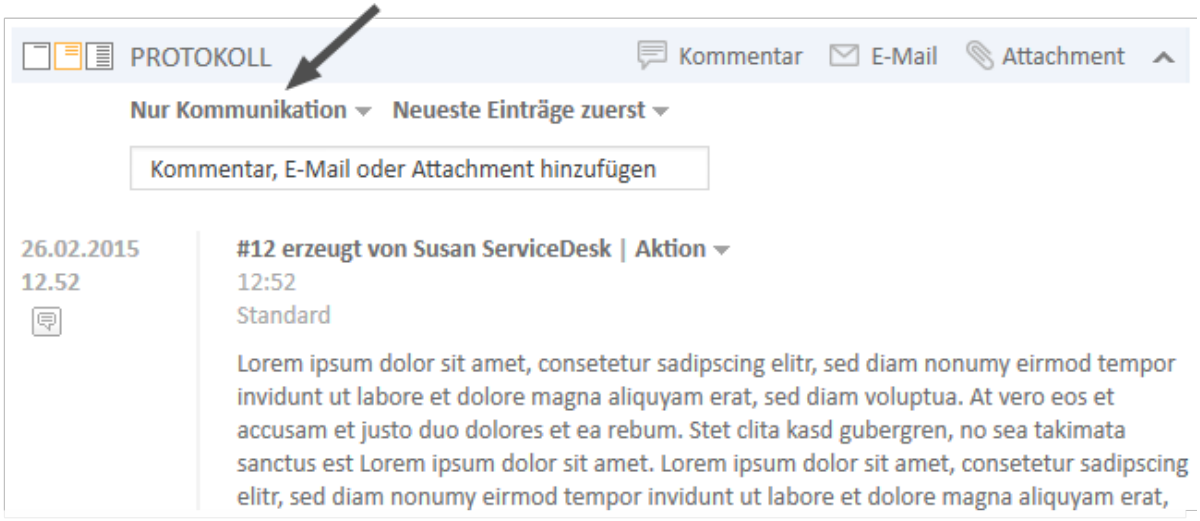


Abbildung 85: *ConSol CM Web Client - Einstellen des Anzeigemodus*

C.4.3.2 Das Sichtbarkeitslevel im Web Client

Es gibt im Web Client drei Levels für das Ticketprotokoll:

- Standard (1. Level)
- Erweitert (2. Level)
- Detail (3. Level)

Sichtbarkeitslevel für Kommunikationseinträge

Auf welchem Sichtbarkeitslevel (Standard, Erweitert oder Detail) die Kommunikationseinträge im Ticketprotokoll angezeigt bzw. ausgeblendet werden, hängt von den Textklassen ab. Sie können diese im Navigationselement *Textklassen* erstellen und verwalten. Die Sichtbarkeit von Einträgen, denen keine Textklasse zugewiesen ist, wird durch die Standardtextklasse gesteuert (in der Standardinstallation von ConSol CM heißt die Standardklasse für E-Mails und Kommentare *default_class* und die Standardklasse für Attachments *default_attachment_class*).

Zusätzlich zur allgemeinen Sichtbarkeit eines Kommunikationseintrags können Sie festlegen, ob ein Eintrag dieser Klasse in voller Länge (vollständig) angezeigt werden soll oder nach einer bestimmten Anzahl an Zeichen verkürzt werden soll (gekürzt).

Sichtbarkeitslevel für andere Einträge

Auf welchem Sichtbarkeitslevel (Standard, Erweitert oder Detail) die anderen Einträge im Ticket-protokoll angezeigt bzw. ausgeblendet werden, kann im Navigationselement Ticketprotokoll konfiguriert werden. Im Gegensatz zu Kommunikationseinträgen wird für diese anderen Einträge kein Unterschied zwischen vollständiger und gekürzter Sichtbarkeit gemacht.

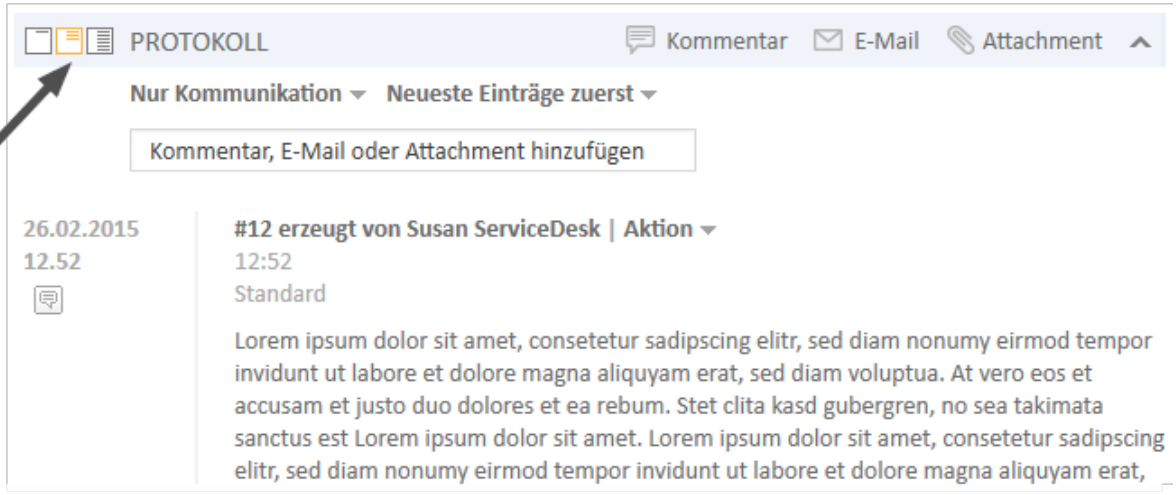


Abbildung 86: *ConSol CM Web Client - Einstellen des Sichtbarkeitslevels*

C.4.4 Speicherung des Ticketprotokolls und Übertragung an das Data Warehouse (DWH)

Bei einer Standardinstallation des CM mit Data Warehouse wird das Ticketprotokoll in der CM-Datenbank gespeichert und an das DWH übertragen.

Sie können mithilfe von Annotationen verhindern, dass das Ticketprotokoll in die CM-Datenbank geschrieben und an das DWH übertragen wird:

Verfügbar in den CM-Versionen	Gruppenannotation	Feldannotation	Ausgelöstes Verhalten
6.6.0 und höher	`no-history`	`no-history-field`	Für Ticketfeldgruppen oder einzelne Ticketfelder mit dieser Annotation werden keine Protokolldaten in die CM-Datenbank geschrieben. Das heißt, es sind keinerlei Protokolldaten verfügbar; die Annotation `dwh-no-history` muss für die entsprechenden Ticketfeldgruppen bzw. Ticketfelder nicht gesetzt werden.
6.10.2 und höher	`dwh-no-history`	`dwh-no-history-field`	Für die Ticketfeldgruppen oder einzelnen Ticketfelder mit dieser Annotation werden keine Protokolldaten an das DWH übertragen.

Das heißt, wenn Sie das Protokoll im Web Client sehen möchten, aber einen guten Grund haben, nicht alle Daten an das DWH zu übertragen, sollten Sie `dwh-no-history` und `dwh-no-history-field` verwenden. Wenn Sie überhaupt kein Protokoll schreiben möchten, z. B. wenn interne Programmierungsdaten in Ticketfelder geschrieben werden, sollten Sie die Kombination `no-history` und `no-history-field` verwenden. Sie können natürlich die beiden Varianten auch kombinieren und verschiedene Werte für die einzelnen Felder oder Feldgruppen verwenden.

Beachten Sie, dass während der Lebensdauer eines CM-Systems Änderungen an der Konfiguration zur Protokollübertragung vorgenommen werden können. Dies kann dazu führen, dass eine CM-Datenbank (alte) Protokolldaten enthält, obwohl die Annotation `no-history` gesetzt ist.



Berücksichtigen Sie folgendes Verhalten bei einem Update auf Version 6.10.2 und höher:

Damit die alten Systemkonfigurationen weiterhin gültig sind, werden bei allen Ticketfeldern und Ticketfeldgruppen, die mit `no-history-field` und `no-history` annotiert sind, automatisch die neuen Annotationen `dwh-no-history-field` und `dwh-no-history` hinzugefügt.

C.5 Konfiguration der Ticketliste

In diesem Kapitel werden folgende Themen behandelt:

C.5.1 Einleitung 169
C.5.2 Konfiguration von Gruppierung und Sortierung der Ticketliste 171
C.5.3 Konfiguration der Parameter, die für ein Ticket angezeigt werden 176
C.5.4 Die Definition von Kunden-Templates 177
C.5.5 Die Annotation von Ticketfeldern 177
C.5.6 Die Seitenanpassung für die Attribute der Ticketliste 178

C.5.1 Einleitung

Im Web Client wird die Ticketliste auf der linken Seite der Benutzeroberfläche angezeigt. Die Liste dient als To-Do-Liste für die Bearbeiter.

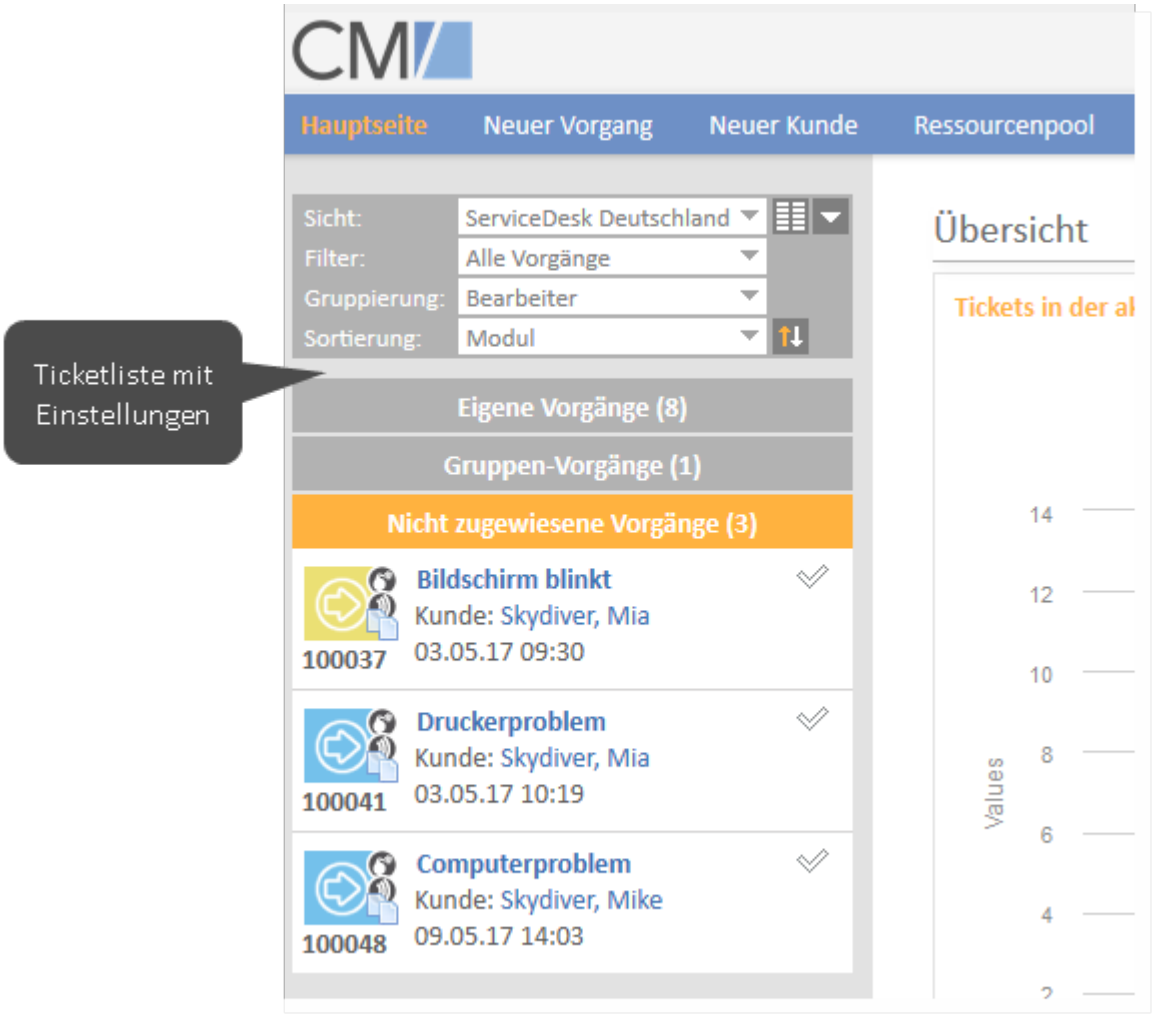


Abbildung 87: *ConSol CM Web Client: Ticketliste*

Die Konfiguration der Sichten und die Zuweisung der Sichten zu bestimmten Rollen definiert, welche Tickets in der Ticketliste angezeigt werden. Eine detaillierte Einführung in die Konfiguration der Sichten durch den Administrator finden Sie im Abschnitt Sichtenverwaltung.

C.5.2 Konfiguration von Gruppierung und Sortierung der Ticketliste

C.5.2.1 Gruppierung

Die Ticketliste kann in Gruppen unterteilt angezeigt werden. In der obigen Abbildung basieren die Gruppen auf dem Bearbeiter, dem das Ticket zugewiesen ist. Es gibt drei Gruppen: *Eigene Vorgänge*, *Gruppen-Vorgänge*, *Nicht zugewiesene Vorgänge*. Die Gruppierung kann vom Bearbeiter im Web Client geändert werden, indem er die gewünschte Gruppierung im Drop-down-Menü *Gruppierung* der Ticketlistenkonfiguration auswählt.

Es stehen immer drei Standardwerte zur Verfügung:

- **Keine Gruppierung**
  Alle Tickets werden in einer Gruppe mit dem Namen *Alle Vorgänge* angezeigt.
- **Bearbeiter**
  Die Tickets sind in drei Gruppen aufgeteilt. Die Namen der Gruppen hängen von der individuellen CM-Konfiguration ab, die Gruppen haben aber immer die gleiche Bedeutung:
  - Tickets, die dem aktuell angemeldeten Bearbeiter zugewiesen sind (z. B. *Eigene Vorgänge*)
  - Tickets, die einem anderen Bearbeiter als dem aktuell angemeldeten Bearbeiter zugewiesen sind (z. B. *Gruppen-Vorgänge*)
  - Tickets, die keinem Bearbeiter zugewiesen sind (z. B. *Nicht zugewiesene Vorgänge*)
- **Queue**
  Die Tickets sind in so viele Gruppen aufgeteilt, wie es Queues in Ihrem CM-System gibt. Sie können allerdings nur die Gruppen sehen, für die Sie mindestens Leseberechtigungen haben. Die Queues werden alphabetisch nach ihren lokalisierten Queue-Namen sortiert.

Zusätzlich zu diesen Werten können Sie als Administrator weitere Parameter definieren, nach denen die Ticketliste gruppiert werden soll. Jedes Ticketfeld des Typs *enum (Sortierte Liste)* kann für diesen Zweck konfiguriert werden. Sie müssen lediglich die Feldannotation `groupable` für den entsprechenden Wert setzen. Im folgenden Beispiel wurde die Annotation für das Ticketfeld *priority* gesetzt, das auf Deutsch *Priorität* heißt.

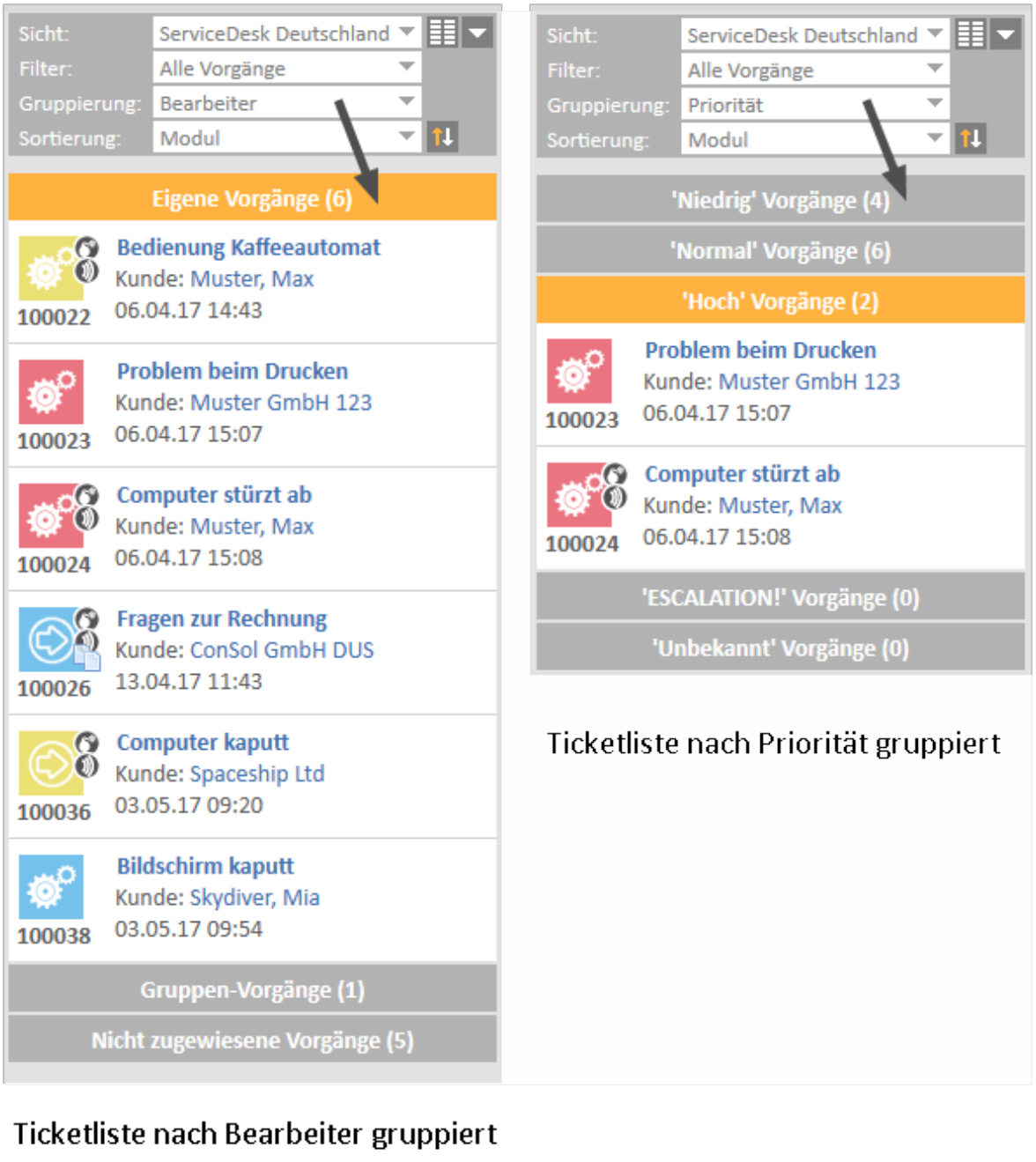


Abbildung 88: *ConSol CM Web Client: Zwei unterschiedliche Gruppierungen der Ticketliste (Bearbeiter: Standard, Priorität: angepasst)*

Im Admin Tool wird die Annotation `groupable` für das Ticketfeld in der Navigationsgruppe *Tickets*, Navigationselement *Ticketfelder* gesetzt.

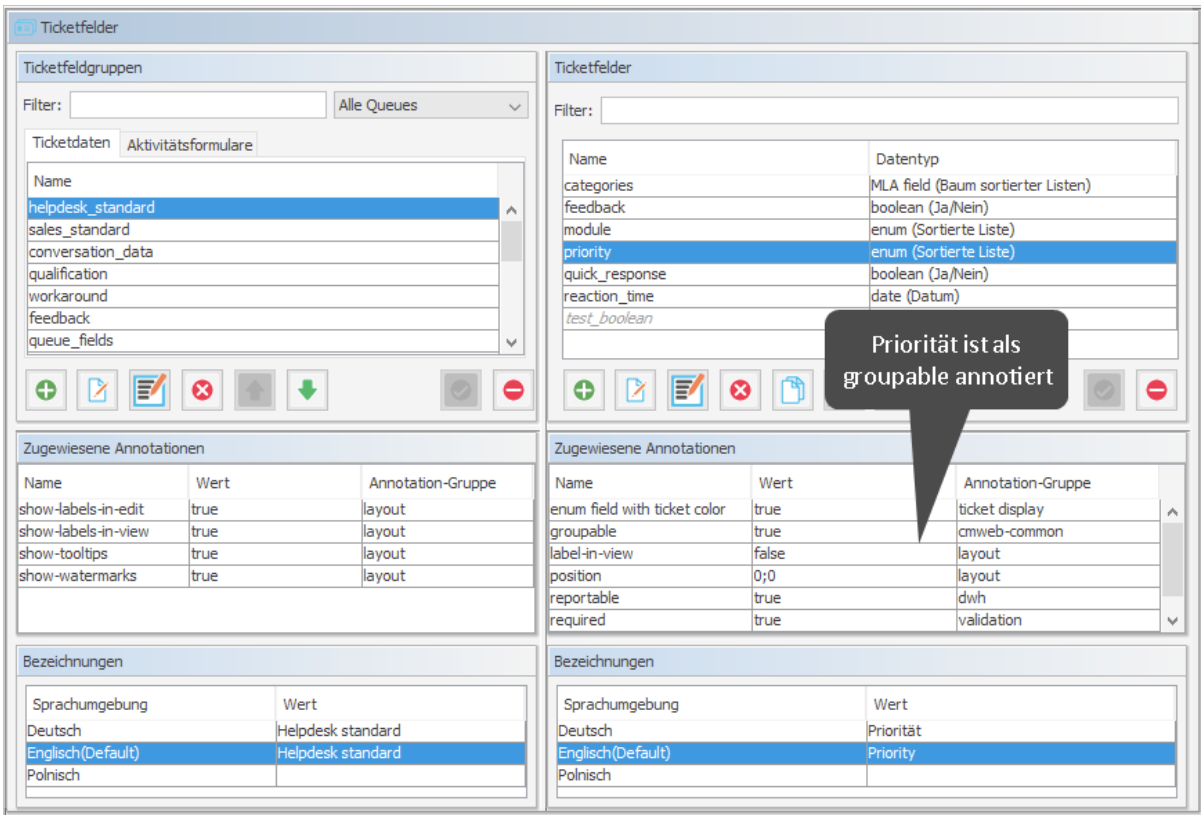


Abbildung 89: *ConSol CM Admin Tool - Tickets, Ticketfelder: Annotation groupable für Ticketfeld gesetzt*

C.5.2.2 Sortierung

Innerhalb einer Gruppe in der Ticketliste können die Tickets sortiert werden. Die Sortierung kann von dem Bearbeiter im Web Client geändert werden, indem er die gewünschte Sortierung im Drop-down-Menü *Sortierung* der Ticketlistenkonfiguration auswählt. Es gibt drei Standardwerte, nach denen die Tickets sortiert werden können.

- **Bereich**
  Sortiert die Tickets innerhalb der Gruppen nach der logischen Reihenfolge der Bereiche, in denen sie sich gerade befinden. Die logische Reihenfolge der Bereiche ist die Reihenfolge der Prozessschritte in Ihrem Geschäftsprozess, z. B. Offenes Ticket - Ticket in Bearbeitung - Frage an Fachabteilung - Dem Kunden die Lösung mitteilen - Ticket schließen.
- **Erstellungsdatum**
  Sortiert die Tickets innerhalb der Gruppen nach ihrem Erstellungsdatum.
- **Änderungsdatum**
  Sortiert die Tickets innerhalb der Gruppe nach dem Datum der letzten Änderung. Eine Änderung ist ein Ereignis der folgenden Art, das manuell (über den Web Client) oder automatisch (über den Workflow) ausgelöst werden kann:

- Bearbeiten eines Ticketfeldes
- Bearbeiten der Ticketdetails
- Hinzufügen eines Zusatzkunden
- Ändern der Rolle eines Kunden
- Setzen eines Kunden als Hauptkunden
- Hinzufügen einer Relation
- Löschen einer Relation
- Hinzufügen eines zusätzlichen Bearbeiters
- Löschen eines zusätzlichen Bearbeiters
- Hinzufügen einer Ressourcenrelation
- Löschen einer Ressourcenrelation
- Hinzufügen eines Kommentars
- Senden einer E-Mail
- Antworten, Weiterleiten, erneutes Senden einer im Protokoll gespeicherten E-Mail
- Hinzufügen eines Attachments
- Löschen eines Attachments
- Hinzufügen einer Zeitbuchung

Zusätzlich zu diesen Werten können Sie als Administrator weitere Parameter definieren, nach denen die Tickets innerhalb einer Gruppe sortiert werden können. Jedes Ticketfeld des Typs *enum (Sortierte Liste)* kann für diesen Zweck konfiguriert werden. Sie müssen lediglich die Feldannotation `sortable` für den entsprechenden Wert setzen. Im folgenden Beispiel wurde die Annotation für das Ticketfeld *priority* gesetzt, das auf Deutsch *Priorität* heißt.

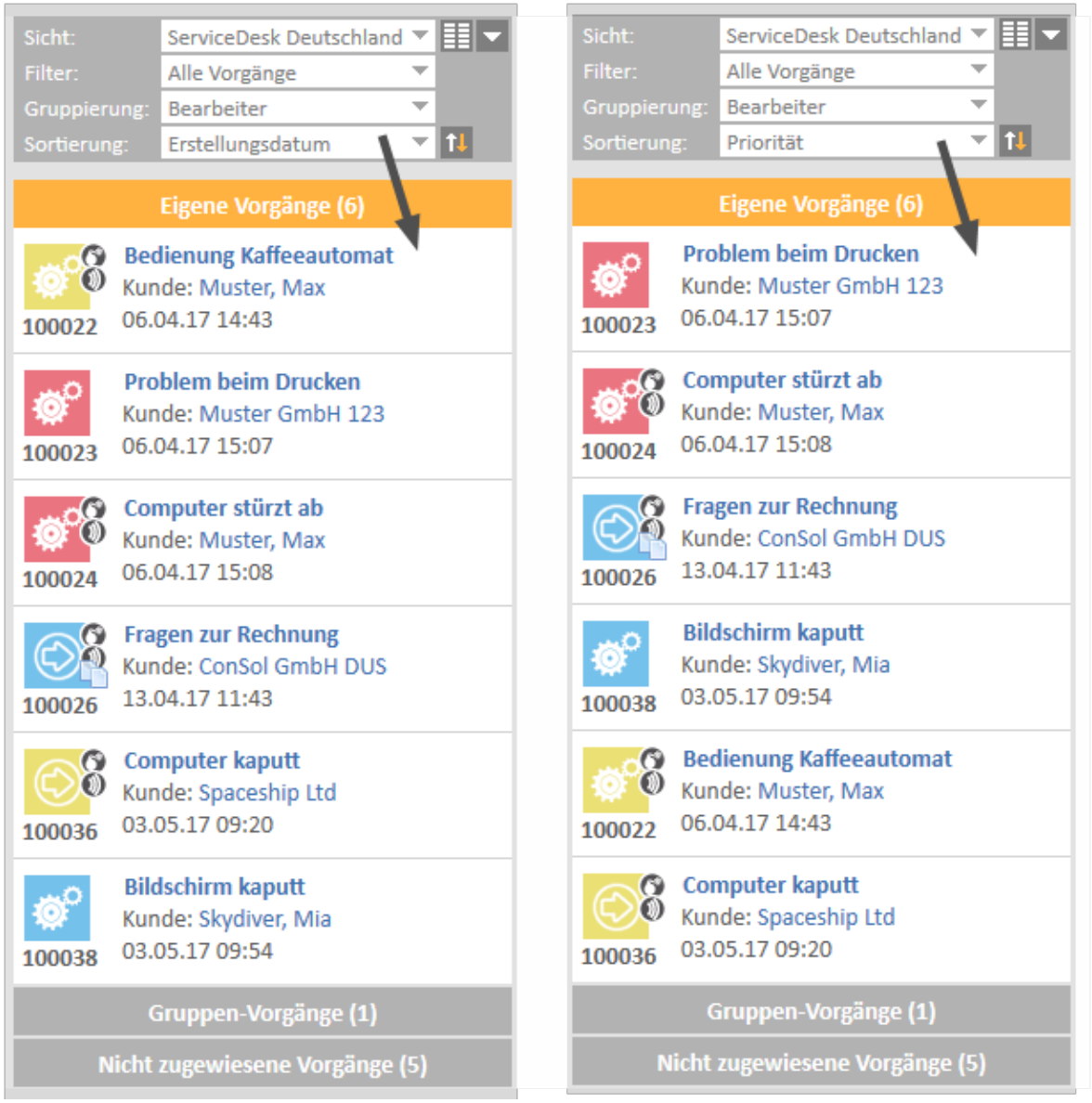


Abbildung 90: *ConSol CM Web Client: Zwei unterschiedliche Sortierungen der Ticketliste, Gruppierung nach Bearbeiter, Gruppe Eigene Vorgänge (Erstellungsdatum: Standard, Priorität: angepasst)*

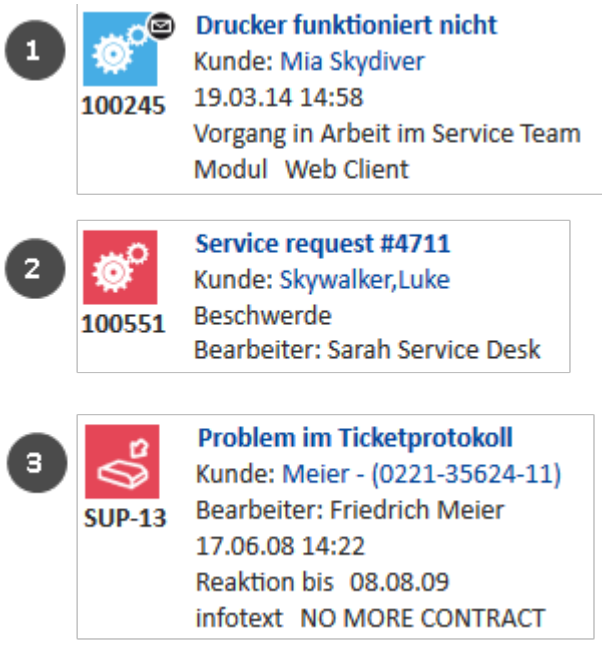
Im Admin Tool wird die Annotation `sortable` für das Ticketfeld in der Navigationsgruppe *Tickets*, Navigationselement *Ticketfelder* gesetzt.

C.5.3 Konfiguration der Parameter, die für ein Ticket angezeigt werden

In diesem Abschnitt erfahren Sie, wie Sie die Parameter, die für ein Ticket angezeigt werden, konfigurieren können. Ein Ticket in der Ticketliste kann in unterschiedlichen Systemen unterschiedlich aussehen. Dies hängt von folgenden Parametern ab:

- die Definition von Kunden-Templates
- die Annotation von Ticketfeldern
- die Seitenanpassung für die Attribute der Ticketliste

Die folgenden Beispiele zeigen unterschiedliche Darstellungen der Tickets:



Im obigen Beispiel werden folgende Daten angezeigt:

(1) Beispiel für Service-Desk-Ticket

- Ticketthema (Standard)
- Kunde (mittels Kunden-Template formatiert)
- Erstellungsdatum (Standard für nicht zugewiesene Tickets)
- Bereich (mittels Seitenanpassung formatiert)
- Ticketfeld (hier: Modul, eine Sortierte Liste; mittels Annotation formatiert)

(2) Beispiel für Customer-Service-Ticket

- Ticketthema (Standard)
- Kunde (mittels Kunden-Template formatiert)
- Ticketfeld (hier: Tickettyp, eine Sortierte Liste)
- Zugewiesen an (Standard für zugewiesene Tickets)

(3) Beispiel für Support-Ticket

- Ticketthema (Standard)
- Kunde (mittels Kunden-Template formatiert, hier mit Telefonnummer)
- Zugewiesen an (Standard für zugewiesene Tickets)
- Erstellungsdatum
- Ticketfeld (hier: Deadline der Reaktionszeit, ein Datumsfeld)
- Ticketfeld (hier: ein String-Feld, infotext)

C.5.4 Die Definition von Kunden-Templates

Die Darstellung der Kundendaten (z. B. nur Name oder Name und Telefonnummer) wird durch das Template definiert, das dem Kundenobjekt zugewiesen ist. Dies kann das Ticketlisten-Template oder, wenn das Ticketlisten-Template nicht definiert ist, das Standard-Template sein. Eine detaillierte Erklärung der Templates für Kundendaten finden Sie im Abschnitt Templates für Kundendaten.

C.5.5 Die Annotation von Ticketfeldern

C.5.5.1 Spezielle Annotationen für die Ticketliste

Es gibt drei Annotationen, mit denen die Darstellung von Ticketfelddaten in der Ticketliste gesteuert wird:

- ticket-list-position
- ticket-list-colspan
- ticket-list-rowspan

ticket-list-position

Die Annotation kann für Ticketfelder gesetzt werden. Sie definiert die Position des angegebenen Ticketfeldes in der Ticketliste. Bedenken Sie, dass alle Felder, die so annotiert sind, in eine gemeinsame Matrix passen müssen. Es gilt das gleiche Prinzip wie für die Positionierung der Ticketfelder im Ticket, siehe Abschnitt Verwaltung von Ticketfeldern (Einrichten des Ticketdatenmodells).

ticket-list-colspan

Die Annotation kann für Ticketfelder gesetzt werden. Sie definiert die Anzahl der Spalten, die für das entsprechende Feld verwendet werden sollen.

ticket-list-rowspan

Die Annotation kann für Ticketfelder gesetzt werden. Sie definiert die Anzahl der Zeilen, die für das entsprechende Feld verwendet werden sollen.

C.5.5.2 Allgemeine Annotationen, die für die Ticketliste relevant sind

Die folgenden Annotationen sind für die Ticketliste relevant, auch wenn sie nicht nur für die Ticketliste gelten, d.h. sie können auch für beliebige andere Ticketfelder verwendet werden.

- show-label-in-view

show-label-in-view

Diese Annotation kann für Ticketfelder, Kundenfelder und Ressourcenfelder gesetzt werden. Für die Konfiguration der Ticketliste sind nur Ticketfelder relevant.

Wenn die Annotation auf „true“ gesetzt ist, wird das Label (der Name des Ticketfeldes) im Ansichtsmodus angezeigt (hier: in der Ticketliste). Wenn das Label nicht angezeigt werden soll, setzen Sie den Wert auf „false“.

C.5.5.3 Beispiel

Das folgende Beispiel zeigt ein Ticket in der Ticketliste, das mithilfe von mehreren Parametern konfiguriert wurde.

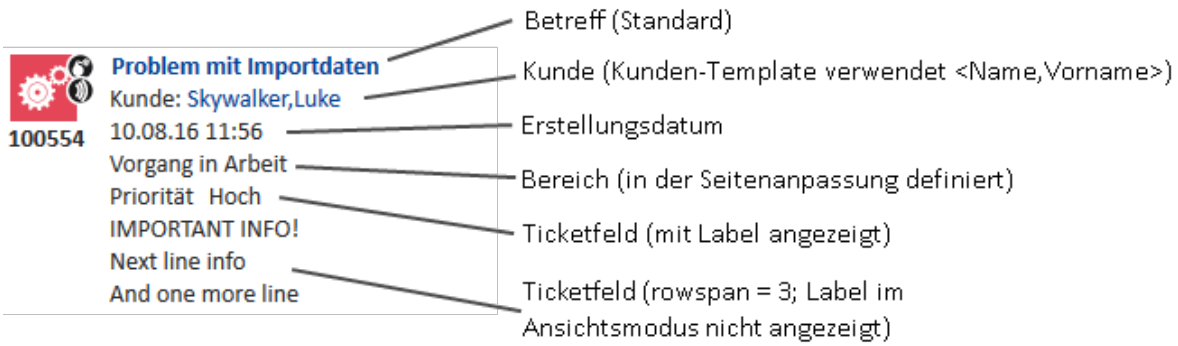


Abbildung 91: *Ticket in der Ticketliste, mithilfe mehrerer Parameter formatiert*

C.5.6 Die Seitenanpassung für die Attribute der Ticketliste

Es gibt vier Attribute der Seitenanpassung, die für die Konfiguration der Darstellung eines Tickets in der Ticketliste relevant sind. Sie werden für den Seitenanpassungs-Typ *accordionTicketList* gesetzt.

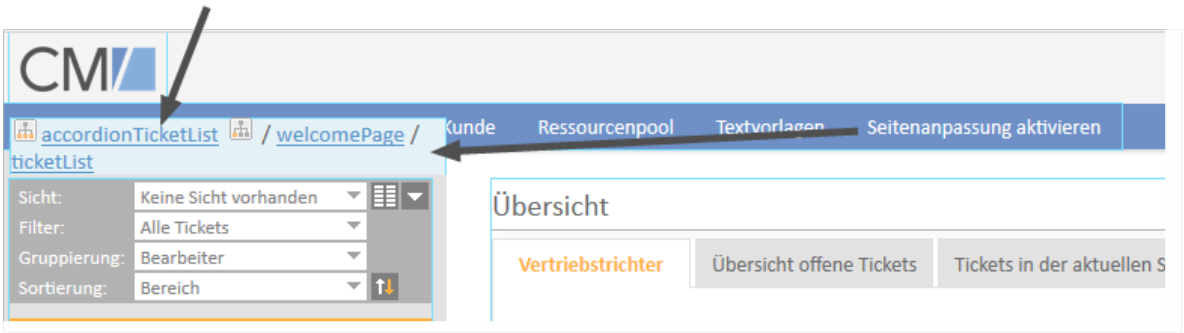


Abbildung 92: *ConSol CM Web Client - Seitenanpassung für die Ticketliste*

Es können folgende Attribute verwendet werden:

- hideEmptyGroups
- ticketDataConfigQueueGroupingScript

- ticketDataConfigEngineerGroupingScript
- ticketDataConfigCustomGroupingScript
- ticketDataConfigNoGroupingScript

Siehe auch Abschnitt accordionTicketList (Typ).

Um das Layout der einzelnen Tickets in der Ticketliste zu definieren, können folgende Schlüsselwörter verwendet werden (eine detaillierte Erklärung der erforderlichen Syntax finden Sie im Abschnitt Formatierungsprinzip und Syntax):

- CREATION_DATE
- QUEUE
- SCOPE
- ENGINEER
- CUSTOMER

Außerdem verwendet jede Art der Gruppierung (siehe folgende Abschnitte) einen zusätzlichen typspezifischen Satz an Schlüsselwörtern.

C.5.6.1 ticketDataConfigQueueGroupingScript

Definiert die Anzeige der Ticketinformationen, wenn die Gruppierung nach Queue ausgewählt ist. Die folgenden zusätzlichen Schlüsselwörter können verwendet werden (eine detaillierte Erklärung der erforderlichen Syntax finden Sie im Abschnitt Formatierungsprinzip und Syntax):

- Queue-Namen

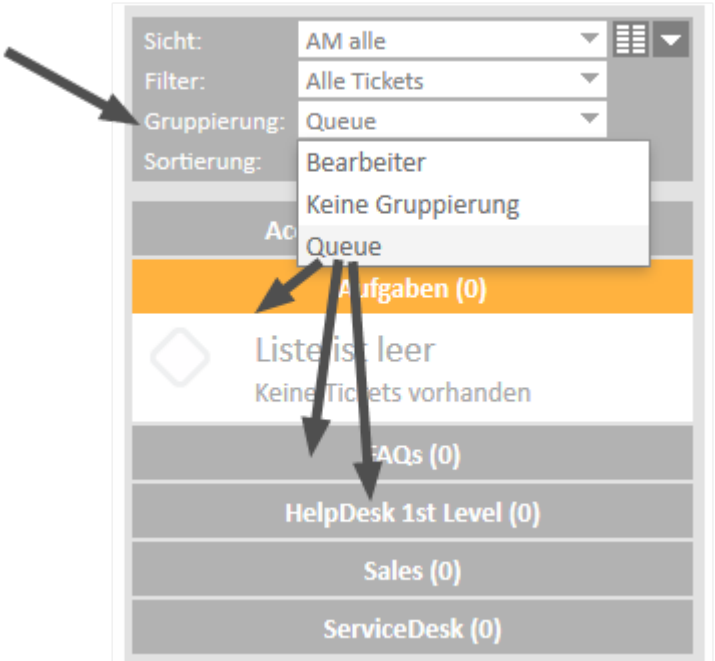


Abbildung 93: *ConSol CM Web Client - Nach Queue gruppierte Ticketliste*

C.5.6.2 ticketDataConfigEngineerGroupingScript

Definiert die Anzeige der Ticketinformationen, wenn die Gruppierung nach Bearbeiter ausgewählt ist. Die folgenden zusätzlichen Schlüsselwörter können verwendet werden (eine detaillierte Erklärung der erforderlichen Syntax finden Sie im Abschnitt Formatierungsprinzip und Syntax):

- MINE
- GROUP
- UNASSIGNED

C.5.6.3 ticketDataConfigCustomGroupingScript

Definiert die Anzeige der Ticketinformationen, wenn die Gruppierung nach einem Ticketfeld ausgewählt ist. Die folgenden zusätzlichen Schlüsselwörter können verwendet werden (eine detaillierte Erklärung der erforderlichen Syntax finden Sie im Abschnitt Formatierungsprinzip und Syntax):

- enum_grp_name.enum_value_name

C.5.6.4 ticketDataConfigNoGroupingScript

Definiert die Anzeige der Ticketinformationen, wenn kein Kriterium für die Gruppierung ausgewählt ist. Es können keine zusätzlichen Schlüsselwörter verwendet werden.

- Es kann nur die Einstellung DEFAULT verwendet werden.

C.5.6.5 Formatierungsprinzip und Syntax

Das Formatierungsprinzip und die Syntax ist für alle vier Attribute identisch.

Der Konfigurations-String kann 2048 Zeichen enthalten. Falls Sie einen längeren Ausdruck verwenden möchten, müssen Sie ein Admin-Tool-Skript (des Typs *Seitenanpassung*) einsetzen.



Abbildung 94: *ConSol CM Web Client - Attribute der Seitenanpassung für accordionTickeList (zwei Beispiele dargestellt)*

Um die Anzeige der Ticketliste in der Seitenanpassung (für die oben erwähnten Attribute) zu definieren, wird eine hierarchische Konfiguration verwendet.

Die **erste** Ebene ist die Definition der Sichten.

- ALL_VIEWS_DEFAULT
- Sichtennamen wie zum Beispiel *2nd_Level_View_Open*

Auf der **zweiten** Ebene können Sie festlegen, für welche Menge der Parameter angezeigt werden soll.

Für die queue-basierte Gruppierung (d. h. in `ticketDataConfigQueueGroupingScript`) können folgende Parameter verwendet werden:

- Queue-Namen

Für die bearbeiterbasierte Gruppierung (d. h. in `ticketDataConfigEngineerGroupingScript`) können folgende Parameter verwendet werden:

- MINE
- GROUP
- UNASSIGNED

Für die benutzerdefinierte Gruppierung (d. h. in `ticketDataConfigCustomGroupingScript`) können folgende Parameter verwendet werden:

- enum_grp_name.enum_value_name, z. B. helpdesk_standard.module.misc.

Für jede Art der Gruppierung kann ein DEFAULT-Wert gesetzt werden, der für alle nicht speziell definierten Konstellationen verwendet wird.

Die **dritte** Ebene sind die anzuzeigenden Parameter. Folgende Parameter sind möglich:

- CUSTOMER
- SCOPE
- ENGINEER
- CREATION_DATE
- QUEUE

Das folgende Beispiel (für die bearbeiterspezifische Gruppierung) vermittelt Ihnen einen ersten Eindruck. Zur besseren Lesbarkeit enthält es Zeilenumbrüche. Verwenden Sie in CM keine Zeilenumbrüche in diesem Ausdruck.

```
[ALL_VIEWS_DEFAULT:
 [MINE: [CUSTOMER, CREATION_DATE]],
 [GROUP: [CUSTOMER, ENGINEER, CREATION_DATE]],
 [UNASSIGNED: [CUSTOMER, CREATION_DATE]],
 ServiceDeskAll:
 [ DEFAULT: [CUSTOMER, CREATION_DATE, SCOPE]],
 [ UNASSIGNED: [CUSTOMER, CREATION_DATE]]]
```

Die obige Vorlage hat folgende Bedeutung:

Für alle Sichten ist eingestellt, dass die dem Bearbeiter zugewiesenen Tickets den Kunden und das Erstellungsdatum enthalten, dass die Gruppentickets den Kunden, den zugewiesenen Bearbeiter und das Erstellungsdatum enthalten, und dass die nicht zugewiesenen Tickets den Kunden und das Erstellungsdatum enthalten. Für Tickets in der ServiceDesk-Queue ist eine spezielle Konfiguration festgelegt: Für alle Arten von Tickets (d.h. dem Bearbeiter zugewiesene Tickets und Gruppentickets) werden die Kundendaten, das Erstellungsdatum und der Bereich angezeigt. Für nicht zugewiesene Tickets werden nur die Kundendaten und das Erstellungsdatum angezeigt.

Die Ticketliste aus diesem Beispiel wird so dargestellt, wie in den folgenden zwei Abbildungen gezeigt.



Abbildung 95: *ConSol CM Web Client - Ticketliste für die Sicht 1st Level Offene Tickets, definiert durch ALL_VIEWS_DEFAULT*

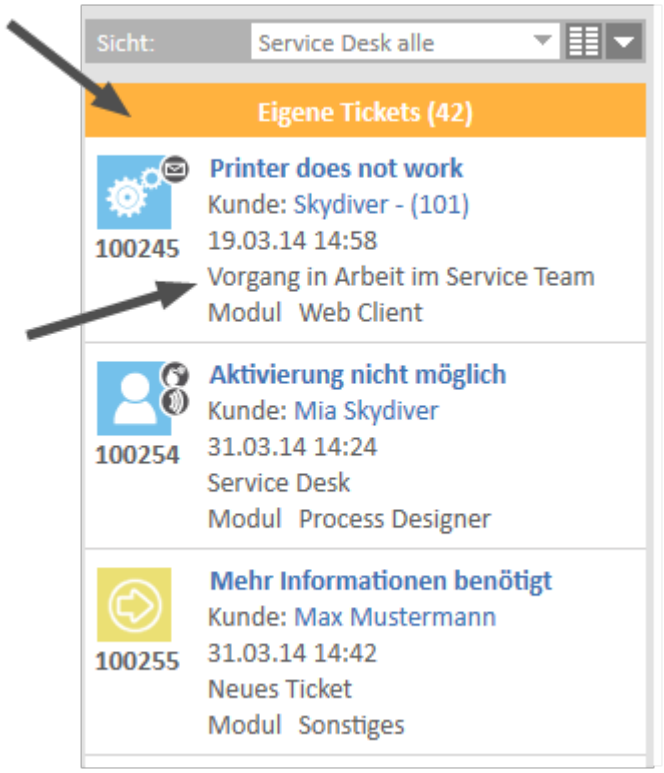


Abbildung 96: *ConSol CM Web Client - Ticketliste für die Sicht Service Desk Alle mit sichtenspezifischer Konfiguration, Bereich wird für die Tickets des Bearbeiters angezeigt*

Da die beiden Definitionen identisch sind, können Sie hier mit dem Wert DEFAULT arbeiten. Die folgende JSON-Anweisung stellt eine Konfiguration dar, die mit dem oben gezeigten Beispiel identisch ist.

```
[ALL_VIEWS_DEFAULT:
 [DEFAULT: [CUSTOMER, CREATION_DATE]],
 [GROUP: [CUSTOMER, ENGINEER, CREATION_DATE]],
 ServiceDeskAll:
 [ DEFAULT: [CUSTOMER, CREATION_DATE, SCOPE]],
 [ UNASSIGNED: [CUSTOMER, CREATION_DATE]]]
```

Die Standardeinstellung für eine Konfiguration wiederherstellen:

- DEFAULT_CONFIGURATION_SCRIPT

C.5.6.6 Standardsystemeinstellungen

Wenn keine Werte geändert wurden oder die Standardeinstellung (mit dem Parameter DEFAULT_CONFIGURATION_SCRIPT) wiederhergestellt wird, werden folgende Werte gesetzt:

- **Queues** (ticketDataConfigQueueGrouping):

```
[ ALL_VIEWS_DEFAULT: [ DEFAULT: [CUSTOMER, CREATION_DATE] ] ]
```

- **Bearbeiter** (ticketDataConfigEngineerGrouping)

```
[ ALL_VIEWS_DEFAULT: [ MINE: [CUSTOMER, CREATION_DATE ] ],
  [ GROUP: [CUSTOMER, ENGINEER, CREATION_DATE] ],
  [ UNASSIGNED: [CUSTOMER, CREATION_DATE ] ] ]
```

- **Ticketfelder** (ticketDataConfigCustomGrouping):

```
[ ALL_VIEWS_DEFAULT: [ DEFAULT: [CUSTOMER, CREATION_DATE] ] ]
```

- Keine Gruppierung (ticketDataConfigNoGrouping):

```
[ ALL_VIEWS_DEFAULT: [ DEFAULT: [CUSTOMER, CREATION_DATE] ] ]
```

C.5.6.7 Konfigurieren der Seitenanpassung für die Ticketliste über ein Skript

Wie bei allen Attributen der Seitenanpassung können Sie die erforderlichen Attribute auch in einem Admin-Tool-Skript setzen, statt die Werte für jedes einzelne Attribut über den Web Client zu setzen. Eine detaillierte Erklärung dazu finden Sie in Seitenanpassung.

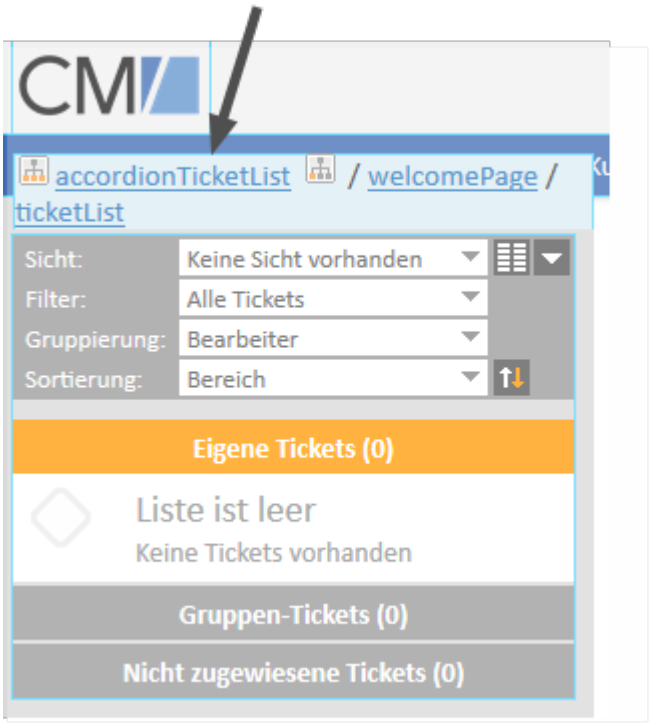


Abbildung 97: *ConSol CM Web Client - Seitenanpassung für die Ticketliste*

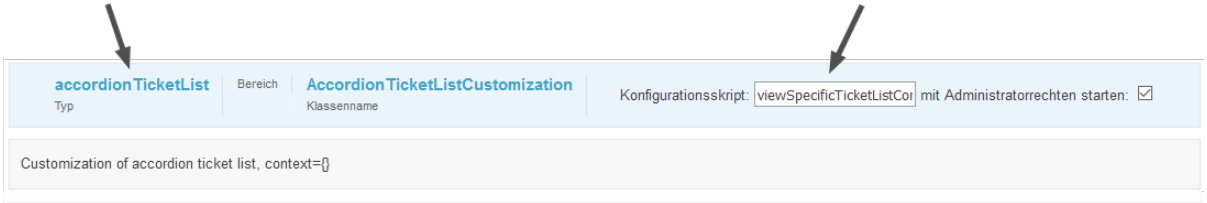


Abbildung 98: *ConSol CM Web Client - Konfigurieren eines Skripts für die Ticketliste in der Seitenanpassung*

Die Attribute für die Ticketliste müssen in einem Skript für den Typ *accordionTicketList* gesetzt werden. Das Skript kann entweder nur eines der Attribute für den Typ, mehrere Attribute oder sogar Werte für alle Attribute zurückgeben. Das folgende Beispiel zeigt ein Skript, das Werte für die beiden Attribute *ticketDataConfigNoGrouping* und *ticketDataConfigQueueGrouping* zurückgibt.

```
return [ticketDataConfigNoGrouping: "[ALL_VIEWS_DEFAULT: [DEFAULT: [CUSTOMER,
 CREATION_DATE]], MultiCGView: [DEFAULT: [ENGINEER]], 1st_Level_View_Open:
 [DEFAULT: [SCOPE]]]",ticketDataConfigQueueGrouping: "[ALL_VIEWS_DEFAULT: [DEFAULT:
 [CUSTOMER, CREATION_DATE, ENGINEER, ENGINEER]], [HelpDesk_1st_Level: [CUSTOMER,
 SCOPE, CREATION_DATE, QUEUE]], [HelpDesk_2nd_Level: [CUSTOMER, QUEUE, SCOPE,
 CREATION_DATE]]]"]
```

Code-Beispiel 1: *Beispielskript für die Konfiguration der Ticketliste (view-SpecificTicketListConfig.groovy)*

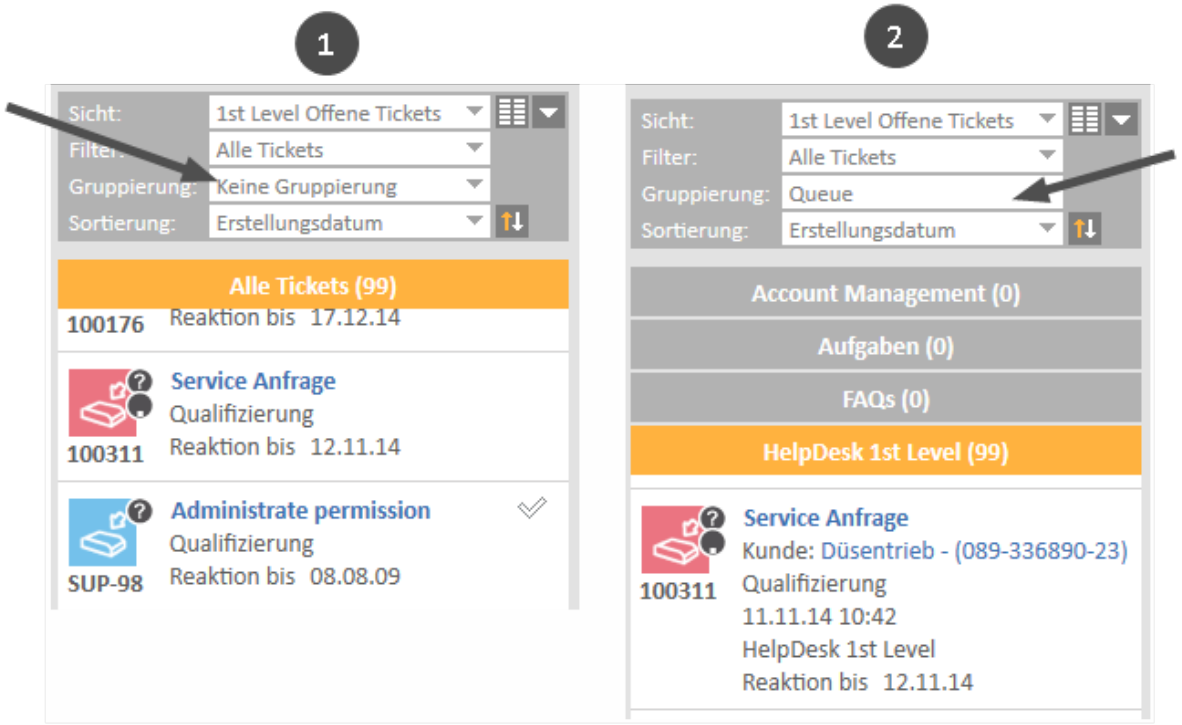


Abbildung 99: *Teile von zwei Ticketlisten entsprechend der Konfiguration im Beispielskript*

- (1) Keine Gruppierung. HelpDesk_1st_Level:
  DEFAULT: SCOPE
  (Reaktionszeit stammt aus Annotation!)
- (2) Gruppierung nach Queue. HelpDesk_1st_Level:
  CUSTOMER, SCOPE, CREATION_DATE, QUEUE angezeigt
  (Reaktionszeit stammt aus Annotation!)

C.6 Konfiguration des Web Client Dashboards

C.6.1 Einleitung

Seit Version 6.9.4 bietet der ConSol CM Web Client für die Bearbeiter ein **Dashboard**, das auf der **Hauptseite** des Web Clients angezeigt wird. Auf dem Dashboard können Sie statistische Daten anzeigen lassen, die die wichtigsten Informationen über die Arbeitsbereiche der Bearbeiter enthalten.

Das Dashboard besteht aus einem oder mehreren sogenannten Widgets. Als Standardkonfiguration hat das Dashboard nur ein Widget, das eine Grafik mit der Anzahl der Tickets in allen Gruppen der ausgewählten Sicht zeigt. Dashboards können interaktive Elemente enthalten, z. B. um Elemente der Grafik ein- oder auszublenden. Eine dauerhafte bearbeiterspezifische (individuelle) Anpassung ist allerdings nicht möglich.

> Nach einem Update von einer älteren Version auf CM-Version 6.9.4 ist das Web Client Dashboard deaktiviert.
>
> In einer neuen Installation von CM in Version 6.9.4 oder höher, ist das Web Client Dashboard standardmäßig aktiviert. Die Standardkonfiguration des Dashboards wird mitgeliefert (siehe Beispiel unten).

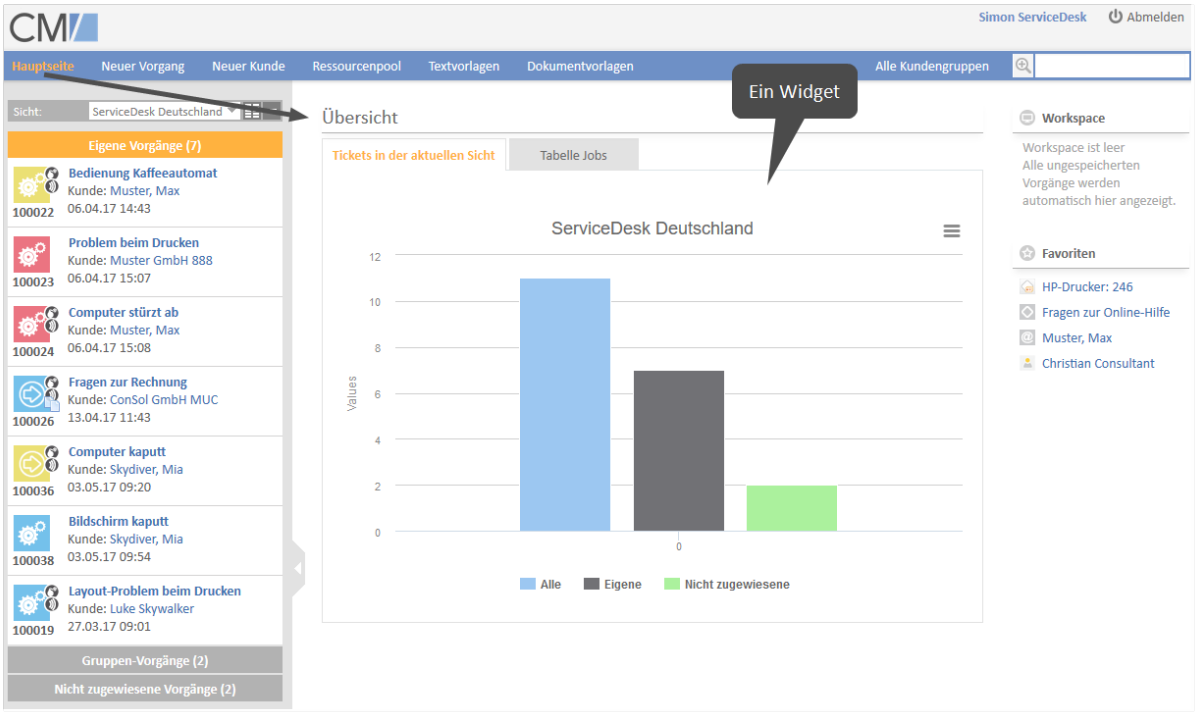


Abbildung 100: *ConSol CM Web Client - Beispiel für Web Client Dashboard*



Abbildung 101: *ConSol CM Web Client - Interaktives Widget (Ein- und Ausblenden von Teilen der Grafik)*

Die für das Dashboard eingesetzten Widgets können von einem der drei folgenden Typen sein:

- **Chart**
  für die grafische Darstellung der Daten
- **Tabelle**
  für die tabellarische Darstellung der Daten
- **KPI**
  für die spezielle Darstellung von Kennzahlen (Key Performance Indicators)
- **Zuletzt angezeigt**
  Standard-Widget, das eine Liste der Objekte anzeigt, die der aktuelle Bearbeiter zuletzt angezeigt hat. Kann ein- oder ausgeschaltet werden. Der Bearbeiter kann auswählen, ob nur seine oder alle Tickets angezeigt werden sollen. In beiden Fällen hängt es von den Zugriffsberechtigungen des Bearbeiters ab, welche Tickets angezeigt werden.
- **Letzte Änderungen**
  Standard-Widget, das eine Liste der zuletzt geänderten Objekte anzeigt. Der Bearbeiter kann auswählen, ob nur seine Änderungen oder die von allen Bearbeitern gemachten Änderungen berücksichtigt werden sollen. Es hängt von den Zugriffsberechtigungen des Bearbeiters ab, welche Tickets angezeigt werden.

Als Administrator können Sie das Layout der Dashboards nach Wunsch gestalten, d. h. Sie können KPI-, Chart- und Tabellen-Widgets auf der Dashboard-Seite platzieren. Die Anordnung auf der Seite basiert auf einem Raster.

C.6.2 Beispiele

Die folgenden Beispiele vermitteln Ihnen einen ersten Eindruck von dem, was auf dem Web Client Dashboard angezeigt werden kann.

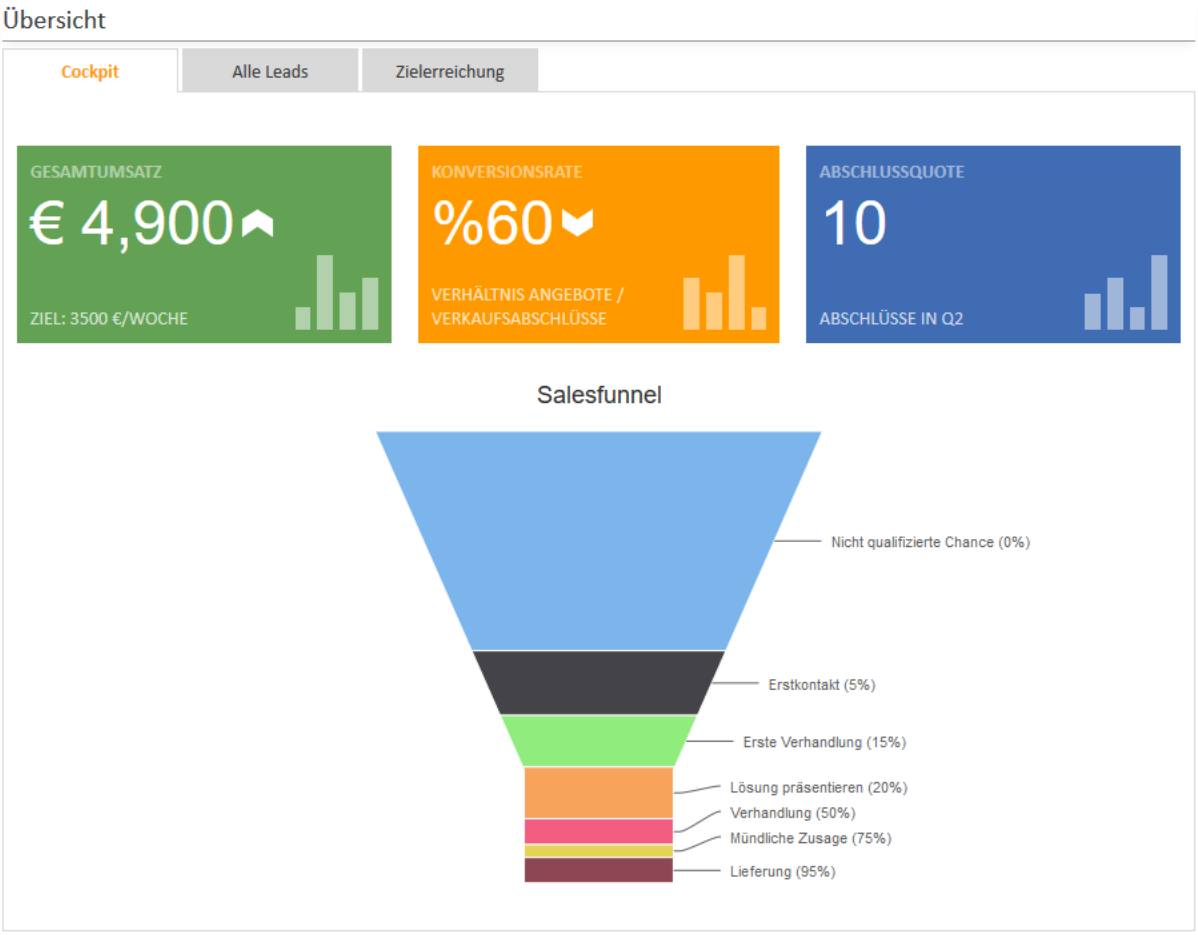


Abbildung 102: *ConSol CM Web Client - Beispiel für Dashboard-Tab mit zwei Chart-Widgets (KPI, Trichter) in einer Spalte*

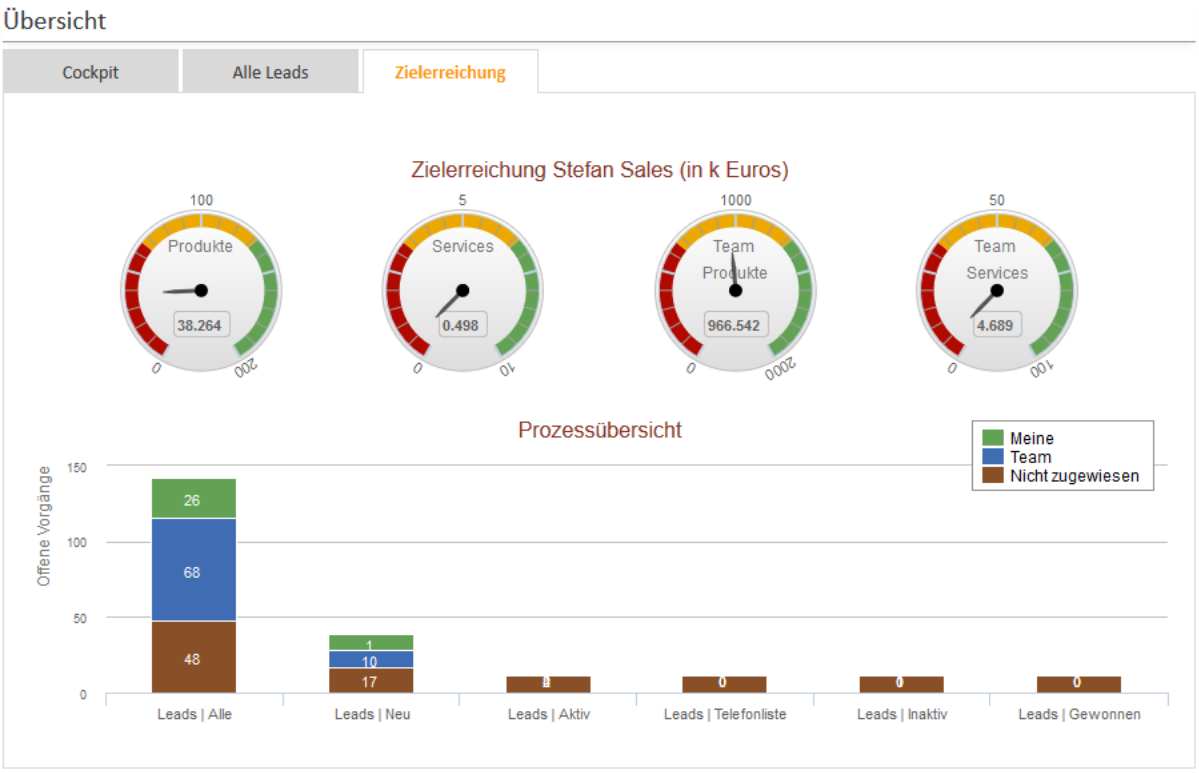


Abbildung 103: *ConSol CM Web Client - Beispiel für Dashboard-Tab mit zwei Chart-Widgets in einer Spalte*



Abbildung 104: *ConSol CM Web Client - Beispiel für Dashboard mit einem Widget*

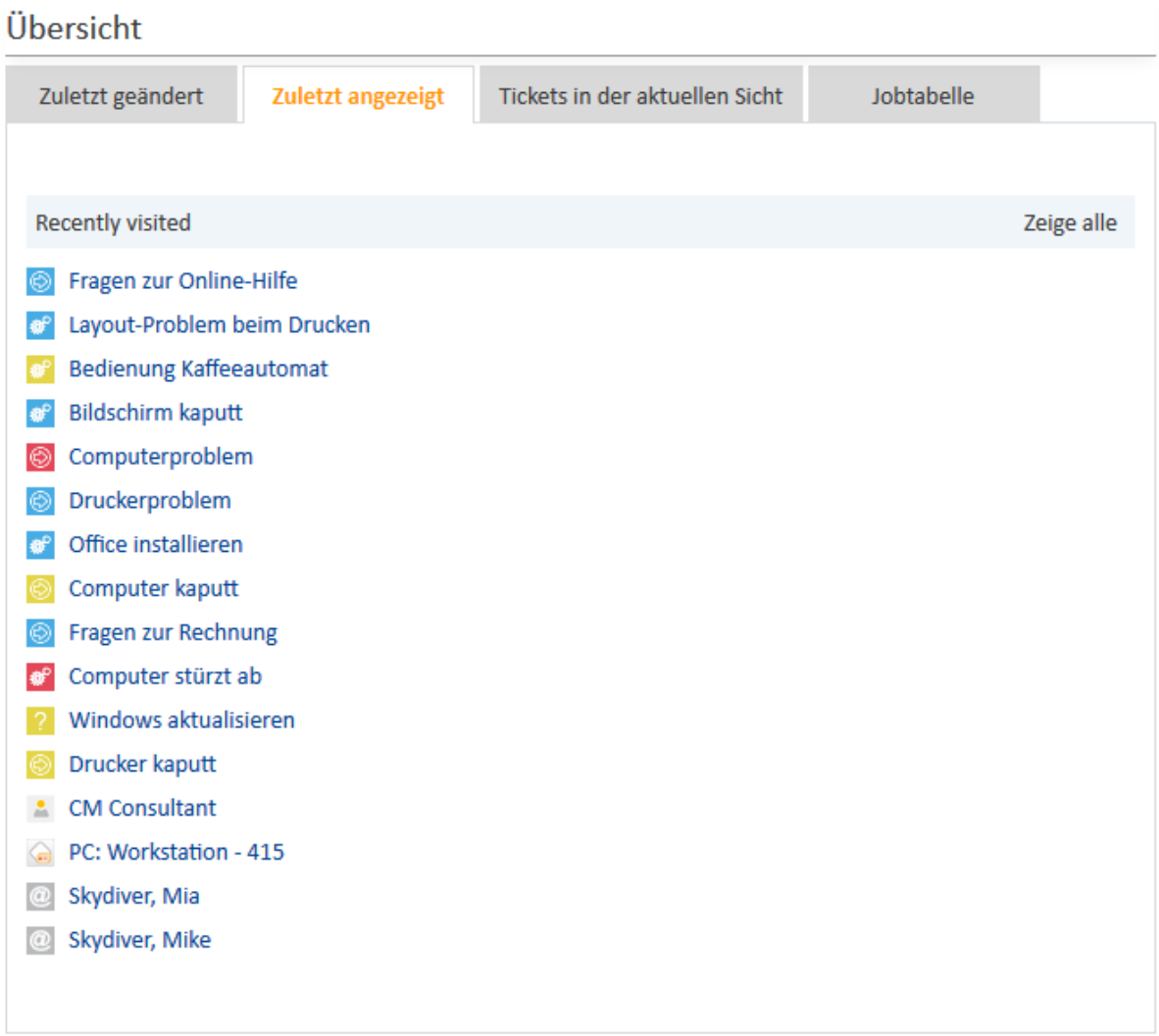


Abbildung 105: *Beispiel für ein recentlyVisitedWidget*

Beachten Sie, dass das Widget *Zuletzt angezeigt* in Ihrem CM-System anders aussehen kann, weil es über Attribute und Einstellungen für das recentlyVisitedWidget anders konfiguriert wurde.

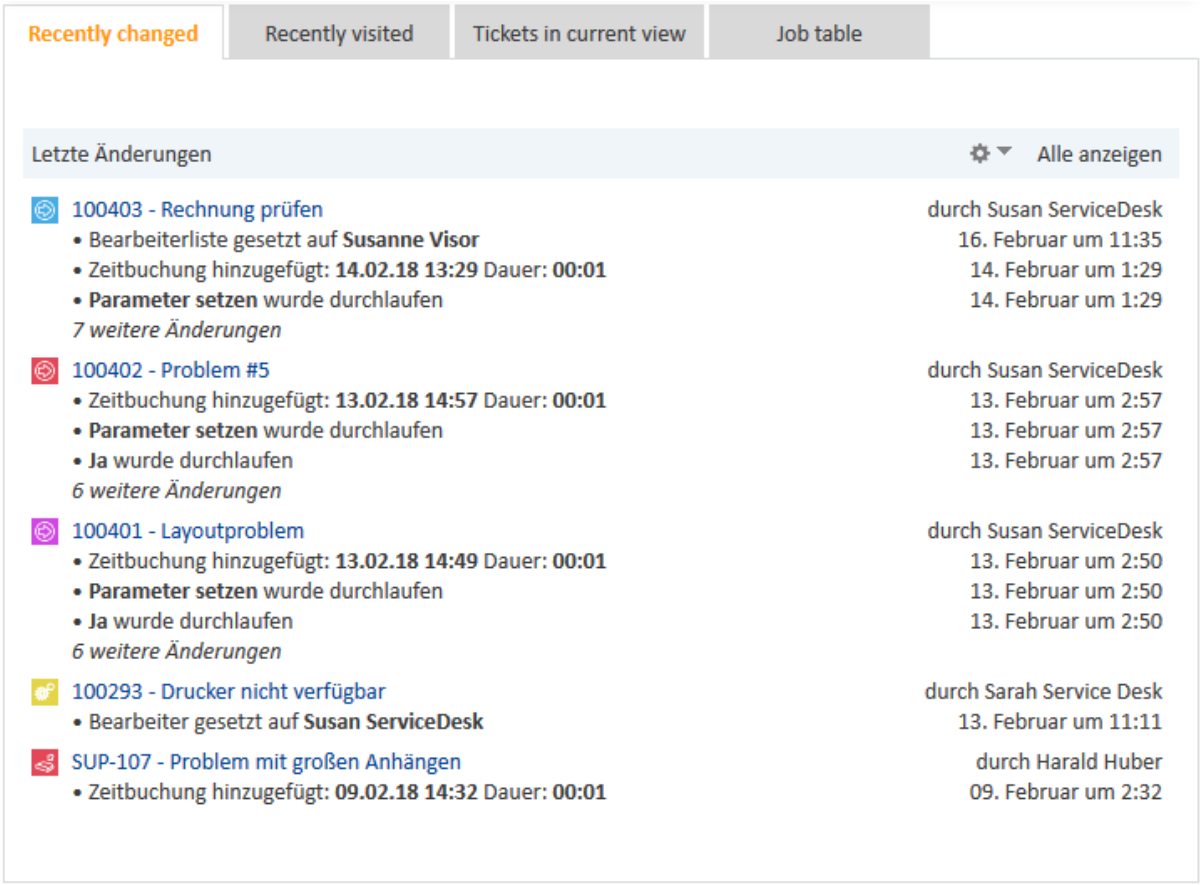


Abbildung 106: *Beispiel für ein recentChangesWidget*

Beachten Sie, dass das Widget *Zuletzt geändert* auf Ihrem CM-System anders aussehen kann, weil es über Attribute für das recentChangesWidget anders konfiguriert wurde.

Die Auslöser der Änderungen im Widget *Zuletzt geändert* werden wie folgt angezeigt:

- Bearbeiter hat Vor- und Nachnamen: Vor- und Nachname werden angezeigt.
- Bearbeiter hat nur Nachnamen: Nachname wird angezeigt.
- Bearbeiter hat nur Vornamen: Vorname und Login werden angezeigt.
- Bearbeiter hat nur Login: Login wird angezeigt.

Hinweis für alle Widgets:

Die Daten für die Graphen und/oder Tabellen werden mittels in Admin-Tool-Skripten gespeicherten Groovy-Statements abgerufen.

Das Dashboard wird in der Seitenanpassung für das Web Client Dashboard konfiguriert.

C.7 Konfiguration der Graphendarstellung von Relationen

C.7.1 Konfiguration der Graphenanzeige im Standardmodus

An mehreren Stellen des ConSol CM Web Clients können Relationen zwischen Objekten hergestellt und angezeigt werden:

- Auf der Ticketseite
    - Relationen des Tickets zu einem oder mehreren Kunden
    - Relationen des Tickets zu anderen Tickets (Referenzen oder Parent/Child- oder Master/Slave-Relationen)
    - Relationen des Tickets zu einer oder mehreren Ressourcen (wenn CM/Resource Pool aktiv ist)
- Auf der Firmenseite:
    - Relationen der Firma zu Tickets
    - Relationen der Firma zu ihren Kontakten
    - Relationen der Firma zu anderen Kunden (Kontakten, Firmen)
    - Relationen der Firma zu einer oder mehreren Ressourcen (wenn CM/Resource Pool aktiv ist)
- Auf der Kontaktseite:
    - Relationen des Kontakts zu Tickets
    - Relationen des Kontakts zu anderen Kunden (Kontakten, Firmen)
    - Relationen des Kontakts zu einer oder mehreren Ressourcen (wenn CM/Resource Pool aktiv ist)

Wenn CM/Resource Pool aktiv ist:

- Auf der Ressourcenseite:
    - Relationen der Ressource zu einer oder mehreren anderen Ressourcen
    - Relationen der Ressource zu Kunden (Kontakten, Firmen)
    - Relationen der Ressource zu Tickets

Die Relationen können als Liste oder Graph angezeigt werden. Die Listenansicht ist standardmäßig aktiviert und kann nicht ausgeschaltet werden. Die Graphenansicht kann zusätzlich in der **Seitenanpassung** aktiviert werden. Wenn sie aktiviert ist, kann der Bearbeiter zwischen Listen- und Graphenansicht wählen.

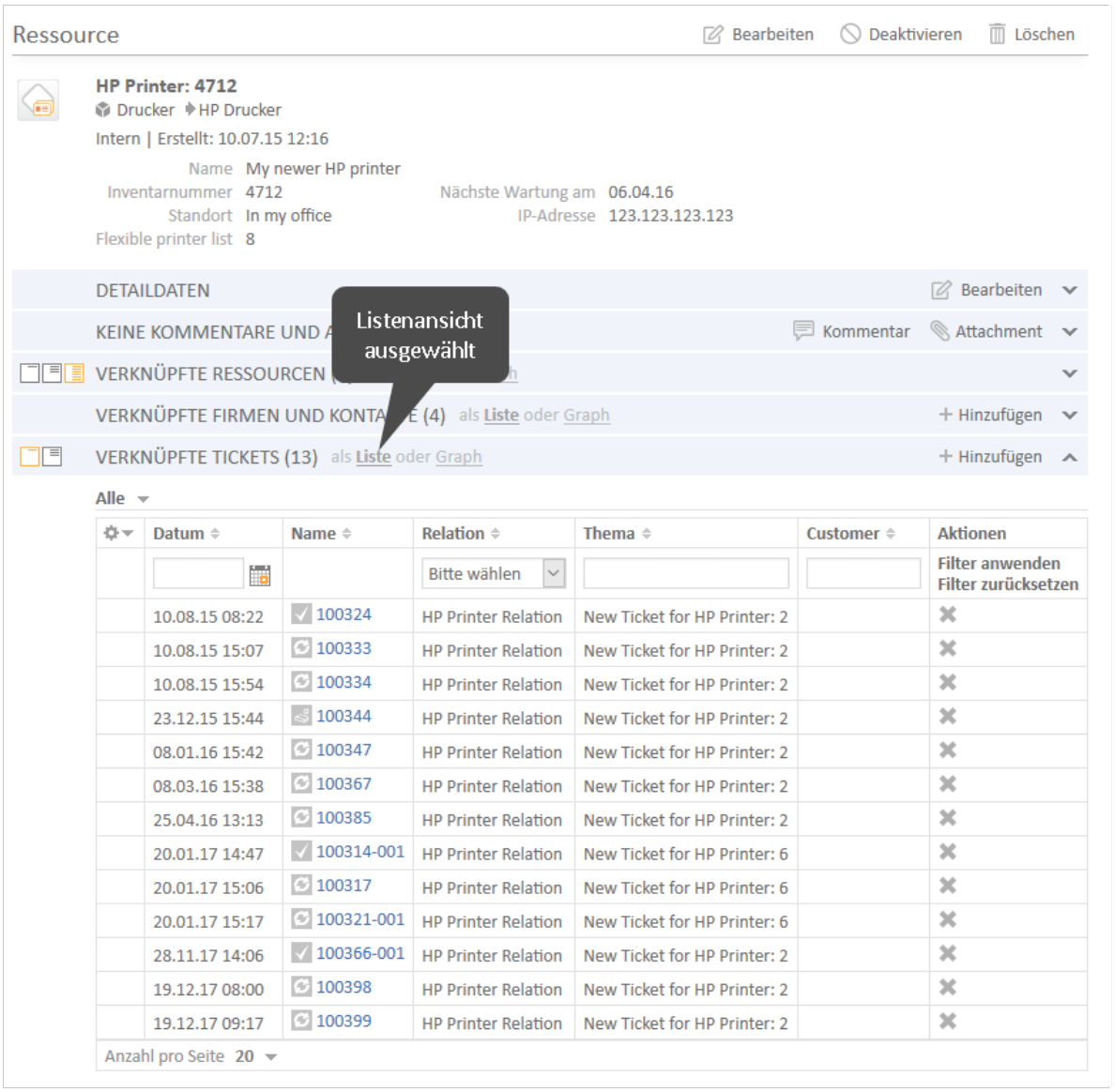


Abbildung 107: *ConSol CM Web Client - Ressourcenseite eines Druckers: Bereich für verknüpfte Tickets im Listenmodus*

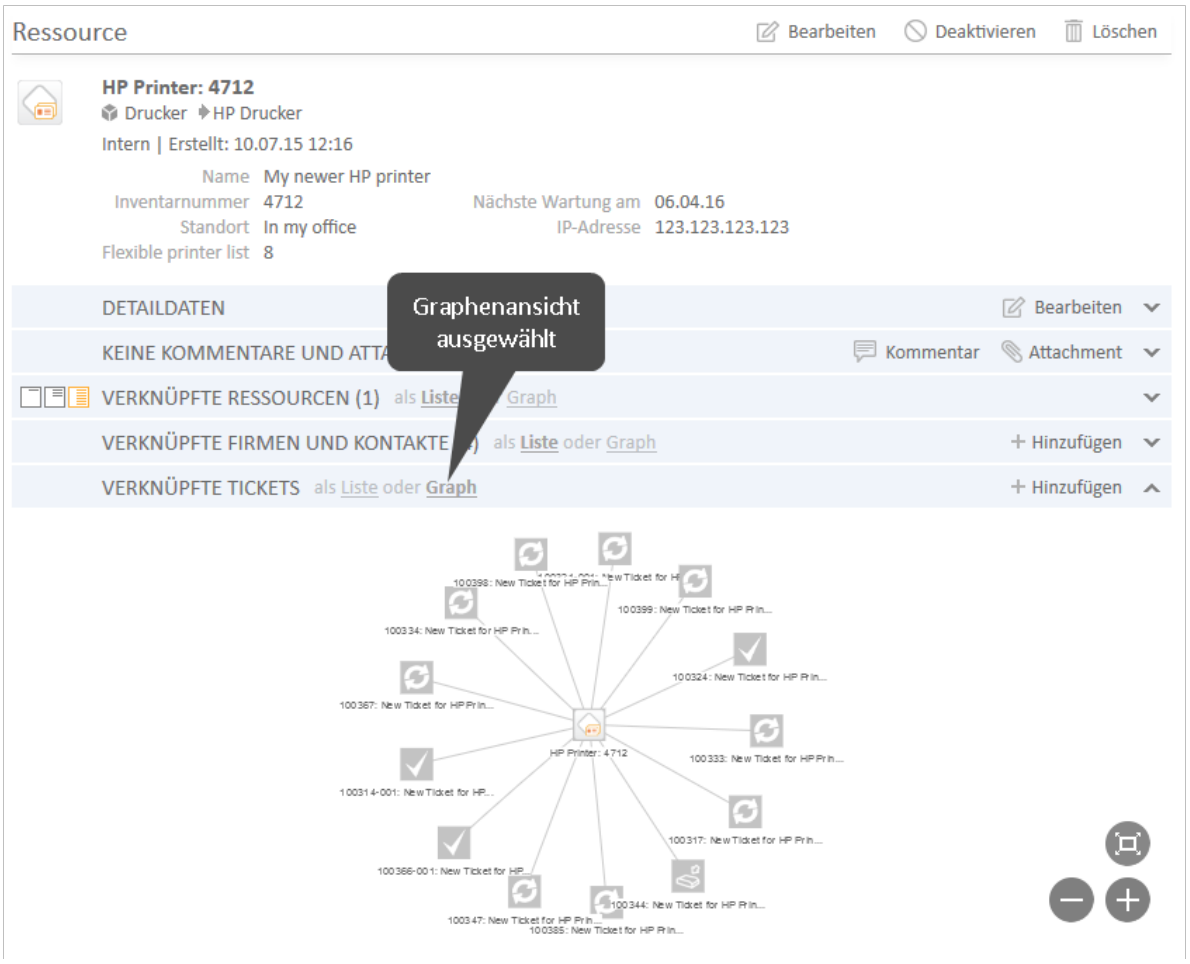


Abbildung 108: *ConSol CM Web Client - Ressourcenseite eines Druckers: Bereich für verknüpfte Tickets im Graphenmodus*

Die Graphenansicht muss für jeden Abschnitt einzeln aktiviert werden (siehe obige Liste). Sie wird mithilfe von Attributen der Seitenanpassung konfiguriert. Eine detaillierte Beschreibung der Konfiguration finden Sie in Seitenanpassung für die grafische Darstellung von Relationen.

C.7.2 Konfiguration weiterer Relationsansichten - Expertenmodus

Zusätzlich zu diesen Standardrelationsbereichen auf Ticket-, Kunden- und Ressourcenseiten kann auf jeder dieser Seiten ein zusätzlicher Bereich angezeigt werden. In diesem Bereich kann ein speziell für das System implementierter, auf einem Skript basierender Relationsgraph angezeigt werden. Eine detaillierte Beschreibung der Konfiguration finden Sie in Seitenanpassung für die grafische Darstellung von Relationen.

C.8 Seitenanpassung

In diesem Kapitel werden folgende Themen behandelt:

C.8.1 Allgemeine Einführung in die Seitenanpassung 197

C.8.2 Seitenanpassung im Web Client 198

C.8.3 Reihenfolge und Prioritäten bei der Seitenanpassung 208

C.8.4 Seitenanpassung mit Attributen 209

C.8.5 Seitenanpassungsattribute für Typen, Bereiche und Unterbereiche in alphabetischer Reihenfolge 219

C.8.6 Seitenanpassung für das Web Client Dashboard 281

C.8.1 Allgemeine Einführung in die Seitenanpassung

Zusätzlich zum Design der Web-Client-GUI, das bei der Definition der Ticketfelder (siehe Abschnitt Verwaltung von Ticketfeldern (Einrichten des Ticketdatenmodells)), Kundenfelder (siehe Abschnitt Verwaltung von Kundenfeldern und GUI-Design für Kundendaten) und Ressourcenfelder (siehe Abschnitt CM/Resource Pool - Einrichten des Ressourcenmodells) vorgenommen wird, kann ein Administrator in der **Seitenanpassung** weitere Features und Funktionen des GUI-Layouts konfigurieren. Die Konfiguration erfolgt durch die Zuweisung von Werten zu bestimmten **Attributen**.

Wie Sie sich als Administrator im Web Client anmelden, sehen Sie im Hauptmenü den Menüpunkt *Seitenanpassung aktivieren*. Je nach Kontext, d. h. der aktuell angezeigten Seite, bietet die Seitenanpassung unterschiedliche seiten- und kontextspezifische Funktionalitäten.

Wenn Sie zum Beispiel ein Ticket geöffnet haben und dann die Seitenanpassung starten, können Sie allgemeine Attribute für Tickets konfigurieren. Wenn der Ticket-E-Mail Editor geöffnet ist, können Sie zusätzlich spezifische Attribute des E-Mail-Editors konfigurieren.

In den folgenden Abschnitten sind die grundsätzlichen Prinzipien der Seitenanpassung und alle verfügbaren Seitenanpassungsattribute detailliert erklärt. In den anderen Abschnitten dieses *ConSol CM Administratorhandbuchs* werden, falls nötig, Verweise auf diese Abschnitte eingefügt.

> Die Einstellungen der Seitenanpassung sind Teil der Konfigurationsdaten von ConSol CM. Sie können diese mit dem Admin Tool exportieren und importieren, siehe Abschnitt Deployment (Import/Export).

C.8.2 Seitenanpassung im Web Client

Wenn Sie den Seitenanpassungsmodus starten, wird in der unteren Hälfte der GUI der Definitionsbereich der Seitenanpassung angezeigt. Auf der rechten Seite sehen Sie einen Baum, der die Struktur der aktuellen Seite widerspiegelt. Auf der Seite selbst ist jedes Seitenelement mit einem blauen Rand markiert. Wenn Sie mit der Maus über ein Element fahren, wird der Name angezeigt. Wenn Sie auf diesen Namen klicken wird der Definitionsbereich für dieses Element geöffnet und das Element wird in der Baumstruktur markiert. Auf diese Weise können Sie die Komponente, die Sie ändern möchten, leicht identifizieren.

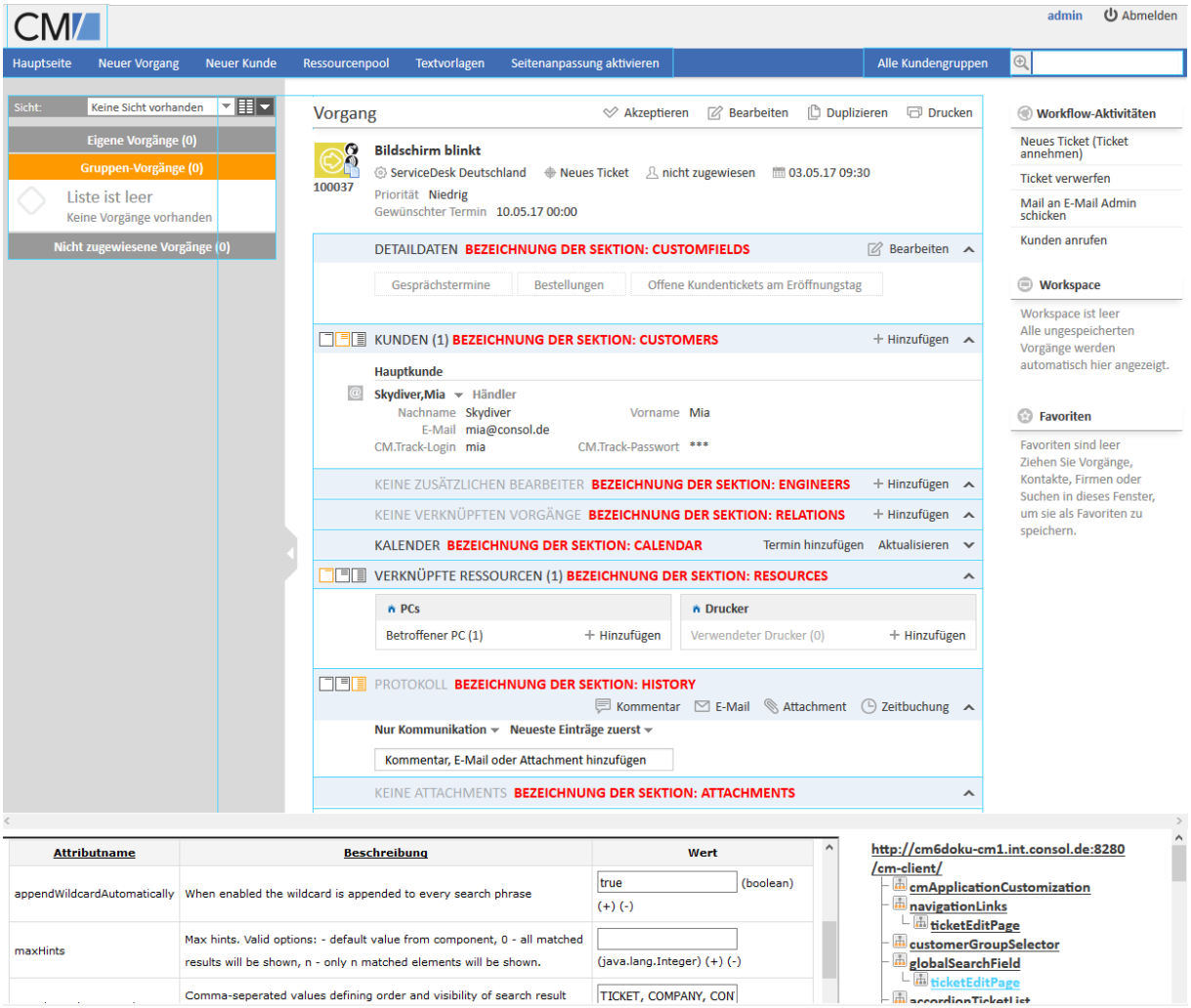


Abbildung 109: *ConSol CM Web Client - Definitionsbereich der Seitenanpassung*

Die Baumstruktur kann folgende Elemente enthalten (siehe nächste Abbildung). Da der Definitionsbereich der Seitenanpassung eher klein ist, müssen Sie unter Umständen scrollen, um alle Elemente sehen zu können. In diesem Beispiel (siehe obige Abbildung) hat der Administrator das Element *ticketEditPage* geöffnet (Details zu diesem Element in den folgenden Abschnitten).

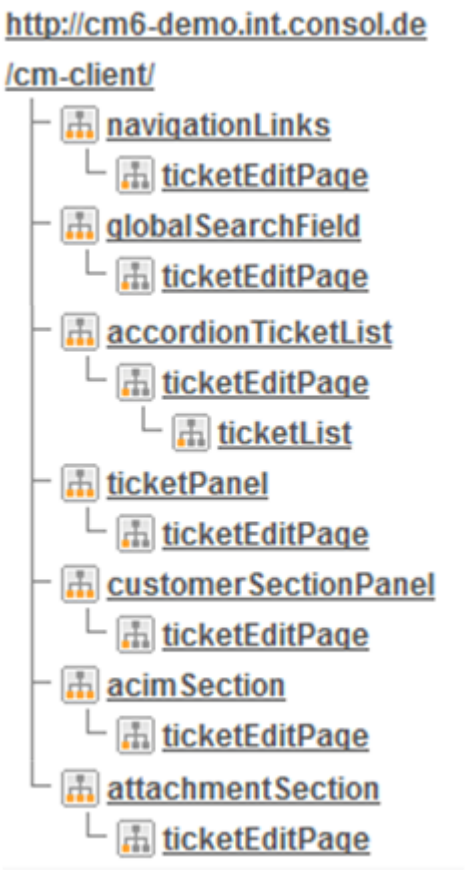


Abbildung 110: *ConSol CM Web Client - Baumstruktur der Seitenanpassung*

Icon	Beschreibung
	Konfiguration aller Elemente dieses Typs
	Konfiguration dieses bestimmten Elements, wenn es sich im ausgewählten Bereich befindet

Sie können in der Baumstruktur auch auf den Namen des Elements klicken, um den Editor für dieses Element im linken Bereich des Definitionsbereichs zu öffnen, z. B. für das Element *navigationLinks* (siehe folgende Abbildung).

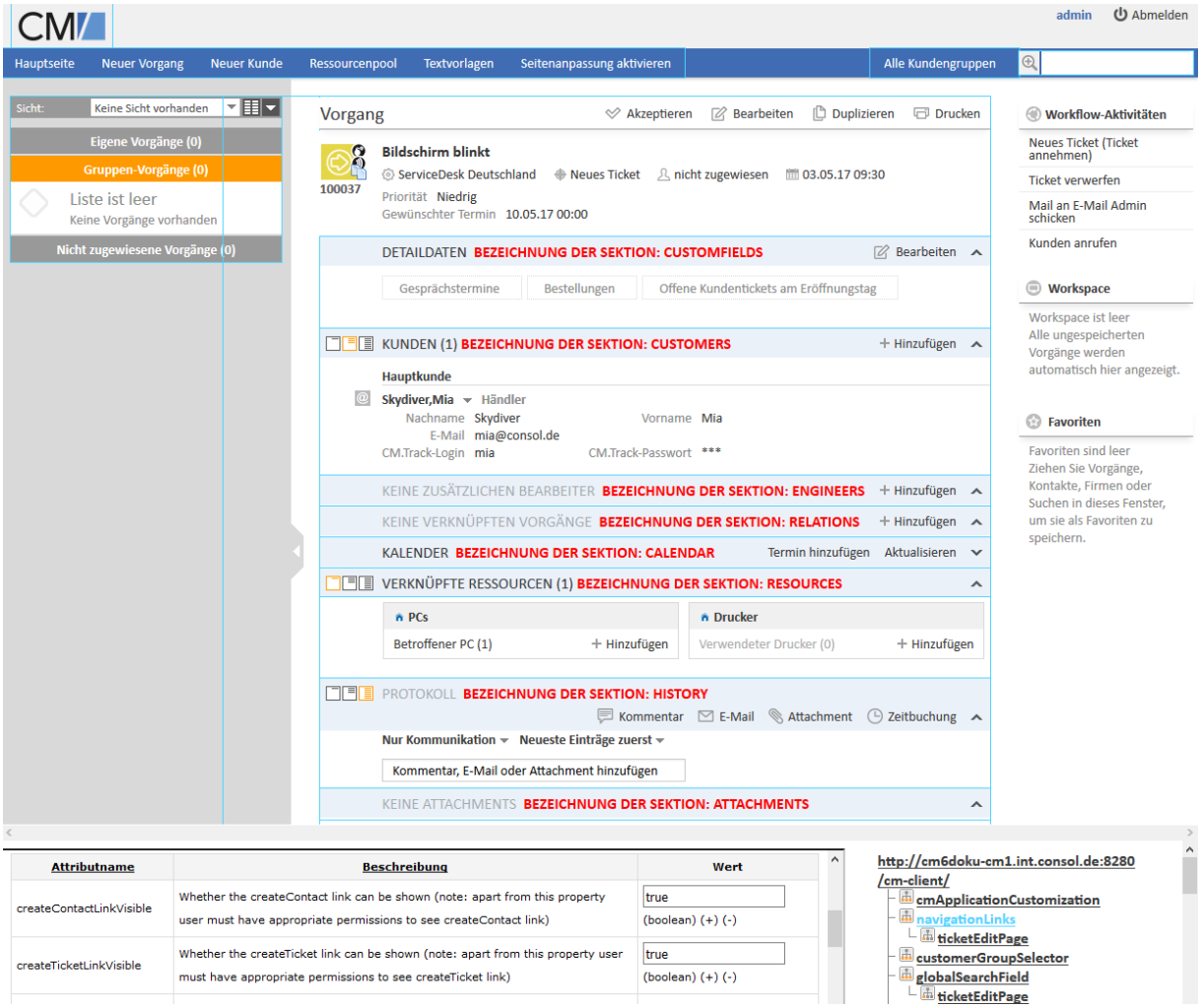


Abbildung 111: *ConSol CM Web Client - Ausgewähltes Element der Baumstruktur im Definitionsbereich*

Die gesamte Seite hat eine streng hierarchische Struktur und jedes Element wird definiert durch:

- einen Typ (Pflicht)
- einen Bereich (Pflicht)
- einen Unterbereich (optional)
- eine Klasse (die entsprechende Java-Klasse, Pflicht)

Wenn Sie ein Element entweder in der Baumstruktur oder auf der GUI markieren, werden diese in der blauen Kopfzeile des Definitionsbereichs angezeigt. In der Seitenanpassung können Sie entscheiden, auf welcher Ebene Sie die Attribute konfigurieren möchten:

- Wenn Sie auf der Ebene *Typ* arbeiten, definieren Sie die Attribute für alle untergeordneten Elemente dieses Typs, d. h. für alle Bereiche (Unterbereiche).
- Wenn Sie auf der Ebene *Bereich* arbeiten, definieren Sie die Attribute für alle (untergeordneten) Elemente dieses Bereichs und alle Unterbereiche.

- Wenn Sie auf der Ebene *Unterbereich* arbeiten, definieren Sie die Attribute nur für diesen Unterbereich.

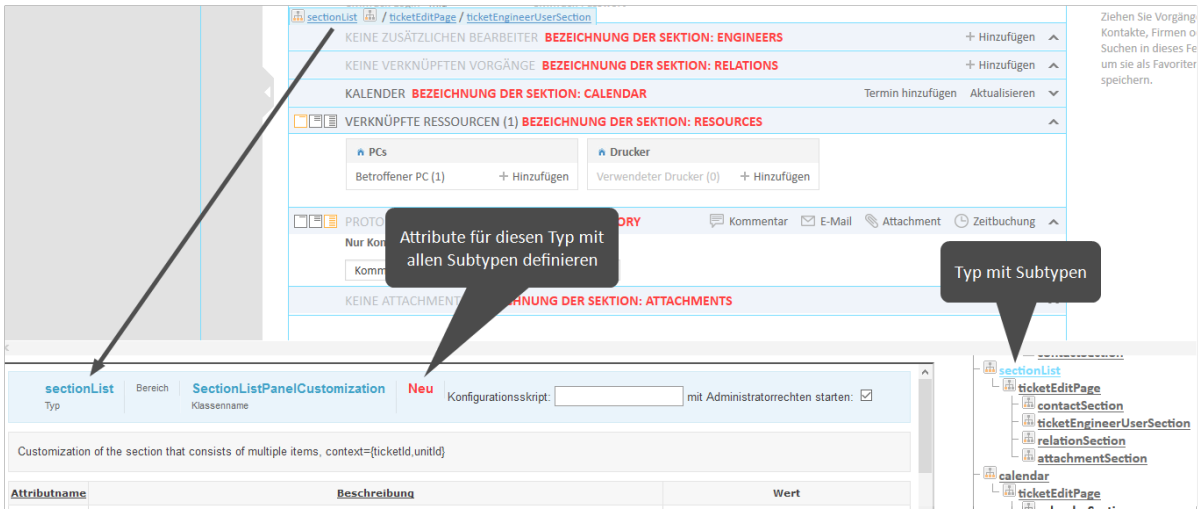


Abbildung 112: *Auswahl der Ebene für die Seitenanpassung, hier: Typ.*

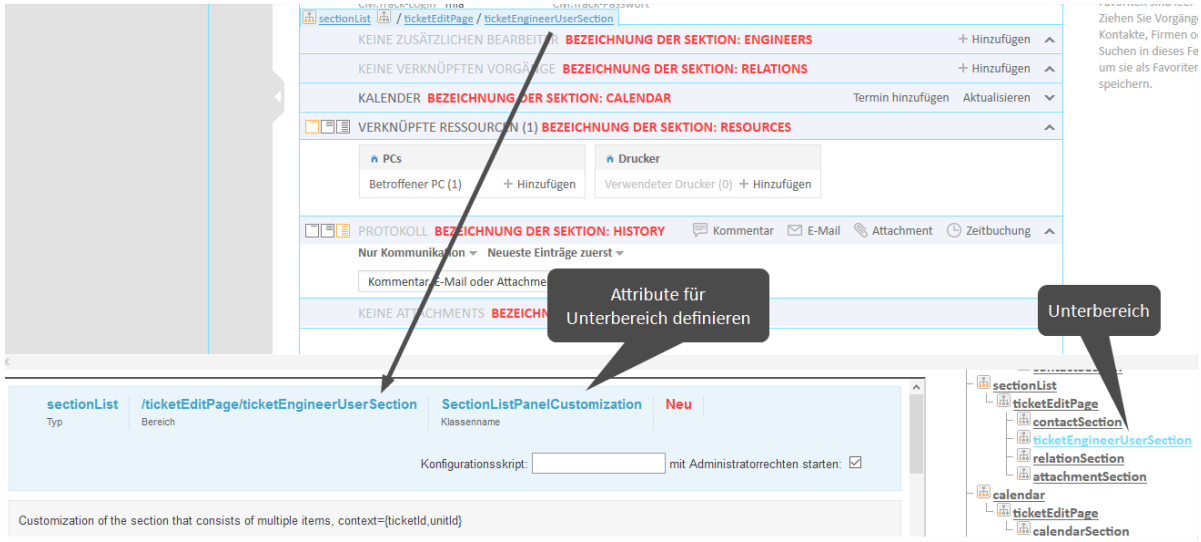


Abbildung 113: *Auswahl der Ebene für die Seitenanpassung, hier: Unterbereich.*

Im folgenden Beispiel ist dies anhand der Konfiguration der Ticketliste dargestellt.

- **Ebene *Typ*:**

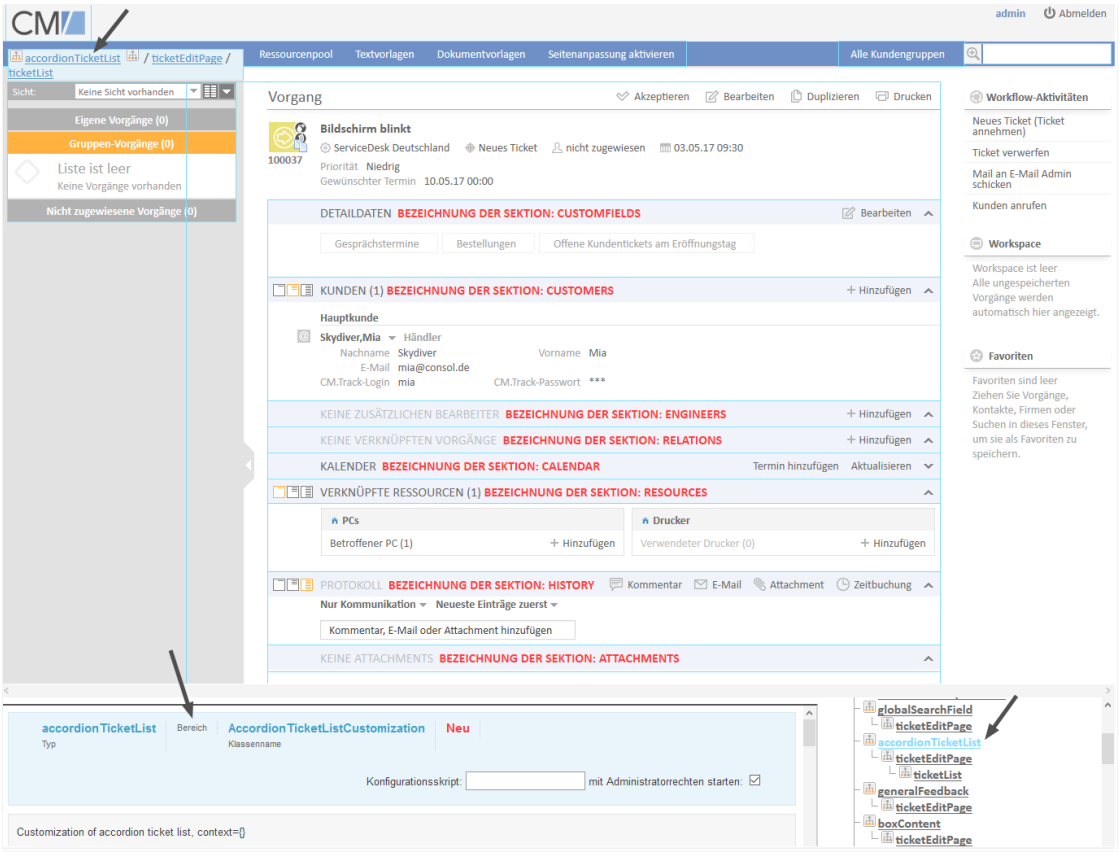


Abbildung 114: *ConSol CM Web Client - Definieren von Attributen auf der Ebene Typ*

- **Ebene *Bereich*:**

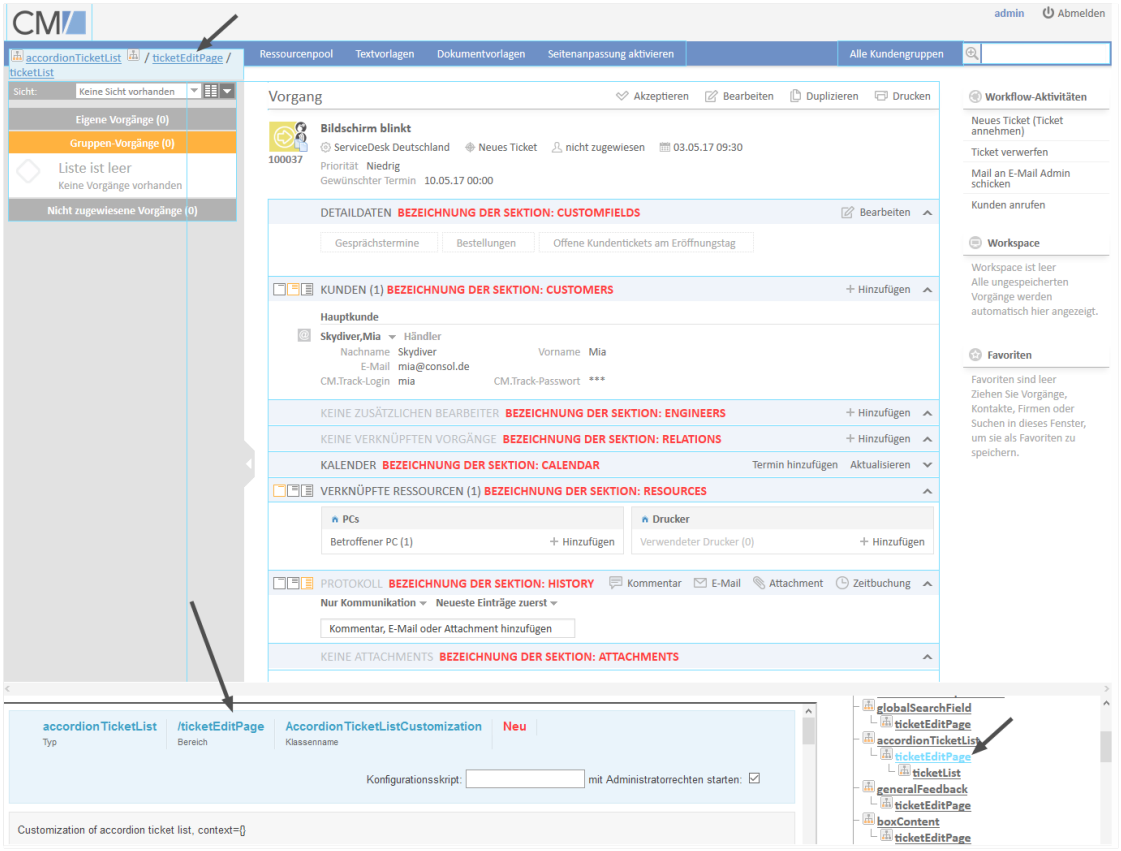


Abbildung 115: *ConSol CM Web Client - Definieren von Attributen auf der Ebene Bereich*

- **Ebene *Unterbereich*:**

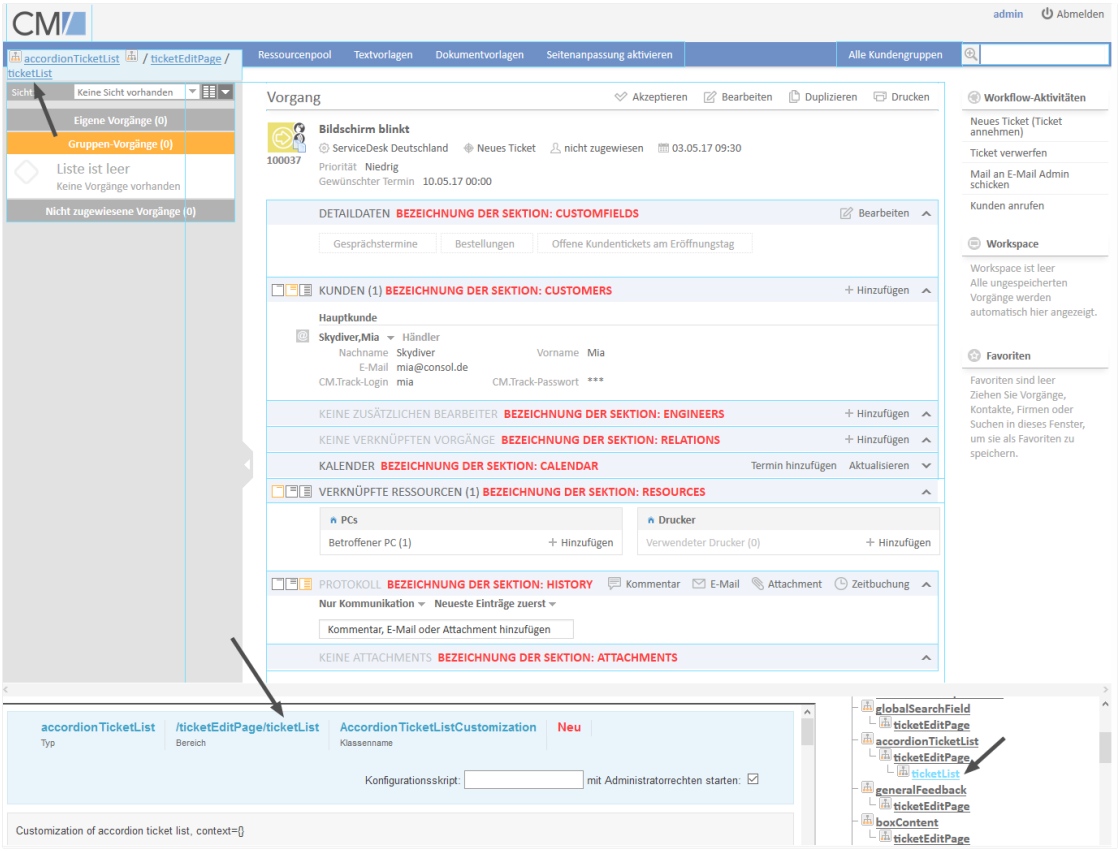


Abbildung 116: *ConSol CM Web Client - Definieren von Attributen auf der Ebene Unterbereich*

Für manche Elemente kann es zwei Pfade in der Hierarchie geben, wie im folgenden Beispiel für den Unterbereich *contactSection* gezeigt:

- Pfad 1: customerSectionPanel / ticketEditPage / contactSection
- Pfad 2: sectionList / ticketEditPage / contactSection

In diesem Beispiel kann der Unterbereich *ticketEditPage / contactSection* also für zwei unterschiedliche Typen konfiguriert werden: *customerSectionPanel* und *sectionList*.

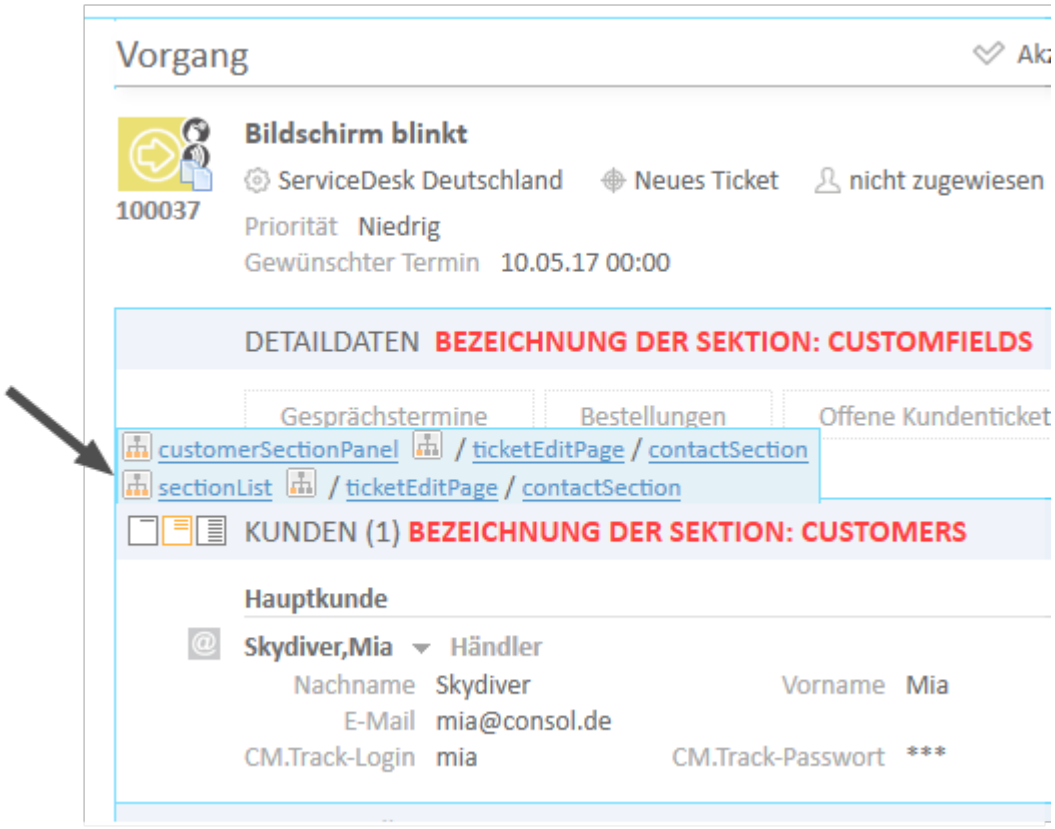


Abbildung 117: *ConSol CM Web Client - Zwei Pfade zum Seitenanpassungselement contactSection*

Für jedes Element gibt zudem ein Konfigurationsskript, mit dem die Werte dynamisch definiert werden können. Das Skript wird im Kontext des aktuell angemeldeten Bearbeiters ausgeführt. Die Berechtigungen des Bearbeiters können das Ergebnis des Skripts also beeinflussen. Wenn Sie zum Beispiel Code verwenden, um **alle Tickets** auszuwählen, wird die Anzahl der Tickets je nach Queue-Berechtigungen des aktuellen Bearbeiters unterschiedlich sein. Wenn Sie innerhalb des Skripts globalen Zugriff benötigen, können Sie es immer auch mit Administratorrechten ausführen, indem Sie die entsprechende Checkbox markieren. Innerhalb des Skripts sind die Objekte des Kontextes verfügbar (siehe folgende Abbildung: Beispiel mit `ticketId`).

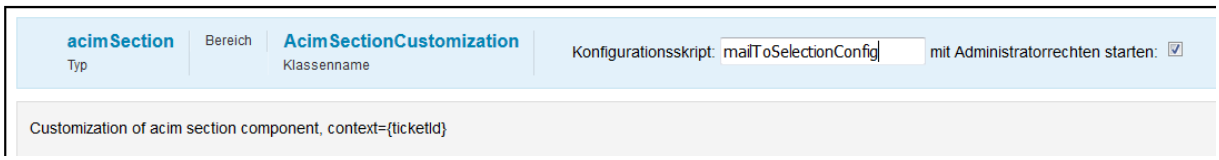


Abbildung 118: *ConSol CM Web Client - Konfigurationsskript zur Definition von Werten*

Es soll zum Beispiel für Tickets mit mittlerer oder niedriger Priorität nur einen E-Mail-Empfänger (hier: den Hauptkunden) geben, bei einem Ticket mit hoher Priorität sollen aber alle Kunden des Tickets eine E-Mail erhalten. Dafür kann folgendes Skript verwendet werden:

```
import com.consol.cmas.common.model.ticket.Ticket
import com.consol.cmas.common.model.customfield.enums.EnumValue

Ticket ticket = ticketService.getById(ticketId);
EnumValue value = ticket.get("helpdesk_standard.priority");
if (value != null && "high".equals(value.getName()))
return [mailToSelection: 'contacts'];
return [mailToSelection: 'contact'];
```

Das Skript muss im Bereich *Skripte und Templates* des Admin Tools gespeichert werden (siehe Abschnitt Admin-Tool-Skripte für Details) und sein Name muss im Feld *Konfigurationsskript* angegeben werden.

Die Rückgabewerte definieren die Attributwerte, die alternativ in den Texteingabefeldern definiert werden können. Beachten Sie die Reihenfolge der beteiligten Komponenten, siehe Abschnitt Reihenfolge und Prioritäten bei der Seitenanpassung.

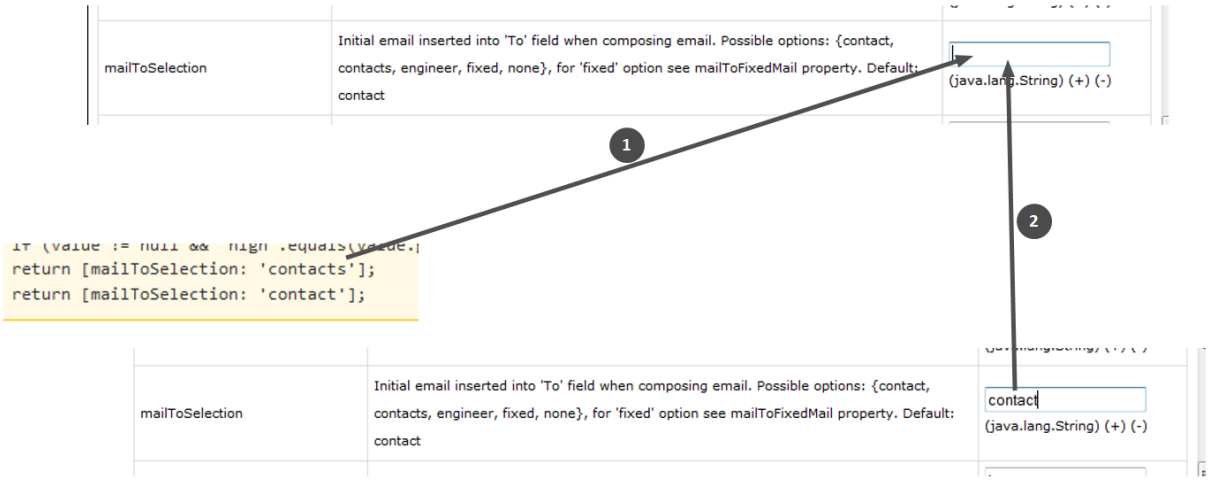


Abbildung 119: *Zwei Alternativen für die Einstellung der Seitenanpassungsattribute*

- Alternative 1: Attribut über Skript zurückgeben (1)
- Alternative 2: Attribut im String-Feld setzen (2)

Beachten Sie, dass in Skripten möglicherweise Boolean-Werte („true“/„false“) zurückgegeben werden müssen. Für ConSol CM-Seitenanpassungsskripte wird eine Syntax verwendet, in der diese Werte als String zurückgeben werden müssen!

Beispiel: Der Wert des Attributs *emailFeature* ist „false“. Im Skript müssen Sie Folgendes schreiben:

```
emailavailable = [emailFeature:'false'] ... return emailavailable
```

C.8.2.1 Buttons zum Setzen von Attributen der Seitenanpassung

Es sind drei Buttons verfügbar:

- **Erzeugen**
  Erzeugt eine neue Seitenanpassung (in der Datenbank). Verwenden Sie diesen Button, um Attribute (neu) zu definieren. Alle Änderungen, die Sie im Formular vorgenommen haben, werden gespeichert.
- **Löschen**
  Löscht alle Einträge für den Typ, Bereich oder Unterbereich aus der Datenbank, d. h. alle im Formular verfügbaren Attribute werden auf ihre ursprünglichen Werte zurückgesetzt.
- **Zurücksetzen**
  Es findet keine Datenbankoperation statt. Es werden lediglich die Einträge rückgängig gemacht, die Sie beim Editieren des Formulars vorgenommen haben und die noch nicht gespeichert wurden.

C.8.3 Reihenfolge und Prioritäten bei der Seitenanpassung

Falls mehr als ein Wert für ein Attribut gesetzt ist, gilt folgende Hierarchie:

1. **Höchste Priorität:** Skript
2. **Mittlere Priorität:** Bereichsdefinition
3. **Niedrigste Priorität:** Typdefinition

Beispiel für den Wert *maxHints* in *GlobalSearchField* auf der *ticketEditPage*:

- **Variante A:**
  - Skript: kein Wert
  - Bereichsdefinition (GlobalSearchField/ticketEditPage): maxHints = 10
  - Typdefinition (GlobalSearchField): maxHints = 5
  - => maxHints ist 10
- **Variante B:**
  - Skript: maxHints = 7
  - Bereichsdefinition (GlobalSearchField/ticketEditPage): kein Wert
  - Typdefinition (GlobalSearchField): maxHints = 5
  - => maxHints ist 7
- **Variante C:**
  - Skript: kein Wert
  - Bereichsdefinition (GlobalSearchField/ticketEditPage): kein Wert
  - Typdefinition (GlobalSearchField): maxHints = 5
  - => maxHints ist 5

C.8.4 Seitenanpassung mit Attributen

In den folgenden Abschnitten werden alle Konfigurationsattribute für die Seitenanpassung erklärt. Dort finden Sie auch auch eine kurze Beschreibung der einzelnen Attribute im Editierbereich.

C.8.4.1 Mögliche Seiten (Bereiche) für die Seitenanpassung

Es stehen folgende Hauptbereiche zur Verfügung, d. h. wenn Sie die entsprechende Seite aufgerufen haben, können Sie die Seitenanpassungsattribute konfigurieren, die nur auf dieser Seite angezeigt werden.

companyEditPage

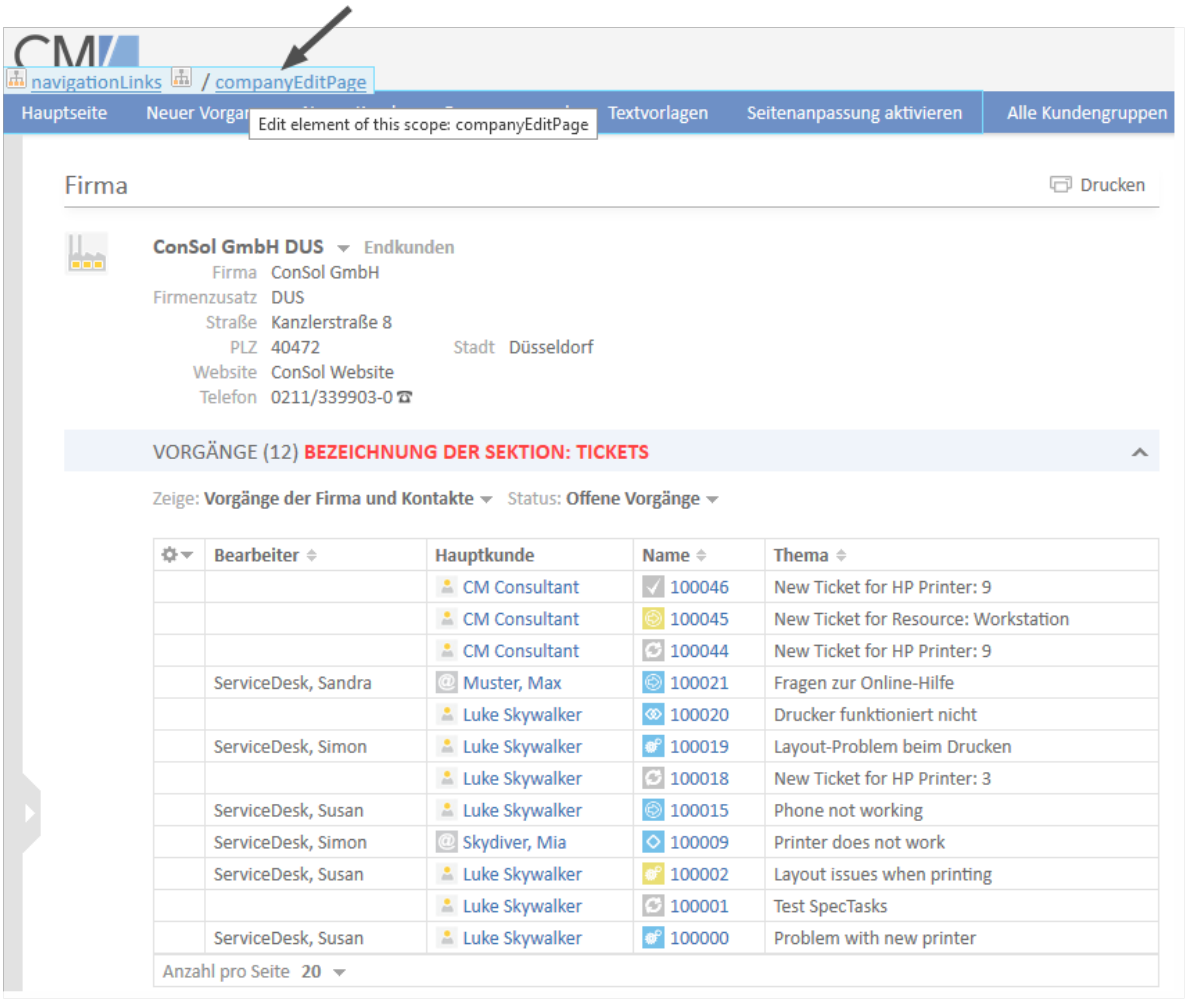


Abbildung 120: *Seitenanpassung für die Firmenseite*

contactCreatePage

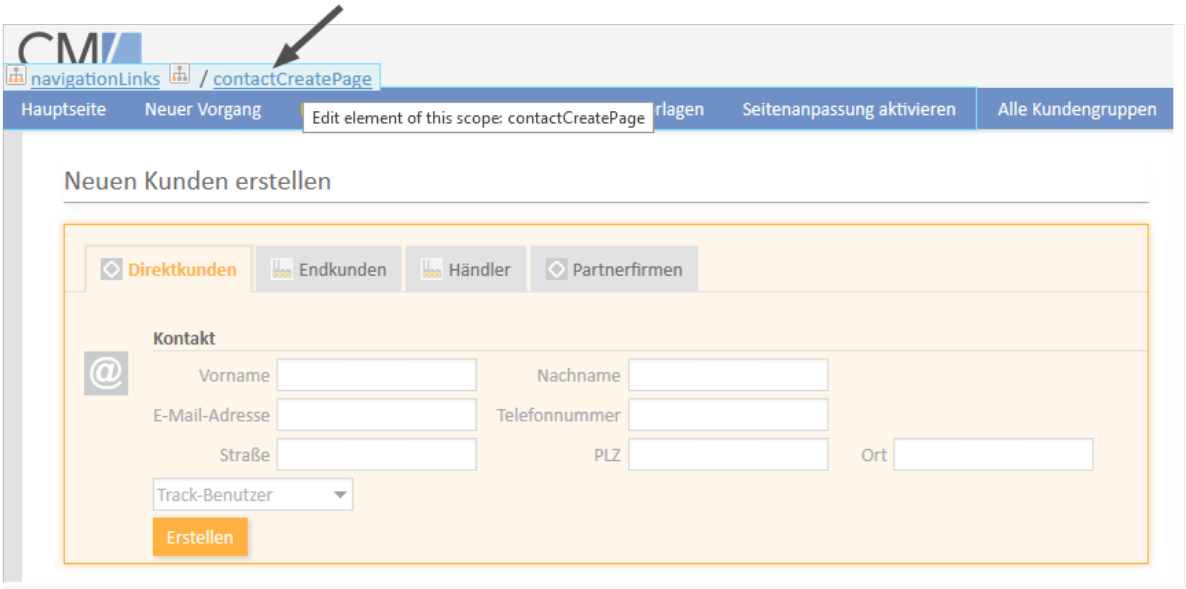


Abbildung 121: *ConSol CM Web Client - Seitenanpassung für die Seite Kunden erstellen*

contactEditPage

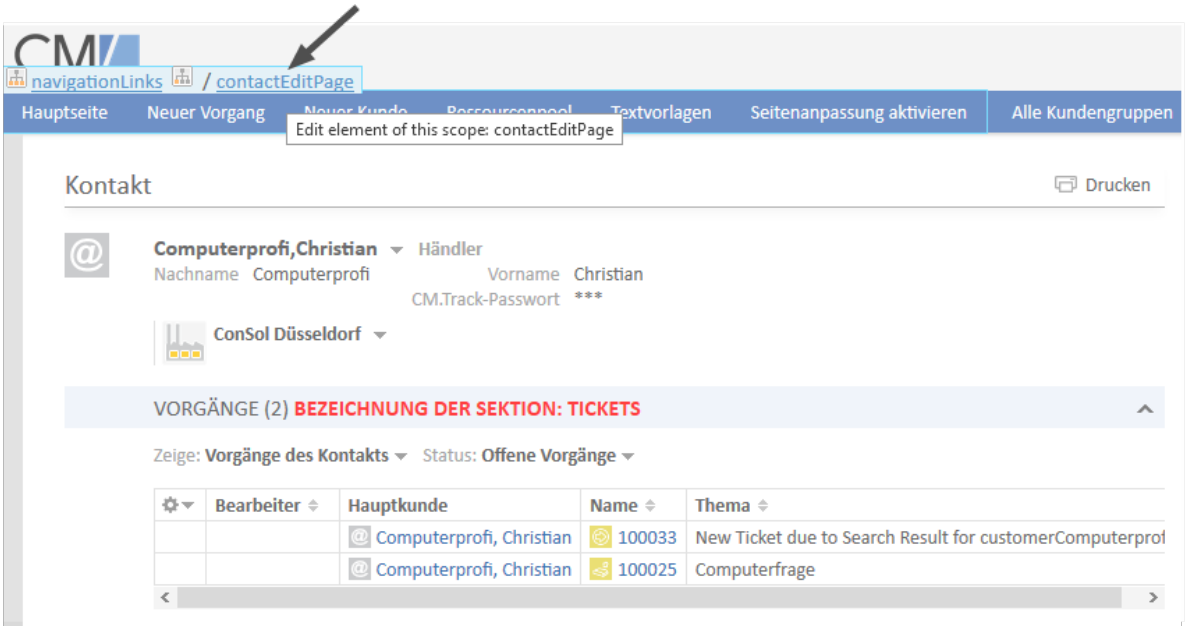


Abbildung 122: *ConSol CM Web Client - Seitenanpassung für die Kontaktseite*

resourceDashboard

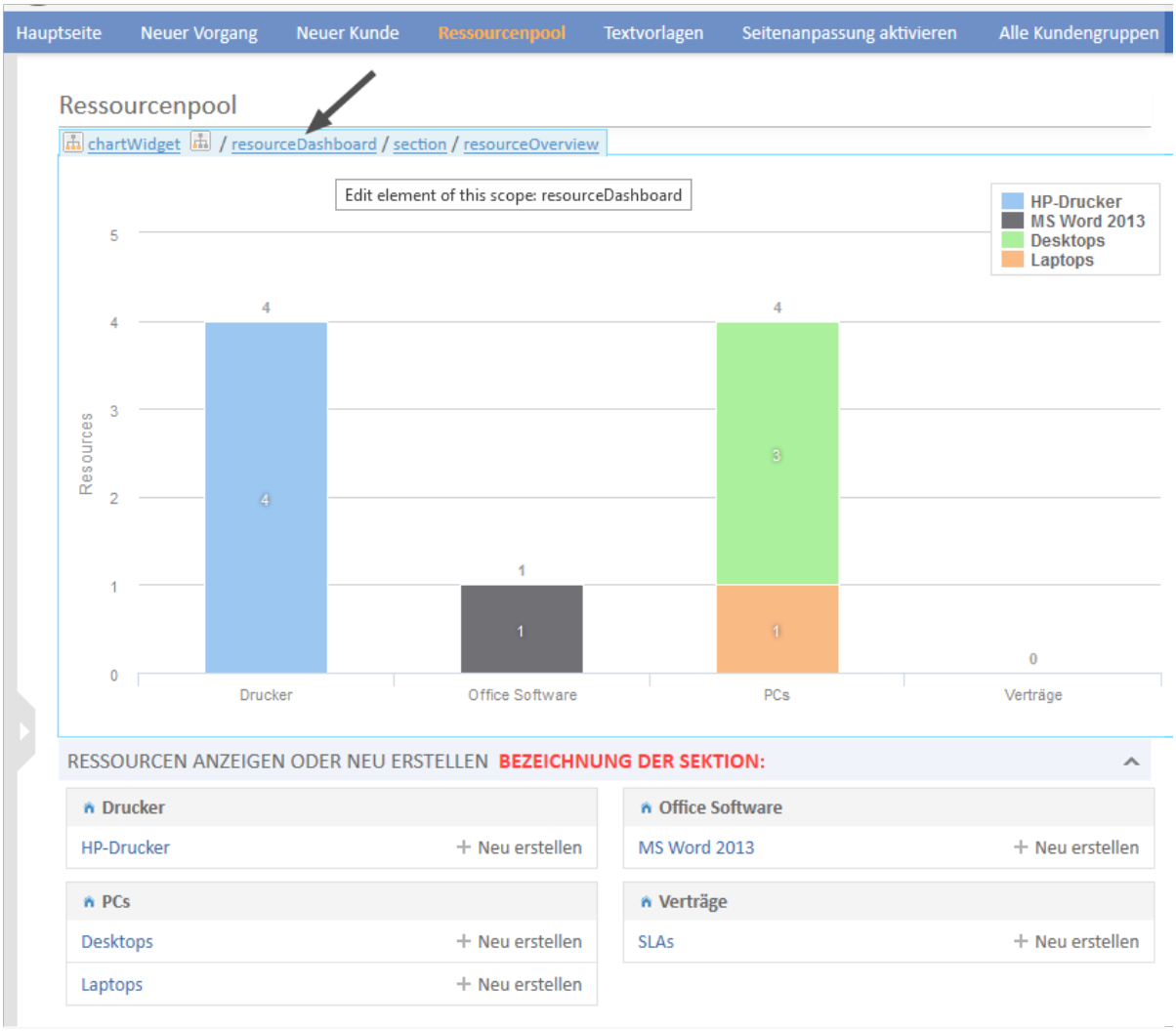


Abbildung 123: *ConSol CM Web Client - Seitenanpassung für das Ressourcenpool-Dashboard*

resourceType (Ressourcentypseite)

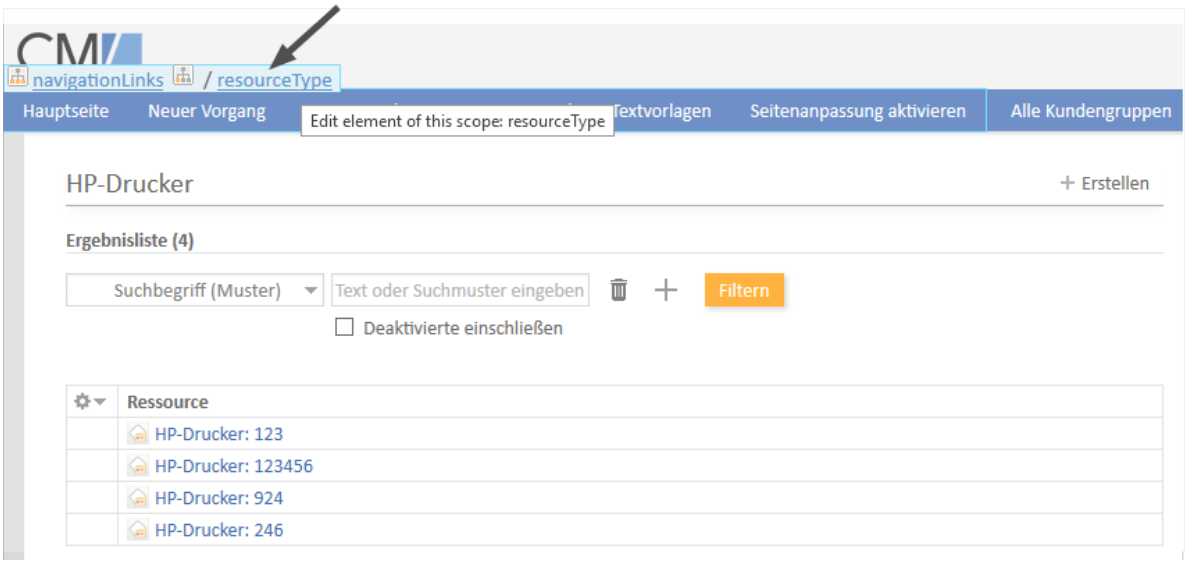


Abbildung 124: *ConSol CM Web Client - Seitenanpassung für die Ressourcentypseite*

resource (Ressourcenseite)

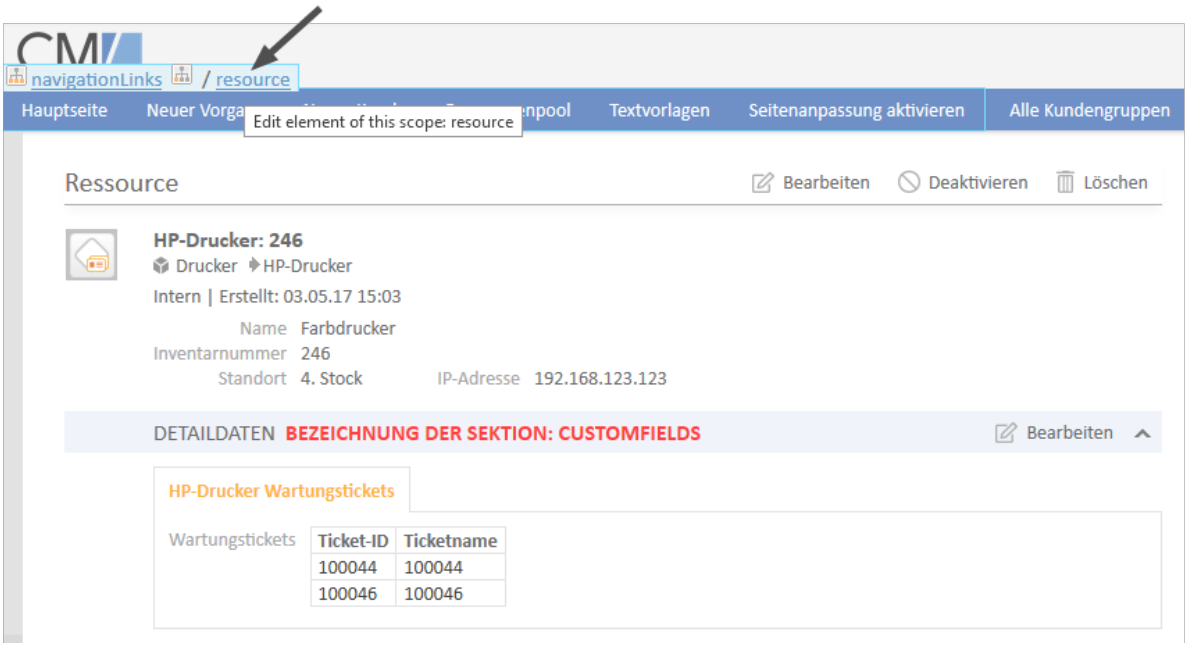


Abbildung 125: *ConSol CM Web Client - Seitenanpassung für die Ressourcenseite*

officeTemplatePage (nur wenn CM/Doc aktiviert ist)

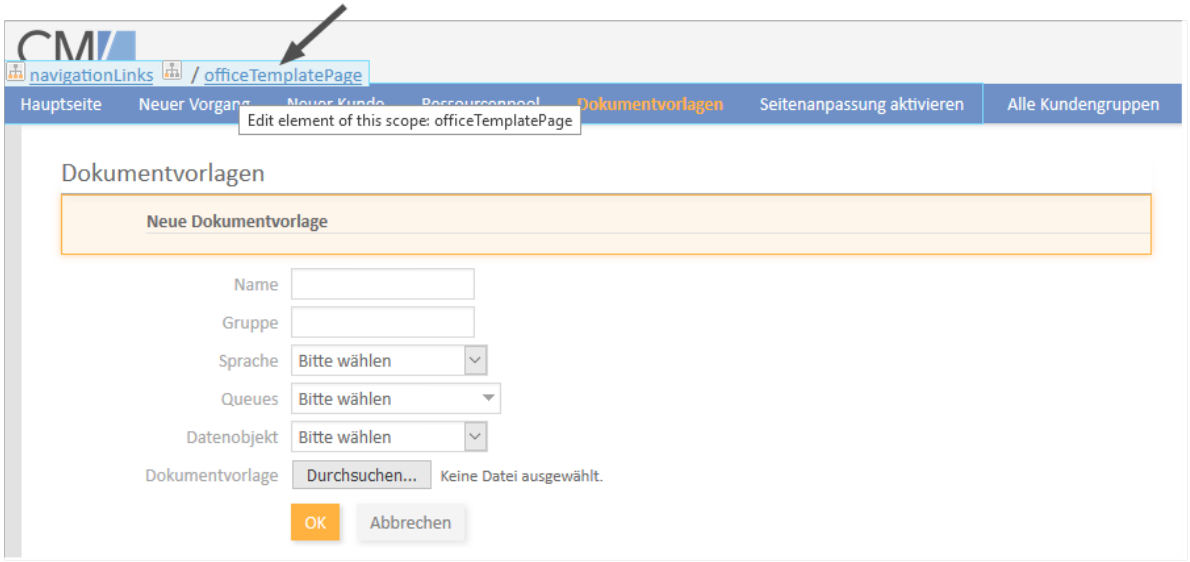


Abbildung 126: *ConSol CM Web Client - Seitenanpassung für die Seite Dokumentvorlagen (nur wenn CM/Doc aktiviert ist)*

searchDetailPage

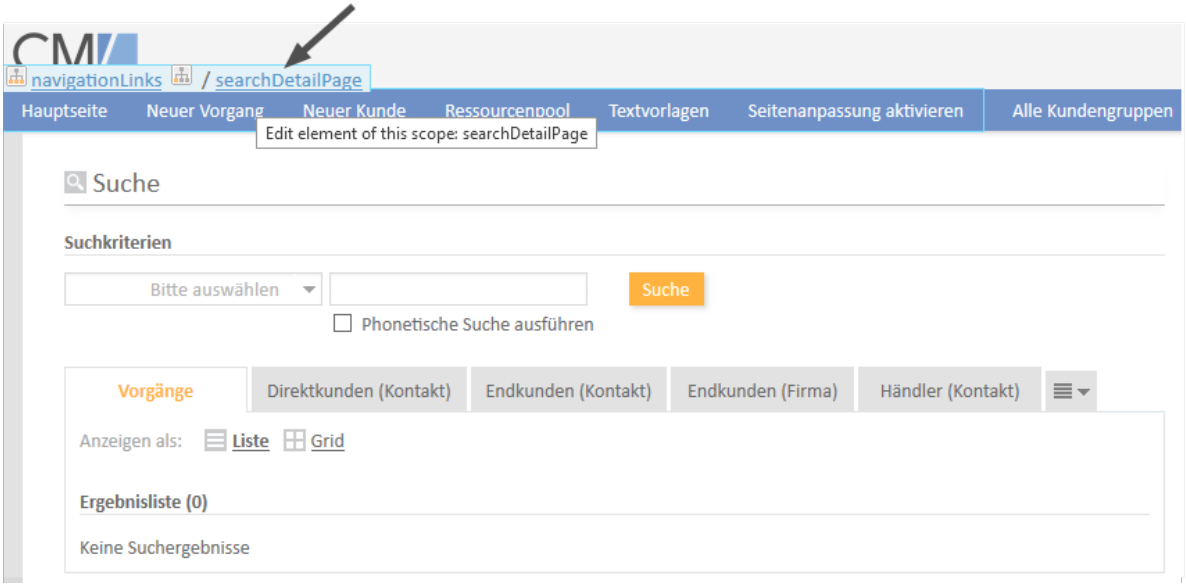


Abbildung 127: *ConSol CM Web Client - Seitenanpassung für die Detailsuche*

templateViewPage

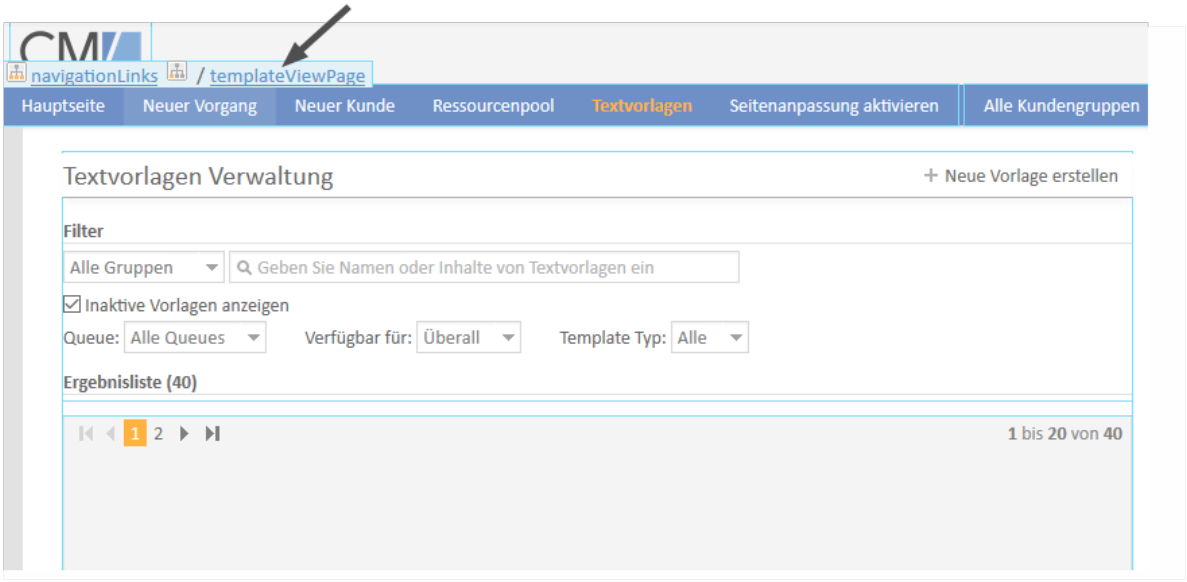


Abbildung 128: *ConSol CM Web Client - Seitenanpassung für die Seite Textvorlagen*

templateEditPage

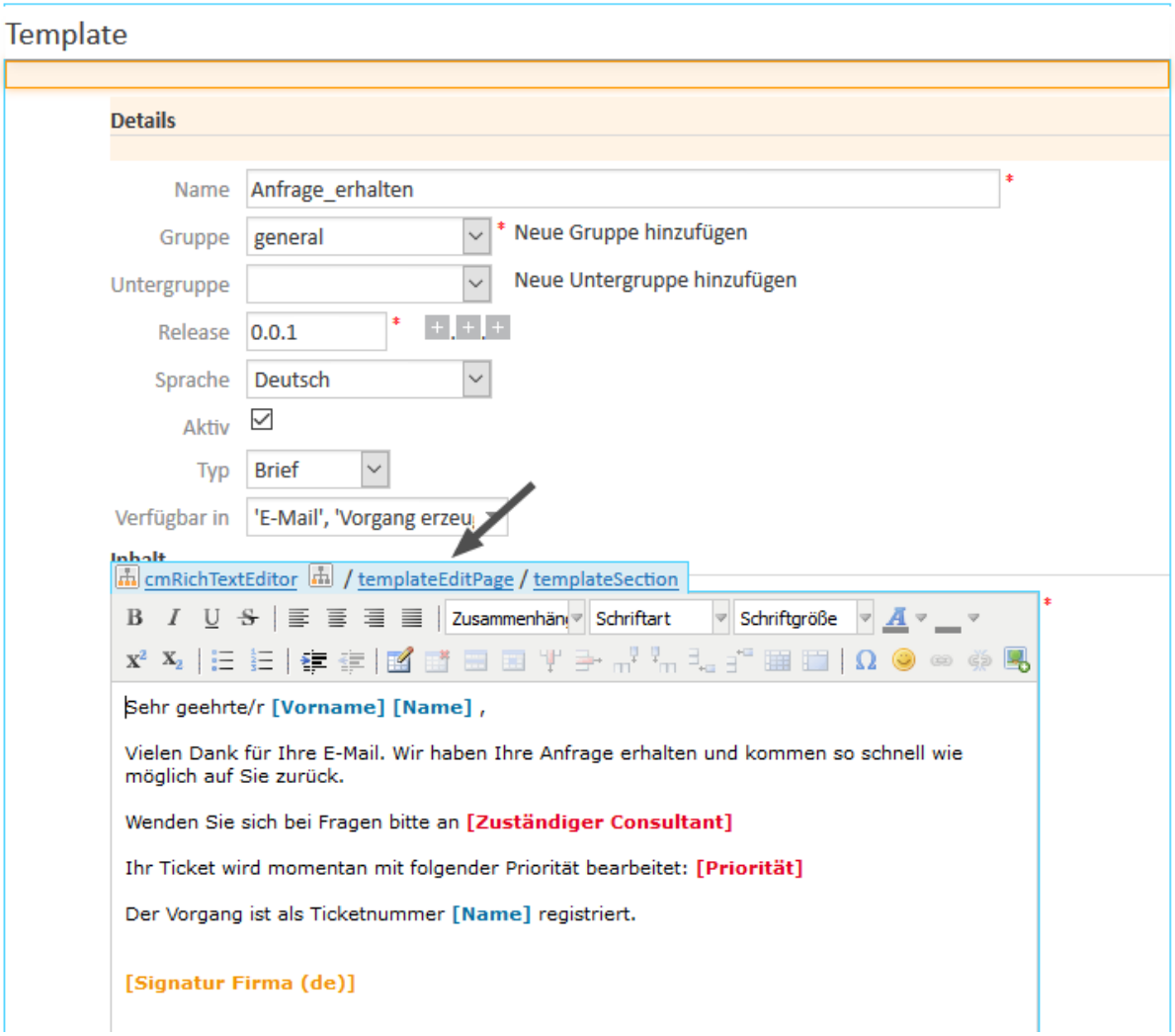


Abbildung 129: *ConSol CM Web Client - Seitenanpassung für die Seite mit den Textvorlagen (Bearbeitungsmodus)*

ticketCreatePage

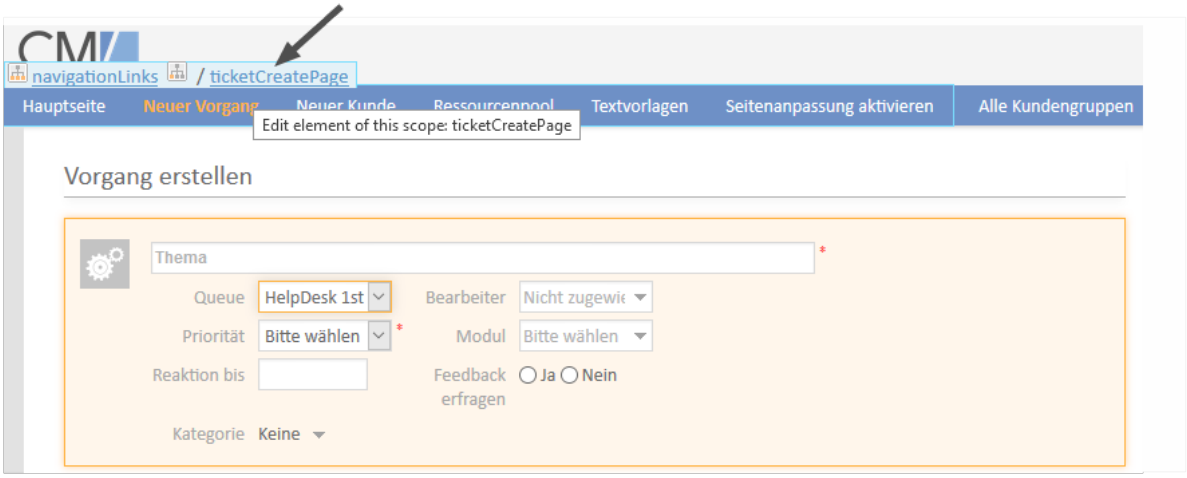


Abbildung 130: *ConSol CM Web Client - Seitenanpassung für die Seite Ticket erstellen*

ticketEditPage

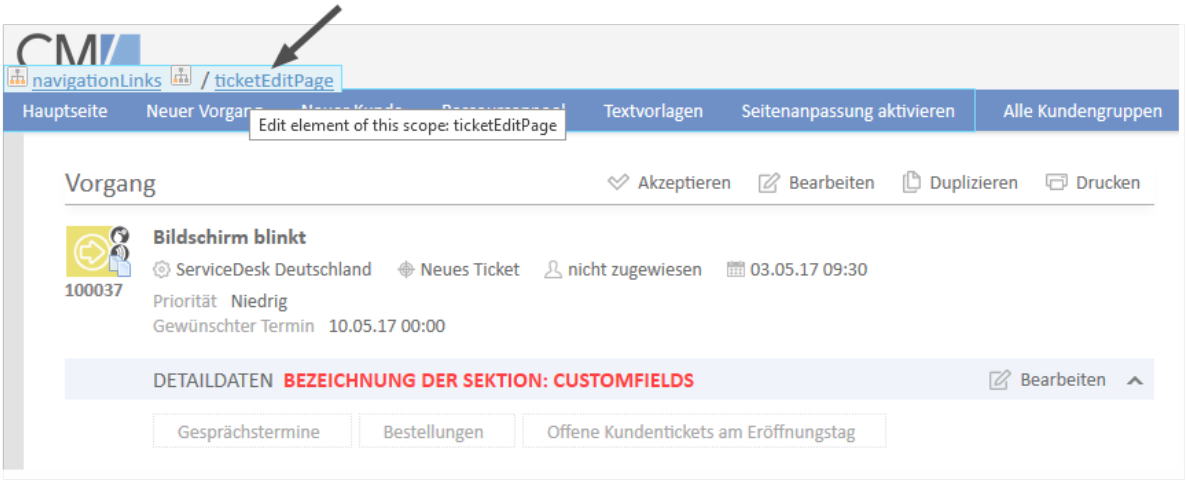


Abbildung 131: *ConSol CM Web Client - Seitenanpassung für die Ticketseite*

userProfilePage

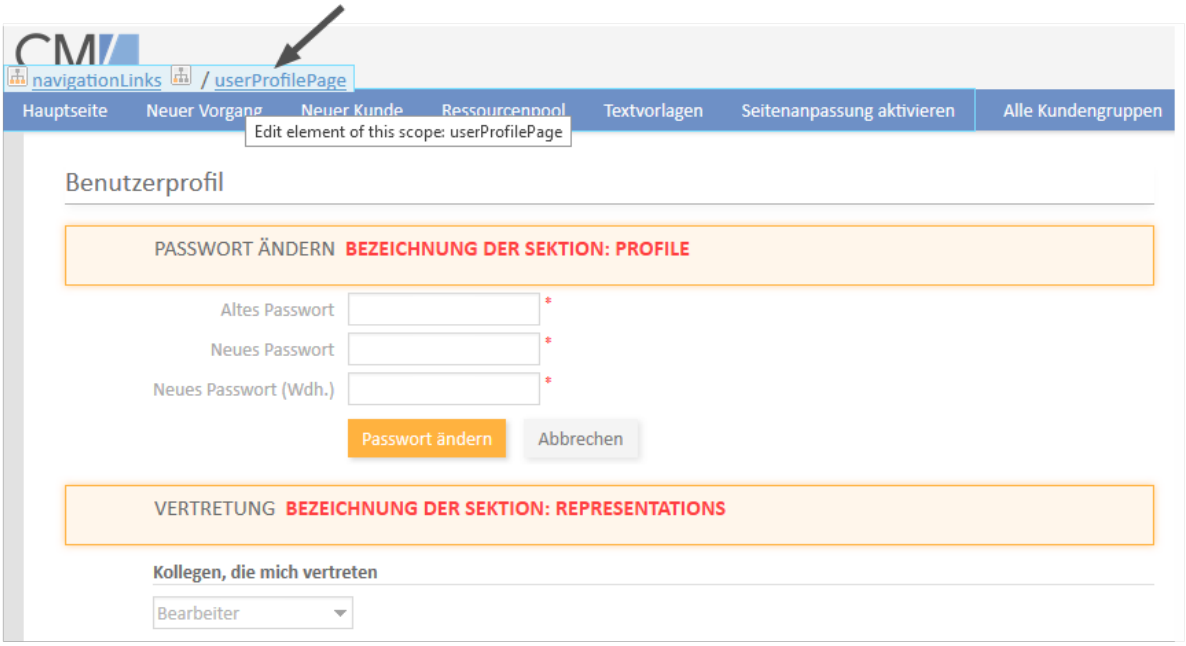


Abbildung 132: *ConSol CM Web Client - Seitenanpassung für das Bearbeiterprofil*

welcomePage (Hauptseite)

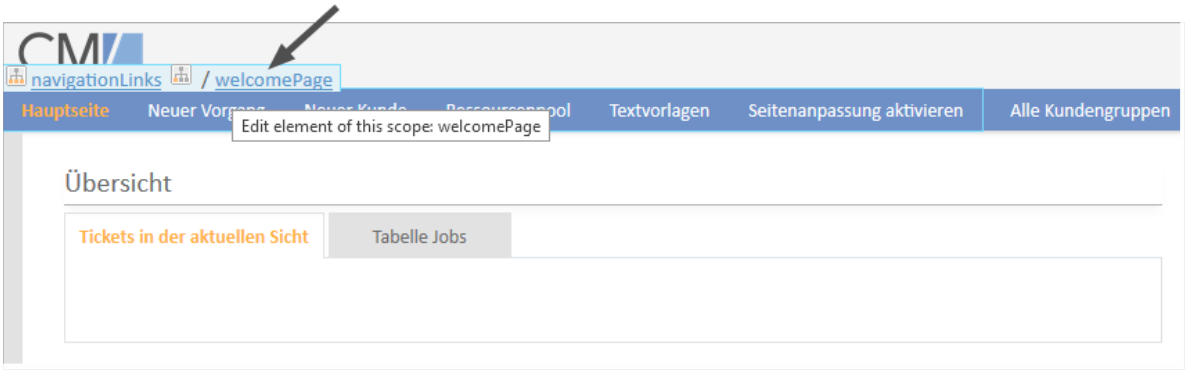


Abbildung 133: *ConSol CM Web Client - Seitenanpassung für die Hauptseite*

Für den Typ *globalSearchField* (siehe folgender Abschnitt) können z. B. folgende seitenspezifische Bereiche konfiguriert werden. Das bedeutet, dass das Verhalten von *globalSearchField* (Typ) für jede der folgenden Seiten (Bereiche) konfiguriert werden kann, auf denen der Typ verfügbar ist:

- globalSearchField/companyEditPage
- globalSearchField/contactCreatePage
- globalSearchField/contactEditPage
- globalSearchField/resource

- globalSearchField/resourceDashboard
- globalSearchField/resourceType
- globalSearchField/searchDetailPage
- globalSearchField/ticketCreatePage
- globalSearchField/ticketEditPage
- globalSearchField/welcomePage

C.8.5 Seitenanpassungsattribute für Typen, Bereiche und Unterbereiche in alphabetischer Reihenfolge

Die folgenden Abschnitte enthalten eine Liste der Typen, Bereiche und Unterbereiche der Seitenanpassung. Die einzelnen Attribute werden auf der jeweiligen Seite beschrieben.

accordionTicketList (Typ) 221
acimSection (Typ oder Unterbereich) 222
attachmentSection (Typ oder Unterbereich) 234
autocomplete 235
boxContent 236
calendarSection 237
cmApplicationCustomization (Typ) 238
cmRichTextEditor 239
customerGroupSelector 241
customerSectionPanel 241
customerTickets 245
detailSearch (Typ) 247
engineerAutocomplete 256
enumAutocomplete 256
generalFeedback 257
globalSearchField (Typ) 257
mailTemplate (Typ) 259
markersLibrary (Typ) 261
navigationLinks (Typ) 261
preview (Typ) 263
resourceRelations 265
resourceRelationsPanel 266
resourceTypes 266
section (Typ) 266
sectionList (Typ) 266
table 269
template 269
ticketList (Unterbereich) 270
ticketPanel 270
TicketRelation 272

ticketsAutocomplete 273
ticketsBookingAutocomplete 273
timeBookingSection 273
unitAutocomplete 274
unitFormPanel 274
unitRelationSection (Typ UnitSection) 275
UnitResourceRelation 276
unitSearch 277
unitSearchHeader 278
viewDiscriminatorsSection (Typ) 278
welcomePage 280

accordionTicketList (Typ)

Hier können Sie Attribute für die Ticketliste definieren. Es ist nur ein Unterbereich verfügbar: *ticketList*

Attribute:

- **hideEmptyGroups**
  Legt fest, ob leere Gruppen in der Ticketliste ausgeblendet werden. Wenn dieses Attribut auf „false“ (Standard) gesetzt ist, werden alle Gruppen angezeigt. Wenn dieses Attribut auf „true“ gesetzt ist, werden leere Gruppen ausgeblendet.

  Beachten Sie, dass eine Gruppe in der Ticketliste nicht ausgeblendet wird, wenn der Fokus auf ihr liegt. In diesem Fall wird die Gruppe angezeigt, wenn die Ticketliste geladen wird, und ausgeblendet, sobald der Benutzer eine andere Gruppe anklickt.
- **loadingTicketListMode**
  Der für die Darstellung der Ticketliste verwendete Modus. Sie können eine von vier Optionen auswählen:
  1. LAZY_SYNCH (Standardwert)
     Das Wartesymbol wird anstelle der Ticketliste angezeigt, solange der Rest der Seite geladen wird. Danach wird die Ticketliste geladen. Der Hauptvorteil dieser Option ist, dass der Inhalt der Hauptseite schneller angezeigt/gelesen werden kann.
  2. LAZY_ASYNCH
     (seit Version 6.8.2 überholt (deprecated), wird als LAZY_SYNCH-Modus interpretiert)
     Eine Abwandlung der LAZY_SYNCH-Strategie. Es wird nicht gewartet, während die Seite geladen wird, sondern es wird ein zweiter Request gesendet, mit dem die Ticketliste geladen wird. Mit dieser Strategie wird die Ticketliste schneller geladen, aber die Hauptseite ist möglicherweise nicht mehr sofort verfügbar.
  3. INCLUDED
     Die Ticketliste wird zusammen mit dem Rest der Seite geladen.
  4. LAZY_ASAP
     Das Wartesymbol wird anstelle der Ticketliste angezeigt, solange der Rest der Seite geladen wird. Der Request für die Ticketliste wird gesendet, sobald die zwei erforderlichen Bibliotheken geladen wurden. Bei dieser Option wird ein Request für die Ticketliste gesendet und gleichzeitig mit dem ersten Request verarbeitet. Die Ticketliste erscheint schneller auf der Seite. (java.lang.String)
- **mainCustomerDescriptionVisible**
  Das Seitenanpassungsattribut *accordionTicketList.**mainCustomerDescriptionVisible***={true, false} ersetzt die Annotation `show-contact-in-ticket-list` (die bis CM-Version 6.8 gültig war). Wenn dieser Wert auf „true“ gesetzt ist, werden die Kundendaten des Hauptkunden in der Ticketlistendarstellung des Tickets angezeigt. Der Standardwert ist „true“. (boolean)
- **quickAssignLinkShowsTicketPageFlag**
  Legt fest, ob das zugewiesene Ticket nach Klicken auf den Schnellzuweisungs-Link (dargestellt durch einen Pfeil am Ticket-Icon aller nicht zugewiesenen Tickets) sofort geöffnet wird (boolean, Standardwert ist „false“).

- **ticketDataConfigQueueGroupingScript**
  Definiert die Anzeige von Ticketinformationen, wenn die Gruppierung nach Queue ausgewählt ist. Siehe Abschnitt Konfiguration der Ticketliste.
- **ticketDataConfigEngineerGroupingScript**
  Definiert die Anzeige von Ticketinformationen, wenn die Gruppierung nach Bearbeiter ausgewählt ist. Siehe Abschnitt Konfiguration der Ticketliste.
- **ticketDataConfigCustomGroupingScript**
  Definiert die Anzeige von Ticketinformationen, wenn die Gruppierung nach einem Ticketfeld ausgewählt ist. Siehe Abschnitt Konfiguration der Ticketliste.
- **ticketDataConfigNoGroupingScript**
  Definiert die Anzeige von Ticketinformationen, wenn kein Gruppierungskriterium ausgewählt ist. Siehe Konfiguration der Ticketliste.

acimSection (Typ oder Unterbereich)

Ein **ACIM** (Aktivitätselement) ist ein Eintrag im Protokollbereich eines Tickets. Das kann Folgendes sein ...

- ein Kommentar
- eine E-Mail, die aus dem Ticket gesendet wurde
- eine E-Mail, die vom Ticket erhalten wurde
- ein Attachment
- ein Zeitbuchungseintrag

Eine ACIM-Gruppe ist eine Gruppe von Einträgen mit einem eigenen Datums-/Zeitstempel. Ein ACIM-Element ist ein einzelner Eintrag in der ACIM-Gruppe. Er hat nur einen Zeitstempel.

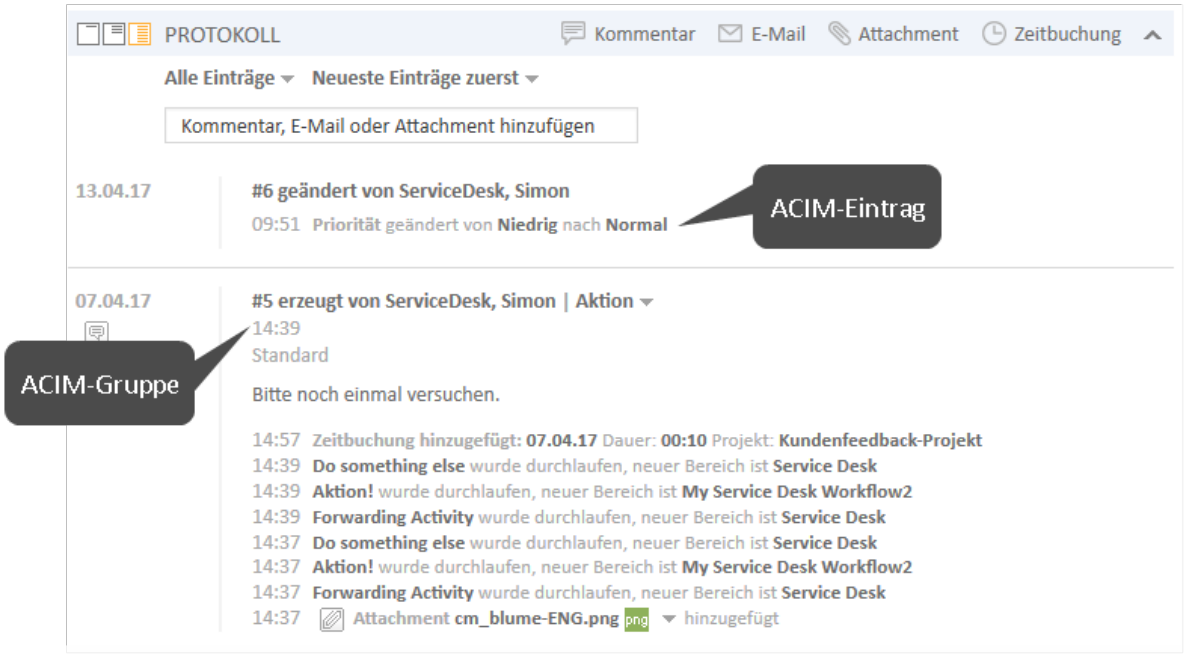


Abbildung 134: *ConSol CM Web Client - ACIM-Gruppe und -Element*

> ⚠ Stellen Sie sicher, dass das Datumsformat, das Sie für eines der folgenden Datumsattribute eingegeben haben, korrekt ist! Wenn das Datumsformat nicht korrekt ist, kann die komplette Seite nicht angezeigt werden! Der Web Client funktioniert in diesem Fall nicht! In der folgenden Tabelle finden Sie die korrekten Datumsformate. Sie können ein Leerzeichen (" ") als Wert eingeben, um den Datums-/Zeitstempel ganz auszublenden.

Letter	Date or Time Component	Examples
G	Era designator	AD
y	Year	1996; 96
M	Month in year	July; Jul; 07
w	Week in year	27
W	Week in month	2
D	Day in year	189
d	Day in month	10
F	Day of week in month	2
E	Day in week	Tuesday; Tue
a	Am/pm marker	PM
H	Hour in day (0-23)	0
k	Hour in day (1-24)	24
K	Hour in am/pm (0-11)	0
h	Hour in am/pm (1-12)	12
m	Minute in hour	30
s	Second in minute	55
S	Millisecond	978
z	Time zone	Pacific Standard Time; PST; GMT-08:00
Z	Time zone RFC 822	-0800

Abbildung 135: *ConSol CM Web Client - Gültige Datumsformate für die ACIM-Datumskonfiguration*

Attribute:

- **acimGroupActionEntryDateFormat**
  Datumsformat für die ACIM-Gruppe ohne Text- oder E-Mail-Eintrag (d. h. für automatische Aktionen). Wenn hier kein Format eingegeben wurde, wird das Standardformat verwendet.
  Syntax: dateFormatFirstLevelOfDetails|secondLevel|thirdLevel
  (java.lang.String, Standardwert = dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm)

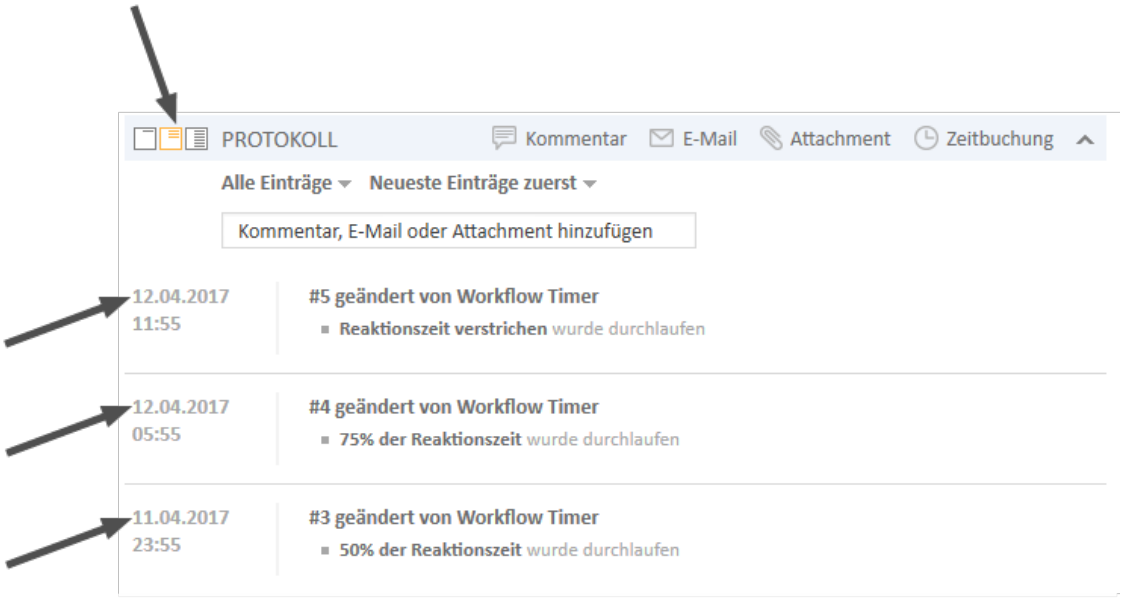


Abbildung 136: *ConSol CM Web Client - Anzeige für Format: dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm*

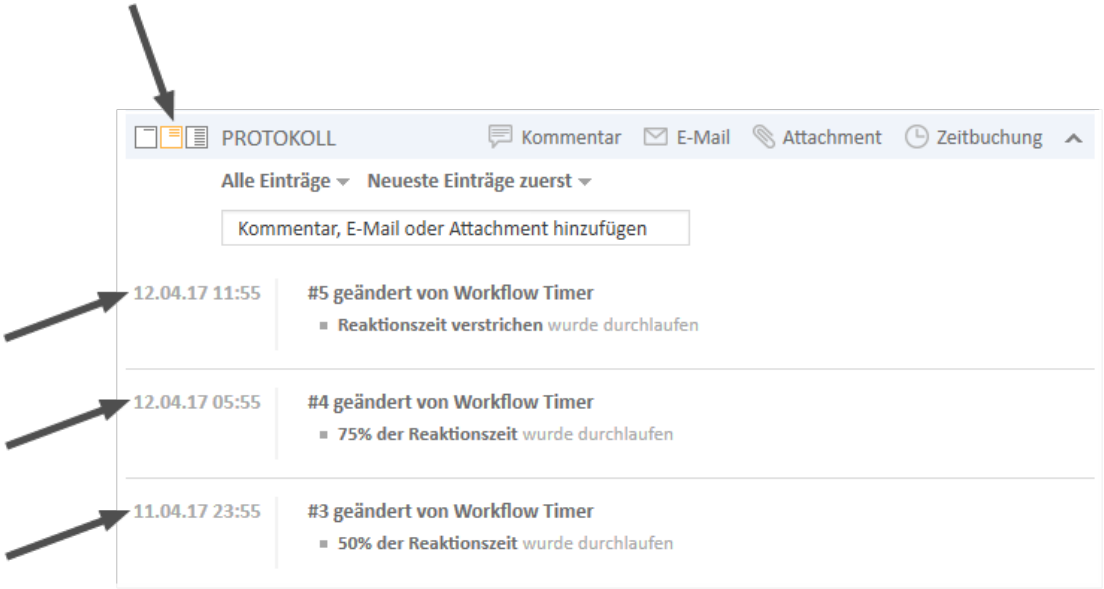


Abbildung 137: *ConSol CM Web Client - Anzeige für Format: dd.MM.yy HH.mm|dd.MM.yy HH.mm|dd.MM.yy HH.mm*

- **acimGroupTextEntryDateFormat**
  Datumsformat für ACIM-Gruppe mit Text-, E-Mail- oder Attachment-Eintrag. Wenn hier kein Format eingegeben wurde, wird das Standardformat verwendet.
  Syntax: dateFormatFirstLevelOfDetails|secondLevel|thirdLevel
  (java.lang.String, Standardwert = dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm|dd.MM.yyyy

HH.mm)

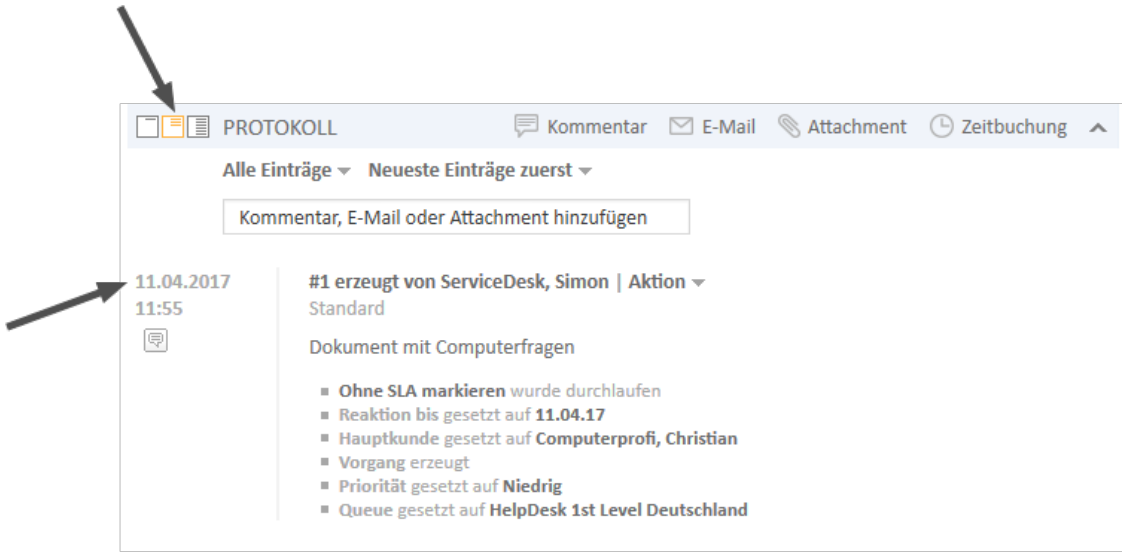


Abbildung 138: *ConSol CM Web Client - Anzeige für Format: dd.MM.yyyy HH.m-m|dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm*

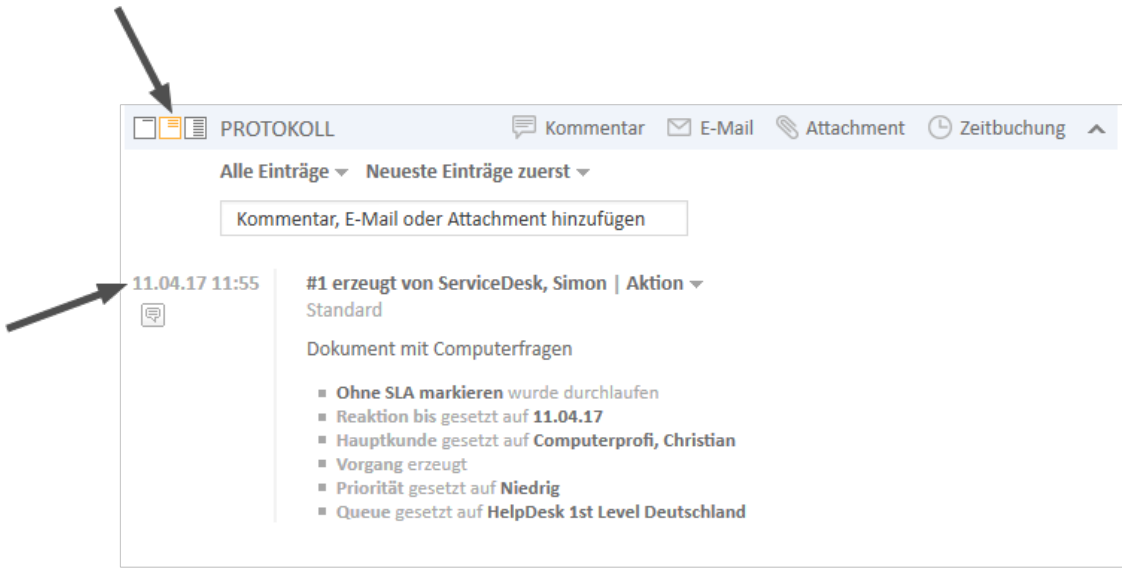


Abbildung 139: *ConSol CM Web Client - Anzeige für Format: dd.MM.yy HH.mm|dd.MM.yy HH.mm|dd.MM.yy HH.mm*

- **acimItemActionEntryDateFormat**
  Datumsformat für ein Element eines ACIM-Eintrags. Wenn hier kein Format eingegeben wurde, wird das Standardformat verwendet.

Syntax: dateFormatFirstLevelOfDetails|secondLevel|thirdLevel
(java.lang.String, Standardwert = dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm)

- **acimItemTextEntryDateFormat**
  Datumsformat für einen Text- oder E-Mail-Eintrag. Wenn hier kein Format eingegeben wurde, wird das Standardformat verwendet.
  Syntax: dateFormatFirstLevelOfDetails|secondLevel|thirdLevel
  (java.lang.String, Standardwert = dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm)
- **showCloneOption**
  Aktiviert die Option *Duplizieren* für einen ACIM-Texteintrag (Kommentar oder E-Mail). (boolean)

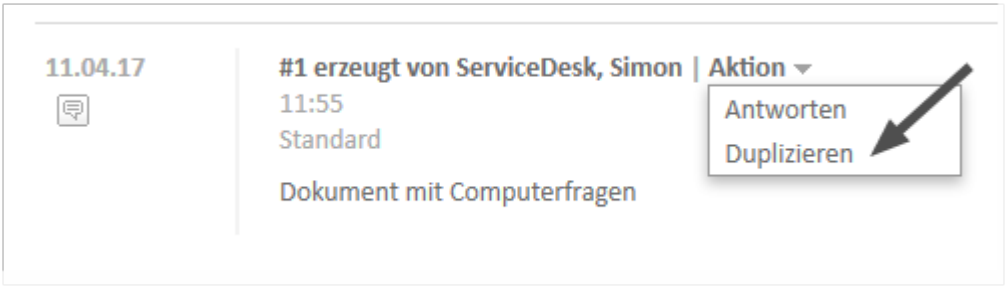


Abbildung 140: *ConSol CM Web Client - Option zum Duplizieren eines ACIM-Texteintrags*

- **appendOrReplaceOnClone**
  Funktioniert nur, wenn die Option *Duplizieren* auf „true“ gesetzt ist. Wenn der Editor geöffnet ist und schon Text enthält, können Sie den Inhalt beim Duplizieren anhängen oder ersetzen, wenn Sie gleichzeitig für ein anderes Element *Duplizieren* auswählen. Mögliche Werte sind „append“, „replace“. Der Standardwert ist „append“. (java.lang.String)
- **attachmentDeletionAllowedManuallyUploaded**
  Boolean. Aktiviert die Löschfunktion (Eintrag im Kontextmenü) für Ticket-Attachments, die manuell als Datei hochgeladen wurden. Standardwert: „true“.
- **attachmentDeletionAllowedIncomingEmail**
  Boolean. Aktiviert die Löschfunktion (Eintrag im Kontextmenü) für Attachments aus eingehenden E-Mails. Standardwert: „false“.
- **attachmentDeletionAllowedOutgoingEmail**
  Boolean. Aktiviert die Löschfunktion (Eintrag im Kontextmenü) für Attachments aus ausgehenden E-Mails aus dem System. Standardwert: „false“

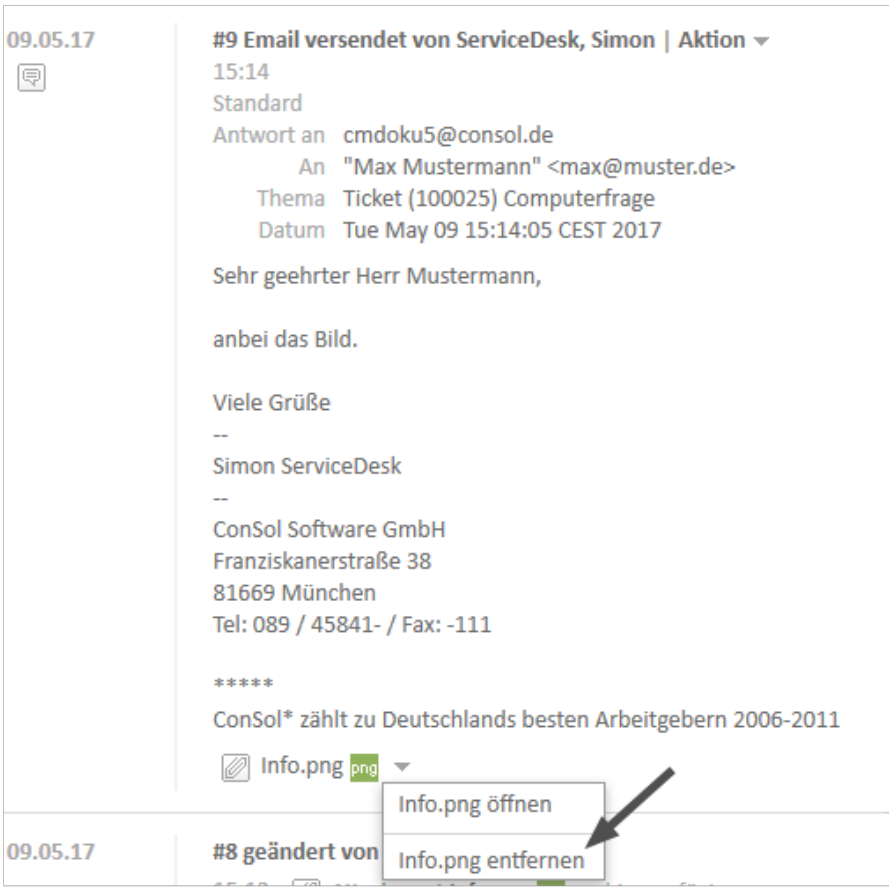


Abbildung 141: *Ausschnitt aus der Ticketseite: attachmentDeletionAllowedManuallyUploaded ist auf true gesetzt, manuell hochgeladenes Attachment kann entfernt/gelöscht werden*

- **headHistoryElementsCount**
  Lazy Loading - Anzahl der Gruppen im ACIM-Bereich, die vom Anfang des Protokolls geladen werden. Der Standardwert ist „0“ (= Lazy Loading ist ausgeschaltet). Die Anpassung funktioniert nur, wenn sie für den Typ konfiguriert wird und nicht für den Bereich. Wenn die Anzahl der Anfangs- und Endelemente „0“ ist, wird das gesamte Protokoll gleichzeitig geladen. (int)
- **tailHistoryElementsCount**
  Lazy Loading - Anzahl der Gruppen im ACIM-Bereich, die vom Ende des Protokolls geladen werden. Der Standardwert ist „0“ (= Lazy Loading ist ausgeschaltet). Die Anpassung funktioniert nur, wenn sie für den Typ konfiguriert wird und nicht für den Bereich. Wenn die Anzahl der Anfangs- und Endelemente „0“ ist, wird das gesamte Protokoll gleichzeitig geladen. (int)

Die Seitenanpassungsattribute *headHistoryElementsCount* und *tailHistoryElementsCount* sind auf drei unterschiedlichen Bereichsebenen verfügbar:

- Typ *acimSection* auf der obersten Ebene ohne Bereich
- Typ *acimSection* mit Bereich */ticketEditPage*
- Typ *acimSection* mit (Unter-)Bereich */ticketEditPage/acimSection*

Ein Wert, der nicht Null ist und Lazy Loading aktiviert, muss entweder nur auf der obersten Ebene ohne Bereich oder auf allen drei Ebenen gesetzt werden.

Andere Verwendungen der Bereiche können zu unerwünschtem Verhalten führen.

Beispiel 1: Lazy Loading ist ausgeschaltet

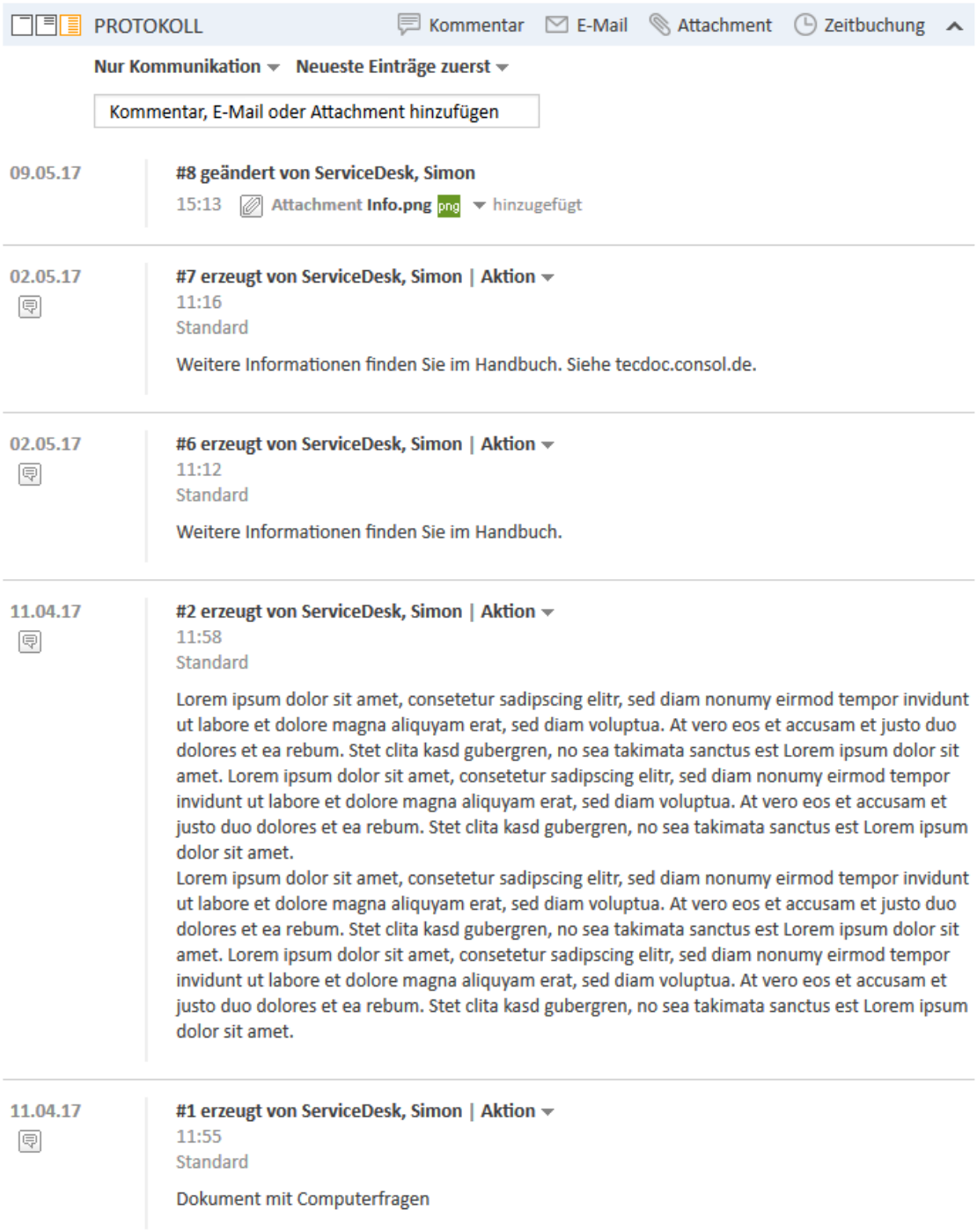


Abbildung 142: *ConSol CM Web Client - Lazy Loading ist ausgeschaltet*

Beispiel 2: headHistoryElementsCount und tailHistoryElementsCount auf „1“ gesetzt

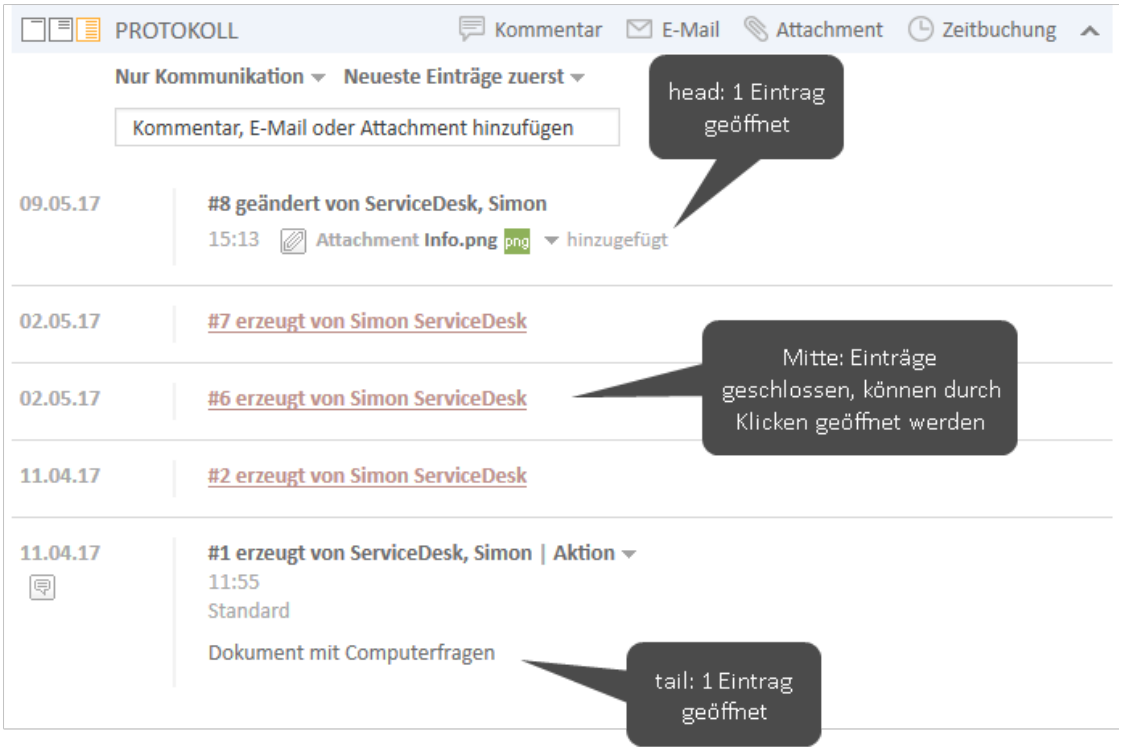


Abbildung 143: *ConSol CM Web Client - headHistoryElementsCount und tailHistoryElementsCount auf 1 gesetzt*

- **mailToSelection**
  Die initiale E-Mail-Adresse, die beim Verfassen einer E-Mail in das Empfängerfeld (An:) eingefügt wird. Standardwert: „contact“ (java.lang.String)
  Mögliche Optionen:
  - **contact**
    Die E-Mail-Adresse des Hauptkontakts wird in das Empfängerfeld kopiert, wenn der Ticket-E-Mail-Editor geöffnet wird.
  - **contacts**
    Die E-Mail-Adressen des Hauptkontakts und aller Zusatzkontakte wird in das Empfängerfeld kopiert, wenn der Ticket-E-Mail-Editor geöffnet wird.
  - **engineer**
    Die E-Mail-Adresse des Bearbeiters wird in das Empfängerfeld kopiert, wenn der Ticket-E-Mail-Editor geöffnet wird.
  - **fixed**
    Details zur Option *fixed* finden Sie beim Attribut *mailToFixedMail*.
  - **none**
    Das Empfängerfeld bleibt nach dem Öffnen des Ticket-E-Mail-Editors leer.

- **mailToFixedMail**
  Wenn das Attribut *mailToSelection* auf „fixed“ gesetzt ist, wird eine einmal festgelegte E-Mail-Adresse verwendet. (java.lang.String)
  Beispiel: E-Mail an eine feste E-Mail-Adresse, *mailToFixedMail* ist auf „foo@bar.de“ gesetzt, *mailToSelection* ist auf „fixed“ gesetzt.

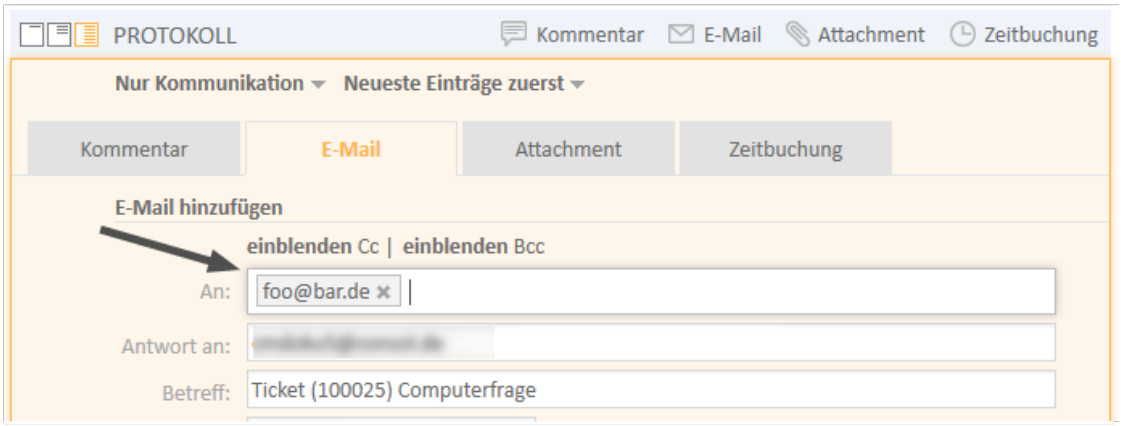


Abbildung 144: *ConSol CM Web Client - E-Mail an eine feste E-Mail-Adresse*

- **recordLastUsedAcimTab**
  Boolean. Der zuletzt verwendete ACIM-Tab wird gespeichert, d. h. wenn Sie den Editor erneut öffnen, wird der Tab (Kommentar/E-Mail/Attachment/Zeitbuchung) geöffnet, der vor dem letzten Schließen des Editors geöffnet und aktiv war (d. h. es wurde eine Aktivität ausgeführt, z. B. Schreiben eines Kommentars).
- **reloadPageIfIE8onAcimSubmit**
  Boolean. Lädt die Seite nach der Übermittlung eines ACIMs neu, nur für IE8. Das ist ein Workaround für das Problem, dass das Hinzufügen von Kommentaren/E-Mails im IE8 lange dauern kann. (boolean)
- **removeContentOnTabSwitch**
  Löscht den Inhalt des Textfeldes jedes Mal, wenn der Bearbeiter im Editor den Tab wechselt. (boolean, Standardwert = „false“, d. h. wenn Sie zum Beispiel vom Ticket-E-Mail-Editor in den Ticket-Kommentar-Editor wechseln, bleibt der im Editor eingegebene Text erhalten und wird nicht gelöscht.
- **timeBookingFeature**
  Boolean. Aktiviert oder deaktiviert die Funktion Zeitbuchung in *acimSection* (d. h. der Link zur Zeitbuchung und der entsprechende Tab im Editor werden angezeigt). Standardwert ist „true“.

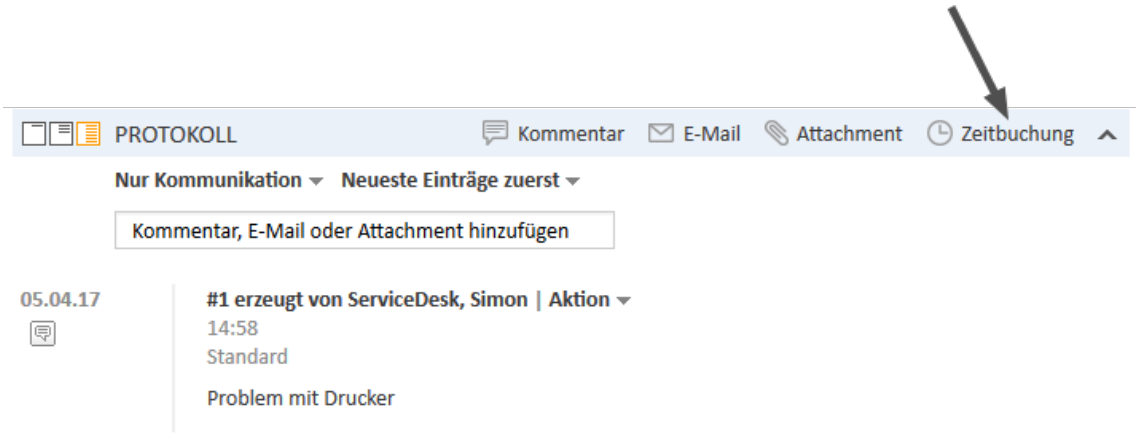


Abbildung 145: *ConSol CM Web Client - Zeitbuchungsfunktion aktiviert (timeBookingFeature = true)*

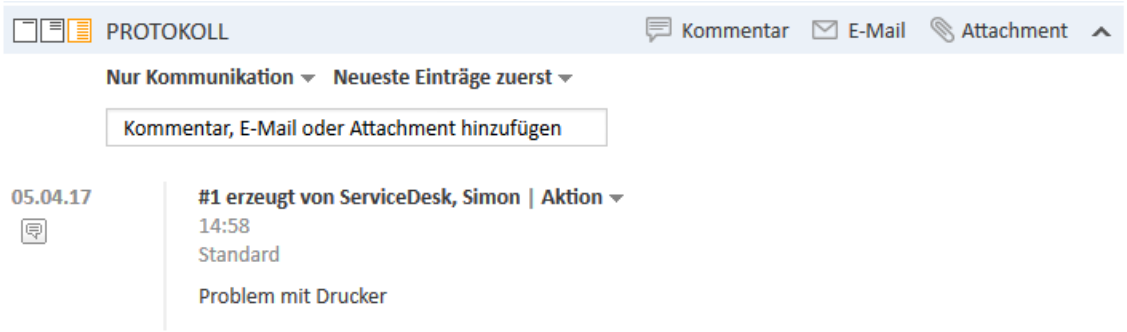


Abbildung 146: *ConSol CM Web Client - Zeitbuchungsfunktion deaktiviert (timeBookingFeature = false)*

Bitte beachten Sie Folgendes:

- Der Wert „false“ in *timeBookingFeature* blendet den Link zum Zeitbuchungseditor aus (siehe obige Abbildung). Der Bearbeiter kann ihn nicht einblenden!
- Die Sichtbarkeit des Bereichs zur Zeitbuchung im Bearbeiterprofil *(userProfilePage)* wird über das Attribut *timeBookingSection - visible* konfiguriert.

- **extendedViewCharactersLimit**
  Integer. Definiert die maximale Anzahl an Zeichen, die im Ticketprotokoll für Kommentare in der erweiterten Ansicht angezeigt werden, wenn dieses Level in der Textklasse, die auf den Eintrag angewendet wurde, als *Erweitert (gekürzt)* definiert ist. Der Standardwert ist 350. Ein Beispiel finden Sie im Abschnitt Ändern des Anzeigeverhaltens für die Sichtbarkeitslevel Standard und Erweitert zum Optimieren der Anzeige des Ticketprotokolls.

- **basicViewCharactersLimit** (früher **standardViewCharactersLimit**)
  Integer. Definiert die maximale Anzahl an Zeichen, die im Ticketprotokoll für Kommentare in der Standardansicht angezeigt werden, wenn dieses Level in der Textklasse, die auf den Eintrag angewendet wurde, als *Standard (gekürzt)* definiert ist. Der Text wird abgeschnitten, wenn die maximale Anzahl an Zeichen erreicht ist und die Formatierung wird entfernt. Für E-Mails werden keine Header-Informationen angezeigt. Die so entstehende Ansicht ist sehr kurz und präzise und verbraucht wenig Platz. Der Standardwert ist 150. Ein Beispiel finden Sie im Abschnitt Ändern des Anzeigeverhaltens für die Sichtbarkeitslevel Standard und Erweitert zum Optimieren der Anzeige des Ticketprotokolls.

attachmentSection (Typ oder Unterbereich)

Attribute:

- **defaultVisibilityFlag**
  Die Sichtbarkeit des Attachment-Bereichs für Bearbeiter, die die Standardsichtbarkeit noch nicht geändert haben (für die anderen Bearbeiter wird der letzte Sichtbarkeitsstatus verwendet, Standardwert = „false“, d. h. der Attachment-Bereich wird nicht angezeigt). Mit diesem Attribut wird nur das initiale Systemverhalten definiert. Der Bearbeiter kann den Attachment-Bereich anzeigen, indem er im Menü des Tickets die Option *Anzeigen* wählt. Der Web Client speichert diese Konfiguration für diesen Bearbeiter.

> Gültig bis CM-Version 6.9.3. Bei den neueren Version (d. h. 6.9.4 und höher) mit dem neuen Konzept für die Ticketbereichsverwaltung verwenden Sie dazu das Attribut *state* in sectionList / ticketEditPage / attachmentSection.



Abbildung 147: *ConSol CM Web Client - Sichtbarkeit des Attachment-Bereichs (defaultVisibilityFlag = true)*



Abbildung 148: *ConSol CM Web Client - Sichtbarkeit des Attachment-Bereichs (defaultVisibilityFlag = false)*

Beachten Sie, dass in ConSol CM-Version 6.11 eine Seitenanpassung für den Attachment-Bereich entfernt wurde. Die Details können Sie der folgenden Abbildung entnehmen.

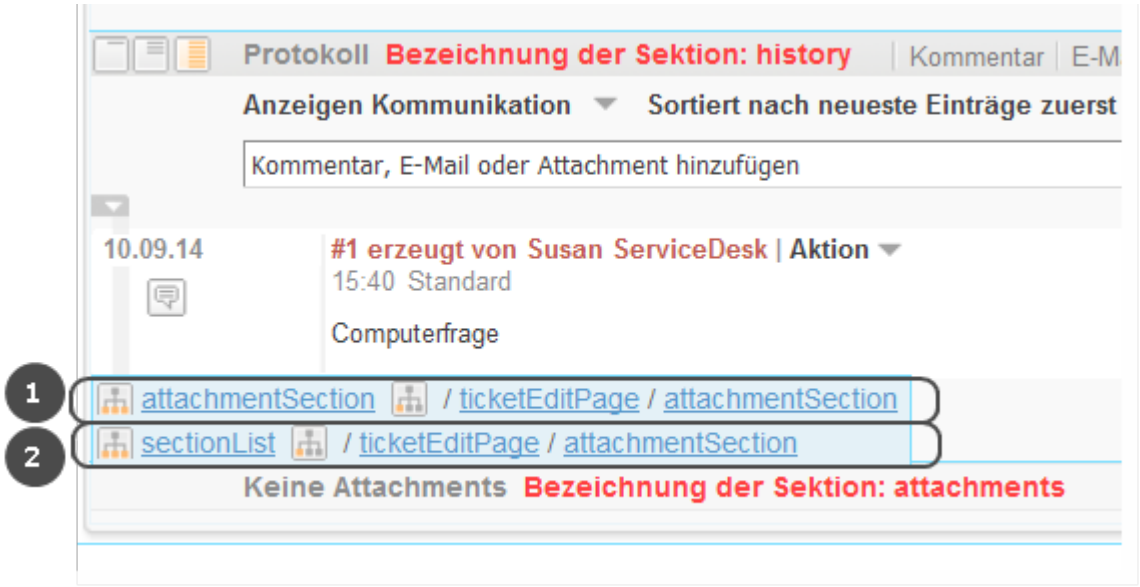


Abbildung 149: *Web Client in ConSol CM-Version 6.10.5. Die Seitenanpassung (1) wurde in Version 6.11 entfernt, die Seitenanpassung (2) ist in Version 6.11 noch verfügbar*

autocomplete

(verfügbar z. B. in *userProfilePage*)

Attribute:

- **suffixCharactersToRemove**
  Jedes der angegebenen Zeichen wird implizit entfernt, wenn es am Ende eines Suchwortes vorkommt. (java.lang.String)
  Beispiel: Wenn Sie mithilfe von Suchmustern suchen, d. h. in der Form "<Nachname>, <Vorname>" gibt die Suche keine Ergebnisse zurück, weil das Wort "<Nachname>," aufgrund des Kommas am Ende des Wortes im Suchindex nicht gefunden wird. Es ist jetzt möglich, einzustellen, dass bestimmte Zeichen vom Ende des Suchwortes entfernt werden. In diesem Fall

wird das Komma (",") bei der Suche ignoriert und der Bearbeiter *Holler* wird gefunden (siehe folgende Abbildung).

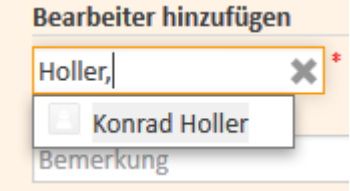


Abbildung 150: *ConSol CM Web Client - Eingabe im Suchfeld, bei der bestimmte Zeichen bei der Suche ignoriert werden sollen*

boxContent

Verfügbar in den folgenden Bereichen:

- ticketEditPage
- userProfilePage

Attribute:

- **order**
  Gibt die Reihenfolge der Bereiche eines Tickets im CSV-Format an (z. B.: Kopfbereich, Protokoll, Relationen).
  Standardwerte für die Standardinstallation:
  - **ticket create**: customfields, contacts, comment
  - **ticket details**: customfields, contacts, engineers, relations, history, attachments
  - **contact details**: customfields, tickets, additional_details, relations, history
  - **company details**: customfields, tickets, contacts, additional_details, relations, history

Für andere Seiten oder kundenspezifische Projekte finden Sie die Bereichsnamen im Header oder in Ihrer ContentBuilder-Implementierung.

Dieses Attribut muss erstellt werden, bevor Sie es verändern können. Klicken Sie dazu in der Seitenanpassung auf den Button *Erzeugen*.

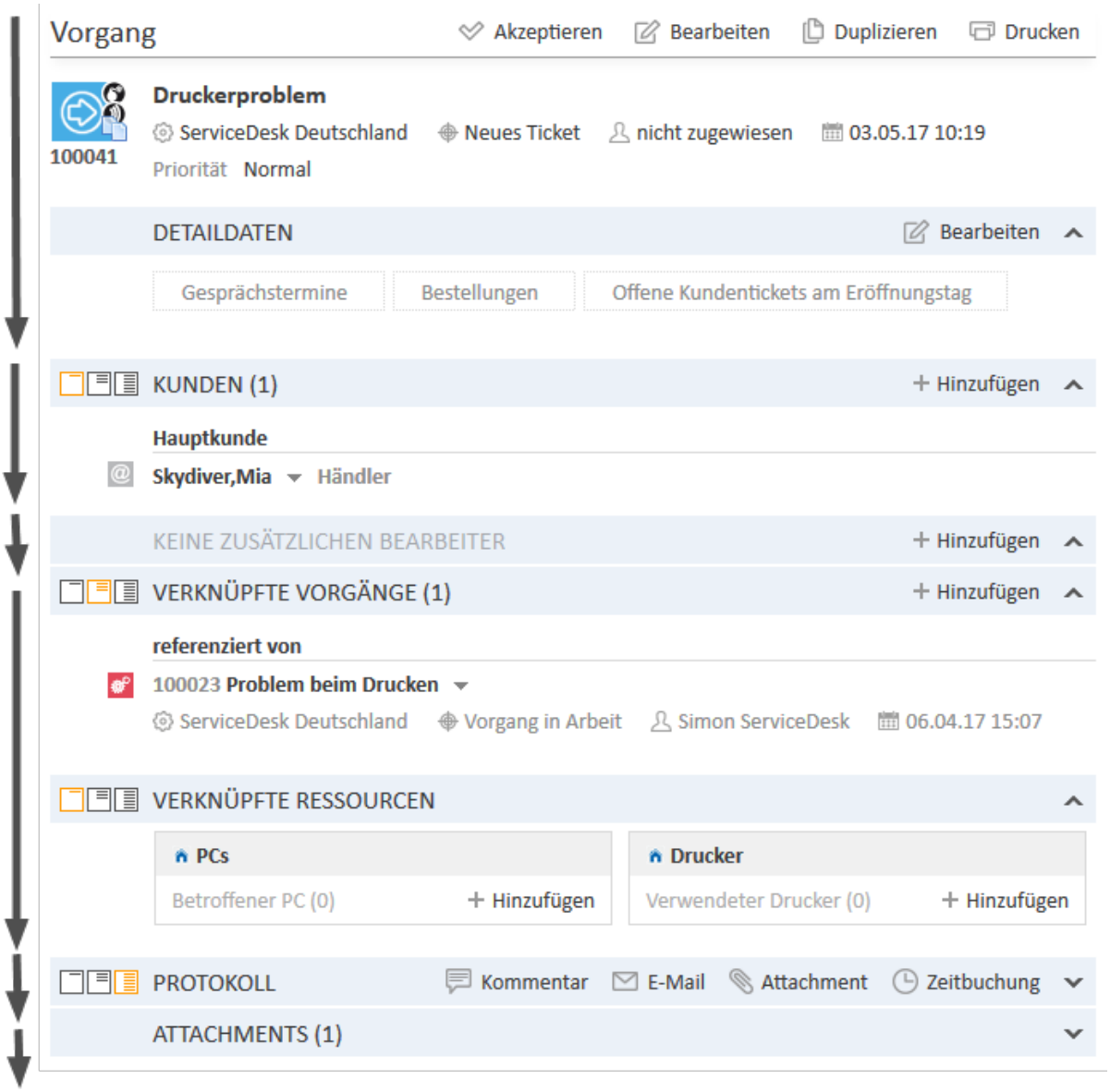


Abbildung 151: *ConSol CM Web Client - Standardwert für das Attribut boxContent / order*

calendarSection

Wird zur Konfiguration des Kalenderbereichs auf Ticketseiten und Kundenseiten verwendet. Aktuell wird mit den entsprechenden Attributen die Integration von Microsoft Exchange-Kalenderansichten konfiguriert. Eine detaillierte Erklärung der Funktionen und eine Liste aller verfügbaren Attribute finden Sie im Abschnitt Integration von Microsoft Exchange-Kalendern.

cmApplicationCustomization (Typ)

Attribute:

- **activityKeyBindingEnabled**
  Boolean. Tastenkombinationen für Aktivitäten/Aktionen aktivieren bzw. deaktivieren. Ab Version 6.10.1 kann der Bearbeiter Tastenkombinationen verwenden, um Workflow-Aktivitäten des Tickets anzusprechen. Der Standardwert ist „true“.
- **submitACFOnEnterEnabled**
  Boolean. Das Absenden von Aktivitätsformularen beim Drücken der Eingabetaste aktivieren bzw. deaktivieren. Ab Version 6.10.1 kann der Bearbeiter die Eingabetaste drücken, um Aktivitätsformulare abzusenden. Der Standardwert ist „true“.

Die folgenden beiden Attribute beeinflussen das Session-Timeout für Bearbeiter im Web Client. Um zu verhindern, dass die Session beendet wird, während der Bearbeiter wichtige Informationen bearbeitet (und somit die eingegebenen Daten mit dem Ende der Session verloren gehen), kann mit den folgenden beiden Attribute eine automatische Erneuerung der Session konfiguriert werden.

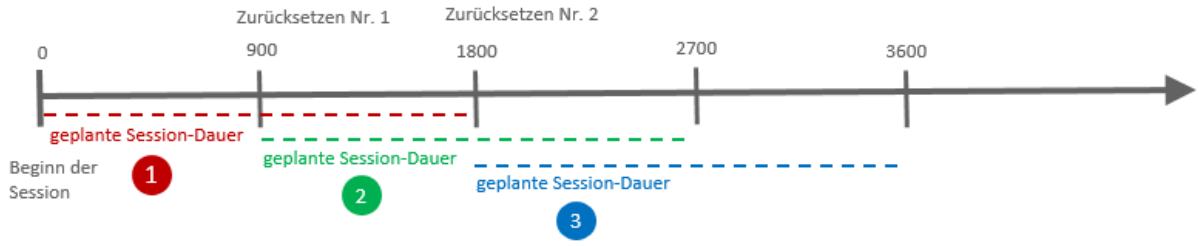


Abbildung 152: *Prinzip der Erneuerung der Bearbeiter-Session, Beispiel mit updateTimeServerSessionActivity = 900.*

- **updateTimeServerSessionActivity**
  Integer. Setzt die Zeit in Sekunden, nach der die Server-Session bei Aktivität automatisch aktualisiert wird. Standardwert ist 900.
- **updateTimeServerSessionActivityEnabled**
  Boolean. Aktiviert/Deaktiviert die automatische Aktualisierung der Server-Session bei Aktivität. Der Standardwert ist „true“. Wenn „false“ gesetzt ist, ist die Erneuerung der Session nicht aktiv.

Die folgenden Aktionen unterliegen der Kontrolle von *updateTimeServerSession*:

- Erstellen eines Tickets, eines Kunden oder einer Ressource
- Erstellen eines Kommentars oder einer E-Mail in einem Ticket
- Erstellen eines Kommentar für einen Kunden oder eine Ressource
- Editieren von Datenfeldern eines Tickets, eines Kunden oder einer Ressource
- Editieren von Datenfeldgruppen
- Aktivitätsformulare

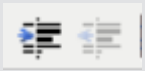
cmRichTextEditor

Verfügbar für folgende (Unter)bereiche:

- acimSection/ticketEditPage
- createTicketAcimSection/ticketCreatePage
- templateSection/templateEditPage (im Textvorlagen-Manager)

Attribute:

- **editorFeatures**
  Zusätzliche Editor-Funktionen. Standardmäßig sind alle Werte gesetzt, d. h. die entsprechenden Menüeinträge sind verfügbar. (java.lang.String)
  Mögliche Werte:

Wert	Icon im Rich-Text-Editor	Beschreibung
SUB_SUP		Ermöglicht hochgestellte und tiefgestellte Zeichen.
INDENTS		Ermöglicht das Einrücken von Text.
LISTS		Ermöglicht das Einfügen und Formatieren von Listen.
TABLES		Ermöglicht das Einfügen von Tabellen.
INSERT		Ermöglicht das Einfügen von Textelementen; zur genaueren Steuerung: INSERT_EMOTICON, INSERT_CHAR und INSERT_IMAGE
INSERT_EMOTION		Ermöglicht das Einfügen von Emoticons.
INSERT_CHAR		Ermöglicht das Einfügen von Sonderzeichen.
INSERT_IMAGE		Ermöglicht das Einfügen von Bildern.

Wert	Icon im Rich-Text-Editor	Beschreibung
INSERT_LINK		Ermöglicht das Einfügen/Bearbeiten und Entfernen von Hyperlinks. Siehe Erklärung unten.

INSERT_LINK: Markieren Sie die Wörter, die als Hyperlink markiert werden sollen, und klicken Sie auf den Button *Link einfügen/verändern*. Dies öffnet ein Dialogfenster, in dem Sie die Adresse, das Fenster (selbes oder anderes Browser-Fenster) und einen optionalen Titel für den Link eingeben können.



Abbildung 153: *Dialogfenster zum Einfügen eines Links*

In der Auswahl des Fenster können Sie festlegen, ob der Link in einem neuen Fenster/Tab oder im aktuellen Fenster/Tab geöffnet werden soll. Die gültigen Präfixe für das Protokoll sind: „http://“, „https://“, „ftp://“, „ftps://“, „file://“, and „mailto:“. Wenn kein Protokoll angegeben wird, wird automatisch das Präfix „http://“ hinzugefügt. Für Dateilinks ist häufig eine Änderung in der Browserkonfiguration erforderlich. Dies ist im Abschnitt Arten von Datenfeldern, *Wie kann ich einen Dateilink anzeigen?* beschrieben.
Diese Linkfunktionalität ist auch für Textvorlagen verfügbar, sodass vordefinierte Links in Textvorlagen verwendet werden können.

- **editorFonts**
  Die Liste der Schriftarten im Editor in der Form <Anzeigename der Schriftart>=<Systemname der Schriftart>.
  Die Schriftarten sind durch ein Semikolon (";") getrennt. Sie können eine durch Kommas getrennte Liste möglicher Systemnamen für jede Schriftart angeben. (java.lang.String) (Standardwert = leerer String)
  Beispiel: Arial=arial,helvetica,sans-serif;Courier New=courier new,-courier;Verdana=verdana,geneva

- **fontSizeValue**
  Das ist die Standardschriftgröße für Text im Rich-Text-Editor. Es muss einer der Werte aus der Liste des Attributs *fontSizeValues* angegeben werden.
- **fontSizeValues**
  Dies ist eine Liste der Werte, die als Auswahl der Schriftgröße im Rich-Text-Editor angeboten werden. Es ist eine durch Kommas getrennte Liste mit gültigen Schriftgrößen einschließlich der Einheit, z. B. "6pt,10pt,12pt". Den Werten kann ein Label vorangestellt werden, das dann in der Auswahl anstatt des eigentlichen Wertes angezeigt wird: "klein=6pt,normale Größe=10pt,groß=12pt".
- **imagePasteEnabled**
  Gibt an, ob das direkte Einfügen von Bildern aus der Zwischenablage in den Editor aktiviert ist. Beachten Sie, dass diese Funktion die Ausführung eines Java Applets erfordert (boolean, Standardwert = „false“). Webbrowser (d. h. Internet Explorer, Firefox) können ein unterschiedliches Verhalten bzgl. der Anzeige von Bildern haben.
  In CM-Versionen bis 6.10.4.x verfügbar, wird ab CM-Versionen 6.10.5.0 und höher nicht mehr unterstützt.

customerGroupSelector

Ist direkt unter dem Stammknoten im Definitionsbereich verfügbar.

Attribute:

- **hiddenCustomerGroups**
  Liste der technischen Namen der Kundengruppen, die nicht im Kundengruppenfilter im Hauptmenü angezeigt werden sollen. Durch Kommas getrennte Liste der Namen der Kundengruppen. Standardwert: leer

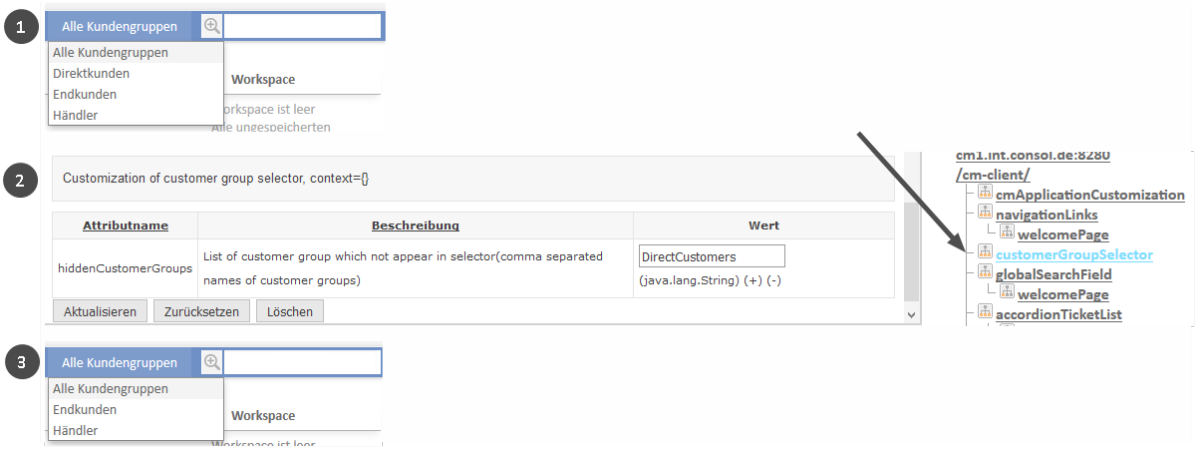


Abbildung 154: *ConSol CM Web Client - hiddenCustomerGroups*

customerSectionPanel

Hier können Sie festlegen, ob das Menüelement *Bearbeiten* im Kontextmenü für Firmen angezeigt werden soll. Beachten Sie, dass der Bearbeiter zusätzlich die entsprechenden Zugangsberechtigungen zum Bearbeiten von Firmendaten haben muss (siehe Abschnitt Rollenverwaltung).

Attribute:

- **referencedUnitEditLinkVisible** (Versionen 6.9.3.3 und niedriger. Ein neues, ähnliches Attribut ist *companyEditLinkVisible*)
  Boolean. Die Sichtbarkeit des Links zum Bearbeiten von referenzierten Units (Kontakten oder Firmen) (boolean, Standardwert ist „true“).

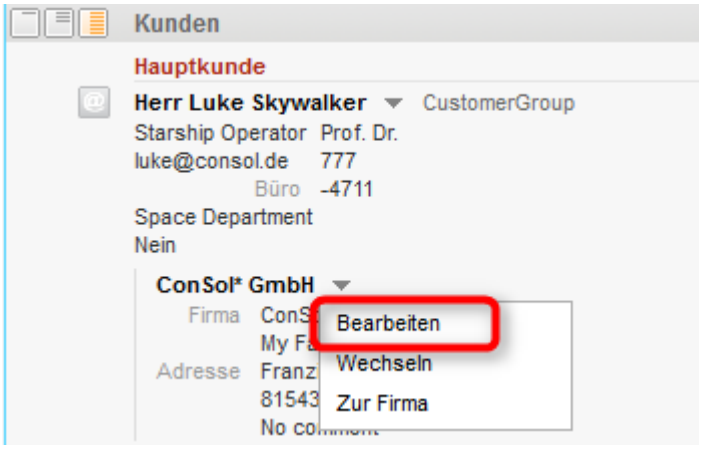


Abbildung 155: *ConSol CM Web Client - Sichtbarkeit des Links Bearbeiten (referencedUnitEditLinkVisible = true)*

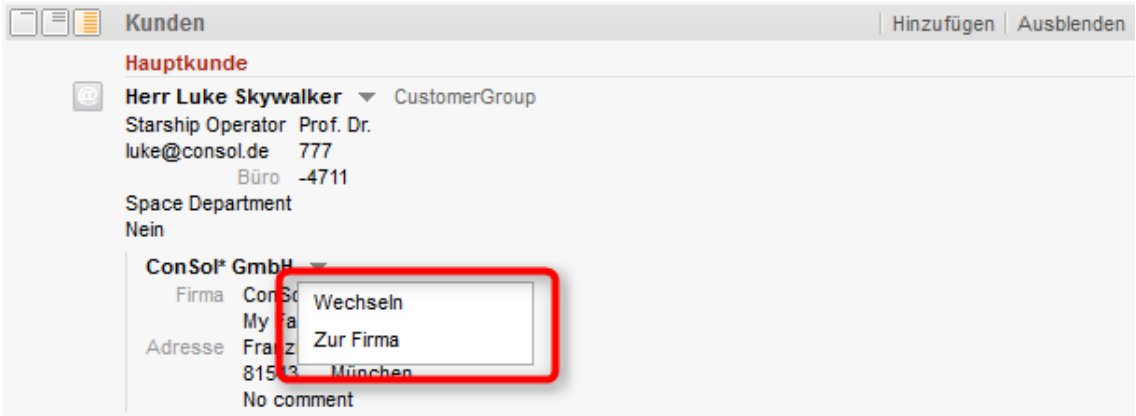


Abbildung 156: *ConSol CM Web Client - Sichtbarkeit des Links Bearbeiten (referencedUnitEditLinkVisible = false)*

- **companyEditLinkVisible** (Versionen 6.9.3.4 und höher. Das ältere Attribut ist *referencedUnitEditLinkVisible*)
  (verfügbar in *customerSectionPanel* im Ticket und in *companyEditPage* und *contactEditPage*)
  Die Sichtbarkeit des Links zum Bearbeiten einer Firma, Standardwert ist „true“.



Abbildung 157: *ConSol CM Web Client - Sichtbarkeit des Links zum Bearbeiten der Firma (companyEditLinkVisible = true)*



Abbildung 158: *ConSol CM Web Client - Sichtbarkeit des Links zum Bearbeiten der Firma (companyEditLinkVisible = false)*

- **additionalCustomersSortStrategy**
  Die Sortierreihenfolge der Zusatzkunden für ein Ticket kann mit diesem Attribut definiert werden. Wenn kein Wert gesetzt ist, werden die Zusatzkunden in der Reihenfolge sortiert, in der sie zum Ticket hinzugefügt wurden.

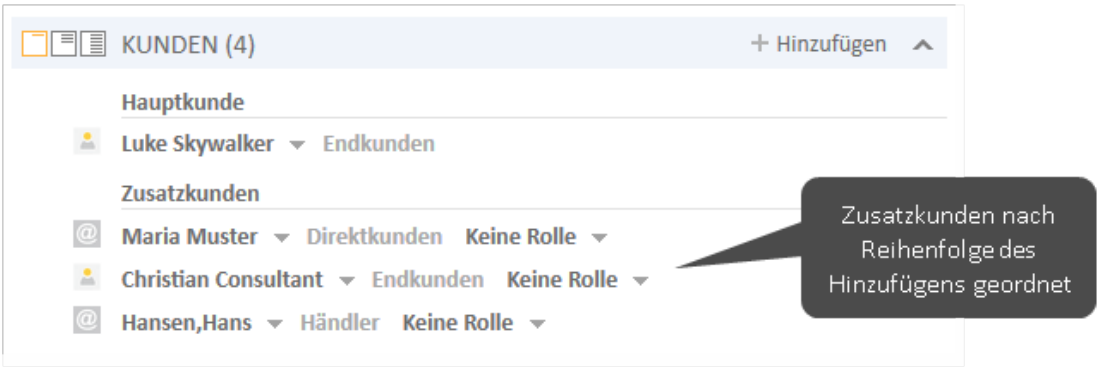


Abbildung 159: *ConSol CM Web Client - Seitenanpassung für die Sortierung der Zusatzkunden (1)*

Die folgenden Werte sind möglich:

- **COMPANY_OF_MAIN_CUSTOMER**
  Die Kunden werden nach Firmenbeschreibung sortiert. Die Firma des Hauptkunden erscheint zuerst.
- **COMPANY**
  Die Kunden werden nach Firmenbeschreibung sortiert.
- **CONTACT**
  Die Kunden werden nach Kontaktbeschreibung sortiert.
- **ROLE**
  Die Kunden werden nach Kundenrolle sortiert.

Es können mehrere Werte in einer durch Kommas getrennten Liste angegeben werden. Die Standardsortierreihenfolge (kein Wert) funktioniert wie vorher: die Kunden werden so wie vorher nach der Reihenfolge der Hinzufügung sortiert.

Die Kundenobjektbeschreibungen, die für die Anzeige von Kontakten und Firmen verwendet werden, stammen aus dem `<Kundenobjekt>.Ticket page`, siehe Abschnitt Templates für Kundendaten.

Im folgenden Beispiel sind die Zusatzkunden nach Namen des Kunden sortiert (Wert: „CONTACT“).



Abbildung 160: *ConSol CM Web Client - Seitenanpassung für die Sortierung der Zusatzkunden (2)*

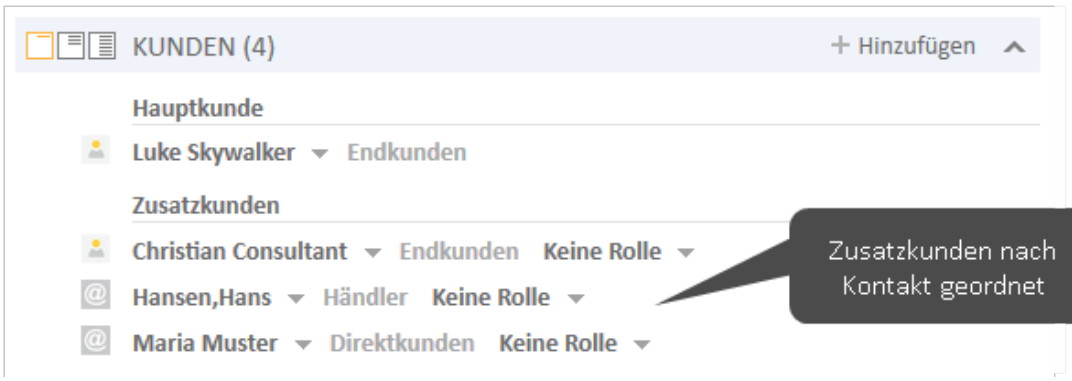


Abbildung 161: *ConSol CM Web Client - Seitenanpassung für die Sortierung der Zusatzkunden (3)*

customerTickets

Mit den Attributen dieses Typs können Sie das Verhalten der Ticketliste auf Kundenseiten (d. h. Kontakt- und Firmenseiten) konfigurieren.

Attribute:

- **enabled**

  Boolean. Legt fest, ob der Bearbeiter- und der Queue-Filter in der Ticketliste auf Kundenseiten (d. h. Kontakt- und Firmenseiten) verfügbar sind. Standardwert: „false“.

  Dies gilt nur, wenn die Anzahl der Tickets in der Liste die für das Attribut *compactViewLimit* gesetzte Anzahl überschreitet.

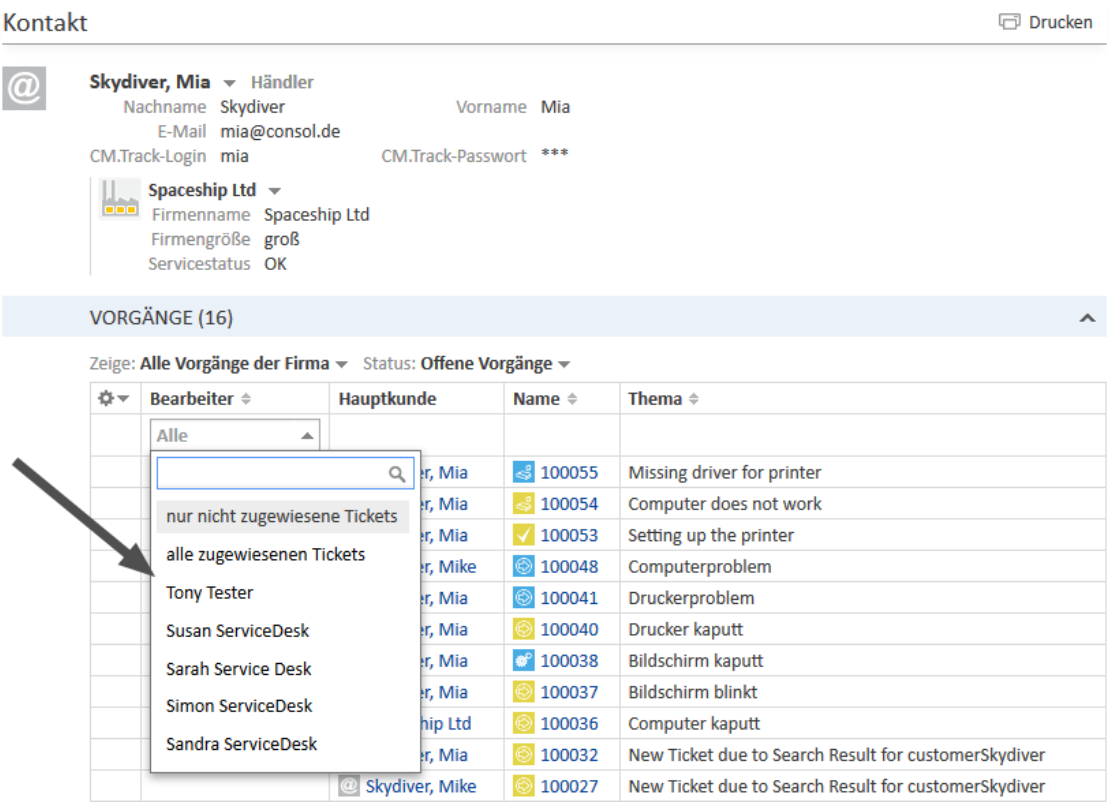


Abbildung 162: *Liste der Kundentickets auf einer Kundenseite (hier: Kontaktseite), Bearbeiter- und Queue-Filter sind verfügbar*

Der Filter wird auf alle Arten von Kundenseiten, d. h. Kontakt- und Firmenseiten, angewendet, wenn das Attribut *enabled* für den Typ *customerTickets* gesetzt ist (siehe Beispiel in der folgenden Abbildung).



Abbildung 163: *Typ und Bereich der Seitenanpassung zum Aktivieren des Bearbeiter- und Queue-Filters in der Ticketliste der Kundenseite*

Der Filter wird nur auf Kontaktseiten angewendet, wenn das Attribut *enabled* für den Bereich *contactEditPage* oder *ContactTicketListSection* gesetzt ist. Der Filter wird nur auf Firmenseiten angewendet, wenn das Attribut *enabled* für den Bereich *companyEditPage* oder *CompanyTicketsSection* gesetzt ist.

> Sie müssen die Tickets neu indizieren, bevor Sie diese Funktion nutzen können. Ansonsten sind die Filter leer und es wird eine Warnung in die Log-Dateien geschrieben.

- **compactViewLimit**

  Integer. Wenn dieses Limit überschritten wird, werden die Ergebnisse mit Filterfunktion angezeigt. Das Standardlimit ist „10“.

detailSearch (Typ)

Verfügbar in folgenden Bereichen:

- searchDetailPage

Attribute:

- **criteriaForAllTypeOfContacts**
  Boolean-Feld. Wenn der Wert auf „true“ gesetzt ist, umfasst die Suche Hauptkunden und Zusatzkunden des Tickets. Wenn das Attribut auf „false“ (Standardwert) gesetzt ist, gelten die Suchkriterien nur für die Hauptkunden des Tickets.

- **duplicatedCustomFieldLabelFormat**
  Ermöglicht die Anpassung des Labelformats, wenn mehrere Ticketfelder den gleichen lokalisierten Namen haben. Es gibt vier Werte, mit denen ein eindeutiges Label erzeugt werden kann:
  - fieldName
  - groupName
  - fieldTechnicalName
  - groupTechnicalName

  Die Werte „fieldName“ und „groupName“ sind lokalisiert.

  Standardformat: ${fieldName} (${groupName}) (java.lang.String)

  Dies hilft Ihnen dabei, unterschiedliche Ticketfelder oder Kundenfelder mit dem gleichen Namen zu unterscheiden, z. B. wenn sie im Drop-down-Menü der Detailsuche angeboten werden.

  Beispiel:

  SalesProcess - Priorität

  HelpdeskProcess - Priorität
- **enableRowSelection**
  Boolean. Schaltet die Zeilenauswahlfunktion in Ergebnistabellen ein, d. h. es ist für jede Zeile eine Checkbox verfügbar und es gibt eine Checkbox, mit der alle Zeilen ausgewählt (nicht ausgewählt) werden können. Der Standardwert ist „true“. Die Checkboxen werden nur angezeigt, wenn für den entsprechenden Objekttyp Suche-Aktionen definiert sind. Das Suche-Aktionsskript wird nur für die Zeilen ausgeführt, die (vom Bearbeiter) mithilfe der Checkboxen ausgewählt wurden.

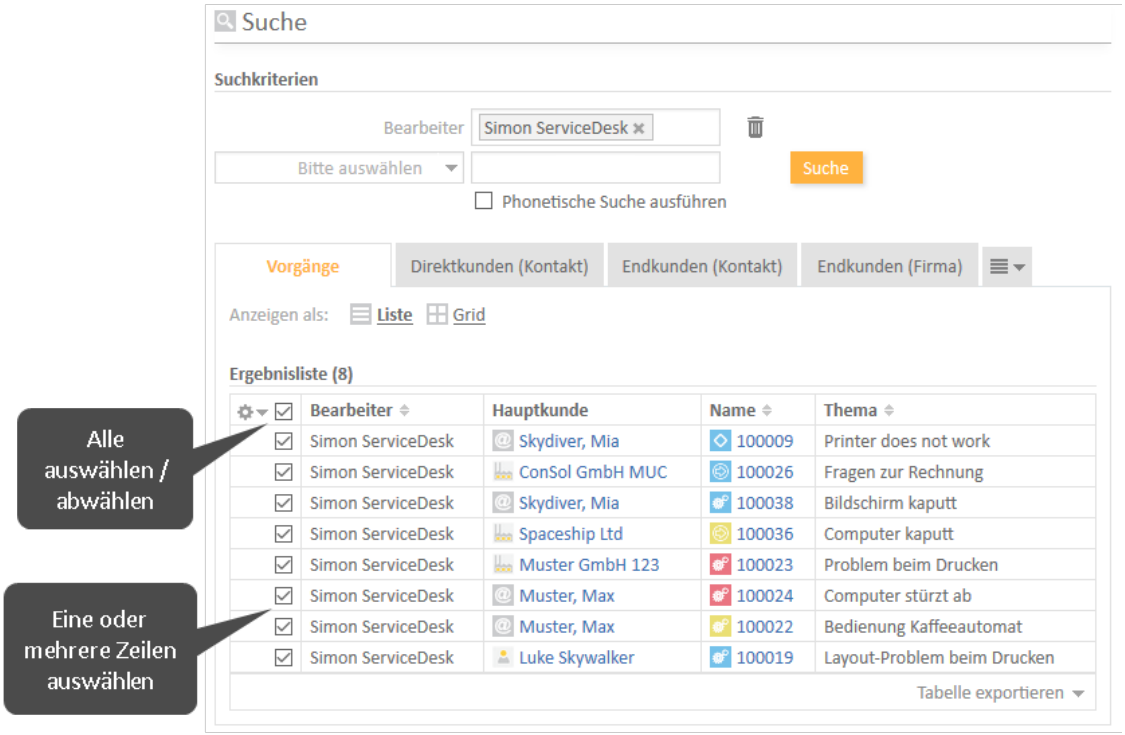


Abbildung 164: *ConSol CM Web Client - Tabelle mit Suchergebnissen mit detailSearch/searchDetailPage, enableRowSelection = true*

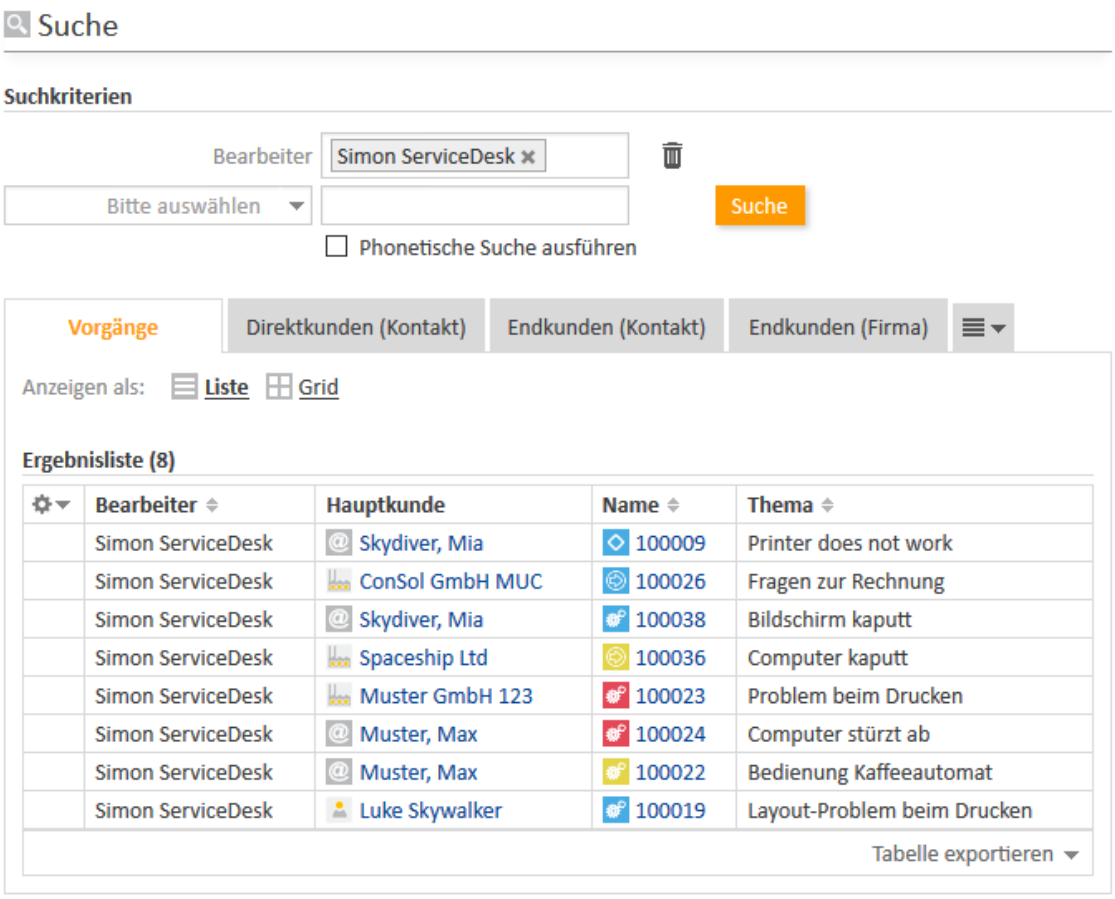


Abbildung 165: *ConSol CM Web Client - Tabelle mit Suchergebnissen mit detailSearch/searchDetailPage, enableRowSelection = false*

- **enableSearchForDeactivatedEnums**

  Boolean. Bei „true“ sind deaktivierte Werte von Sortierten Listen und MLAs im Drop-down-Menü in der Detailsuche verfügbar.

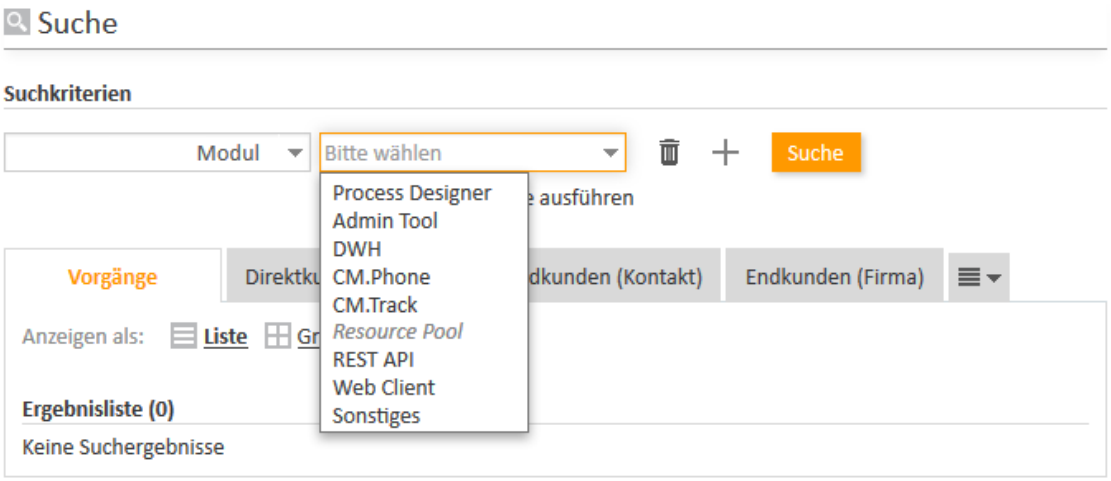


Abbildung 166: *"enableSearchForDeactivatedEnums" in searchDetailPage auf "true" gesetzt: deaktivierter Listenwert wird angezeigt*

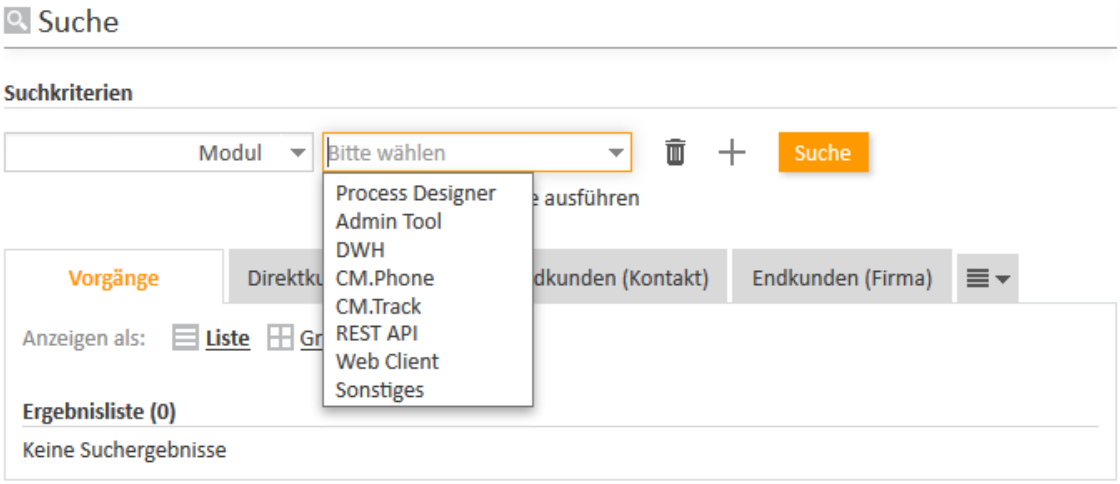


Abbildung 167: *"enableSearchForDeactivatedEnums" in searchDetailPage auf "false" gesetzt: deaktivierter Listenwert wird nicht angezeigt, stattdessen wird der nächste aktive Listenwert angezeigt*

- **enableSearchForOtherUsersTickets**
  Boolean. Entscheidend, ob ein Bearbeiter in CM nach allen Tickets („true“) oder nur nach ihm selbst zugewiesenen Tickets („false“) suchen kann. Der Standardwert ist „true“.
  Dieses Attribut muss auf Bereichs-/Unterbereichsebene gesetzt werden, d. h. für den Bereich bzw. Unterbereich *detailSearch/searchDetailPage*.
  Der Wert dieses Attributs wirkt sich lediglich auf das Layout des Drop-down-Menüs in der Detailsuche aus (siehe Abbildungen unten). Er hat keine Auswirkungen auf die Ergebnisse der Schnellsuche.

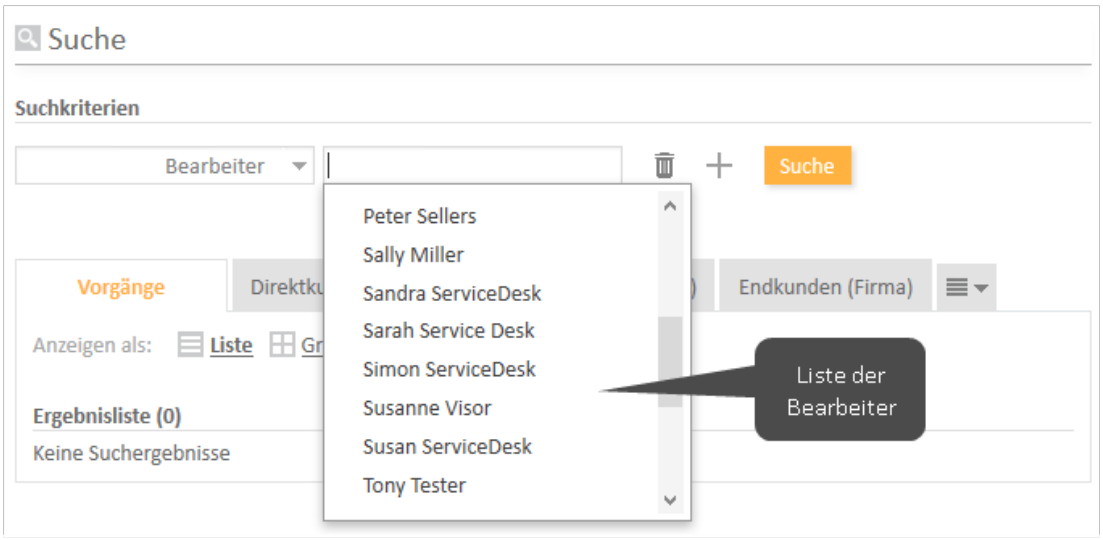


Abbildung 168: *ConSol CM Web Client - Detailsuche nach Bearbeitern mit enableSearchForOtherUsersTickets = true*

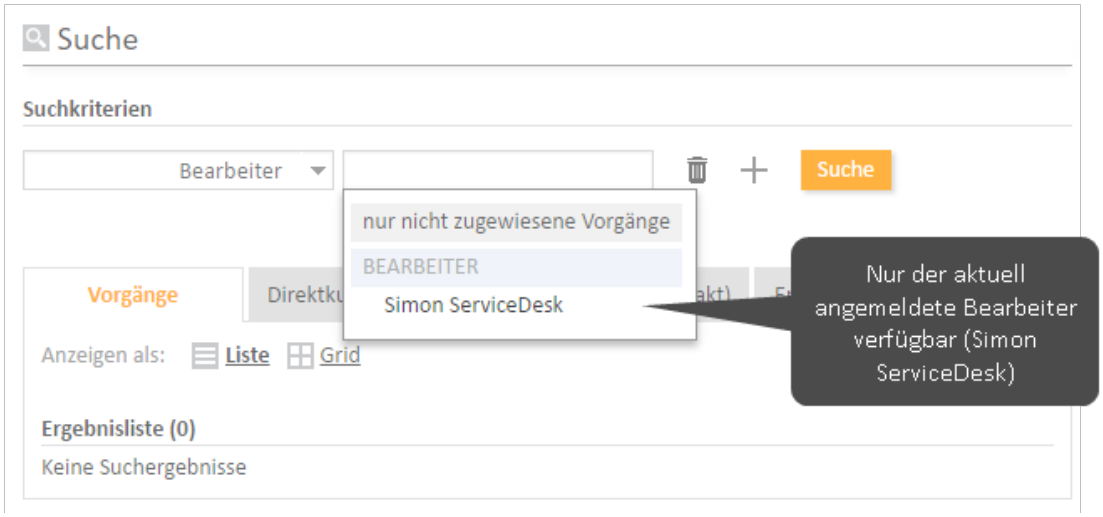


Abbildung 169: *ConSol CM Web Client - Detailsuche nach Bearbeitern mit enableSearchForOtherUsersTickets = false*

- **excludedUserNames**

  (muss normalerweise für searchDetailPage gesetzt werden)
  Die dort angegebenen Login-Namen werden in der Auswahlliste der Bearbeiter herausgefiltert.

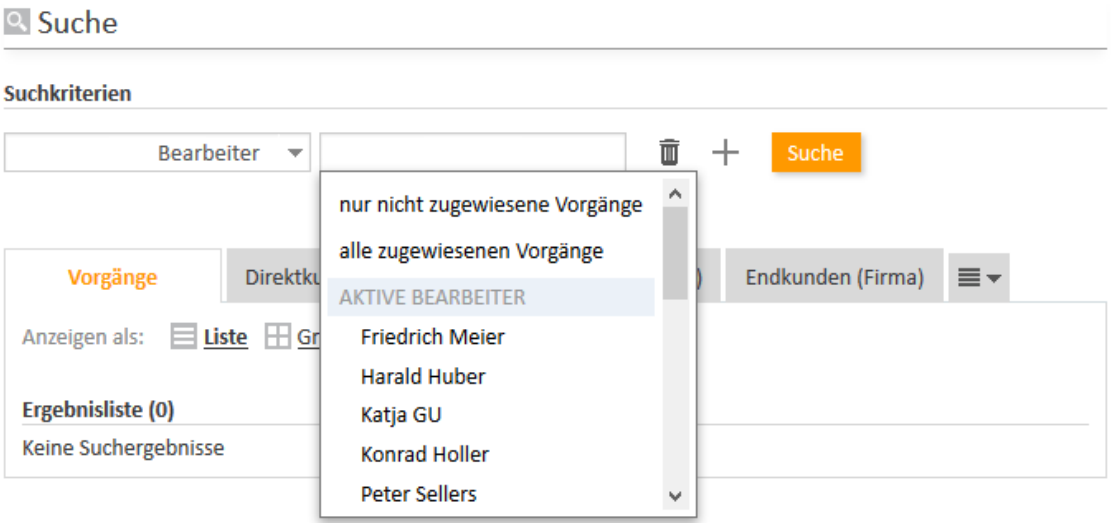


Abbildung 170: *Drop-down-Menü in der Detailsuche nach Bearbeitern vor dem Ausschließen von Benutzernamen*



Abbildung 171: *Verwendung des Attributs der Seitenanpassung excludedUserNames für einen Benutzernamen*

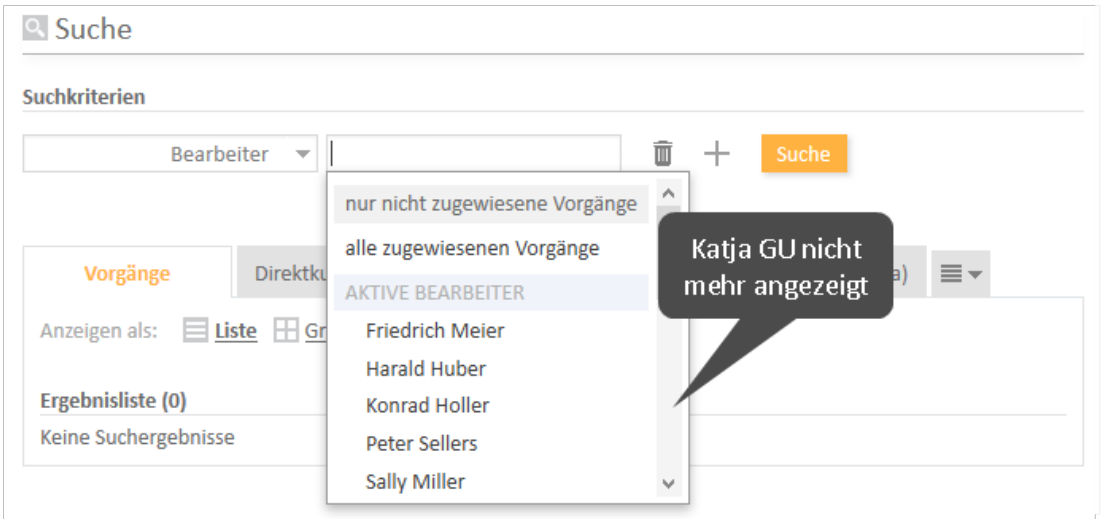


Abbildung 172: *Drop-down-Menü in der Detailsuche nach Bearbeitern nach dem Ausschließen von einem Benutzernamen aus der Liste*

- **maxGridTicketsNumber**
  Maximale Anzahl an Tickets, die in der Grid-Darstellung angezeigt werden können, d. h. im gesamten Grid, nicht in einer Spalte. Der Standardwert ist „120“.

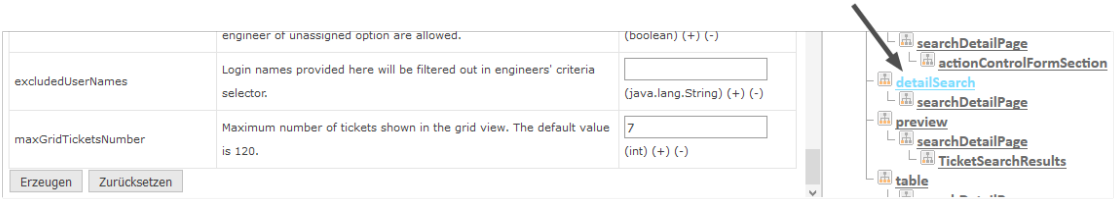


Abbildung 173: *ConSol CM Web Client - Seitenanpassung zur Anpassung der maximal im Grid angezeigten Tickets*

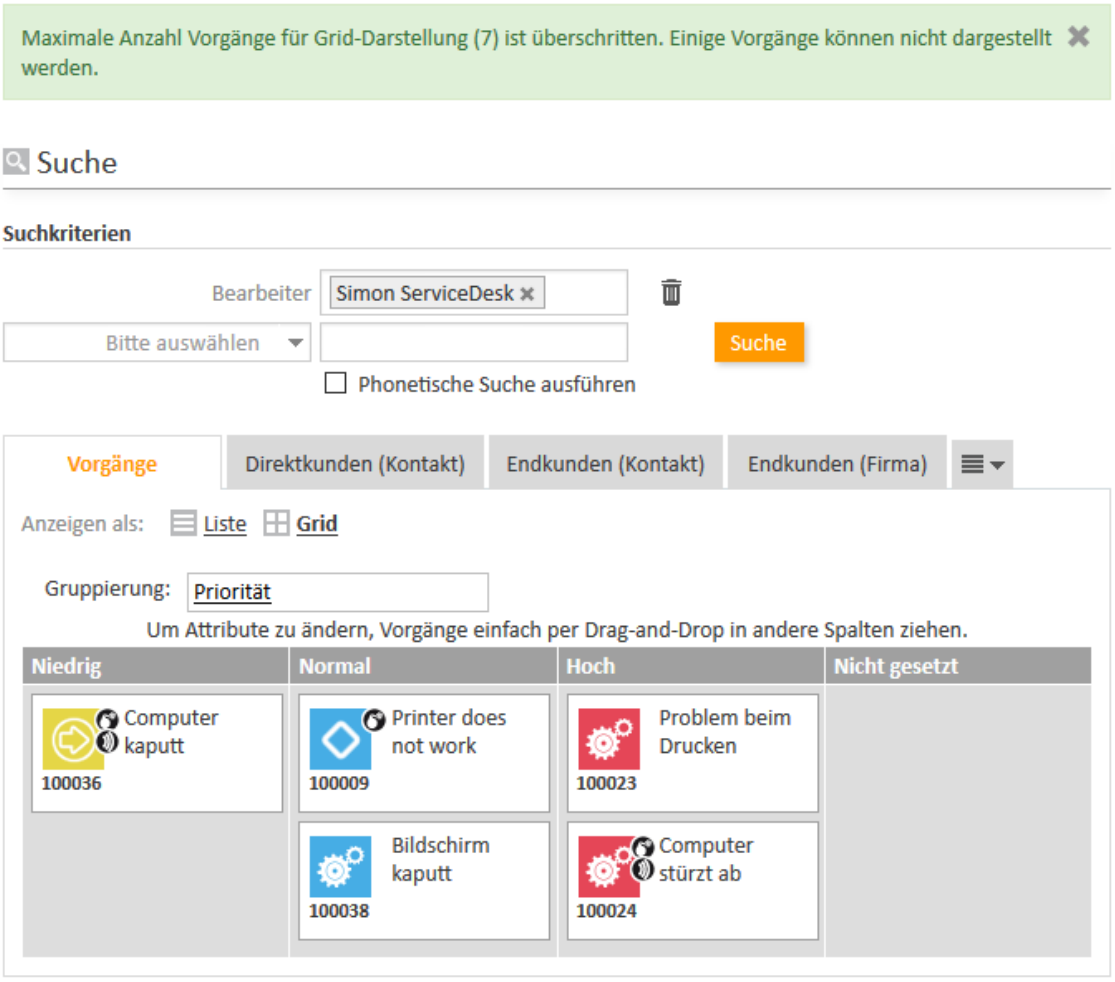


Abbildung 174: *ConSol CM Web Client - Maximale Anzahl der Tickets in Grid mit Maximalwert 7*

Die maximale Anzahl der in der Grid-Darstellung angezeigten Tickets kann zusätzlich von der Rolle des Bearbeiters abhängen. Ein Bearbeiter mit der Rolle *Teamleiter* würde zum Beispiel 100 Tickets sehen und ein Bearbeiter mit der Rolle *Helpdesk* würde 50 Tickets sehen. Im folgenden einfachen Beispiel sehen alle Bearbeiter mit der Rolle *ServiceDesk* 10 Tickets, während die anderen Bearbeiter 5 sehen. Diese Konfiguration verwendet ein Skript, das in der Seitenanpassung für den Typ definiert wird. Das Skript wird im Menü *Skripte und Templates* des Admin Tools gespeichert.

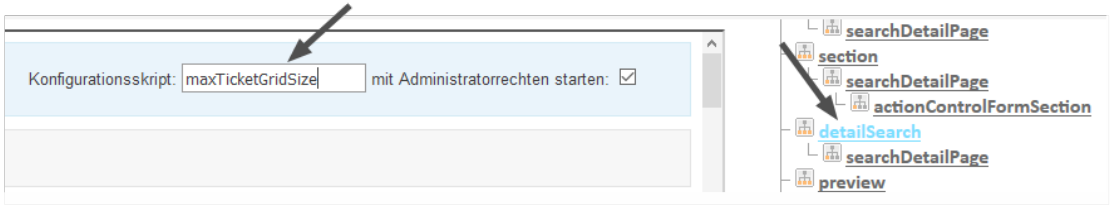


Abbildung 175: *ConSol CM Web Client - Definieren der maximalen Anzahl der Tickets in der Grid-Darstellung mittels Seitenanpassung*

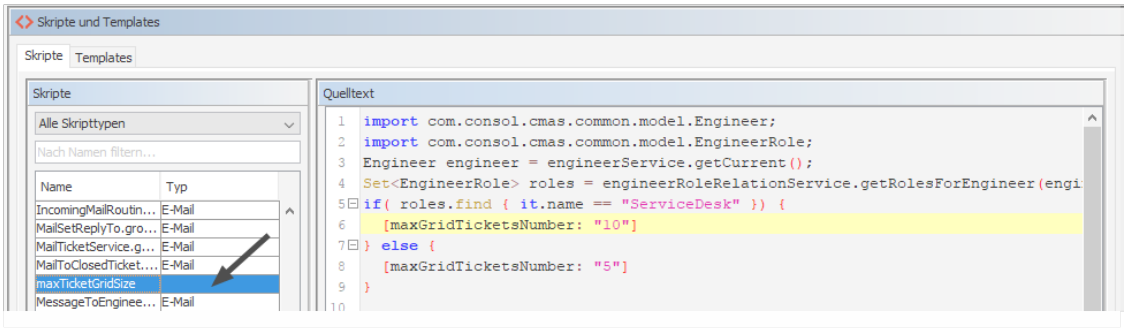


Abbildung 176: *ConSol CM Admin Tool - System, Skripte und Templates: Skript für die maximale Anzahl der Tickets im Grid*

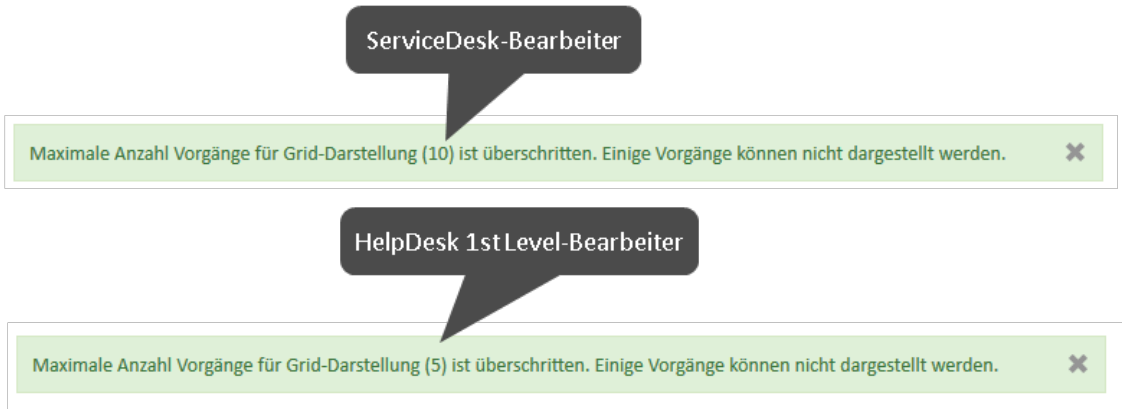


Abbildung 177: *ConSol CM Web Client - Anzahl der Tickets im Ticket-Grid für verschiedene Bearbeiterfunktionen*

engineerAutocomplete

Verfügbar für

- ticketCreatePage
- ticketEditPage
- userProfilePage
- contactEditPage

Attribute:

- **maxHints**
  Definiert die maximale Anzahl an angezeigten Vorschlägen. Wenn dieses Attribut auf „0“ gesetzt ist, werden alle Vorschläge ohne Beschränkung angezeigt.
- **suffixCharactersToRemove**
  Jedes der angegebenen Zeichen wird implizit entfernt, wenn es am Ende eines Suchwortes vorkommt. (String, Standardwert: leer)

enumAutocomplete

(verfügbar in *ticketEditPage* für Sortierte Listen mit der Annotation `enum type` = „autocomplete“)

Attribute:

- **maxHints**
  Definiert die maximale Anzahl an angezeigten Vorschlägen. Wenn dieses Attribut auf „0“ gesetzt ist, werden alle Vorschläge ohne Beschränkung angezeigt.
- **suffixCharactersToRemove**
  Jedes der angegebenen Zeichen wird implizit entfernt, wenn es am Ende eines Suchwortes vorkommt. (String, Standardwert: leer)

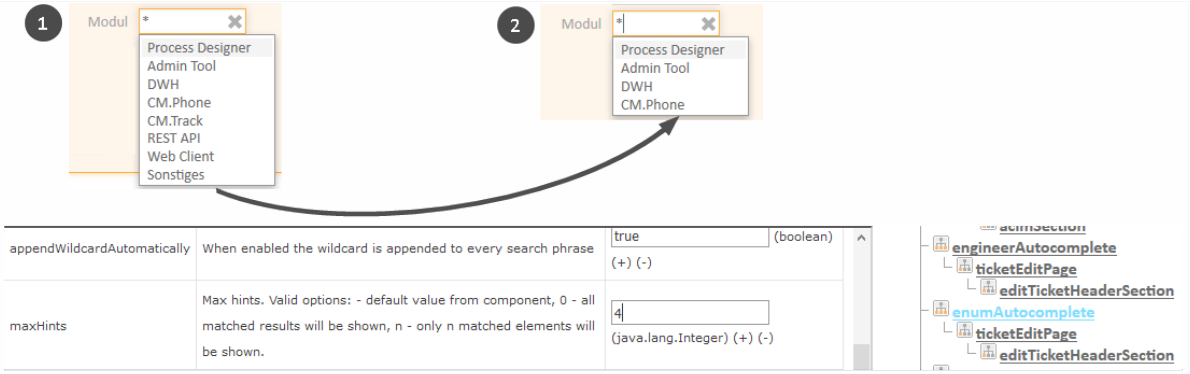


Abbildung 178: *ConSol CM Web Client - Verringern der Anzahl der Vorschläge mit dem enumAutocomplete-Parameter*

generalFeedback

Dieser Typ wird verwendet, um Log-Einträge in einer Cluster-Umgebung zu verändern.

Attribute:

- **appendHostInfo**
  Konfiguriert, ob die Informationen zum Cluster-Node zusammen mit der Fehlermeldung angezeigt werden.
- **appendHostInfoSystemProperty**
  Identifiziert die System-Property, die zum Abrufen des in *appendHostInfo* angezeigten Wertes verwendet wird. Es können zwei Properties verwendet werden:
  - cmas.http.host.port (Standardwert)
  - cmas.clusternode.id
- **successMessageTimeout**
  Integer. Definiert die Zeit in Sekunden bevor eine Erfolgsmeldung (z. B. "Ticket XY wurde erstellt") im Web Client ausgeblendet wird. Wenn 0 gesetzt ist, wird die Meldung nie automatisch ausgeblendet, sondern muss vom Bearbeiter geschlossen werden. Der Standardwert ist 0.

globalSearchField (Typ)

Hier können Sie das Layout und das Verhalten des Eingabefeldes der Schnellsuche definieren.



Abbildung 179: *ConSol CM Web Client - Eingabefeld der Schnellsuche*

Attribute:

- **appendWildcardAutomatically**
  Boolean. Wenn aktiviert, wird an jeden Suchbegriff, der aus mehr als einem Buchstaben oder

mehr als einer Zahl besteht, automatisch ein Platzhalter ("*") angehängt. Das heißt, ein Wort wird auch gefunden, wenn Sie nur einen Teil des Wortes eingeben. Der Standardwert ist „true“.



Abbildung 180: *Ergebnisse der Schnellsuche mit automatisch angefügtem Platzhalter (appendWildcardAutomatically = true)*

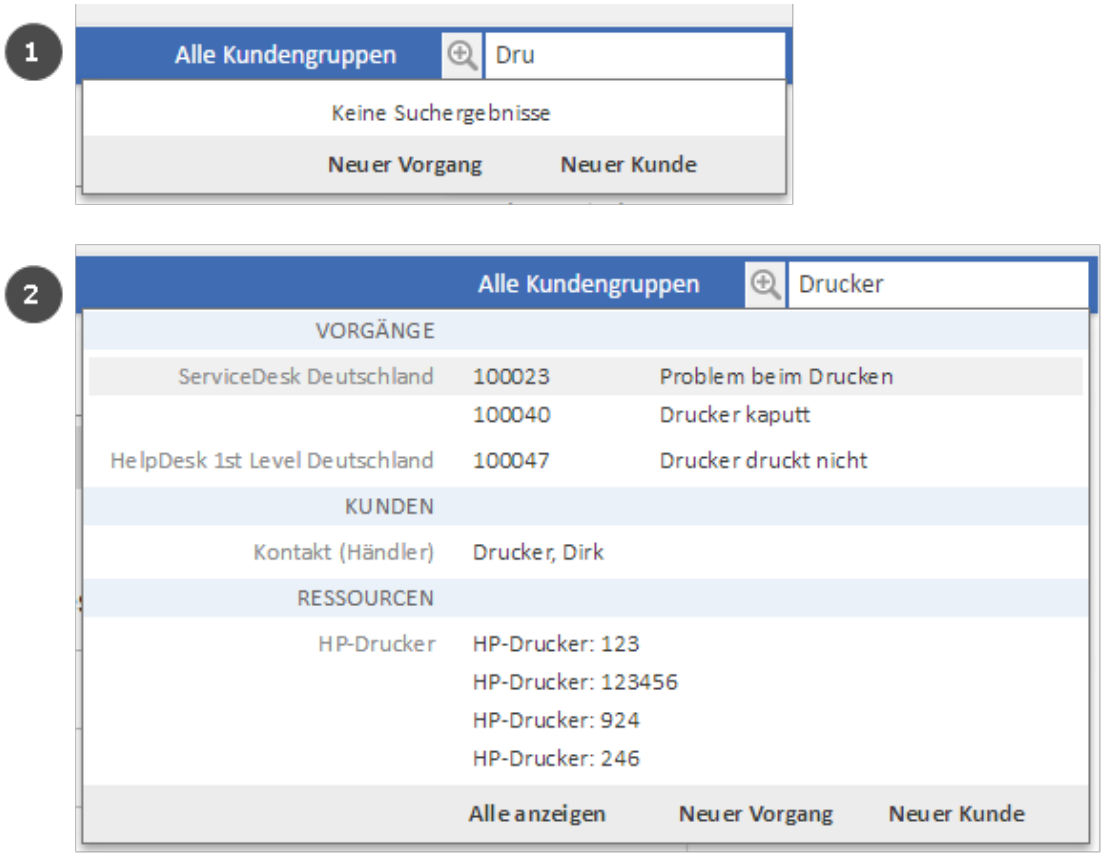


Abbildung 181: *Ergebnisse der Schnellsuche ohne automatisch angefügten Platzhalter (appendWildcardAutomatically = false)*

- **maxHints**
  Definiert die maximale Anzahl an angezeigten Vorschlägen. Wenn dieses Attribut auf „0“ gesetzt ist, werden alle Vorschläge ohne Beschränkung angezeigt.
- **searchResultItemsOrder**
  Durch Kommas getrennte Werte, mit denen die Reihenfolge und die Sichtbarkeit der Suchergebnisse definiert wird. (java.lang.String)
  Mögliche Werte: CONTACT, COMPANY, TICKET
- **suffixCharactersToRemove**
  Jedes der angegebenen Zeichen wird implizit entfernt, wenn es am Ende eines Suchwortes vorkommt. (java.lang.String)

mailTemplate (Typ)

(verfügbar in *ticketEditPage/acimSection*)

Attribute:

- **addingManyTemplateEnabled**
  Boolean. Ermöglicht das Verfassen einer E-Mail mithilfe von mehr als einer Vorlage.
  Obsolet ab CM-Version 6.10.5.0, da in dieser CM-Version ein neuer Textvorlagen-Manager

eingeführt wurde, der (neben anderen Funktionen) die Möglichkeit bietet, mehr als eine Vorlage in eine E-Mail im Ticket-E-Mail-Editor bzw. in einen Kommentar im Ticket-Kommentar-Editor einzufügen.

- **emailDisplayMode**
  String. Definiert das Format der E-Mail-Adresse, die im Ticket-E-Mail-Editor in den Feldern An/Cc/Von/Bcc angezeigt wird. Mögliche Werte: „FULL“ (Name und E-Mail-Adresse), „NAME“, „EMAIL“. Standardwert: „FULL“.
- **engineerPersonalMailsIncluded**
  Boolean. Aktiviert die Funktion zum Anhängen der persönlichen E-Mail. Wenn der Wert von *engineerPersonalMailsIncluded* „false“ ist, werden neue E-Mail-Adressen, die in die Felder *An*, *Cc*, *Bcc* eingegeben werden, nicht gespeichert und werden beim nächsten Mal nicht als Vorschläge angezeigt.
- **includeTextBlocksInEmailTemplate**
  Boolean. Gibt an, ob die Textblöcke aus einer E-Mail-Vorlage standardmäßig eingefügt sind.
- **mailBodyLockedAfterTemplateSelection**
  Boolean. Gibt an, ob der Textteil der E-Mail nach Auswahl einer Vorlage gesperrt ist.
- **mailEncryptionAvailable**
  Boolean. Macht die Option zur E-Mail-Verschlüsselung verfügbar.
- **mailSelectionComponentWidth**
  Die Breite der Komponente zur E-Mail-Auswahl (in Pixel). (java.lang.Integer)
- **mailTemplateSortStrategy**
  Strategie zur Sortierung der E-Mail-Vorlagenliste. (java.lang.String)
  Standardwert: USAGE, NAME. Mögliche, durch Kommas getrennte Optionen sind: USAGE, NAME
- **maxElementLength**
  Die maximale Länge eines einzelnen Elements. Wenn der Wert der Variable auf „0“ gesetzt ist, werden die Elemente nicht abgeschnitten. (java.lang.Integer)
- **minMailInputLength**
  Minimale Eingabelänge, bei der das Drop-down-Menü mit E-Mail-Vorschlägen geöffnet wird, wenn der Bearbeiter mit der Eingabe von E-Mail-Empfängern beginnt. Integer. Standardwert ist 1.
- **quotingFeature**
  Aktiviert die Zitatfunktion im E-Mail-Text. (boolean)
- **selectionComponentWidth**
  Die Breite der Komponente zur E-Mail-/Kommentar-Auswahl (in Pixel). Integer.
- **showBcc**
  Boolean. Gibt an, ob das Bcc-Feld im Ticket-E-Mail-Editor angezeigt wird. Der Standardwert ist „true“.
- **showCc**
  Boolean. Gibt an, ob das Cc-Feld im Ticket-E-Mail-Editor angezeigt wird. Der Standardwert ist „true“.

- **showReplyTo**
  Boolean. Gibt an, ob das Reply-To-Feld im Ticket-E-Mail-Editor angezeigt wird. Der Standardwert ist „true“.

  > Beachten Sie, dass das Attribut *showReplyTo* nicht der einzige Parameter ist, der in ConSol CM die Anzeige und Verwendung der Reply-To-Adresse einer E-Mail beeinflusst! Eine detaillierte Erklärung hierzu finden Sie im Abschnitt Skripte des Typs E-Mail.

- **showUniqueEmails**
  Boolean. Ergebnisse in den E-Mail-Feldern mit Autovervollständigen werden anhand der E-Mail-Adresse verglichen. In den Treffern wird nur die erste E-Mail-Adresse angezeigt. Wenn dieses Attribut auf „true“ gesetzt ist, werden die Ergebnisse mit der kompletten Beschreibung der E-Mail verglichen.
- **templateSortStrategy**
  String. Strategien für die Sortierung von Vorlagen. Der Standardwert ist: USAGE, NAME. Mögliche, durch Kommas getrennte Optionen sind: USAGE, NAME. Standardwert ist USAGE,NAME.

markersLibrary (Typ)

Dieser Typ der Seitenanpassung ist auf der folgenden Seite verfügbar:

- templateEditPage (im Textvorlagenmanager)

Attributes:

- **excludedFields**
  Die Liste der ausgeschlossenen Ticketfelder, z. B. *helpdesk.module,sales.priority*. Hier aufgeführte Felder werden nicht im Menü *Kriterien wählen* angezeigt. In großen Umgebungen kann das Laden von *templateEditPage* durch Ausschließen einiger Felder beschleunigt werden. Es muss allerdings klar sein, dass diese Felder nie für die Zuweisung oder Zuordnung von Vorlagen benötigt werden.
  Das Attribut kann auf Typebene oder für den Unterbereich *templateEditPage/templateSection/bindings* gesetzt werden.

navigationLinks (Typ)

Hier können Sie die Darstellung von mehreren Hyperlinks definieren, die im Hauptmenü des Web Clients angezeigt werden.

Attribute:

- **createContactLinkVisible**
  Definiert, ob der Link (Hauptmenüeintrag) *createContact* (*Neuer Kunde*) angezeigt werden kann. Standardwert ist „true“. (boolean)

> Zusätzlich zu diesem auf „true“ gesetzten Attribut müssen die Bearbeiter die entsprechenden Berechtigungen haben, um den Hauptmenüeintrag *Neuer Kunde* sehen zu können.

- **createTicketLinkVisible**
  Definiert, ob der Link (Hauptmenüeintrag) *createTicket* (Neues Ticket) angezeigt werden kann. Standardwert ist „true“. (boolean)

> Zusätzlich zu diesem auf „true“ gesetzten Attribut müssen die Bearbeiter die entsprechenden Berechtigungen haben, um den Hauptmenüeintrag *Neues Ticket* sehen zu können.

- **externalLinks**
  Externe Links (Hauptmenüeinträge), die zum Hauptmenü hinzugefügt werden. Mit diesem Attribut können mehrere externe Links konfiguriert werden (die Reihenfolge ist wichtig, keine Zeichen zwischen den Link-Ausdrücken).
  Format (kompatibel mit wiki): [http://Beschreibung Link] oder [http://Beschreibung Link]@target
  Mit diesem Attribut können Sie Hyperlinks zur Website der Firma, einer Reporting-Anwendung, einer Hilfeseite oder eine beliebige gültige URL integrieren.
  Beispiel: [http://www.consol.com ConSol*][http://www.somewhere.com Somewhere]

> Links werden standardmäßig in einem neuen Tab geöffnet. Wenn der Link im gleichen Tab geöffnet werden soll, fügen Sie nach dem Link @ ein, z. B. [http://www.consol.com ConSol]@

- **manageTemplateLinkVisible**
  Definiert, ob der Link (Hauptmenüeintrag) *Textvorlagen* (für den Start des ConSol CM Textvorlagen-Managers) angezeigt werden kann. Standardwert ist „true“. (boolean). Siehe auch Abschnitt Der ConSol CM Textvorlagen-Manager.

> Zusätzlich zu diesem auf „true“ gesetzten Attribut müssen die Bearbeiter die entsprechenden Berechtigungen (*Allgemeine Berechtigungen - Template speichern*) haben, um den Hauptmenüeintrag *Textvorlagen* sehen zu können.

- **officeTemplateLinkVisible**
  Definiert, ob der Link (Hauptmenüeintrag) *Dokumentvorlagen* (für den Start des Dokumentvorlagen-Managers) angezeigt werden kann. Standardwert ist „true“. (boolean).

> Zusätzlich zu diesem auf „true “gesetzten Attribut müssen die Bearbeiter die entsprechenden Berechtigungen (*Allgemeine Berechtigungen - Template speichern*) haben, um den Link *Dokumentvorlagen* sehen zu können. Außerdem muss CM/Doc im System aktiviert sein (siehe Abschnitt CM/Doc).

- **overviewLinkVisible**
  Definiert, ob der Link (Hauptmenüeintrag) *Hauptseite* angezeigt werden kann.

preview (Typ)

Mit den Attributen wird das Verhalten des CM Web Clients bezüglich der Vorschaufunktion für Ergebnistabellen mit Tickets definiert.

Verfügbar auf

- der Detailseite der Suche (Tab *Tickets*): preview **/searchDetailPage/TicketSearchResults**
- der Ressourcenseite (Ticketbereich): preview**/resource/ticketRelationsSection/ResourceTicketRelationSearchResults**

Setzen Sie die Attribute für den fettgedruckten Unterbereich.

Attribute:

- **enabled**
  Boolean. Aktiviert/Deaktiviert die Vorschaufunktion innerhalb der Tabelle. Der Standardwert ist „true“.
- **previewHeight**
  Integer. Die Höhe der Attachment-Vorschau in Pixeln. Standardwert ist 200.

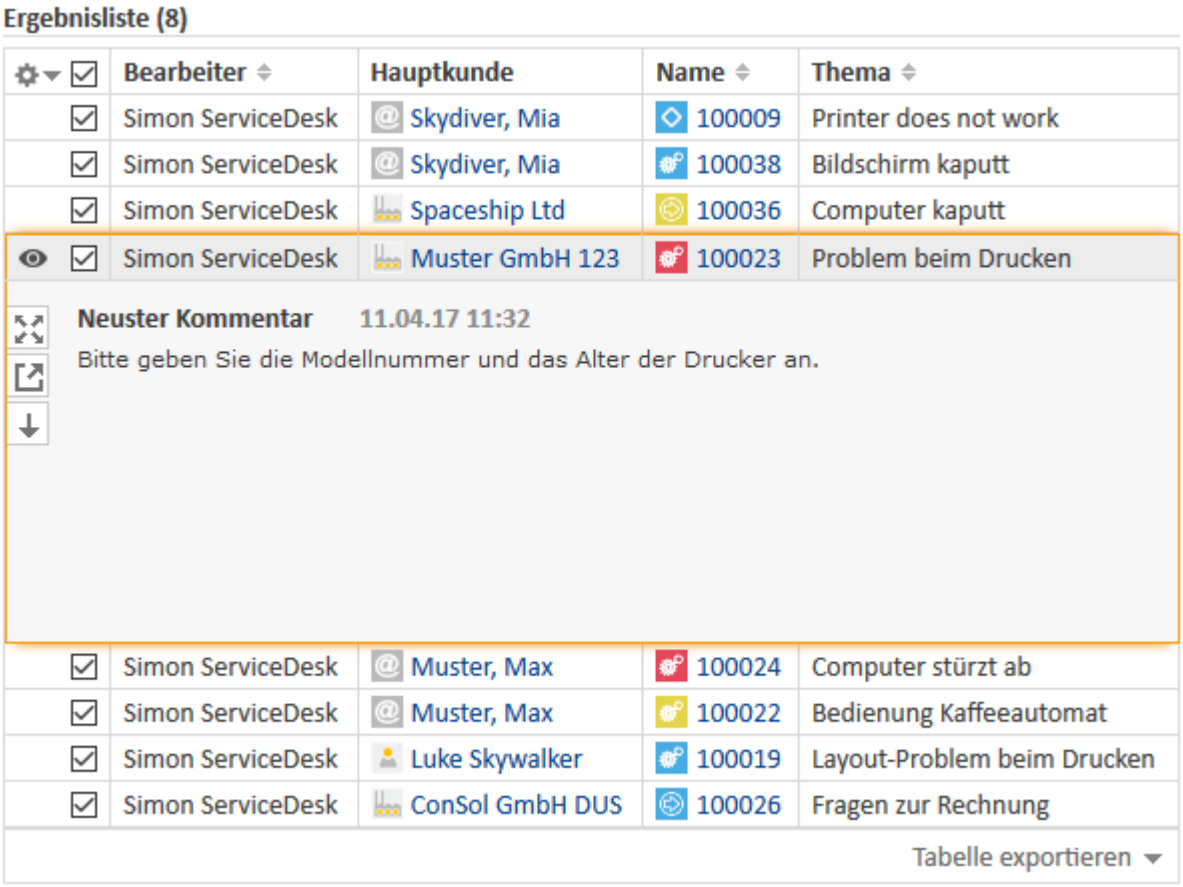


Abbildung 182: *ConSol CM Web Client - Vorschau mit /searchDetailPage/TicketSearchResults, enabled = true*

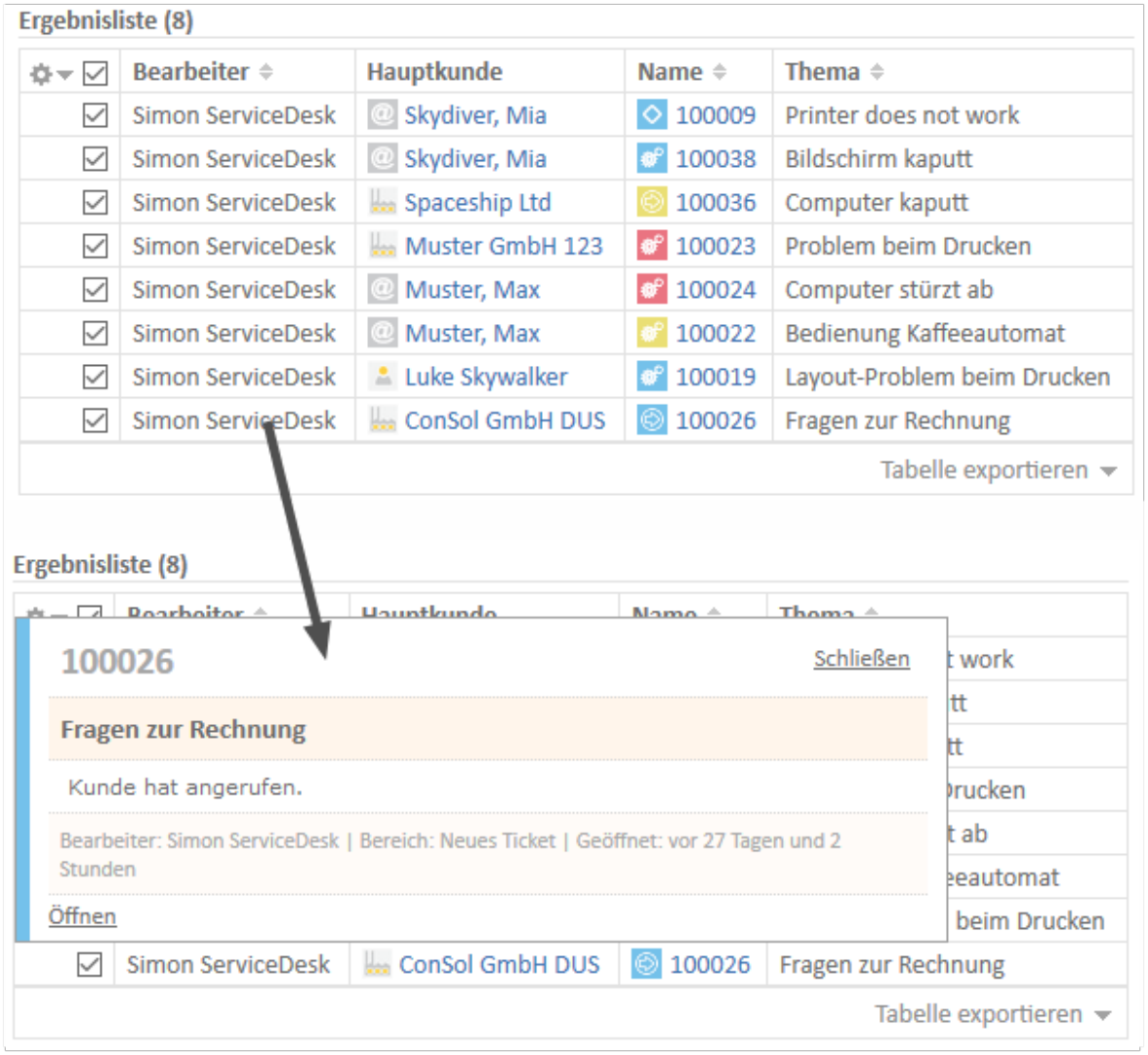


Abbildung 183: *ConSol CM Web Client - Vorschau mit /searchDetailPage/TicketSearchResults, enabled = false*

Eine detaillierte Beschreibung der Vorschaufunktion finden Sie im *ConSol CM Benutzerhandbuch*.

resourceRelations

Verfügbar z. B. auf ticketEditPage.

Attribute:

- **counterVisible**
  Definiert, ob der Zähler in der Kopfzeile des Bereichs angezeigt werden soll. Der Standardwert ist „true“.

- **defaultRelationSortStrategy**
  Zusätzliche Strategien zum Sortieren von Ressourcenrelationen. Mögliche Optionen sind: CREATION_DATE, DESCRIPTION, Standardwert ist CREATION_DATE.
- **state**
  Der Sichtbarkeitsmodus dieses Bereichs, mögliche Werte sind [expanded, collapsed, collapsed_and_preload, hidden], Standardwert: "expanded".

resourceRelationsPanel

Verfügbar z. B. auf ticketEditPage.

Attribute:

- **preserveOrder**
  Ermöglicht die Beibehaltung der Reihenfolge der Elemente. Standardmäßig werden Elemente basierend auf dem freien Platz (vertikal) optimal angeordnet. Der Standardwert ist „false“.

resourceTypes

Verfügbar z. B. auf dem Ressourcen-Dashboard

Attribute:

- **preserveOrder**
  Ermöglicht die Beibehaltung der Reihenfolge der Elemente. Standardmäßig werden Elemente basierend auf dem freien Platz (vertikal) optimal angeordnet. Der Standardwert ist „false“.

section (Typ)

Verfügbar in folgenden Bereichen:

- ticketEditPage
- contactEditPage
- companyEditPage
- userProfilePage

Attribute:

- **state**
  Der Sichtbarkeitsmodus dieses Bereichs. Eine detaillierte Beschreibung finden Sie unter *sectionList*.

sectionList (Typ)

Verfügbar in folgenden Bereichen:

- ticketEditPage
- contactEditPage
- companyEditPage

Attribute:

- **counterVisible**
  Definiert, ob der Zähler in der Kopfzeile des Bereichs angezeigt werden soll. Der Standardwert

ist „true“.

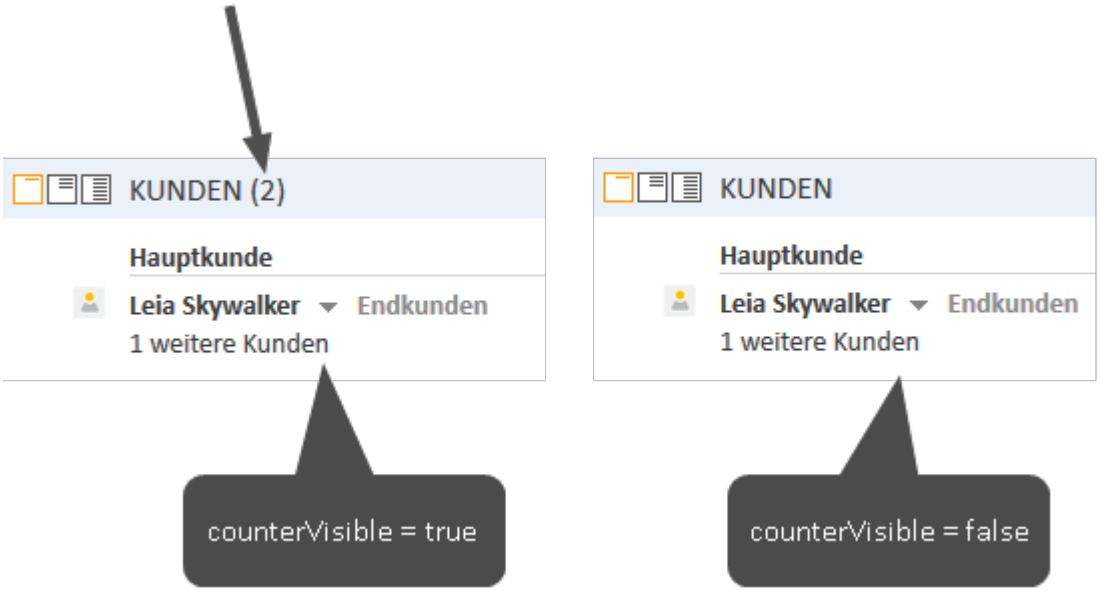


Abbildung 184: *ConSol CM Web Client - Attribut counterVisible*

- **state**
  Der Sichtbarkeitsmodus dieses Bereichs. Mögliche Werte sind [expanded, collapsed, collapsed_and_preload, hidden]. Der Standardwert ist „expanded“. Mit diesem Modus wird die initiale Sichtbarkeit nach dem Start einer Session definiert. Der Bearbeiter kann die Sichtbarkeit des Bereichs danach ändern. Seine Einstellung wird allerdings nicht gespeichert. Wenn die nächste Session gestartet wird (nächste Anmeldung), wird der initiale Sichtbarkeitsmodus erneut angewendet.
  - **expanded**
    Standardwert, die Daten werden am Anfang angezeigt
  - **collapsed**
    Die Daten werden am Anfang nicht angezeigt und auf Anfrage geladen. Damit kann die Geschwindigkeit leicht verbessert werden.
  - **collapsed_and_preload**
    Die Daten werden am Anfang nicht angezeigt, aber geladen.
  - **hidden**
    Dieser Bereich ist verborgen und kann nicht wieder eingeblendet werden. Diese Einstellung kann nur mit Tools wie der Administratorkonsole des Application Servers rückgängig gemacht werden.

Mit dem Attribut HIDDEN werden alle Kundendaten ausgeblendet !

Wenn das Attribut *state* =„ hidden“ für einen Abschnitt/Bereich gesetzt wurde, wird dieser Bereich nicht mehr in der Baumstruktur der Seitenanpassung angezeigt und ist daher in der Seitenanpassung nicht mehr verfügbar. Sie können diese Einstellung nur mittels Tools, die direkt auf die Java-Beans zugreifen, zurücksetzen, z. B. **JConsole** für JBoss. Überlegen Sie es sich also zweimal, bevor Sie diesen Wert setzen.



Abbildung 185: *ConSol CM Web Client - Attribut state (Beispiel: Kundenbereich eines Tickets)*

table

Mit diesem Attribut kann die CSV-Exportfunktion der Liste mit den Suchergebnissen konfiguriert werden. Eine Erklärung dazu finden Sie im Abschnitt CSV-Export von Suchergebnissen.

template

Verfügbar auf allen Seiten, auf denen es einen Rich-Text-Editor mit der Möglichkeit zum Einfügen einer Textvorlage gibt. Das Format wird immer im niedrigsten Bereich, für den das Attribut verfügbar ist, definiert.

Attribute:

- **templateNumberFormat**
  Zur Definition von numerischen Variablen verwendet, d. h. Integer (NUMBER) oder Dezimalzahl (FIXED-POINT NUMBER) für die Platzhalter (Marker) in Textvorlagen.
  Beispiel:

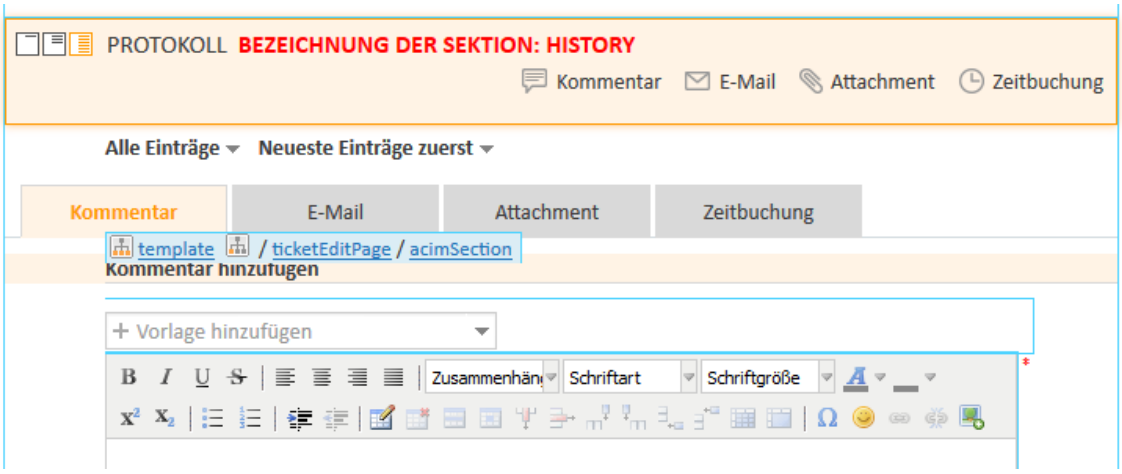


Abbildung 186: *Seitenanpassung im Web Client aktiviert, um templateNumberFormat zu setzen. acimSection verwenden!*

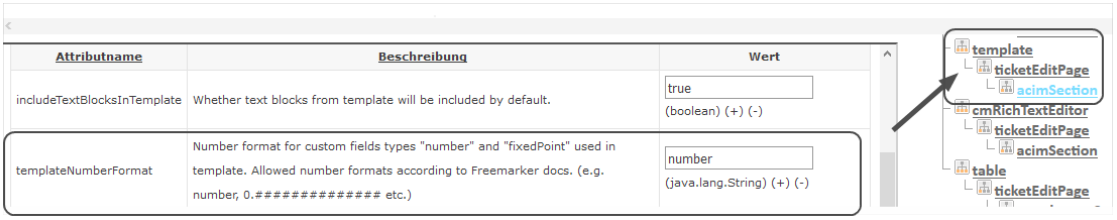


Abbildung 187: *Attribut der Seitenanpassung templateNumberFormat in acimSection*

Der aktuell angezeigte Wert „number“ stellt das Standardformat dar und kann verwendet werden, um wieder die Standardeinstellung zu verwenden, nachdem vorher eine eigene Formatkonfiguration im Einsatz war. Die Syntax zur Formatdefinition ist in der öffentlichen Java-Dokumentation für die Klasse DecimalFormat erklärt.

Die wichtigsten Elemente sind:

- „0“ - Zeigt an dieser Stelle immer eine Ziffer an. Zeigt 0 an, wenn es an dieser Stelle keine Ziffer gibt.
- „#“ - Wird nur angezeigt, wenn die Zahl an dieser Stelle eine Ziffer hat.
- „.“ - Dezimaltrennzeichen
- „,“ - Internes Gruppierungstrennzeichen für Ganzzahl-/Dezimalpositionen

Der Attributwert „00000.#####““ führt zum Beispiel dazu, dass immer fünf Ziffern vor dem Dezimaltrennzeichen und bis zu fünf Zeichen nach dem Dezimaltrennzeichen angezeigt werden, je nachdem wie viele Nachkommastellen die Dezimalzahl hat. Ganzzahlen werden immer mit fünf Ziffern angezeigt. Dieser Effekt ist in der nachfolgenden Abbildung zu sehen. Die erste Zahl in Blau ist eine Ganzzahl, die zweite Zahl ist eine Dezimalzahl mit drei Nachkommastellen.

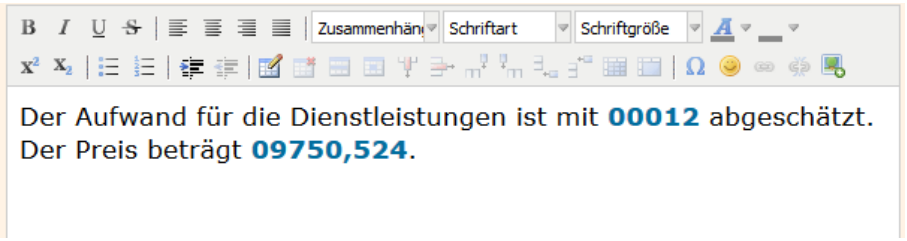


Abbildung 188: *Textvorlage mit Zahlen im Textvorlagen-Manager*

ticketList (Unterbereich)

Siehe accordionTicketList (Typ).

ticketPanel

Attribute:

- **scrollSpeed**
  Scroll-Geschwindigkeit in Millisekunden. Mit diesem Attribut wird festgelegt, wie schnell die Seite gescrollt wird, wenn Sie am rechten Rand der Hauptseite auf den Scroll-Button klicken.

  Dieser Wert legt fest, wie lange die Animation ausgeführt wird. Typische Werte: 200, 600, 1000 ... (je höher der Wert, desto langsamer) (java.lang.String, Standardwert = 200), z. B. heißt 200, dass das Scrollen zum Ende/Anfang der Seite 200 ms dauert.

- **topBottomPageButtonVisible**
  Definiert, ob der Scroll-Button sichtbar ist (boolean, der Standardwert ist „false“).
  - **topBottomPageButtonVisible = false:**

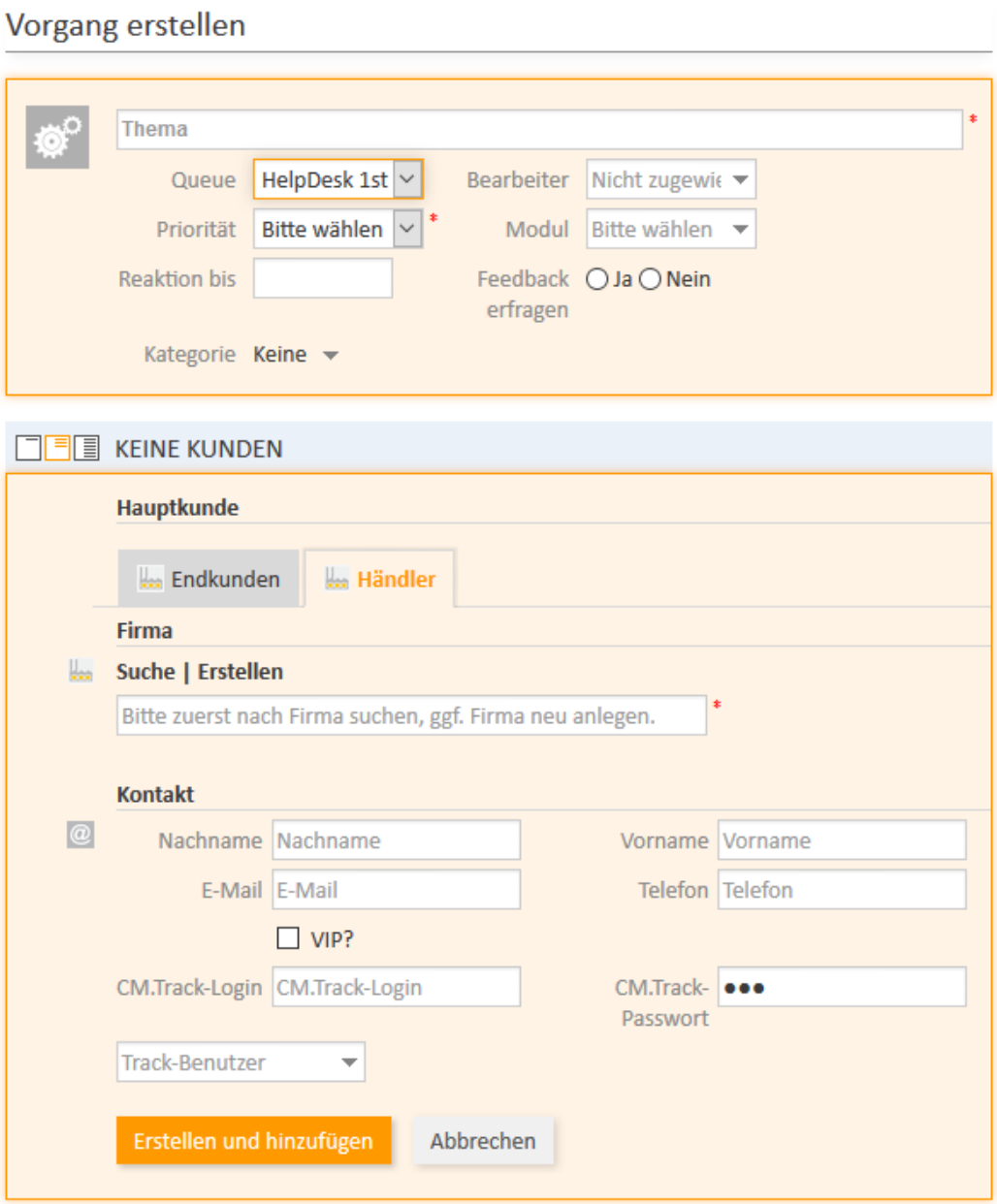


Abbildung 189: *ConSol CM Web Client - Scroll-Button nicht sichtbar (topBottomPageButtonVisible = false)*

- **topBottomPageButtonVisible = true:**

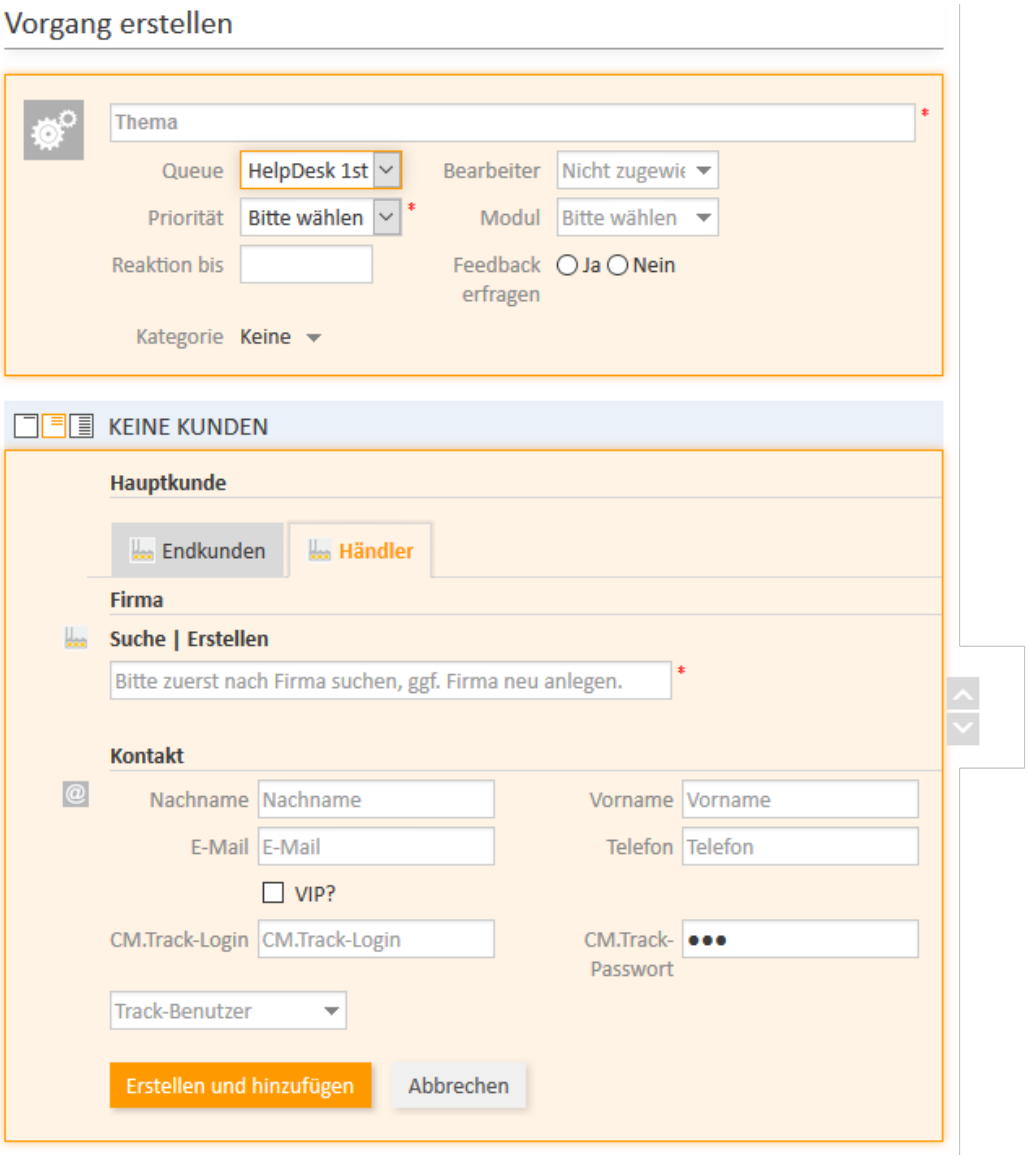


Abbildung 190: *ConSol CM Web Client - Scroll-Button sichtbar (topBottomPageButtonVisible = true)*

TicketRelation

Verfügbar in folgenden Bereichen:

- TicketRelationSection (z. B. auf der Ressourcenseite)

Attribute:

- **compactViewLimit**
  Wenn die Grenze überschritten ist, wird die Tabelle zusammen mit einer Filterfunktion

angezeigt. Die Standardgrenze ist 10.

- **defaultRelationSortStrategy**
  Zusätzliche Strategien zum Sortieren von Ressourcenrelationen. Mögliche Optionen sind: CREATION_DATE (Standardwert), FIRST_SORTABLE_COLUMN (nach der ersten sortierbaren Spalte sortieren)

ticketsAutocomplete

(verfügbar im Formular zum Hinzufügen von Relationen)

Attribute:

- **maxHints**
  Definiert die maximale Anzahl an anzuzeigenden Vorschlägen. Wenn dieses Attribut auf „0“ gesetzt ist, werden alle Vorschläge ohne Beschränkung angezeigt.

ticketsBookingAutocomplete

(verfügbar im Formular zum Hinzufügen von Zeitbuchungen auf der Seite *userProfilePage*)

Attribute:

- **maxHints**
  Definiert die maximale Anzahl an anzuzeigenden Vorschlägen. Wenn dieses Attribut auf „0“ gesetzt ist, werden alle Vorschläge ohne Beschränkung angezeigt.

timeBookingSection

(verfügbar z. B. in *userProfilePage*)

Attribute:

- **visible** (bis CM-Version 6.9.3.x)
  Boolean. Die Sichtbarkeit des Zeitbuchungsbereichs auf der Seite *userProfilePage*. (Standardwert = „true“)
  Denken Sie daran, dass die Sichtbarkeit des Zeitbuchungsbereichs auf der Ticketseite in acimSection, Attribut *timeBookingFeature* konfiguriert wird!
- **state** (CM-Version 6.10.4 und höher)
  Mögliche Werte sind:
  - **expanded**
    Standardwert, die Daten werden initial angezeigt.
  - **collapsed**
    Die Daten werden initial nicht angezeigt und auf Anfrage geladen.
  - **collapsed_and_preload**
    Die Daten werden initial nicht angezeigt, aber schon geladen.
  - **hidden**
    Der Bereich ist verborgen und kann nicht eingeblendet werden). Kann nur mit Tools wie der Administratorkonsole des Application Servers rückgängig gemacht werden.

> ⚠ Mit dem Attribut **hidden** wird der Zeitbuchungsbereich endgültig ausgeblendet.
>
> Wenn das Attribut *state* = „hidden“ für einen Abschnitt/Bereich gesetzt wurde, wird dieser Bereich nicht mehr in der Baumstruktur der Seitenanpassung angezeigt und ist daher in der Seitenanpassung nicht mehr verfügbar. Sie können diese Einstellung nur mittels Tools, die direkt auf die Java-Beans zugreifen, zurücksetzen, z. B. ***JConsole*** für JBoss. Überlegen Sie es sich also zweimal, bevor Sie diesen Wert setzen.

unitAutocomplete

(verfügbar in Formularen zum Hinzufügen und Erstellen von Kunden)

Attribute:

- **maxHints**
  Definiert die maximale Anzahl an angezeigten Vorschlägen. Wenn dieses Attribut auf „0“ gesetzt ist, werden alle Vorschläge ohne Beschränkung angezeigt.

unitFormPanel

Verfügbar auf:

- contactCreatePage
- ticketEditPage
- contactEditPage, z. B. unitFormPanel / ticketCreatePage / contactSection

Attribute:

- **maxSuggestions**
  Dieses Attribut bezieht sich auf den Kundenbereich, der beim Erstellen eines neuen Tickets angezeigt wird. Hier werden die Vorschläge für Kunden angezeigt, wenn in der Datenbank passende Treffer gefunden werden. Die Anzahl der angezeigten Vorschläge kann mit diesem Attribut konfiguriert werden.

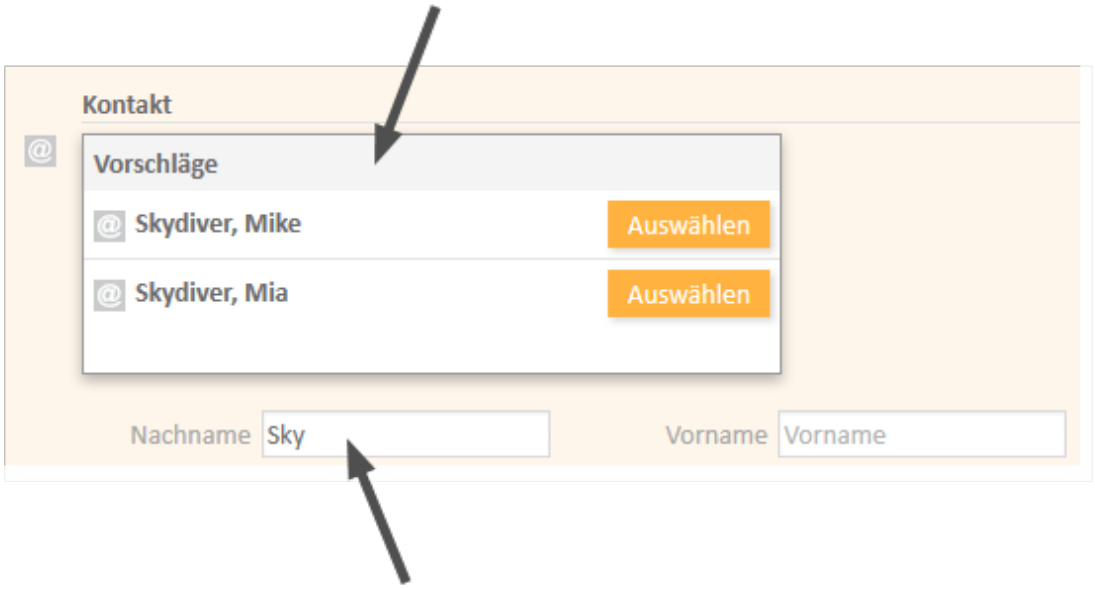


Abbildung 191: *ConSol CM Web Client - Vorschläge für den Ticketkontakt*

unitRelationSection (Typ UnitSection)

Verfügbar in:

- contactEditPage
- companyEditPage

Attribute:

- **compactViewLimit**
  Wenn die Grenze überschritten ist, werden die Relationen zusammen mit einer Filterfunktion angezeigt. Die Standardgrenze ist „10“.



Abbildung 192: *ConSol CM Web Client - Effekt des Attributs compactViewLimit = 3 (contactEditPage / unitRelationSection)*

- **numberOfRelations**
  Die Standardanzahl der anzuzeigenden Relationen. Der Standardwert ist „10“.
- **unitPreviewLayout**
  Die Konfiguration der Unit-Vorschau. Diese wird für das Feld verwendet, das angezeigt wird, wenn man im Bereich *Verknüpfte Firmen und Kontakte* auf den Namen eines Kunden oder einer Firma klickt. Das ist ähnlich wie das Feld, das angezeigt wird, wenn man in einer Liste mit Tickets auf einen Ticketnamen klickt.
  Dazu muss ein JSON-Objekt zurückgegeben werden, siehe folgendes Code-Beispiel.

```
{ "layout": [ ["firstname", "lastname", "lastname"], ["email", "email", null]
 ]}
```

Code-Beispiel 2: *JSon-Objekt für das Kundendatenformat in der Vorschau*

Für unterschiedliche Unit-Definitionen in FlexCDM können Sie folgende Syntax verwenden:

```
{"unit_definition_1": "<json description>", "unit_definition_2": "component_
 name_used_to_display_preview"}.
```

Code-Beispiel 3: *Beispielwert für das Layout der Kundendatenvorschau*

In JSON wird die Konfiguration für eine bestimmte Unit-Definition festgelegt. Das JSON-Objekt kann die Unit-Vorschau oder nur den Namen der Komponente (Spring-Bean) enthalten, mit der die Vorschau erzeugt wird.

UnitResourceRelation

Verfügbar in unitRelationsSection (z. B. auf der Ressourcenseite).

Attribute:

- **compactViewLimit**
  Wenn die Grenze überschritten ist, werden die Relationen zusammen mit einer Filterfunktion angezeigt. Die Standardgrenze ist 10.
- **defaultRelationSortStrategy**
  Zusätzliche Strategien zum Sortieren von Ressourcenrelationen. Mögliche Optionen sind: CREATION_DATE (Standardwert), FIRST_SORTABLE_COLUMN (nach der ersten sortierbaren Spalte sortieren)
- **numberOfRelations**
  Die Standardanzahl der anzuzeigenden Relationen. Der Standardwert ist 10.

unitSearch

(verfügbar in *ticketCreatePage* im Firmenbereich)

Attribute:

- **aidLevel**
  Hilfestufe für CM-Einsteiger:
  - NONE
  - BASIC (breiteres Suchfeld mit Erklärung)
  - EXTENDED (wie BASIC plus ein zusätzliches Hilfesymbol mit Tooltip)

  (java.lang.String, Standardwert = „BASIC“)

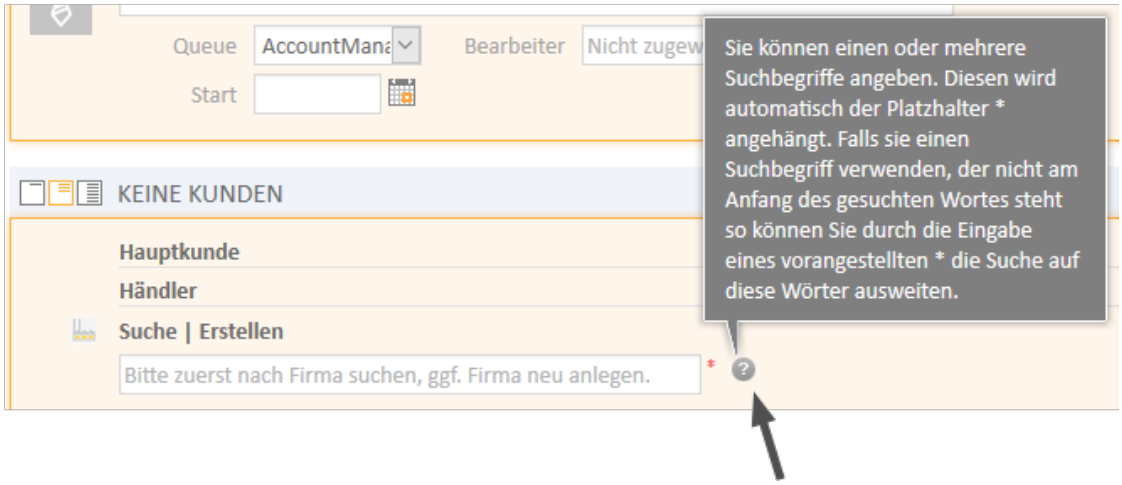


Abbildung 193: *ConSol CM Web Client - aidLevel EXTENDED in Firmensuche*

unitSearchHeader

(verfügbar in *ticketCreatePage* im Firmenbereich)

Attribute:

- **companyCreateLinkVisible**
  Boolean. Die Sichtbarkeit des Links zur Erstellung von referenzierten Firmen.

viewDiscriminatorsSection (Typ)

(verfügbar z. B. in *userProfilePage*)

Attribute:

- **visibilityFlag**
  Boolean. Die Sichtbarkeit des Bereichs *Sichtenmerkmale* (Bereich für die Einstellungen für dynamische Sichten in *userProfilePage*). (Standardwert = „true“)

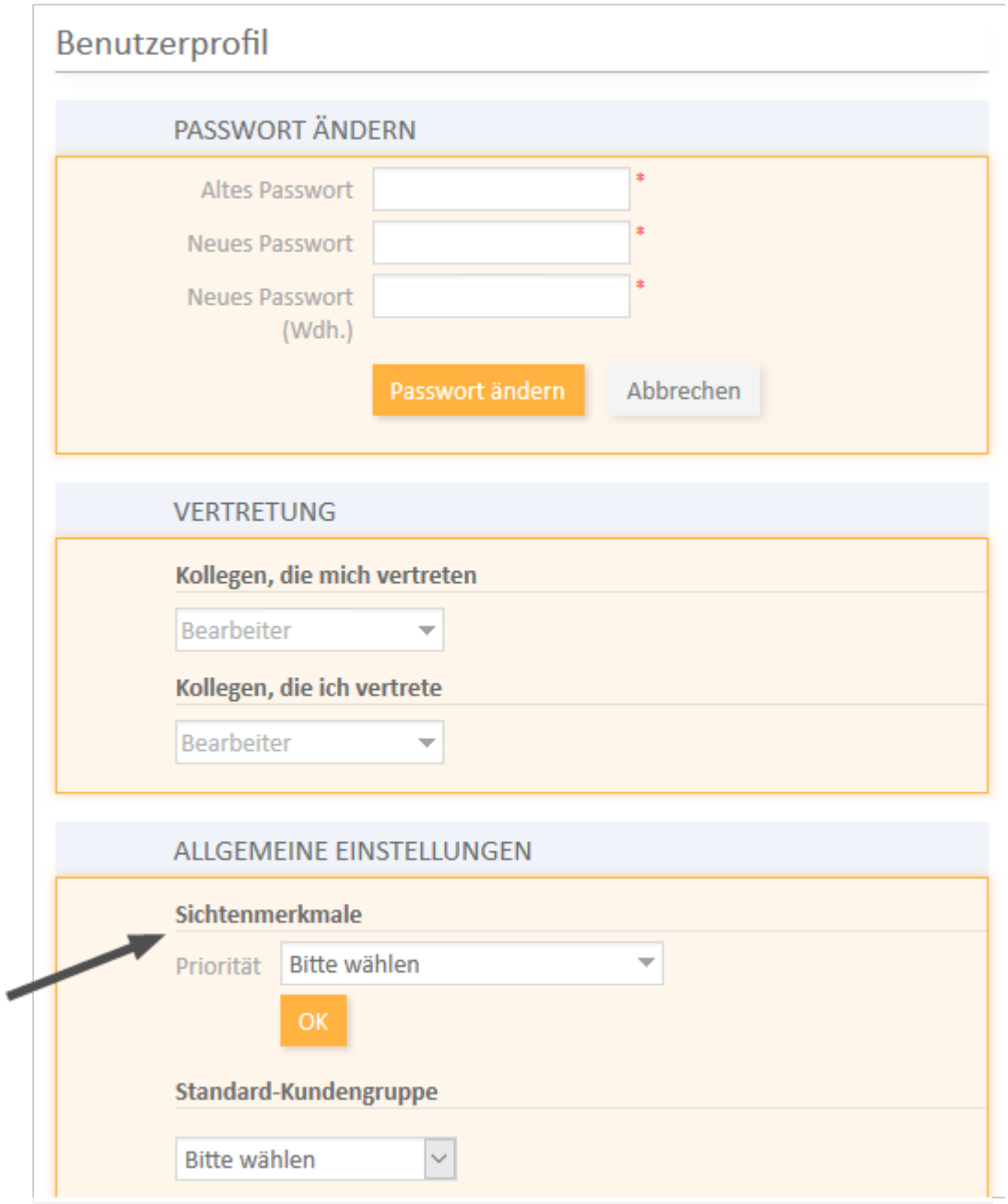


Abbildung 194: *ConSol CM Web Client - Sichtbarkeit des Bereichs Sichtenmerkmale (visibilityFlag=true)*

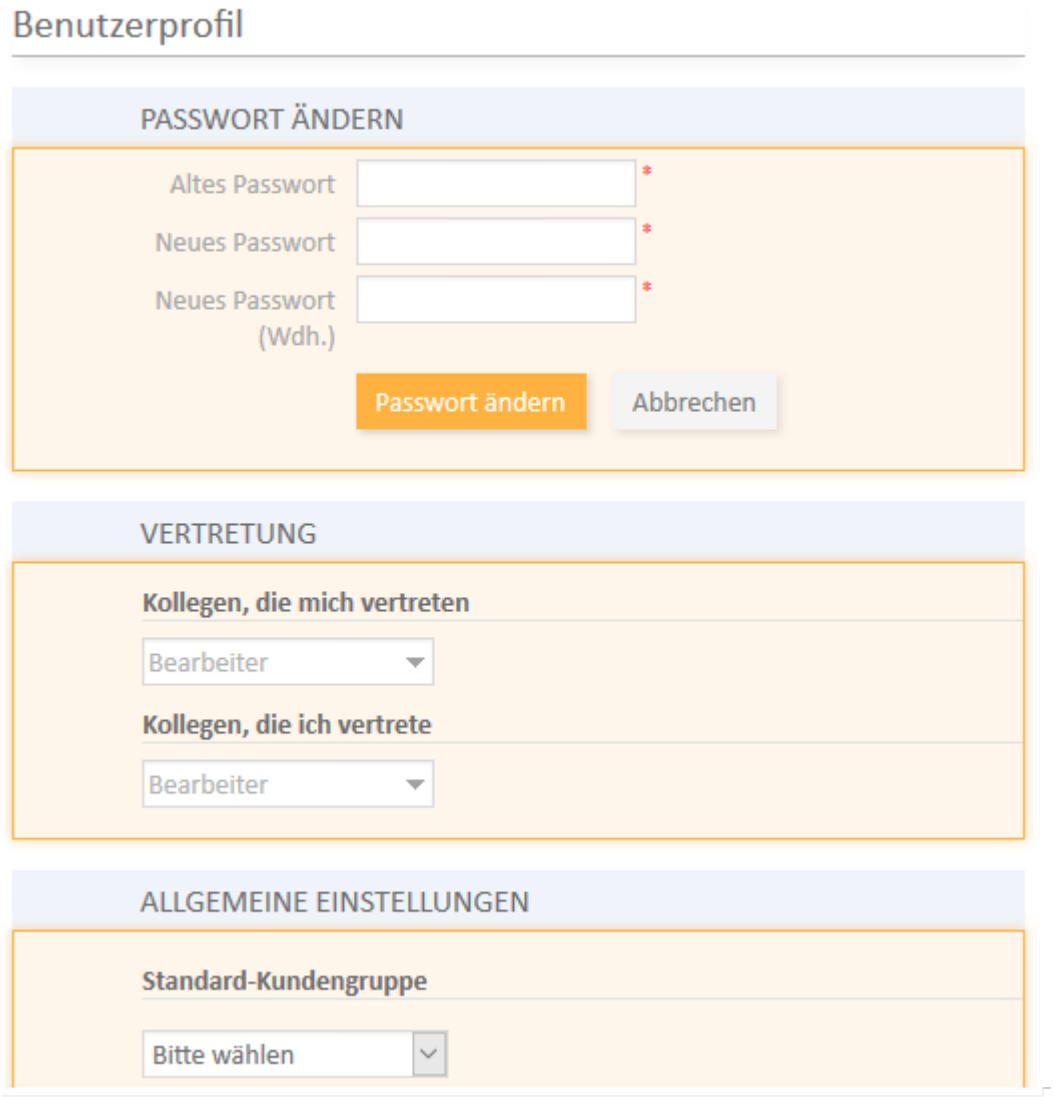


Abbildung 195: *ConSol CM Web Client - Sichtbarkeit des Bereichs Sichtenmerkmale (visibilityFlag=false)*

welcomePage

Die wichtigste Konfiguration der *welcomePage* (Hauptseite) ist die Seitenanpassung für das Web Client Dashboard.

C.8.6 Seitenanpassung für das Web Client Dashboard

C.8.6.1 Einleitung

Das **Web Client Dashboard** wird mittels Seitenanpassung konfiguriert.

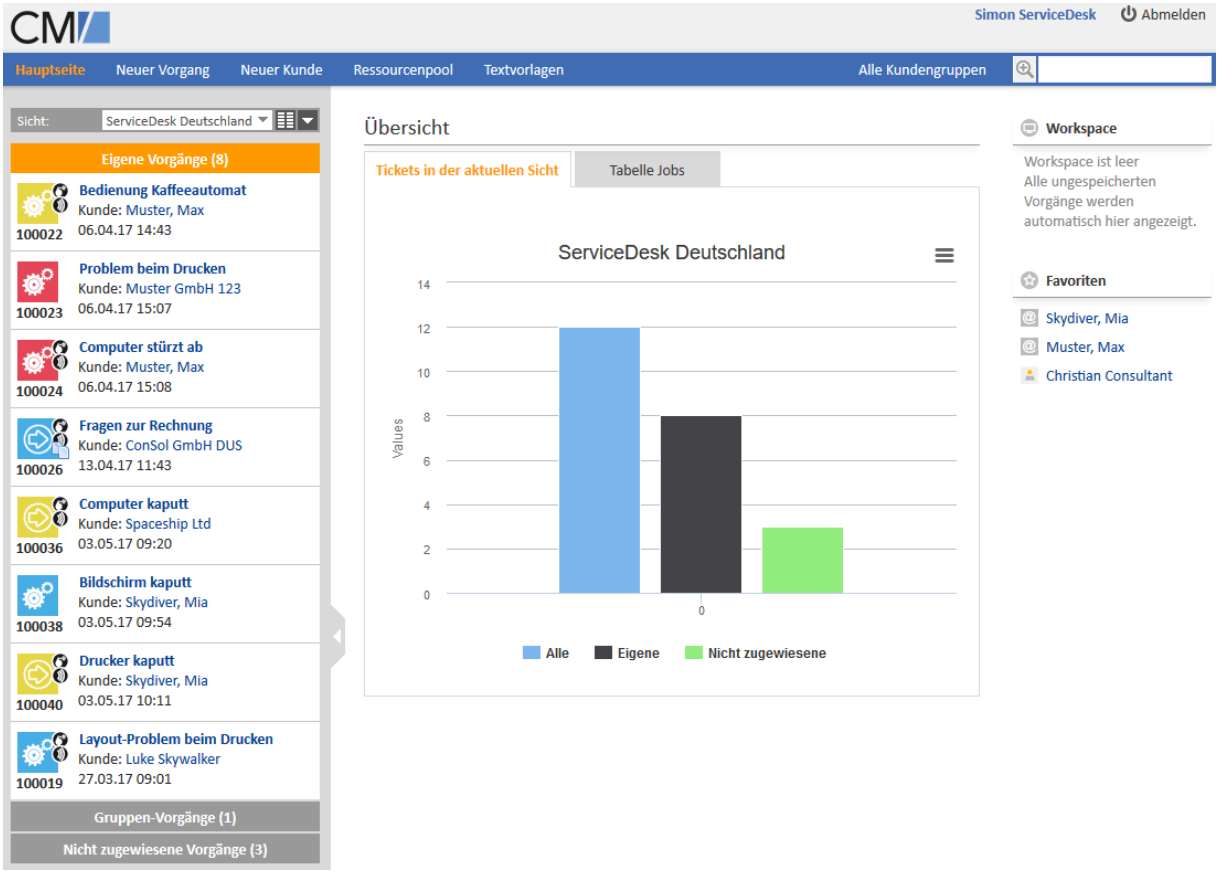


Abbildung 196: *ConSol CM Web Client - Web Client Dashboard mit mehreren Tabs. Offener Tab mit Chart-Widget*

Melden Sie sich als *admin* an, öffnen Sie die *Hauptseite* und wählen Sie im Hauptmenü *Seitenanpassung aktivieren*. Die folgenden (für das Dashboard relevanten) Elemente können hier konfiguriert werden:

1. ***widgetsGrid / welcomePage***
   Hier kann das Web Client Dashboard ein- und ausgeschaltet werden. Wenn das Feld *layout* einen korrekten Wert enthält, wird das Dashboard angezeigt (**Vorsicht:** Wenn der Wert nicht korrekt ist, startet der Web Client nicht!). Wenn das Feld leer ist, wird kein Dashboard angezeigt.
   Hier können folgenden Konfigurationen vorgenommen werden:

   a. Das Layout des Dashboards, d. h. das Layout des Rasters, auf dem das Dashboard basiert (siehe Abschnitt Definition des übergreifenden Dashboard-Layouts). Dies beinhaltet:
      i. Die Widgets, die angezeigt werden sollen.
      ii. Die Reihenfolge und Anordnung dieser Widgets auf der Dashboard-Seite.

2. ***chartWidget / welcomePage*** (nur verfügbar, wenn es Chart-Widgets gibt)
   a. Die Definition bzw. das Layout für alle Chart-Widgets unter *chartWidget* in der Baumstruktur.
   b. Jedes Widget wird durch einen Knoten dargestellt, der den Namen des Widgets trägt, z. B. *chartWidget / welcomePage / ticketsInView*
   c. Ein neues Chart-Widget kann hinzugefügt werden, indem der Name im Wert von *layout* hinzugefügt wird.
   d. Attribute können für alle Chart-Widgets gemeinsam auf der Ebene *chartWidget* oder *chartWidget / welcomePage* definiert werden. Alternativ können sie für jedes einzelne Chart-Widget individuell über die Werte der Attribute des Chart-Widgets konfiguriert werden, z. B. *chartWidget / welcomePage / ticketsInView*.
   e. Die Attribute für Chart-Widgets sind im Abschnitt Attribute für Chart-Widgets detailliert beschrieben.

3. ***tableWidget / welcomePage*** (nur verfügbar, wenn es Tabellen-Widgets gibt)
   a. Die Definition aller Tabellen-Widgets unter *tableWidget* in der Baumstruktur.
   b. Jedes Widget wird durch einen Knoten dargestellt, der den Namen des Widgets trägt, z. B. *tableWidget / welcomePage / ticketsOverview*
   c. Ein neues Tabellen-Widget kann hinzugefügt werden, indem der Name im Wert von *layout* hinzugefügt wird.
   d. Attribute können für alle Tabellen-Widgets gemeinsam auf der Ebene *tableWidget* oder *tableWidget / welcomePage* definiert werden. Alternativ können sie für jedes einzelne Tabellen-Widget individuell über die Werte der Attribute des Tabellen-Widgets konfiguriert werden, z. B. *tableWidget / welcomePage / ticketsOverview*.
   e. Die Attribute für Tabellen-Widgets sind im Abschnitt Attribute für Tabellen-Widgets detailliert beschrieben.

4. ***kpiWidget / welcomePage*** (nur verfügbar, wenn es KPI-Widgets gibt)
   a. Die Definition aller KPI-Widgets in der Unterstruktur *kpiWidget*
   b. Jedes Widget wird durch einen Knoten dargestellt, der den Namen des Widgets trägt, z. B. *kpiWidget / welcomePage / ticketsGlobalOverview*
   c. Ein neues KPI-Widget kann hinzugefügt werden, indem der Name im Wert von *layout* hinzugefügt wird.

   d. Attribute können für alle KPI-Widgets gemeinsam auf der Ebene *kpiWidget* oder *kpiWidget / welcomePage* definiert werden. Alternativ können sie für jedes einzelne KPI-Widget individuell über die Werte der Attribute des KPI-Widgets konfiguriert werden, z. B. *kpiWidget / welcomePage / ticketsGlobalOverview*.
   e. Attribute für KPI-Widgets sind im Abschnitt Attribute für KPI-Widgets detailliert beschrieben.

5. **recentlyVisitedWidget / welcomePage/**(nur verfügbar, wenn *recentlyVisitedWidget* im Attribut *layout* konfiguriert wurde)
   a. Die Konfiguration der Darstellung von *recentlyVisitedWidget* mithilfe von Attributen.
   b. Die Attribute des recentlyVisitedWidget sind im Abschnitt Attribute und Einstellungen für das recentlyVisitedWidget detailliert beschrieben.
6. **recentChangesWidget / welcomePage** (nur verfügbar, wenn *recentChangesWidget* im Attribut *layout* konfiguriert wurde)
   a. Die Konfiguration der Darstellung von *recentChangesWidget* mithilfe von Attributen.
   b. Die Attribute des *recentChangesWidget* sind im Abschnitt Attribute für das recentChangesWidget detailliert beschrieben.

Wie im Abschnitt Seitenanpassung erklärt, wird jedes der vier Elemente durch eine Unterstruktur in der Baumstruktur der Seitenanpassung konfiguriert. Die folgende Abbildung zeigt ein Beispiel für eine Baumstruktur der Seitenanpassung mit den entsprechenden Unterstrukturen für das Web Client Dashboard. Eine detaillierte Beschreibung finden Sie in den folgenden Abschnitten.



Beachten Sie, dass bei der Arbeit mit Dashboards mit Tabs in der Seitenanpassung immer nur die Unterstrukturen des aktuell aktiven (d. h. gerade geöffneten) Tabs angezeigt werden.

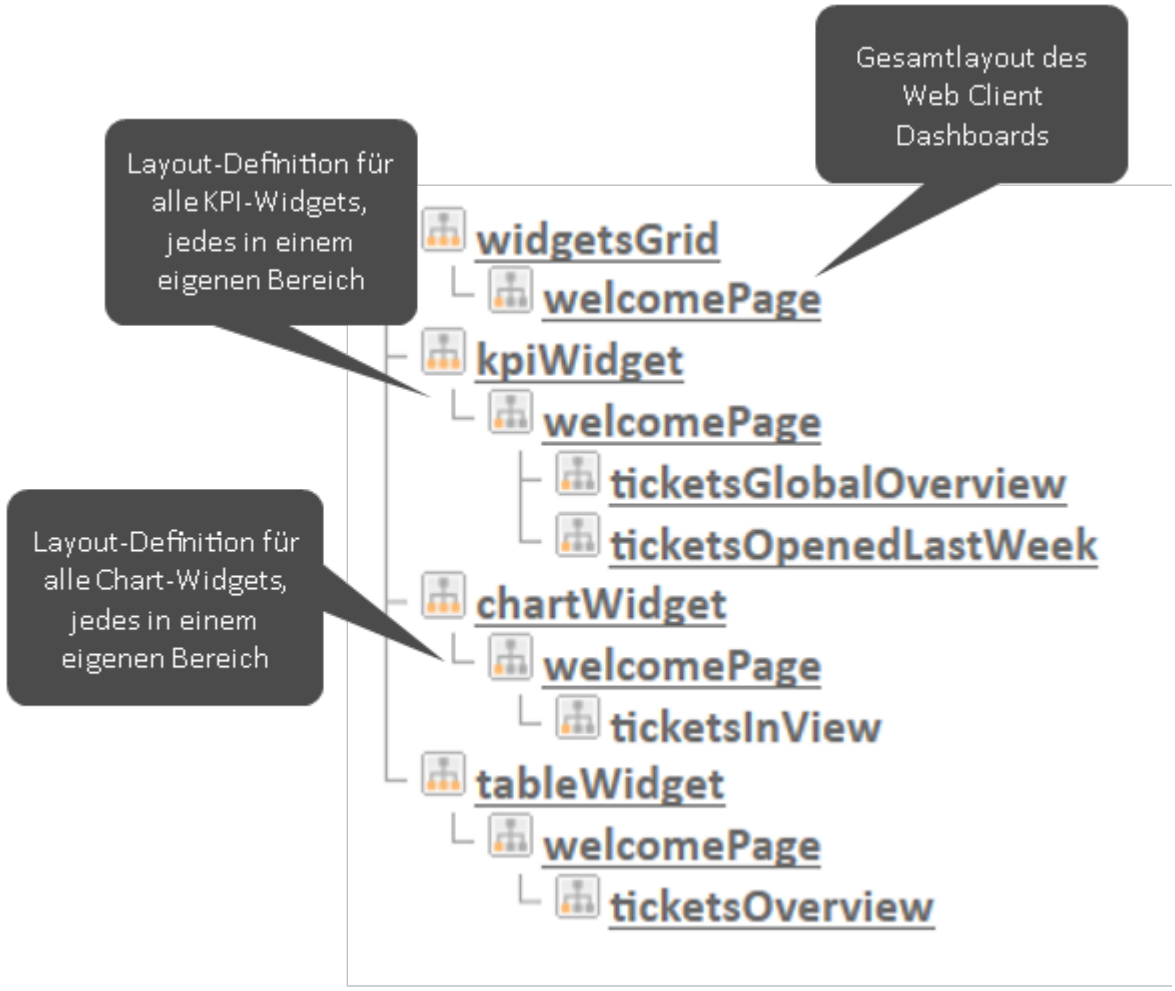


Abbildung 197: *ConSol CM Web Client - Unterstrukturen der Seitenanpassung für das Layout des Web Client Dashboards*

C.8.6.2 Definition des übergreifenden Dashboard-Layouts

Das übergreifende Layout des gesamten Web Client Dashboards wird mit dem Seitenanpassungsattribut *widgetsGrid / welcomePage* definiert.

Variante 1: Anzeigen aller Widgets auf einer Seite

Attribute:

- **layout**
  Definiert das Layout des gesamten Dashboards auf der Hauptseite anhand der folgenden Prinzipien:
  - Ein Widget wird durch seinen Namen und Typ beschrieben. Diese sind durch einen Doppelpunkt getrennt, z. B. "ticketsInView:Chart". Der Name eines Widgets muss eindeutig sein.

- Der Typ eines Widgets ist *Chart* oder *Table*. Der Typ muss nur bei der ersten Nennung des Widget-Namens angegeben werden. Danach kann er weggelassen werden, z. B. [ticketsInView:Chart, ticketsInView, ticketsInView].
- Jede Zeile des Dashboard-Rasters wird als Array aus Elementen dargestellt: [x,y,z]. Ein neues Widget-Objekt wird in der Baumstruktur der Seitenanpassung automatisch hinzugefügt, nachdem es als Wert im Attribut *layout* hinzugefügt wurde, d. h. wenn der Wert **zuvor** [ticketsInView:Chart, ticketsInView] war und in [ticketsInView:Chart, ticketsInView, myTickets:Table] geändert wird, wird ein weiteres Tabellen-Widget mit dem Namen *myTickets* in der Baumstruktur der Seitenanpassung (siehe obige Abbildung) und auf dem Dashboard hinzugefügt. Auf diese Weise können Widgets auch aus dem Dashboard gelöscht werden. Entfernen Sie einfach den Namen und Typ des Widgets aus dem Attribut *layout*. Nachdem Sie die Seite gespeichert und neu geladen haben, sind alle Layout-Änderungen für die weitere Konfiguration in der Baumstruktur verfügbar.
- Das Raster beginnt in der oberen linken Ecke (0,0) und wird Zeile um Zeile aufgebaut. Ein *layout*-Wert mit zwei Paaren von eckigen Klammern [ ] beschreibt zwei Zeilen wie in der Abbildung und im Code unten gezeigt.
- *null* ist das reservierte Schlüsselwort für eine leere Zelle.
- Das Widget kann mehrere nebeneinander liegende Zeilen und Spalten belegen.
- Das ganze Dashboard kann deaktiviert werden, indem der Wert im Attribut *layout* entfernt wird.

Die folgenden Abbildungen zeigen zwei mögliche Layout-Konfigurationen und die entsprechende Darstellung im Web Client.

Beispiel 1

Es werden drei Widgets angezeigt, zwei Charts und eine Tabelle.

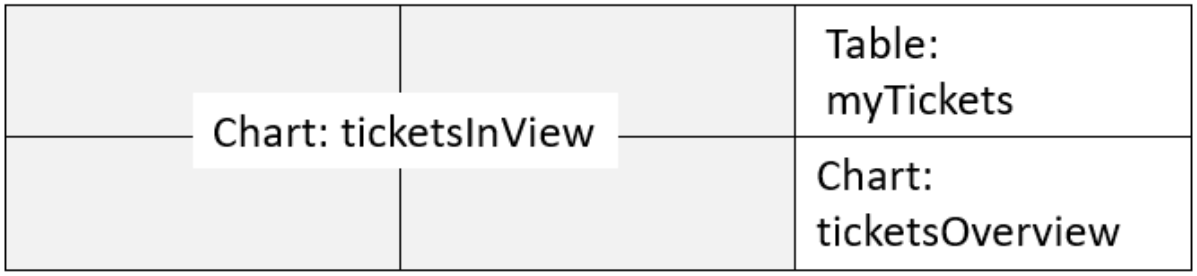


Abbildung 198: *Organisation eines Beispielrasters auf der welcomePage*

Der Wert des entsprechenden Attributs *layout* wäre in diesem Fall:

```
[ticketsInView:Chart, ticketsInView, myTickets:Table], [ticketsInView,
 ticketsInView, ticketsOverview:Chart]
```

Beispiel 2

Es werden vier Widgets angezeigt, zwei KPI-Widgets, ein Chart und eine Tabelle.

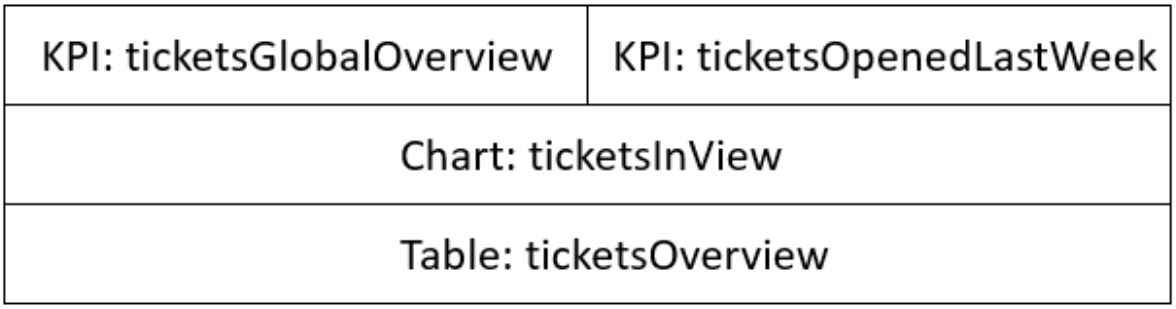


Abbildung 199: *Organisation eines Beispielrasters auf der welcomePage mit KPI-Widgets*

Der Wert des entsprechenden Attributs *layout* wäre in diesem Fall:

```
[ticketsGlobalOverview:KPI,ticketsOpenedLastWeek:KPI],[ticketsInView:Chart],
 [ticketsOverview:Table]
```

Die Dashboard-Seite im Web Client sieht so aus, wie in der folgenden Abbildung dargestellt.



Abbildung 200: *ConSol CM Web Client: Dashboard mit vier Widgets unterschiedlicher Typen*

Variante 2: Widgets in Tabs

Die Widgets können auch in Tabs angezeigt werden. In diesem Fall wird jedes Widget in einem eigenen Tab angezeigt. Für die Konfiguration wird das layout-Attribut verwendet.

Variante 2.1: Verwendung von festen Kopfzeilen

Die folgende Notation für den Wert des Attributs *layout* setzt eine feste Kopfzeile für jeden Tab, d. h. die im Browser eingestellte Sprache wird nicht berücksichtigt.

```
[tabName:'Tickets in current view',widgets:[ticketsInView:Chart]],[tabName:'Job
 table',widgets:[ticketsOverview:Table]]
```

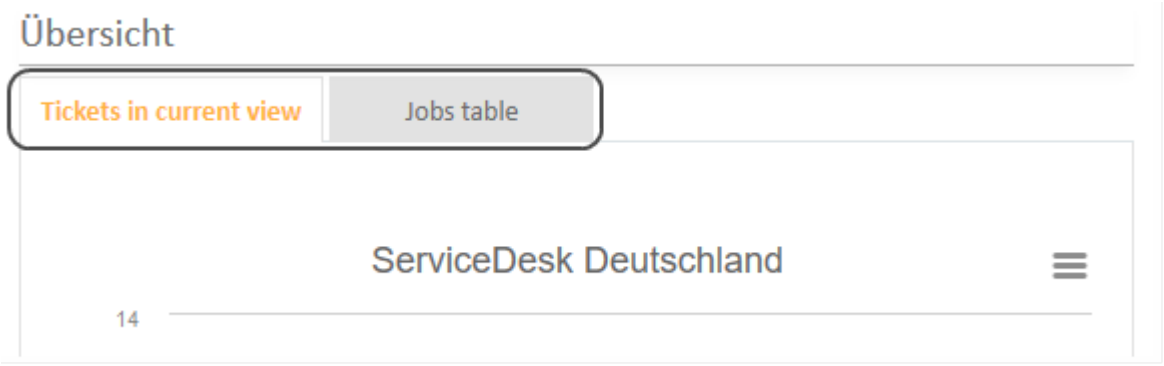


Abbildung 201: *Setzen von festen Kopfzeilen als Titel für die Tabs*

Variante 2.2: Verwenden von lokalisierten Kopfzeilen

Die folgende Notation für den Wert des Attributs *layout* setzt lokalisierte Kopfzeilen, d. h. die Kopfzeile hängt von der im Browser eingestellten Sprache ab. Im folgenden Beispiel sind Kopfzeilen für Englisch und Deutsch angegeben. Wenn im Browser keine Sprache eingestellt ist, wird die Standardsprache von CM verwendet.

Lokalisierte Kopfzeilen, Beispiel 1

```
[tabName:'Tickets in current view',i18n:{en:'Tickets in current view',de:'Tickets
 in der aktuellen Sicht'},widgets:[ticketsInView:Chart]],[tabName:'Job table',i18n:
 {en:'Job table',de:'Tabelle Jobs'},widgets:[ticketsOverview:Table]]
```

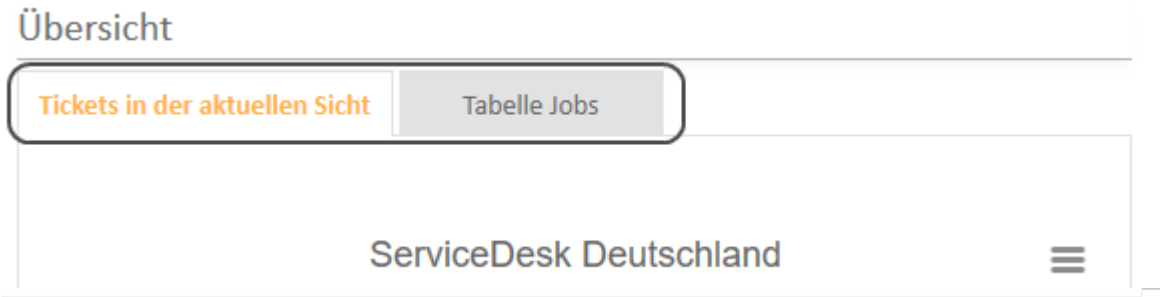


Abbildung 202: *Verwendung von lokalisierten Kopfzeilen (im Beispiel en und de, angezeigt wird de)*

Lokalisierte Kopfzeilen, Beispiel 2

Ein Beispiel für ein etwas komplexeres Dashboard-Layout mit drei Tabs wird im folgenden Code-Beispiel und der folgenden Abbildung gezeigt.

Wert für das layout-Attribut in widgetsGrid/welcomePage:

```
[tabName:'Open tickets global overview',i18n:{en:'Open tickets global
 overview',de:'Übersicht offene Tickets'},widgets:
 [ticketsGlobalOverview:KPI,ticketsOpenedLastWeek:KPI]],[tabName:'Tickets in
 current view',i18n:{en:'Tickets in current view',de:'Tickets in der aktuellen
 Sicht'},widgets:[ticketsInView:Chart]],[tabName:'Job table',i18n:{en:'Job
 table',de:'Tabelle Jobs'},widgets:[ticketsOverview:Table]]
```

Anzeige im Web Client mit Sprache DE:

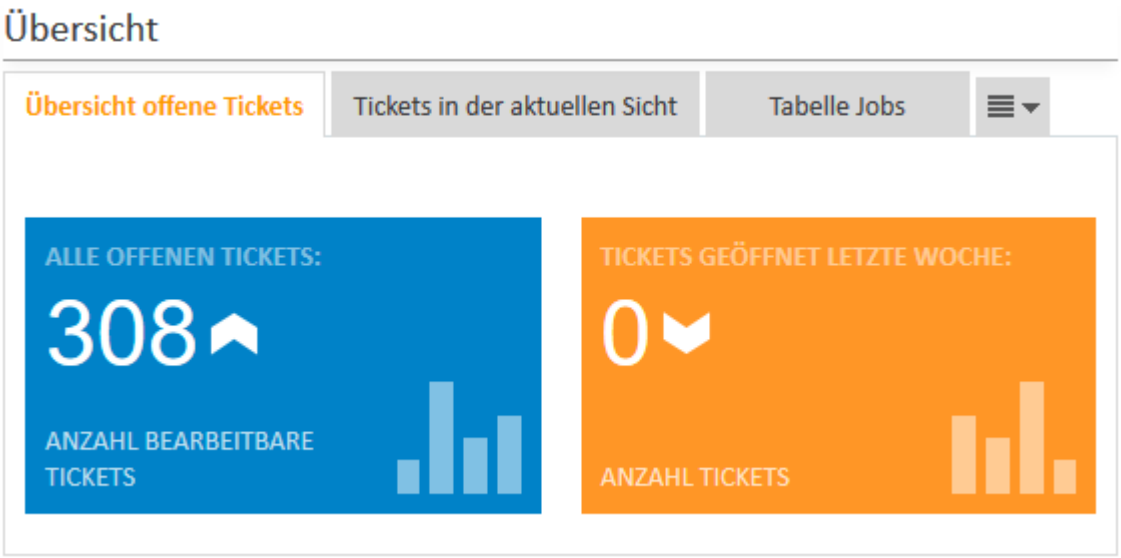


Abbildung 203: *ConSol CM Web Client: Dashboard mit Tabs, zwei KPI-Widgets im ersten Tab*

C.8.6.3 Konfiguration der Widgets

Konfigurationsskript für Widgets

Jedes Chart-Widget und jedes Tabellen-Widget hat ein Konfigurationsskript. Dabei handelt es sich um ein Groovy-Skript, das im Admin-Tool-Bereich *Skripte* hinterlegt ist und durch seinen Namen referenziert wird. Das Skript muss den Typ *Seitenanpassung* haben. Wählen Sie das Widget im Definitionsbereich der Seitenanpassung aus und geben Sie den Namen des Skripts ein:

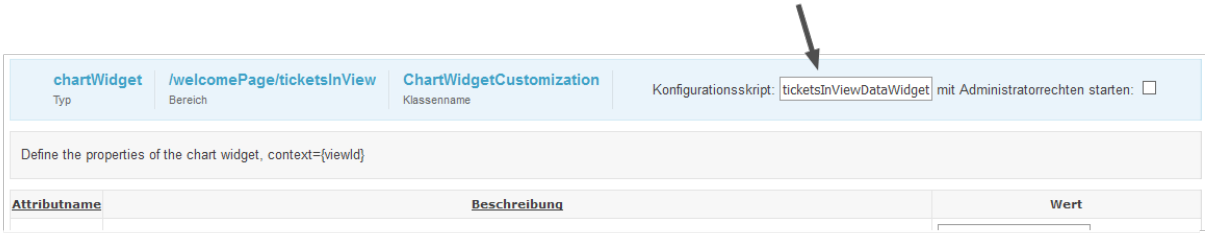


Abbildung 204: *ConSol CM Web Client - Definition des Skripts für ein Chart-Widget*

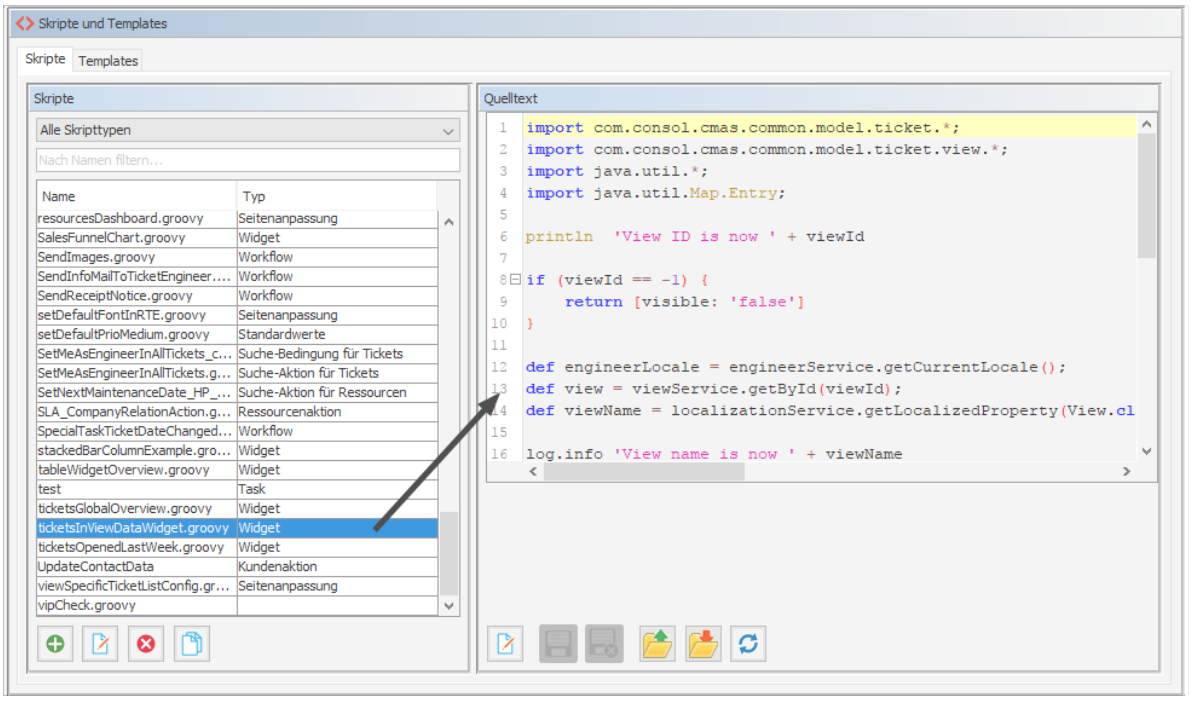


Abbildung 205: *ConSol CM Admin Tool - Admin-Tool-Skript für ein Widget des Web Client Dashboards*

Im Konfigurationsskript des Widgets werden die Statements, mit denen die erforderlichen Daten aus dem CM-System abgerufen werden, und das Layout des Widgets definiert. Die Ausführung dieses Groovy-Skripts ist ein Kernstück der Anpassung. Das Skript muss eine Map mit Variablen zurückgeben, die zu den definierten Widget-Eigenschaften passen.

⚠ Ein **falsches Skript** liefert eine Datenstruktur, die im Web Client Dashboard nicht angezeigt werden kann. Da sich das Dashboard auf der beim Start geöffneten Hauptseite befindet, kann der Web Client in diesem Fall **nicht starten**! Um den Web Client wieder starten zu können, müssen Sie das Skript deaktivieren oder auskommentieren.

⚠ Das Skript überschreibt die in der Seitenanpassung angegebenen Konfigurationsdaten. Die Werte werden **nicht** zusammengeführt! Das Skript überschreibt also alle in der grafischen Seitenanpassung gesetzten Werte der Widget-Attribute. Stellen Sie daher sicher, dass das gewünschte Attribut nicht mittels Skript gesetzt wird, wenn Sie das Attribut in der grafischen Seitenanpassung definieren möchten.

Ein **Skript**, das mit einem Widget verknüpft ist, wird normalerweise mit den Berechtigungen des Benutzers (=Bearbeiters) ausgeführt, z. B. zeigt das Standard-Chart-Widget eine grafische Darstellung der ausgewählten Sicht. Manchmal müssen allerdings Werte verwendet werden, die im Kontext des Bearbeiters nicht verfügbar sind, z. B. eskalierte Tickets (aller Bearbeiter) in einer bestimmten Queue. Um das Skript mit Administratorberechtigungen auszuführen, markieren Sie die Checkbox *mit Administratorrechten starten*. Denken Sie daran, dass die Ergebnisse der Java- oder Groovy-Methoden,

mit denen die Daten abgerufen werden, je nach Kontext unterschiedlich sein können. Die Methode `ticketService.getAll()` gibt zum Beispiel nur Tickets zurück, für die der aktuelle Bearbeiter mindestens Leseberechtigungen hat. Wenn die gleiche Methode mit Administratorberechtigungen ausgeführt wird, gibt sie aber alle im System vorhandenen Tickets zurück.

Die **Chart-Darstellung** im Web Client Dashboard basiert auf der Highcharts-Bibliothek. Für Chart-Widgets muss das Admin-Tool-Skript die zu setzenden Attribute daher als HashMap zurückgeben (siehe Return-Statement im unten dargestellten Code-Beispiel, in dem die Attribute *series*, *visible*, *chart*, *title*, *tooltip* und *localization* verwendet werden). Eine detaillierte Beschreibung aller Attribute und der zugehörigen Links finden Sie im Abschnitt Attribute für Chart-Widgets.

Die **Tabellendarstellung** im Web Client Dashboard basiert auf der Datatables-Bibliothek. Für Tabellen-Widgets muss das Admin-Tool-Skript die zu setzenden Attribute daher als HashMap zurückgeben. Siehe Abschnitt Attribute für Tabellen-Widgets.

Die KPI-Darstellung im Web Client Dashboard basiert auf der Bibliothek `jquery-kpiwidget`. Für KPI-Widgets muss das Admin-Tool-Skript eine Hashmap mit dem Attribut, das die KPI-Darstellung definiert, zurückgeben. Eine detaillierte Erklärung dazu finden Sie im Abschnitt Attribute für KPI-Widgets.

⚠ Sehr komplexe Skripte können die Systemleistung beeinträchtigen!

Das folgende Beispiel zeigt das Skript `ticketsInViewDataWidget.groovy`, das in der Standardinstallation von ConSol CM enthalten ist.

```
import com.consol.cmas.common.model.ticket.*;
import com.consol.cmas.common.model.ticket.view.*;
import java.util.*;
import java.util.Map.Entry;

if (viewId == -1) {
   return [visible: 'false']
}

def engineerLocale = engineerService.getCurrentLocale()
def view = viewService.getById(viewId)
def viewName = localizationService.getLocalizedProperty(View.class, "name", viewId,
 engineerLocale)

ViewCriteria allCriteria = new ViewCriteria(view,
   ViewAssignmentParameter.allTickets(),
   ViewGroupParameter.allTickets(),
   new ViewOrderParameter())


def allTickets = ticketService.getCountForView(allCriteria)


ViewCriteria ownCriteria = new ViewCriteria(view,
   ViewAssignmentParameter.allTickets(engineerService.getCurrent()),
   ViewGroupParameter.onlyOwnTickets(),
   new ViewOrderParameter())
```

```
def ownTickets = ticketService.getCountForView(ownCriteria)

ViewCriteria unassignedCriteria = new ViewCriteria(view,
   ViewAssignmentParameter.allUnassignedTickets(),
   ViewGroupParameter.onlyUnassignedTickets(),
   new ViewOrderParameter())

def unassignedTickets = ticketService.getCountForView(unassignedCriteria)


def data = []

data.add("{name: _('all'), data:[${allTickets}]}" as String)
data.add("{name: _('own'), data:[${ownTickets}]}"as String)
data.add("{name: _('unassigned'), data:[${unassignedTickets}]}"as String)


return [series: "[${data.join(',')}]" as String,
   visible: 'true',
   chart: "{type: 'column'}", title: "{text: '${viewName}'}" as String,
   tooltip:"{headerFormat:''}" ,
   localization:"de: {all:'Alle',own:'Eigene',unassigned:'Unzugewiesene'},"+ "en:
    {all:'All', own:'Own', unassigned: 'Unassigned'}"];
```

Code-Beispiel 4: *ConSol CM-Standardskript für Chart-Widgets ticketsInViewDataWidget.groovy*

Das folgende Chart wird mit dem obigen Skript definiert. Eine detaillierte Beschreibung finden Sie im Abschnitt Beispiel für ein Chart-Widget.

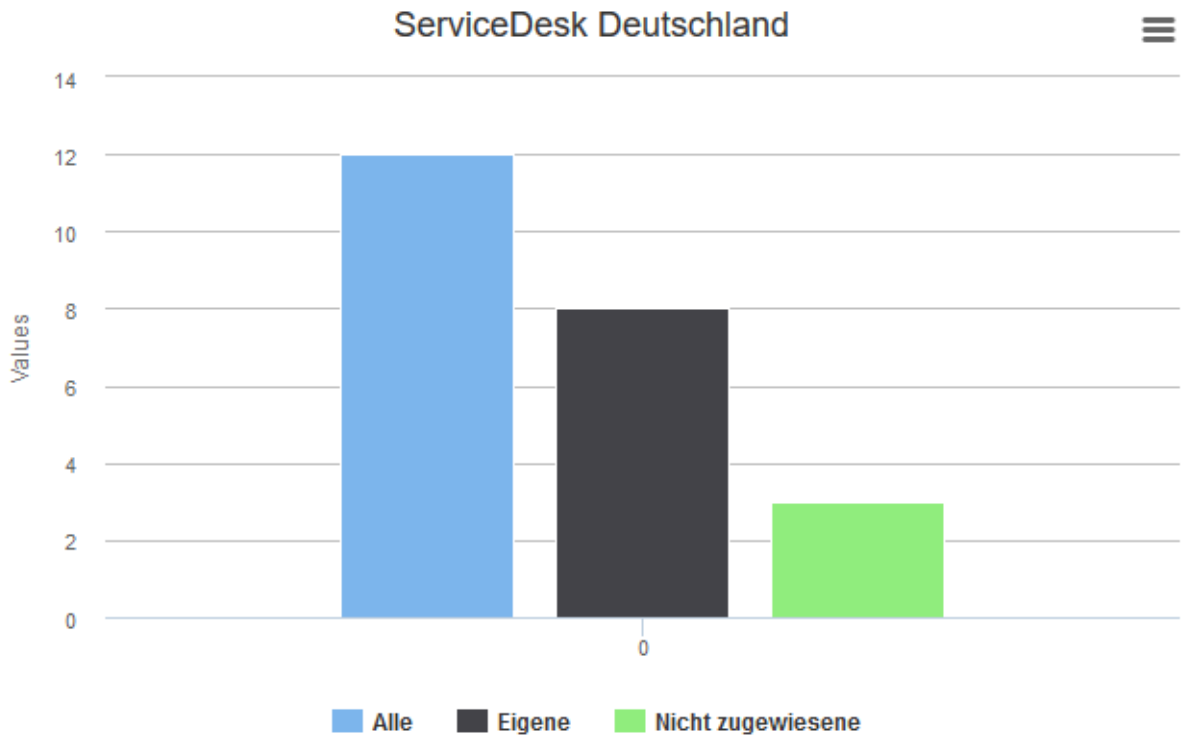


Abbildung 206: *ConSol CM Web Client - Beispiel für Chart-Widget*

C.8.6.4 Konfigurationsattribute für Widgets

Alle Definitionen, die für ein Widget verwendet werden können, können mithilfe der Attribute und Werte in der Seitenanpassung gesetzt werden. Es gibt drei Arten von Attributen:

- Allgemeine Attribute (für jeden Widget-Typ verfügbar)
- Attribute für Chart-Widgets
- Attribute für Tabellen-Widgets

⚠ Denken Sie daran, dass der Wert, der in einem Admin-Tool-Skript für ein Attribut gesetzt wird, immer den Wert überschreibt, der für dieses Attribut in der Seitenanpassung gesetzt wurde!

Beispiel: Für das Chart-Widget *ticketsInView* wurde das Attribut *visibility* auf „true“ gesetzt. Das mit diesem Widget verknüpfte Admin-Tool-Skript (`ticketsInViewDataWidget.groovy`) enthält das Statement

```
return [visible: 'false']
```

In diesem Fall wird das Widget nicht angezeigt!

Allgemeine Attribute

Diese Attribute gelten für alle Widgets und können an zwei Stellen der Baumstruktur der Seitenanpassung gesetzt werden:

- widgetsGrid
- widgetsGrid/welcomePage

Attribute:

- **layout**
  Siehe Allgemeine Attribute.
- **refreshOnViewChange**
  Gibt an, ob das Dashboard aktualisiert werden soll, wenn der Bearbeiter die Sicht ändert, d. h. in der Drop-down-Liste *Sicht* eine andere Sicht für die Ticketliste auswählt. Der Standardwert ist „true“.

Beispiele für die Konfiguration der Sichtbarkeit

Beispiel 1: Das Chart soll nicht angezeigt werden.

```
return [visible: 'false']
```

Code-Beispiel 5: *Sichtbarkeit ausgeschaltet (im Admin-Tool-Skript definiert)*

Beispiel 2: Das Chart soll nur angezeigt werden, wenn die Sicht *service_customer* ausgewählt ist und der Bearbeiter die Funktion *consultant* hat.

```
view = viewService.getById(view_id) // view_id is passed in context
if (!view.getName().equals("service_customer"))
{
   return {"visible": false}
}

def role = roleService.getById('consultant');
def engineer = engineerService.getById(engineer_id);
if(!getRolesForEngineer(engineer).contains(role))
{
   return {"visible": false}
}
```

Code-Beispiel 6: *Sichtbarkeit abhängig von der Bearbeiterfunktion (im Admin-Tool-Skript definiert)*

Lokalisierung von String-Werten von Attributen

- **localization**

  Lokalisierte Werte d. h. „de: {subject:'Thema', yes:'Ja'}, en: {subject:'Subject', yes:'Yes'}“

  Das Attribut *localization* kann die lokalisierten Werte für alle im Widget-Skript verwendeten String-Attribute liefern.

Wenn Sie (wie in der folgenden Abbildung gezeigt) Werte für die Attribute *title* und *footer* brauchen, können Sie drei Attribute setzen:

- Attribut: *title*, Wert _('titlestring')
- Attribut *footer*, Wert _('footerstring')

Attribut *localization* liefert die lokalisierten Werte für beide Attribute. Wert „de:{titlestring:'Offene Tickets: ',footerstring:'Anzahl bearbeitbare Tickets'}, en:{titlestring:'Open tickets: ', footerstring:'Number of accessible tickets'}“

Sie können die Werte für die Attribute im Admin-Tool-Skript setzen, wie im folgenden Skript für die drei Strings *all*, *own* und *unassigned* gezeigt.

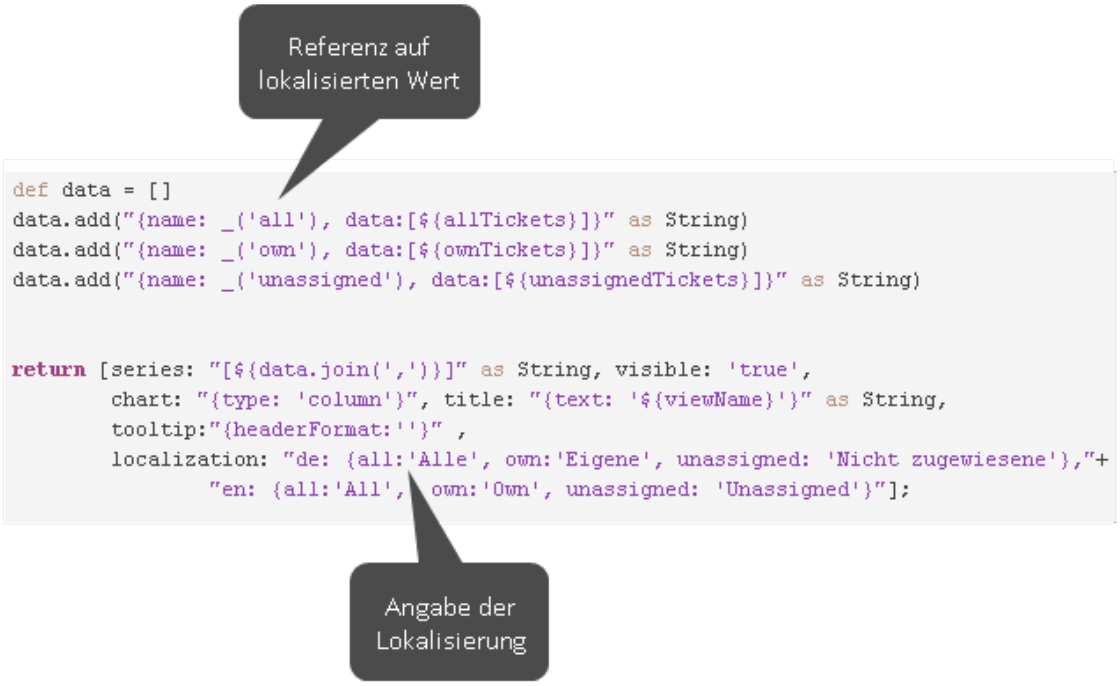


Abbildung 207: *Lokalisieren von Attributen der Seitenanpassung in einem Admin-Tool-Skript*

Es macht keinen Unterschied, ob Sie die Werte für die Attribute in einem Skript angeben oder auf der graphischen Benutzeroberfläche der Seitenanpassung. Sie können Werte auch folgendermaßen lokalisieren (Beispiel eines KPI-Widgets):

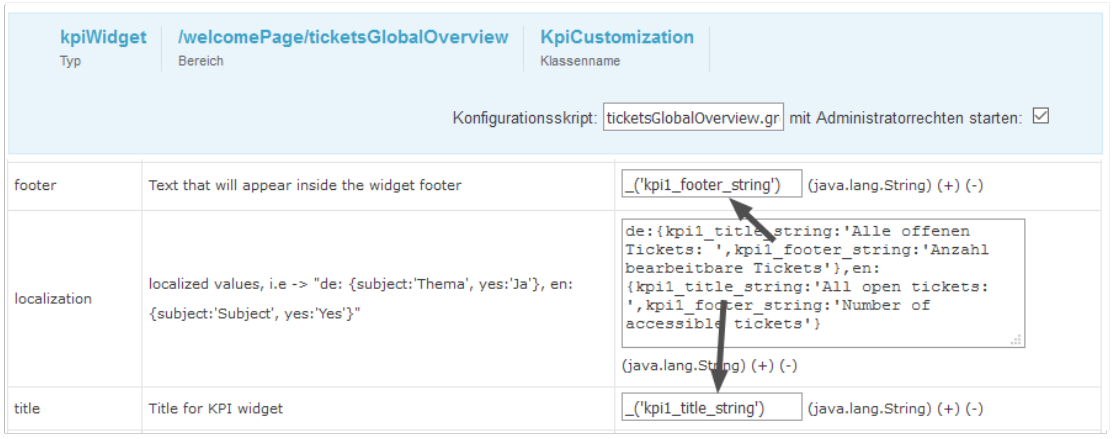


Abbildung 208: *Lokalisieren von String-Attributen der Seitenanpassung über die graphische Benutzeroberfläche*

Attribute für Chart-Widgets

Chart-Widgets verwenden die Highcharts-Bibliothek. Alle Attribute sind JSON-Objekte.

Folgende Attribute können gesetzt werden:

- Allgemeine Attribute wie die Sichtbarkeit.
- Die grundlegenden Konfigurationsoptionen der Highcharts-Bibliothek. Die Werte ...
  - können in den Attributen der Seitenanpassung gesetzt werden.
  - können im mit dem Chart-Widget verknüpften Admin-Tool-Skript gesetzt werden (siehe Abschnitt oben). Die Attribute müssen als HashMap zurückgegeben werden.
  - können leer gelassen werden.

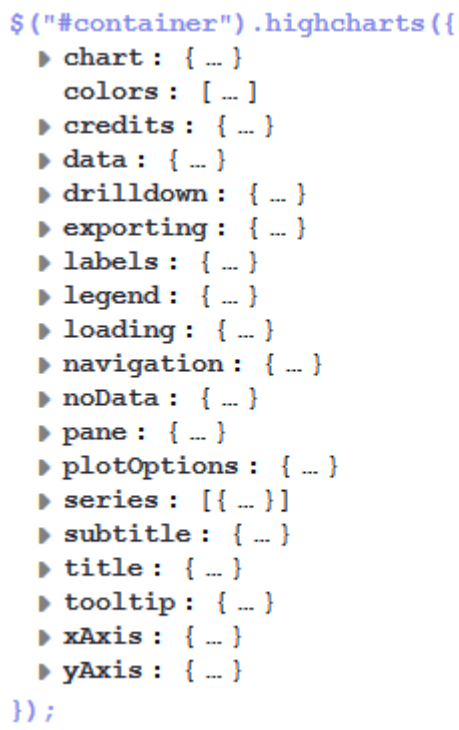


Abbildung 209: *Highcharts-Konfigurationsoptionen*

Allgemeine Attribute:

- **localization**

  Ermöglicht das Festlegen von lokalisierten (d. h. landesspezifischen Werten für die Strings, die als Werte in Attributen verwendet werden. Eine detaillierte Erklärung dazu finden Sie im Abschnitt Lokalisierung von String-Werten von Attributen.

- **visible**

  Definiert, ob das Widget angezeigt wird, „true“ oder „false“.

Highchart-spezifische Attribute:

- **chart**
  Optionen, die den Chart-Bereich und den Plot-Bereich betreffen, sowie allgemeine Chart-Optionen (http://api.highcharts.com/highcharts#chart). Beispiel:

  ```
  chart = "type:'column', pltShadow:false, backgroundColor:'#4dc245', height:
  300";"items: [{html:'sometext', style: { left: '100px'; }}]"
  ```

  Code-Beispiel 7: *Chart-Objekt*

- **colors**
  Ein Array, das die Standardfarben für die Datenreihen des Charts enthält. Wenn alle Farben verwendet wurden, werden benötigte neue Farben wieder vom Anfang genommen. Standardwerte: http://api.highcharts.com/highcharts#colors.
- **credits**
  Standardmäßig setzt Highchart in der unteren rechten Ecke des Charts ein Credits-Label. Das kann mit folgenden Optionen geändert werden: http://api.highcharts.com/highcharts#credits.
- **drilldown**
  Optionen für Drill-Down, dem Konzept zum Einsehen von immer höher aufgelösten Daten durch Klicken auf die Chart-Elemente wie Balken oder Tortenstücke (http://api.highcharts.com/highcharts#drilldown).
- **exporting**
  Optionen für das Exportmodul (http://api.highcharts.com/highcharts#exporting).
- **global**
  Globale Optionen, die nicht für jedes Chart gelten (http://api.highcharts.com/highcharts#global). Es kann nur für einen Typ gesetzt werden, d. h. für *chartWidget* oder *tableWidget*, nicht aber für Bereiche oder einzelne Widgets!
- **labels**
  HTML-Label, die an einer beliebigen Stelle des Chart-Bereichs positioniert werden können (http://api.highcharts.com/highcharts#labels).

```
labels = "items: [{html:'sometext', style: { left: '100px'; }}]"
```

Code-Beispiel 8: *Label-Objekt*

- **lang**
  Sprachobjekt. Das Sprachobjekt ist global und kann nicht für jedes Chart einzeln gesetzt werden (http://api.highcharts.com/highcharts#lang). Es kann nur für einen Typ gesetzt werden, d. h. für *chartWidget* oder *tableWidget*, nicht aber für Bereiche oder einzelne Widgets!
- **legend**
  Die Legende ist ein Feld, das für jedes Element der Datenreihe oder des Punktes im Chart ein Symbol und einen Namen enthält (http://api.highcharts.com/highcharts#legend).
- **loading**
  Die Ladeoptionen steuern das Aussehen der Ladeseite, die den Plot-Bereich während Chart-Operationen bedeckt (http://api.highcharts.com/highcharts#loading).
- **localization**
  Lokalisierte Werte d. h. „de: {subject:'Thema', yes:'Ja'}, en: {subject:'Subject', yes:'Yes'}“.
- **navigation**
  Eine Sammlung von Optionen für die Buttons und Menüs im Exportmodul (http://api.highcharts.com/highcharts#navigation).

- **noData**
  Optionen für die Anzeige einer Meldung wie "Keine Daten verfügbar" (http://api.highcharts.com/highcharts#noData).
- **pane**
  Gilt für Netzdiagramme und Tachos. Diese Konfigurationsobjekte enthalten allgemeine Optionen für die kombinierten X- und Y-Achsen (http://api.highcharts.com/highcharts#pane).
- **plotOptions**
  Die Plot-Optionen sind ein Wrapper-Objekt für die Konfigurationsobjekte für jeden Datenreihentyp (http://api.highcharts.com/highcharts#plotOptions).
- **series**
  Die Datenreihe, die an das Chart angehängt wird (http://api.highcharts.com/highcharts#series).
- **subtitle**
  Der Untertitel des Charts (http://api.highcharts.com/highcharts#subtitle).
- **title**
  Der Titel des Charts (http://api.highcharts.com/highcharts#title).
- **tooltip**
  Optionen für den Tooltip, der erscheint, wenn der Benutzer mit der Maus über eine Datenreihe oder einen Punkt fährt (http://api.highcharts.com/highcharts#tooltip).
- **visible**
  Gibt an, ob das Widget angezeigt werden soll.
- **xAxis**
  Die X-Achse oder die Kategorie-Achse (http://api.highcharts.com/highcharts#xAxis) .
- **yAxis**
  Die Y-Achse oder die Werte-Achse (http://api.highcharts.com/highcharts#yAxis) .

Beispiel für ein Chart-Widget

Das folgende Beispiel zeigt das Widget *TicketsInView* und erklärt die Logik des verknüpften Admin-Tool-Skripts `ticketsInViewDataWidget.groovy`. Das komplette Skript finden Sie im Codeblock weiter oben (Konfigurationsskript für Widgets). Hier werden die einzelnen Codezeilen mit den GUI-Elementen, die sie konfigurieren, in Verbindung gebracht.



Abbildung 210: *Beispiel für ein Chart-Widget mit Skriptcode*

Trichter-Charts

Mithilfe von Charts des Typs *funnel* können trichterartige Darstellungen, so wie sie häufig in Vertriebstrichtern verwendet werden, implementiert werden.

Ein Trichter-Chart wird genauso implementiert wie ein anderer Chart, es muss lediglich das Unterattribut *type* des Chart-Attributs auf „funnel“ gesetzt werden. Das folgende Beispiel zeigt eine Implementierung eines Vertriebstrichter-Charts, bei der im Skript feste Werte verwendet werden. In ein einer echten Implementierung müssen diese Zahlen durch dynamisch aus dem aktuellen Inhalt der Datenbank erzeugte Werte ersetzt werden.

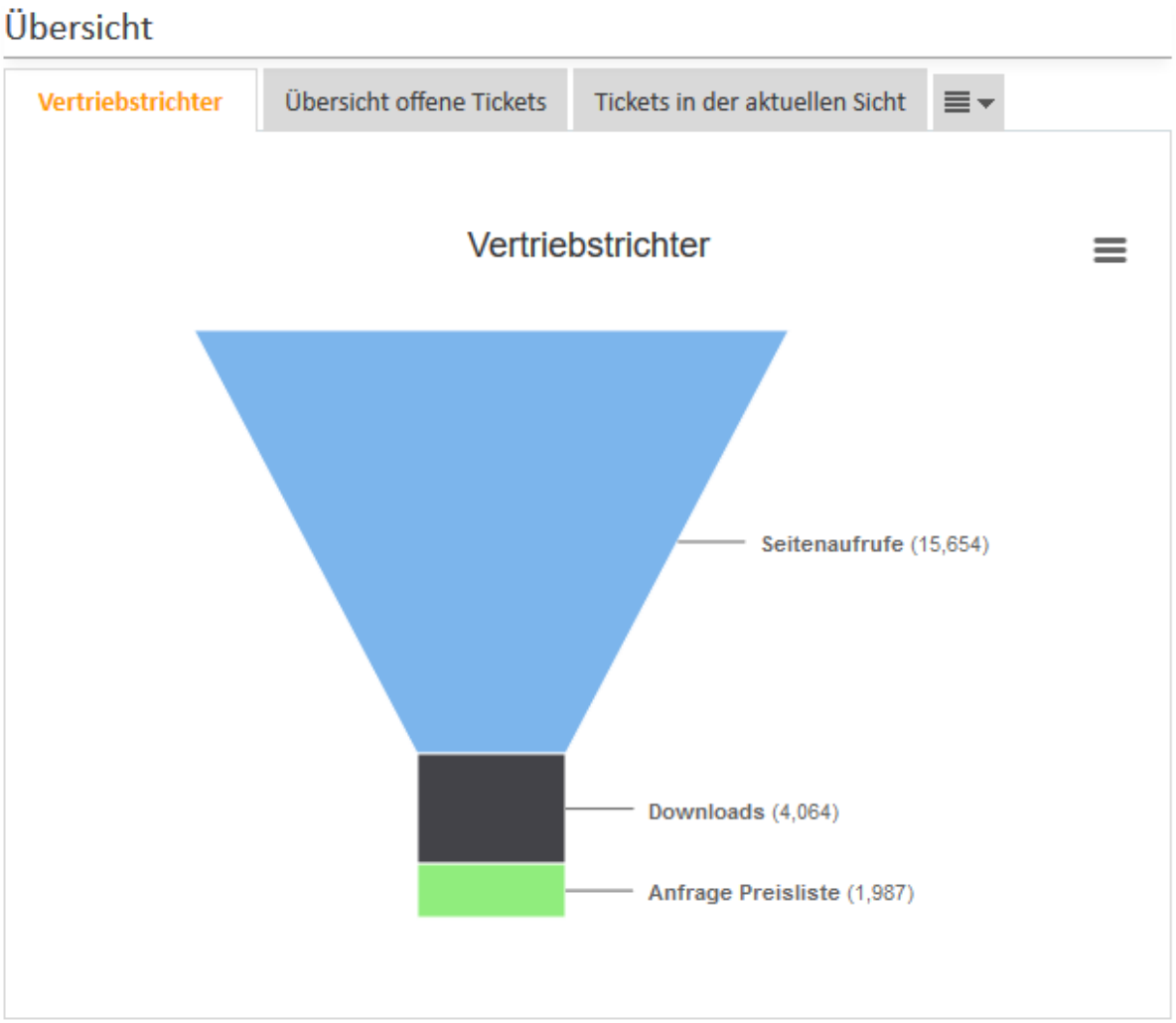


Abbildung 211: *ConSol CM Web Client: Vertriebstrichter-Widget*

Es wird folgendes Skript verwendet:

```
return [title: "{text: _('title')}",
   chart: "{type: 'funnel', \
      marginLeft: '50', \
      marginRight: '150'}",
   plotOptions: "{series:{ height: '90%', \
      width: '85%', \
      neckWidth: '20%', \
      neckHeight: '20%', \
      dataLabels: {enabled:'true', \
      format: '<b>{point.name}</b> ({point.y:,.0f})', \
      softConnector: 'true'}}}",
   visible: "true",
   series: "[{name: _('users'), \
      data: [[_('visits') , 15654], \
      [_('downloads'), 4064], \
      [_('requests') , 1987]]}]",
   localization: "de: {title: 'Vertriebstrichter', \
      users: 'Individuelle Benutzer', \
      visits: 'Seitenaufrufe', \
      downloads: 'Downloads', \
      requests: 'Anfrage Preisliste'}, \
   en: {title: 'Salesfunnel', \
      users: 'Unique users', \
      visits: 'Page visits', \
      downloads: 'Downloads', \
      requests: 'Requests for price list'}"
]
```

Code-Beispiel 9: *Groovy-Skript für die Implementierung eines Vertriebstrichter-Charts auf dem Web Client Dashboard (mit festen Zahlen als Beispiel)*

Attribute für Tabellen-Widgets

Tabellen-Widgets verwenden die Datatables-Bibliothek.

Die Attribute können entweder in der Seitenanpassung oder in den verknüpften Admin-Tool-Skripten gesetzt werden. Denken Sie daran, dass der Wert, der in einem Admin-Tool-Skript für ein Attribut gesetzt wird, immer den Wert überschreibt, der für dieses Attribut in der Seitenanpassung gesetzt wurde!

Folgende Attribute können gesetzt werden:

- Allgemeine Attribute
- Datatables-spezifische Attribute

Allgemeine Attribute:

- **localization**
  Lokalisierte Werte d. h. „de: {subject:'Thema', yes:'Ja'}, en: {subject:'Subject', yes:'Yes'}“
- **visible**
  Definiert, ob das Widget angezeigt wird, „true“ oder „false“.

Datatables-spezifische Attribute:

- **columns**
  Optionen, die auf Spaltenobjekte angewendet werden können (http://datatables.net/reference/option/#Columns)
- **data**
  Daten (http://datatables.net/reference/option/#Data)
- **localization**
  Lokalisierte Werte d. h. „de: {subject:'Thema', yes:'Ja'}, en: {subject:'Subject', yes:'Yes'}“. Eine detaillierte Erklärung dazu finden Sie im Abschnitt Lokalisierung von String-Werten von Attributen.
- **options**
  Optionen (http://datatables.net/reference/option/)
- **visible**
  Gibt an, ob das Widget angezeigt werden soll

Beispiel für ein Tabellen-Widget

Im Folgenden werden die Grundprinzipien der Implementierung eines Tabellen-Widgets basierend auf der Datatables-Bibliothek gezeigt.

```
// provide some dummy data for display
def rawdata = [
   [firstname:'Homer' , lastname:'Simpson' , title:'Nuclear disaster' , level:'3' ,
    hired:'25.03.1989'],
   [firstname:'Zaphod' , lastname:'Beeblebrox' , title:'President of the Galaxy',
    level:'0' , hired:'12.09.1979'],
   [firstname:'Sheldon' , lastname:'Cooper' , title:'Mad scientist' , level:'321',
    hired:'01.04.2006'],
   [firstname:'Robin' , lastname:'Scherbatsky', title:'Anchorwoman' , level:'25' ,
    hired:'10.09.2004'],
   [firstname:'Elmer' , lastname:'Fudd' , title:'Duck hunter' , level:'1' ,
    hired:'15.12.1962'],
   [firstname:'Eric' , lastname:'Cartman' , title:'Pupil' , level:'10' ,
    hired:'23.02.1995'],
   [firstname:'Mickey' , lastname:'Mouse' , title:'Private investigator' ,
    level:'111', hired:'04.11.1932'],
   [firstname:'Wilma' , lastname:'Flintstone' , title:'Housewife' , level:'64' ,
    hired:'07.01.1964'],
   [firstname:'Charlie' , lastname:'Harper' , title:'Composer' , level:'12' ,
    hired:'16.07.2001'],
   [firstname:'Daenerys', lastname:'Targaryen' , title:'Mother of dragons' ,
    level:'238', hired:'08.05.2010'],
   [firstname:'Lara' , lastname:'Croft' , title:'Tomb Raider' , level:'239',
    hired:'10.12.1991'],
   [firstname:'Henry' , lastname:'Jones' , title:'Archeologist' , level:'109',
    hired:'08.06.1942']
]
```

```
// prepare the data for display
def tabledata = []

rawdata.each { element ->
   tabledata.add("""
      {'firstname': '${element['firstname']}',
      'lastname' : '${element['lastname']}' ,
      'jobtitle' : '${element['title']}' ,
      'expertise': '${element['level']}' ,
      'hiredate' : '${element['hired']}' }
   """)
}

// return the table information including the data
return [
   "columns": """[
      {title: 'First name' , data: 'firstname'},
      {title: 'Last name' , data: 'lastname' },
      {title: 'Job title' , data: 'jobtitle' },
      {title: 'Expertise level', data: 'expertise'},
      {title: 'Hire date' , data: 'hiredate' }
      ]""",
   "options": """{
      'order': []
      }""",
   "data": "[${tabledata.join(",")}]" as String
]
```

Code-Beispiel 10: *Admin-Tool-Skript für Tabellen-Widget*

Im Web Client wird die Tabelle folgendermaßen angezeigt (alle anderen Widgets wurden auf "unsichtbar" gesetzt).

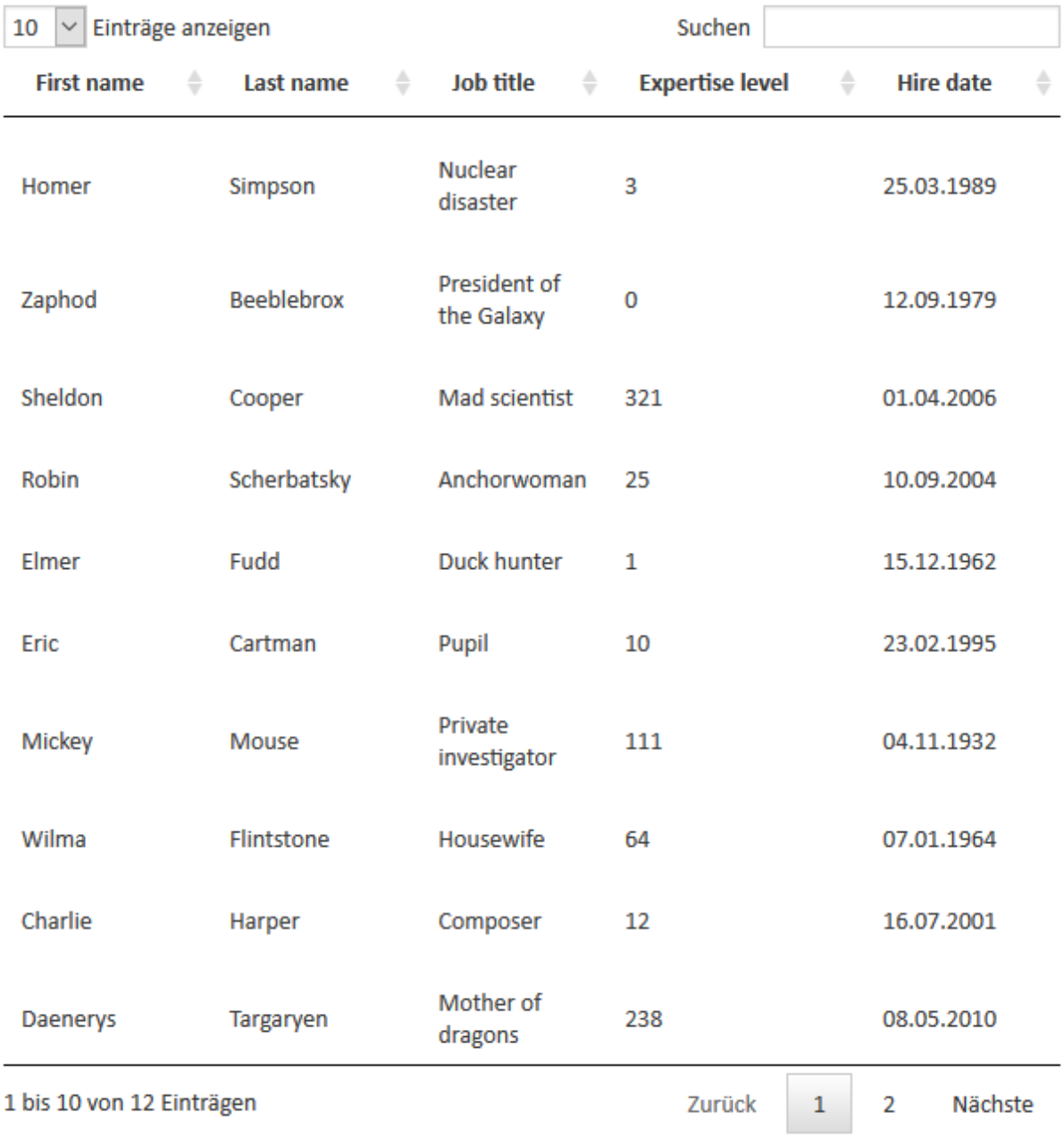


Abbildung 212: *ConSol CM Web Client - Tabellen-Widget auf dem Web Client Dashboard*

Attribute für KPI-Widgets

Für KPI-Widgets können folgende Attribute verwendet werden:

- **color**
  Hintergrundfarbe für die rechteckige Widget-Fläche, zum Beispiel #A0B0C0.

- **footer**
  String, der im unteren Bereich des Widgets angezeigt wird (im linken Beispiel "Anzahl bearbeitbare Tickets"), kann lokalisiert werden, siehe Lokalisierung von String-Werten von Attributen.
- **height**
  Höhe des Widgets-Rechtecks in Pixeln (Integer-Wert).
- **localization**
  Lokalisierungsdefinition für die angezeigten String-Werte, siehe Lokalisierung von String-Werten von Attributen.
- **maxValueForSize**
  Numerischer Widget-Wert ab dem die Schriftgröße verkleinert wird, damit der Wert in die Zeile des Rechtecks passt, Standardwert: 100000.
- **symbol**
  Buchstabe, Symbol oder String als Präfix für den numerischen Wert.
- **title**
  String, der als oberste Zeile im Widget angezeigt wird (im linken Beispiel "Alle offenen Tickets"), kann lokalisiert werden, siehe Lokalisierung von String-Werten von Attributen.
- **trend**
  Bezeichnung für das Trend-Symbol, das nach dem numerischen Wert angezeigt wird (siehe linkes Beispiel), gültige Bezeichnungen sind 'up', 'down' und 'flat'.
- **value**
  Anzuzeigender numerischer Wert, normalerweise sollte dies kein fester Wert sondern ein dynamisch aufgelöster Wert sein. Dies kann nur mithilfe eines Skripts erreicht werden, Details dazu weiter unten. Wenn hier ohne Skript ein Wert eingetragen wird, wird immer nur dieser Wert angezeigt.
- **visible**
  Boolean-Wert, der festlegt, ob das Widget angezeigt werden soll; identisch wie bei den anderen Widget-Typen.

Der folgende Code zeigt ein Beispiel für ein KPI-Widget-Skript.

```
import com.consol.cmas.common.model.ticket.*
import java.util.*
import com.consol.cmas.common.model.DateRange
TicketCriteria crt = new TicketCriteria()
def to_date = new Date()
def from_date = to_date - 7
def range = new DateRange(from_date,to_date)
crt.setCreationDateRange(range)
// crt.setStatus(TicketCriteria.Status.OPEN)
ticketcount = ticketService.getByCriteria(crt).size()
switch (ticketcount) {
   case 0..25:
      trendline = 'down'
      break
   case 26..50:
      trendline = 'flat'
      break
   default:
      trendline = 'up'
}
return[value: ticketcount as String, trend: trendline, visible: 'true']
```

Code-Beispiel 11: *Admin-Tool-Skript für ein KPI-Widget, mit dem die in der letzten Woche geöffneten Tickets berechnet werden*

Das Widget wird im Web Client wie in der folgenden Abbildung angezeigt.

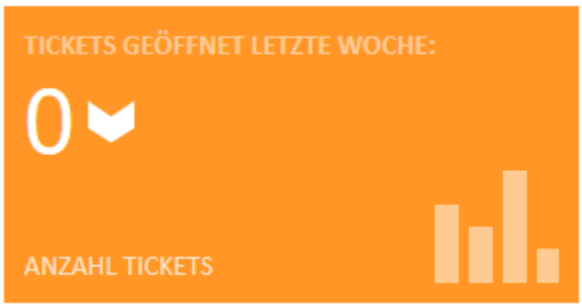


Abbildung 213: *ConSol CM Web Client: KPI-Widget auf der Dashboard-Seite*

Attribute und Einstellungen für das recentlyVisitedWidget

Attribute der Seitenanpassung

- **excludedCustomerGroups**: String. Eine durch Kommas getrennte Liste der durch ihre technischen Namen identifizierten Kundengruppen. Die genannten Kundengruppen und ihre Mitglieder werden aus der Liste im Widget ausgeschlossen.

- **excludedItemTypes**: String. Eine durch Kommas getrennte Liste der ausgeschlossenen Objekttypen. Die Typen werden über folgende Schlüsselwörter identifiziert:
  - TICKET: Alle Tickets werden aus der Liste im Widget ausgeschlossen.
  - UNIT: Alle Kunden (Firmen und Kontakte) werden aus der Liste im Widget ausgeschlossen.
  - RESOURCE: Alle Ressourcen werden aus der Liste im Widget ausgeschlossen.
- **excludedQueues**: String. Eine durch Kommas getrennte Liste der durch ihre technischen Namen identifizierten Queues. Die genannten Queues und ihre Tickets werden aus der Liste im Widget ausgeschlossen.
- **excludedResources**: String. Eine durch Kommas getrennte Liste der durch ihre technischen Namen identifizierten Ressourcentypen. Die genannten Ressourcentypen und ihre Ressourcen werden aus der Liste im Widget ausgeschlossen.
- **localization**: Momentan nicht in Verwendung.
- **maxItems**: Integer. Die maximale Anzahl der in der Hauptansicht des Widgets angezeigten Elemente.
- **title**: String. Die in der Kopfzeile angezeigte Überschrift. Die Lokalisierung dieser Überschrift ist noch nicht verfügbar.
- **visible**: Boolean. Widget-Sichtbarkeit, muss auf „true“ gesetzt sein, damit das Widget auf der Startseite angezeigt wird. Wenn der Wert auf „false“ gesetzt ist, sind das Widget und der Inhalt ausgeblendet.

Weitere Einstellung mit CM-System-Properties

Das CM-Systemverhalten bezüglich der Datensammlung für das `recentlyVisitedWidget` kann mithilfe vom CM-System-Properties konfiguriert werden.

Datensammlung ein-/ausschalten

Je nach Verwendung des `recentlyVisitedWidget` kann die Datensammlung, die die Grundlage für die Anzeige des Widgets darstellt, ein- oder ausgeschaltet werden. Wenn das Widget nicht angezeigt wird, macht es keinen Sinn, dass der Thread zur Datensammlung im Hintergrund läuft und Systemressourcen verbraucht. In diesem Fall können Sie den Datensammlungsprozess mit folgender CM-System-Property ausschalten:

- cmas-core-server, recent.items.persistence.enabled

Vergessen Sie nicht, die Datensammlung (wieder) einzuschalten, wenn das Widget angezeigt werden soll.

Verhalten der Datensammlung konfigurieren

Wenn die Datensammlung eingeschaltet ist, d. h. wenn `cmas-core-server`, `recent.items.persistence.enabled` auf „true“ gesetzt ist, kann das Systemverhalten bezüglich der Datensammlung mit den folgenden CM-System-Properties konfiguriert werden:

- recent.items.max.per.engineer
- recent.items.cleanup.interval.minutes

- recent.items.cleanup.cluster.node.id

Attribute für das recentChangesWidget

Attribute der Seitenanpassung

Die Liste der „excludedHistoryTypes“ zeigt ebenfalls, welche Aktionen als `recent change` gelten.

- **excludedHistoryTypes**: String. Die Protokolleintragstypen, die aus der Liste der Änderungen ausgeschlossen werden sollen. Der Wert wird als durch Kommas getrennte Liste der folgenden Bezeichnungen für die Eintragstypen angegeben:
  - „OPEN“: Das Ticket wurde erstellt.
  - „REOPEN“: Das Ticket wurde neu geöffnet.
  - „UPDATE“: Das Ticket wurde aktualisiert.
  - „SUBJECT_CHANGE“: Der Betreff des Tickets wurde aktualisiert.
  - „ENGINEER_CHANGE“: Der verknüpfte Bearbeiter wurde geändert.
  - „TICKET_USER_ADDED“: Ein zusätzlicher Bearbeiter wurde hinzugefügt.
  - „TICKET_USER_REMOVED“: Ein zusätzlicher Bearbeiter wurde entfernt.
  - „TICKET_USER_MOVED“: Die Funktion eines zusätzlichen Bearbeiters wurde in eine andere Funktion geändert.
  - „QUEUE_CHANGE“: Die Queue des Tickets wurde geändert.
  - „ACTIVITY_CHANGE“: Der Status des Tickets wurde geändert.
  - „TRANSFERED_TO_ACTIVITY“: Das Ticket wurde in eine andere Aktivität überführt.
  - „CONTENT_ADD“: Es wurde neuer Inhalt zum Ticket hinzugefügt.
  - „ATTACHMENT_DELETE“: Ein Ticket-Attachment wurde gelöscht.
  - „ATTACHMENT_UNDO_DELETION“: Das Löschen eines Ticket-Attachments wurde rückgängig gemacht.
  - „EMAIL_ATTACHMENT_DELETE“: Ein Attachment aus einer E-Mail wurde gelöscht.
  - „EMAIL_ATTACHMENT_UNDO_DELETION“: Das Löschen eines E-Mail-Attachments wurde rückgängig gemacht.
  - „BOOKING“: Eine Zeitbuchung wurde hinzugefügt.
  - „CONTACT_ADD“: Ein Kunde wurde zum Ticket hinzugefügt.
  - „CONTACT_REMOVE“: Ein Kunde wurde vom Ticket entfernt.
  - „CONTACTROLE_SWAP“: Die Kundenrolle im Ticket wurde geändert.
  - „CONTACTROLE_UPDATE“: Die Kundenrolle im Ticket wurde aktualisiert.
  - „TRANSFERRED_TO_CONTACT“: Das Ticket wurde an einen neuen Kontakt übertragen.
  - „TICKETRELATION_CREATE“: Eine Ticketrelation wurde erstellt.
  - „TICKETRELATION_REMOVE“: Eine Ticketrelation wurde entfernt.

- „TICKETRELATION_UPDATE“: Eine Ticketrelation wurde aktualisiert.
- „TICKETRELATION_MOVE“: Der Typ einer Ticketrelation wurde geändert.
- „RESOURCE_RELATION_CREATED“: Eine Ressourcenrelation wurde erstellt.
- „RESOURCE_RELATION_UPDATED“: Eine Ressourcenrelation wurde aktualisiert.
- „RESOURCE_RELATION_REMOVED“: Eine Ressourcenrelation wurde entfernt.
- „RESOURCE_ON_THE_FLY_RELATION_CREATED“: Eine Ressource auf Anforderung wurde erstellt.
- „RESOURCE_ON_THE_FLY_RELATION_UPDATED“: Eine Ressource auf Anforderung wurde aktualisiert.
- „RESOURCE_ON_THE_FLY_RELATION_REMOVED“: Eine Ressource auf Anforderung wurde entfernt.

- **excludedQueues**: String. Eine durch Kommas getrennte Liste der durch ihre technischen Namen identifizierten Queues. Die genannten Queues und ihre Tickets werden aus der Liste im Widget ausgeschlossen.
- **localization**: Momentan nicht in Verwendung.
- **maxItems**: Integer. Die maximale Anzahl der in der Hauptansicht des Widgets angezeigten Elemente.
- **title**: String. Die in der Kopfzeile angezeigte Überschrift. Die Lokalisierung dieser Überschrift ist noch nicht verfügbar.
- **visible**: Boolean. Widget-Sichtbarkeit, muss auf „true“ gesetzt sein, damit das Widget auf der Startseite angezeigt wird. Wenn der Wert auf „false“ gesetzt ist, sind das Widget und der Inhalt ausgeblendet.

Beispiel für ein zusammengesetztes Dashboard

Die folgenden beiden Abbildungen zeigen ein Beispiel für ein zusammengesetztes Dashboard:

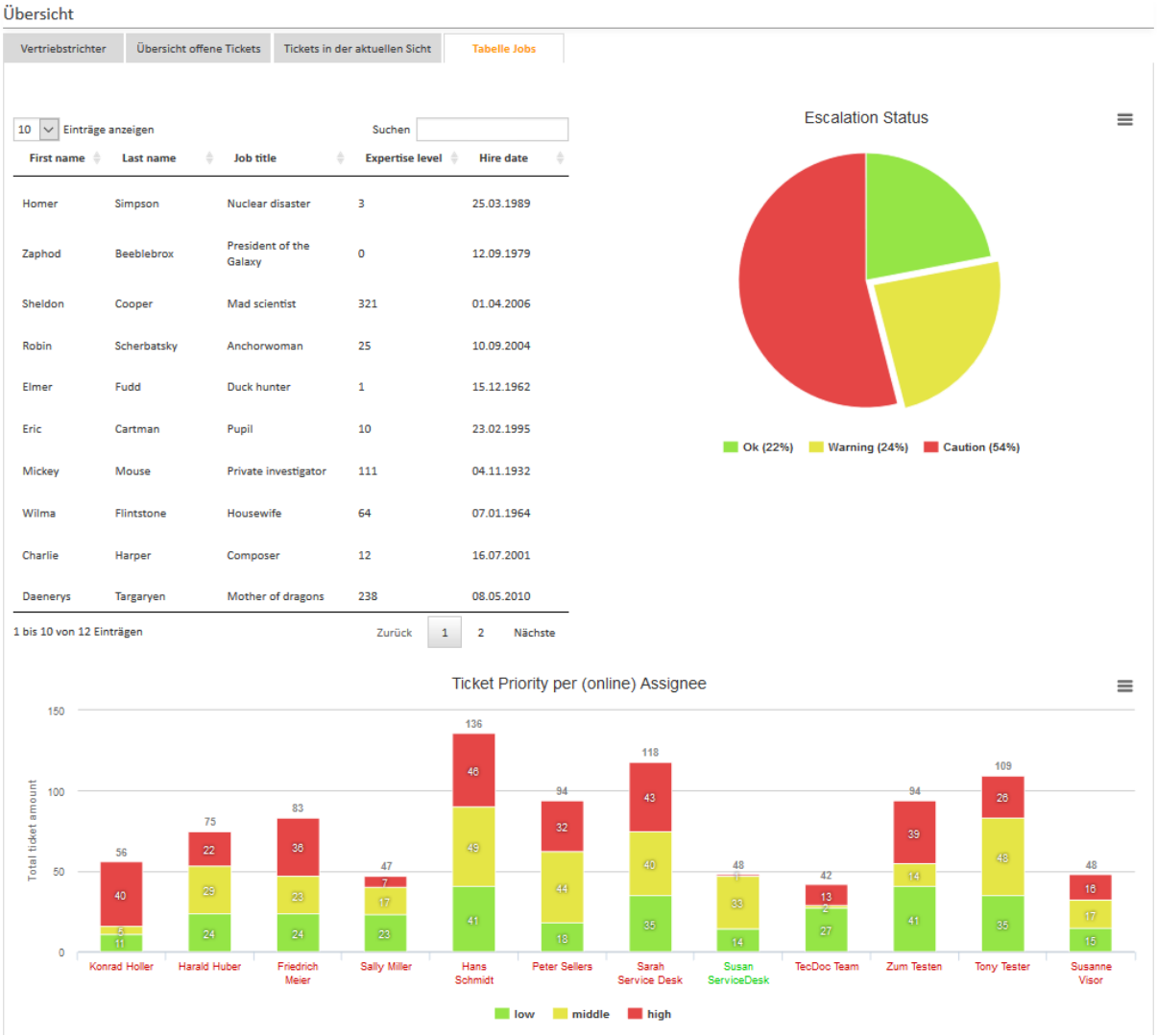


Abbildung 214: *ConSol CM Web Client - Beispiel für ein zusammengesetztes Dashboard, oberer Teil*

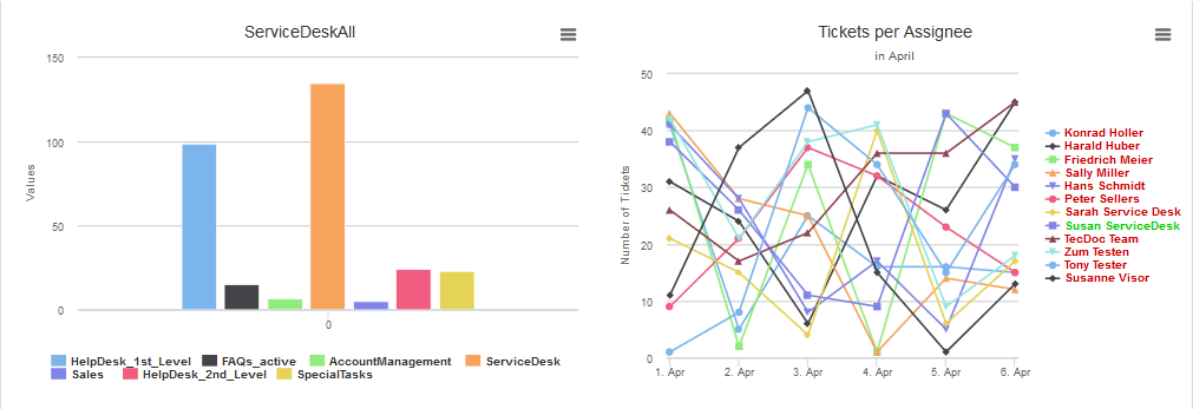


Abbildung 215: *ConSol CM Web Client - Beispiel für ein zusammengesetztes Dashboard, unterer Teil*

Das Seitenanpassungsattribut *widgetsGrid / welcomePage / layout* hat folgenden Wert:

```
[tabName:'Sales funnel',i18n:{en:'Sales funnel',de:'Vertriebstrichter'},widgets:
 [SalesFunnel:chart]],[tabName:'Open tickets global overview',i18n:{en:'Open
 tickets global overview',de:'Übersicht offene Tickets'},widgets:
 [ticketsGlobalOverview:KPI,ticketsOpenedLastWeek:KPI]],[tabName:'Tickets in
 current view',i18n:{en:'Tickets in current view',de:'Tickets in der aktuellen
 Sicht'},widgets:[ticketsInView:Chart]],[tabName:'Job table',i18n:{en:'Job
 table',de:'Tabelle Jobs'},widgets:[[ticketsOverview:Table,pieChartExample:Chart],
 [stackedBarColumnExample:Chart,stackedBarColumnExample:Chart],
 [chartWidgetExample:Chart,lineChartExample:Chart]]]
```

Code-Beispiel 12: *Layout-Attribut für zusammengesetztes Dashboard*

> Beachten Sie, dass immer die Unterstruktur der Seitenanpassung für den aktiven Tab angezeigt wird.

Im folgenden Beispiel bedeutet dies, dass nur die Unterstruktur des Tabs *Tabelle Jobs* angezeigt wird. Die Unterstrukturen für *Vertriebstrichter*, *Übersicht offene Tickets* und *Tickets in der aktuellen Sicht* werden nicht angezeigt.

Die Seitenanpassungsattribute für den Tab *Tabelle Jobs* sehen folgendermaßen aus:

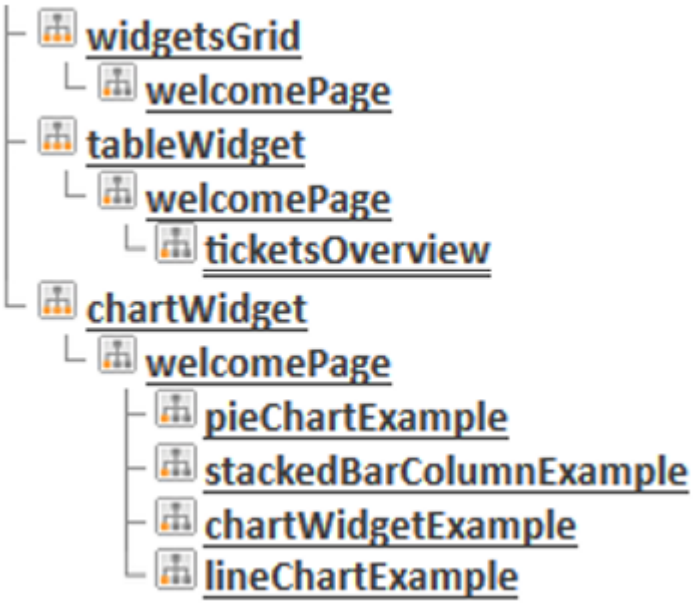


Abbildung 216: *ConSol CM Web Client - Beispiel für die Unterstruktur der Attribute der Seitenanpassung*

Allen Widgets ist jeweils ein Admin-Tool-Skript zugeordnet. Die folgende Abbildung zeigt die Konfiguration des Tabellen-Widgets *ticketsOverview*.

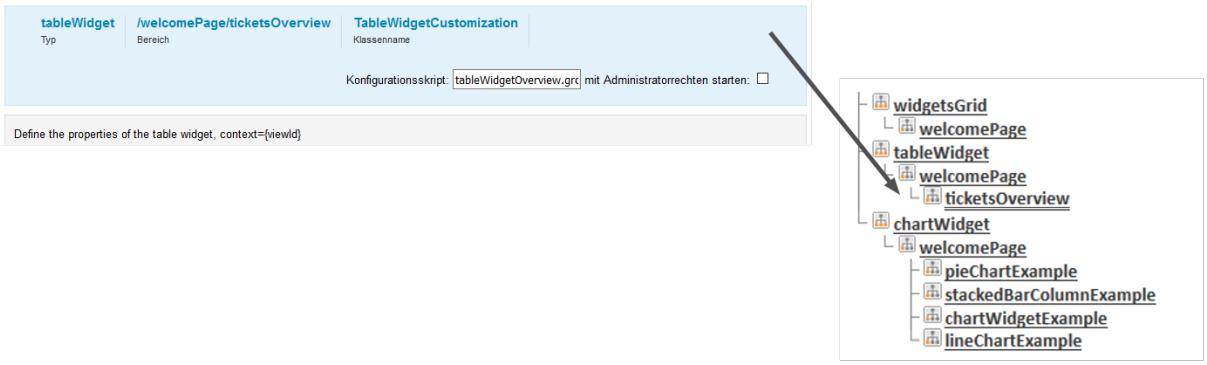


Abbildung 217: *ConSol CM Web Client - Beispiel für die Konfiguration eines Tabellen-Widgets*

Das Admin-Tool-Skript mit dem Namen `tableWidgetOverview.groovy` für das Tabellen-Widget (mit dem Namen *tickesOverview*):

```
// provide some demo data for display
def rawdata = [
[firstname:'Homer' , lastname:'Simpson' , title:'Nuclear disaster' , level:'3' ,
 hired:'25.03.1989'],
```

```
[firstname:'Zaphod' , lastname:'Beeblebrox' , title:'President of the Galaxy',
 level:'0' , hired:'12.09.1979'],
[firstname:'Sheldon' , lastname:'Cooper' , title:'Mad scientist' , level:'321',
 hired:'01.04.2006'],
[firstname:'Robin' , lastname:'Scherbatsky', title:'Anchorwoman' , level:'25' ,
 hired:'10.09.2004'],
[firstname:'Elmer' , lastname:'Fudd' , title:'Duck hunter' , level:'1' ,
 hired:'15.12.1962'],
[firstname:'Eric' , lastname:'Cartman' , title:'Pupil' , level:'10' ,
 hired:'23.02.1995'],
[firstname:'Mickey' , lastname:'Mouse' , title:'Private investigator' ,
 level:'111', hired:'04.11.1932'],
[firstname:'Wilma' , lastname:'Flintstone' , title:'Housewife' , level:'64' ,
 hired:'07.01.1964'],
[firstname:'Charlie' , lastname:'Harper' , title:'Composer' , level:'12' ,
 hired:'16.07.2001'],
[firstname:'Daenerys', lastname:'Targaryen' , title:'Mother of dragons' ,
 level:'238', hired:'08.05.2010'],
[firstname:'Lara' , lastname:'Croft' , title:'Tomb Raider' , level:'239',
 hired:'10.12.1991'],
[firstname:'Henry' , lastname:'Jones' , title:'Archeologist' , level:'109',
 hired:'08.06.1942']
]

// prepare the data for display
def tabledata = []

rawdata.each { element ->
   tabledata.add("""
      {'firstname': '${element['firstname']}',
      'lastname' : '${element['lastname']}' ,
      'jobtitle' : '${element['title']}' ,
      'expertise': '${element['level']}' ,
      'hiredate' : '${element['hired']}' }
      """)
   }


// return the table information including the data
return [
"columns": """[
   {title: 'First name' , data: 'firstname'},
   {title: 'Last name' , data: 'lastname' },
   {title: 'Job title' , data: 'jobtitle' },
   {title: 'Expertise level', data: 'expertise'},
   {title: 'Hire date' , data: 'hiredate' }
   ]""",
"options": """{
   'order': []
   }""",
"data": "[${tabledata.join(",")}]" as String
]
```

Code-Beispiel 13: *Admin-Tool-Skript, das mit dem Tabellen-Widget verknüpft ist*

Mit dem ersten Chart-Widget, einem Tortendiagramm mit dem Namen *pieChartExample*, ist ebenfalls ein Admin-Tool-Skript verknüpft (genannt `pieChartExample.groovy`):

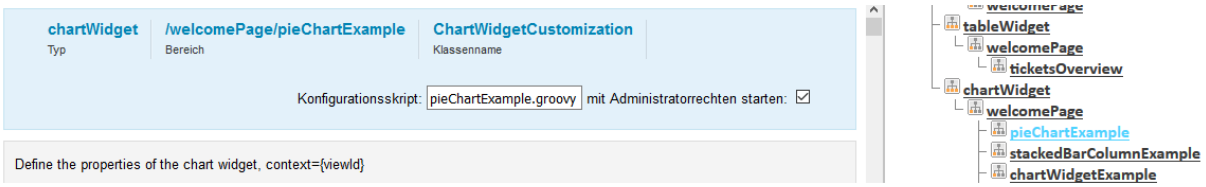


Abbildung 218: *ConSol CM Web Client - Beispiel für die Konfiguration eines Chart-Widgets*

Das Admin-Tool-Skript für das erste Chart-Widget (oben rechts, genannt *pieChartExample*):

```
def kpiGroups = ['Ok', 'Warning', 'Caution']
def escalationColor = [ ok: "#95E546", warning: "#E5E546", danger: "#E54646"]
def data = []
def name = [:]
def size = [:]
def percent = [:]
percent[kpiGroups[0]] = Math.round(Math.random() * 50)
percent[kpiGroups[1]] = Math.round(Math.random() * 50)
percent[kpiGroups[2]] = 100 - percent[kpiGroups[1]] - percent[kpiGroups[0]]
def prio = [:]
prio[kpiGroups[0]] = "39-A"
prio[kpiGroups[1]] = "40-B"
prio[kpiGroups[2]] = "41-C%2B"
for(def kpiGroup : kpiGroups){
   name[kpiGroup] = kpiGroup
   size[kpiGroup] = percent[kpiGroup] * 3
   sliced = kpiGroup.equals(kpiGroups[1]) ? ", sliced: true, selected: true" : ""
   def link = '/cm-client/search?c_queue=1'
   link += "&sales_standard:priority=" + prio[kpiGroup]
   link += "&c_status=OPEN"
   def dataSet = "{name: '${name[kpiGroup]} (${percent[kpiGroup]}%)', y: ${size
    [kpiGroup]} ${sliced}, link: '${link}'}" as String
   data.add(dataSet)
   //println dataSet
}
def chartData = [
   series: "[{name:'Escalation-Type', data: [${data.join(',')}]}]" as String,
   visible: 'true',
   chart: """{
   type: 'pie',
   }""",
   title: "{text: 'Escalation Status'}" as String,
   tooltip:"{headerFormat:'{point.name}', pointFormat: '{point.name}:
    {point.y}<br/>'}",
   plotOptions: """{
      pie: {
         allowPointSelect: true,
         cursor: 'pointer',
         dataLabels: {
```

```
        enabled: false
        },
      showInLegend: true
      },
      series: {
         cursor: 'pointer',
         point: {
            events: {
               click:
               function () {
                  window.open(this.link,'_blank')
               }
            }
         }
      }
   }""" as String,
   colors: "['${escalationColor['ok']}', '${escalationColor['warning']}',
    '${escalationColor['danger']}']" as String
]
//println chartData
return chartData
```

Code-Beispiel 14: *Admin-Tool-Skript, das mit dem Tortendigramm-Widget verknüpft ist*

Das gleiche Prinzip, das für die ersten beiden Widgets (die erste Zeile des Dashboard-Tabs) gezeigt ist, ist auch für die anderen Widgets umgesetzt. Die zweite Zeile des Dashboards ist vollständig mit dem Chart *stackedBarColumnExample* gefüllt, und in der dritten Zeile befinden sich zwei Chart-Widgets: *chartWidgetExample* und *lineChartExample*.

Für jedes Chart-Widget können spezifische Attribute definiert werden, z. B. *title* oder *subtitle*. Diese Attribute können im Skript definiert und in die zurückgegebene Datenstruktur eingefügt werden. Alternativ können sie auch über die grafische Oberfläche definiert werden. Die Attribute für Chart-Widgets sind im Abschnitt Attribute für Chart-Widgets erklärt und die Attribute für Tabellen-Widgets im Abschnitt Attribute für Tabellen-Widgets.

C.8.6.5 Druckfunktion im Web Client Dashboard

Ab Version 6.9.4.2 bietet ConSol CM eine Druckfunktion für Chart-Widgets im Web Client Dashboard. Mit dem Drucken-Button wird das Druckfenster des Betriebssystems geöffnet.

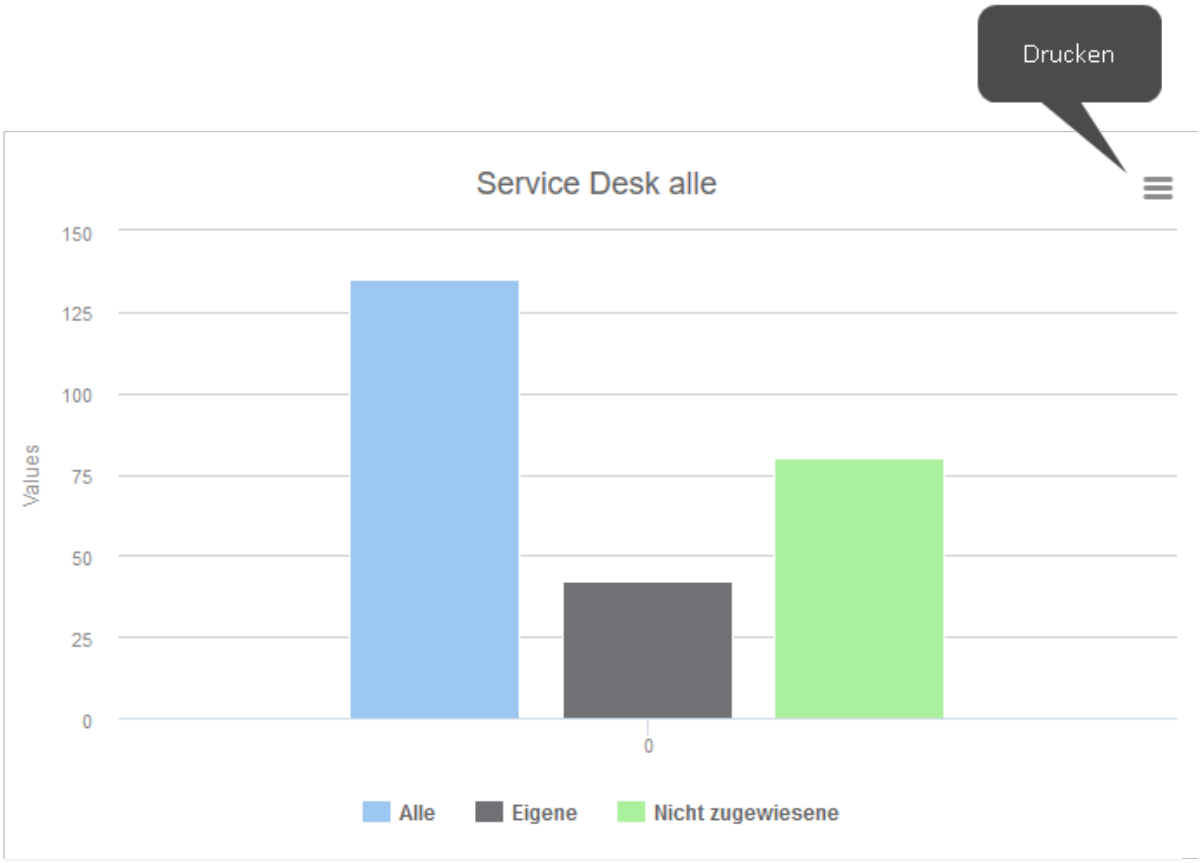


Abbildung 219: *ConSol CM Web Client - Drucken-Button im Web Client Dashboard*

Sie können die Druckfunktion deaktivieren, d. h. den Button ausblenden, indem Sie das Seitenanpassungsattribut *exporting* auf *enabled* = „false“ setzen.

C.8.6.6 3D-Ausgabe für Grafiken (Chart-Widgets)

Für Chart-Widgets können Sie die 3D-Ausgabe verwenden. Es wird folgende Bibliothek verwendet: http://api.highcharts.com/highcharts#chart.options3d.

Weitere Informationen für die Implementierung von 3D und das Lösungskonzept finden Sie in http://www.highcharts.com/docs/chart-concepts/3d-charts.

> ⚠ Die Verwendung einer solchen Darstellung sorgt für eine hohe Last des Browsers. Stellen Sie sicher, dass die Client-Umgebungen leistungsfähig genug für die Anzeige von 3D-Charts sind, bevor Sie sie implementieren!

Ein Beispielskript, mit dem das Standard-Chart in 3D angezeigt wird, finden Sie unten. Die 3D-Optionen werden im Rückgabewert des Skripts gesetzt.

```
import com.consol.cmas.common.model.ticket.*;
import com.consol.cmas.common.model.ticket.view.*;
import java.util.*;
import java.util.Map.Entry;


if (viewId == -1) {
   return [visible: 'false']
}
def engineerLocale = engineerService.getCurrentLocale()
def view = viewService.getById(viewId)
def viewName = localizationService.getLocalizedProperty(View.class, "name", viewId,
 engineerLocale)


ViewCriteria allCriteria = new ViewCriteria(view,
   ViewAssignmentParameter.allTickets(),
   ViewGroupParameter.allTickets(),
   new ViewOrderParameter())
def allTickets = ticketService.getIdsByView(allCriteria)


ViewCriteria ownCriteria = new ViewCriteria(view,
   ViewAssignmentParameter.allTickets(engineerService.getCurrent()),
   ViewGroupParameter.onlyOwnTickets(),
   new ViewOrderParameter())
def ownTickets = ticketService.getIdsByView(ownCriteria)


ViewCriteria unassignedCriteria = new ViewCriteria(view,
   ViewAssignmentParameter.allUnassignedTickets(),
   ViewGroupParameter.onlyUnassignedTickets(),
   new ViewOrderParameter())
def unassignedTickets = ticketService.getIdsByView(unassignedCriteria)


def data = []
data.add("{name: _('all'), data:[${allTickets.size()}]}" as String)
data.add("{name: _('own'), data:[${ownTickets.size()}]}" as String)
data.add("{name: _('unassigned'), data:[${unassignedTickets.size()}]}" as String)


return [series: "[${data.join(',')}]" as String,
   visible: 'true',
   chart: "{type: 'column',
      options3d: {enabled: 'true', alpha: '15', beta: '15', depth: '50',
         viewDistance: '25'}}",
      plotOptions: "{column: {depth: '25'}}",
      title: "{text: '${viewName}'}" as String,
      tooltip:"{headerFormat:''}" ,
      localization: "de: {all:'Alle', own:'Eigene', unassigned:'Nicht
       zugewiesene'},"
         + "en: {all:'All', own:'Own', unassigned: 'Unassigned'}"];
```

Code-Beispiel 15: *Anzeigen des Standard-Charts in der 3D-Ansicht*

C.8.6.7 Drill-down-Funktion für Grafiken (Chart-Widgets)

ConSol CM bietet eine Drill-down-Funktion für Charts im Dashboard. Das Seitenanpassungsattribut *drilldown* kann für die allgemeinen Einstellungen verwendet werden. Es wird folgende Bibliothek verwendet: http://api.highcharts.com/highcharts#drilldown.

Für einen sinnvollen Einsatz dieser Funktion sollte allerdings das Datenskript erweitert werden. Daten im Rückgabeskript müssen durch die im Drill-down angezeigten Daten ergänzt werden. Diese zusätzlichen Daten müssen mit den Daten, die sie ergänzen, referenziert werden. Eine Beschreibung der Konzepte finden Sie in der Highcharts-Dokumentation: http://www.highcharts.com/docs/chart-concepts/drilldown.

Mit dieser Funktion kann eine zweite Datenebene im gleichen Chart angezeigt werden, wenn Sie auf eine Teilmenge klicken. Im linken Screenshot sind die Balken anklickbar und die Label der Balkennamen sind Links. Nachdem Sie entweder auf einen Balken oder einen Link geklickt haben, wird eine Detailansicht dieser Teilmenge angezeigt, wie im unteren Screenshot zu sehen ist.

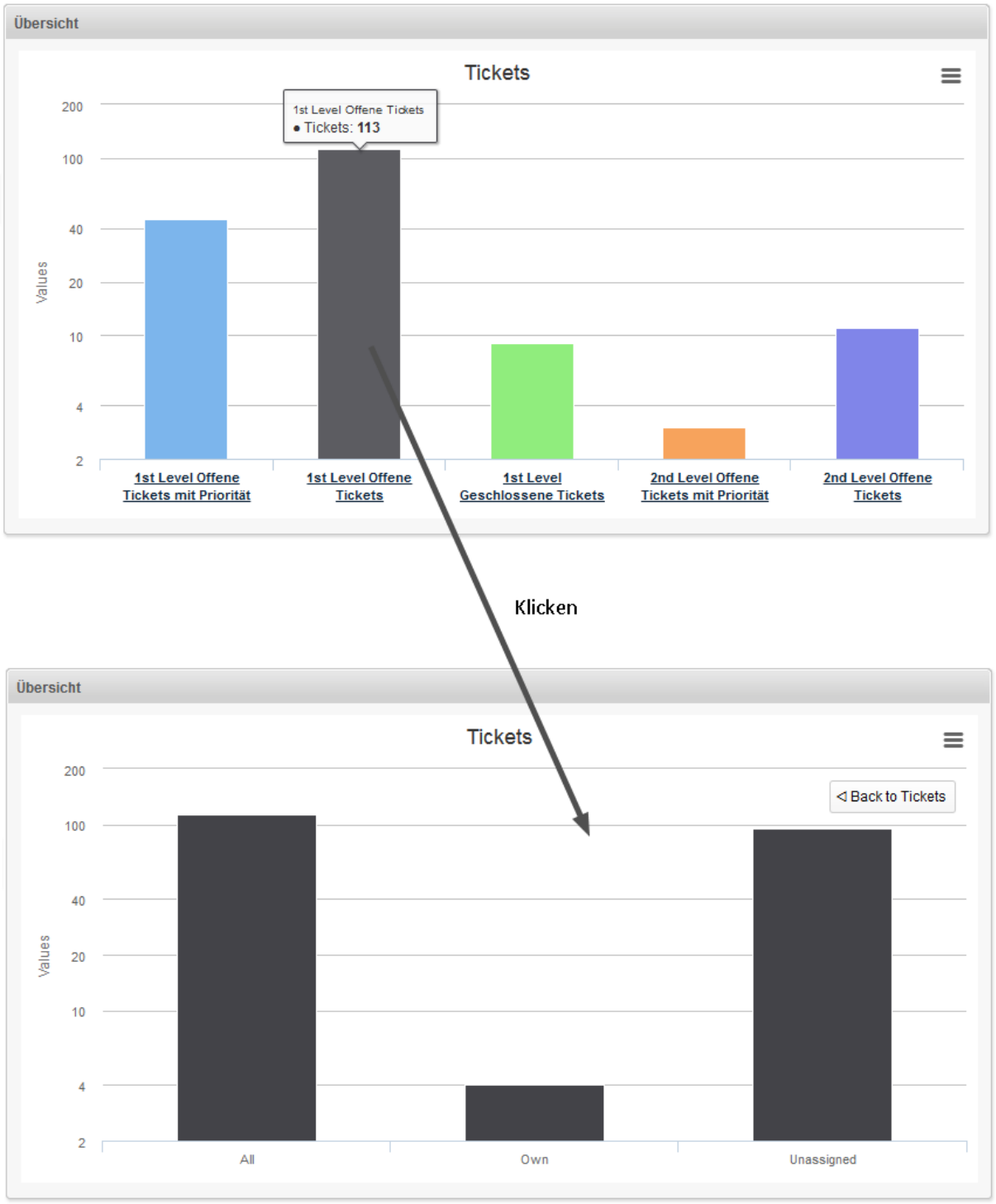


Abbildung 220: *ConSol CM Web Client - Drill-down-Funktion in Chart-Widgets*

Die Farbe der Balken in der Detailansicht ist die Farbe der in der Hauptansicht ausgewählten Teilmenge. Die Detailansicht enthält zudem einen Zurück-Button in der oberen rechten Ecke des Charts. Das folgende Datenskript ruft die Daten für dieses Drill-down dynamisch ab.

```
import com.consol.cmas.common.model.ticket.*;
import com.consol.cmas.common.model.ticket.view.*;
import java.util.*;
import java.util.Map.Entry;

if (viewId == -1) {
   return [visible: 'false']
}

def engineerLocale = engineerService.getCurrentLocale()

def views = []

def seriesdata = []
def drilldownseries = []
def allTicketsCounter = 0
views = viewService.getByEngineer(engineerService.getCurrent())

for (view in views){
   def viewName = localizationService.getLocalizedProperty(View.class,
      "name",
      view.getId(),
      engineerLocale)

   ViewCriteria allCriteria = new ViewCriteria(view,
      ViewAssignmentParameter.allTickets(),
      ViewGroupParameter.allTickets(),
      new ViewOrderParameter())
   def allTickets = ticketService.getIdsByView(allCriteria)
   ViewCriteria ownCriteria = new ViewCriteria(view,
      ViewAssignmentParameter.allTickets(engineerService.getCurrent()),
      ViewGroupParameter.onlyOwnTickets(),
      new ViewOrderParameter())
   def ownTickets = ticketService.getIdsByView(ownCriteria)

   ViewCriteria unassignedCriteria = new ViewCriteria(view,
      ViewAssignmentParameter.allUnassignedTickets(),
      ViewGroupParameter.onlyUnassignedTickets(),
      new ViewOrderParameter())
   def unassignedTickets = ticketService.getIdsByView(unassignedCriteria)

   seriesdata.add("{name: '${viewName}',
      y: ${allTickets.size()},
      drilldown: '${view.getName()}'}")

   def data = []
   data.add("['All', ${allTickets.size()}]")
   data.add("['Own', ${ownTickets.size()}]")
   data.add("['Unassigned', ${unassignedTickets.size()}]")

   drilldownseries.add("{id: '${view.getName()}',
      data:[${data.join(',')}]}" as String)
```

```
   allTicketsCounter += allTickets.size()
}

return [series: "[{name: 'Tickets', colorByPoint: true,
   data: [${allTicketsCounter}, ${seriesdata.join(',')}]}]" as String,
   drilldown: "{series: [${drilldownseries.join(',')}]}" as String,
   visible: 'true',
   chart: "{type: 'column'}",
   title: "{text: 'Tickets'}",
   xAxis: "{type: 'category'}",
   yAxis: "{type: 'linear'}",
   legend: "{enabled: false}"
];
```

Code-Beispiel 16: *Drill-down-Funktion für Chart-Widget*

C.9 Seitenanpassung für die grafische Darstellung von Relationen

Die Darstellung der Relationen als Graphen basiert auf der Seitenanpassung. Es sind zwei Modi verfügbar:

Der **Standardmodus** ermöglicht das Konfigurieren der Graphenansicht für die Relationsbereiche von Tickets, Kundenseiten und Ressourcenseiten. Die Graphenansicht wird dann als Link *Graph* angeboten und der Bearbeiter kann zwischen der Listen- und der Graphenansicht wechseln. Eine detaillierte Erklärung finden Sie im Abschnitt Seitenanpassung für die Visualisierung von Relationen, Standard.

Der **Expertenmodus** ermöglicht das Einblenden eines neuen Bereichs auf Ticket-, Kunden- und Ressourcenseiten, in dem ein vollständig angepasster Graph, der auf einem Skript basiert, angezeigt wird. Eine detaillierte Beschreibung finden Sie im Abschnitt Seitenanpassung zur Visualisierung von Relationen, Experten.

Das **grundlegende Konfigurationsprinzip** ist in beiden Modi gleich und wird im Abschnitt Konfiguration der Graphenansicht von Relationen erklärt.

C.9.1 Seitenanpassung für die Visualisierung von Relationen, Standard

C.9.1.1 Einleitung

In allen Bereichen, die Relationen darstellen (siehe Abschnitt Konfiguration der Graphenanzeige im Standardmodus) ist es möglich, den ConSol CM Web Client so zu konfigurieren, dass er neben der Listenansicht auch eine Graphenansicht der Relationen zeigt. Dies wird in der Seitenanpassung gemacht.

Sie können einfach die Graphenansicht aktivieren und die Standardeinstellungen verwenden. Dies ist ein sehr einfacher Schritt. Alternativ können Sie auch eine etwas komplexere Konfiguration vornehmen.

Das folgende Beispiel zeigt, wie die Anzeige von Relationsgraphen auf einer Ticketseite konfiguriert wird. Da CM/Resource Pool aktiv ist, umfasst das Beispiel die Konfiguration von Relationen zwischen Tickets und Ressourcen als Graph. Dasselbe Prinzip, das anhand einer

- Ticketseite

gezeigt wird, gilt auch für eine

- Firmenseite
- Kontaktseite
- Ressourcenseite (verfügbar, wenn CM/Resource Pool aktiv ist)

C.9.1.2 Beispiel für die Seitenanpassung (Standard): contactSection auf Ticketseite

Seitenanpassung des Kontaktbereichs auf der Ticketseite beginnen

Auf einer Ticketseite gibt es drei Bereiche, in denen Relationen angezeigt werden können.

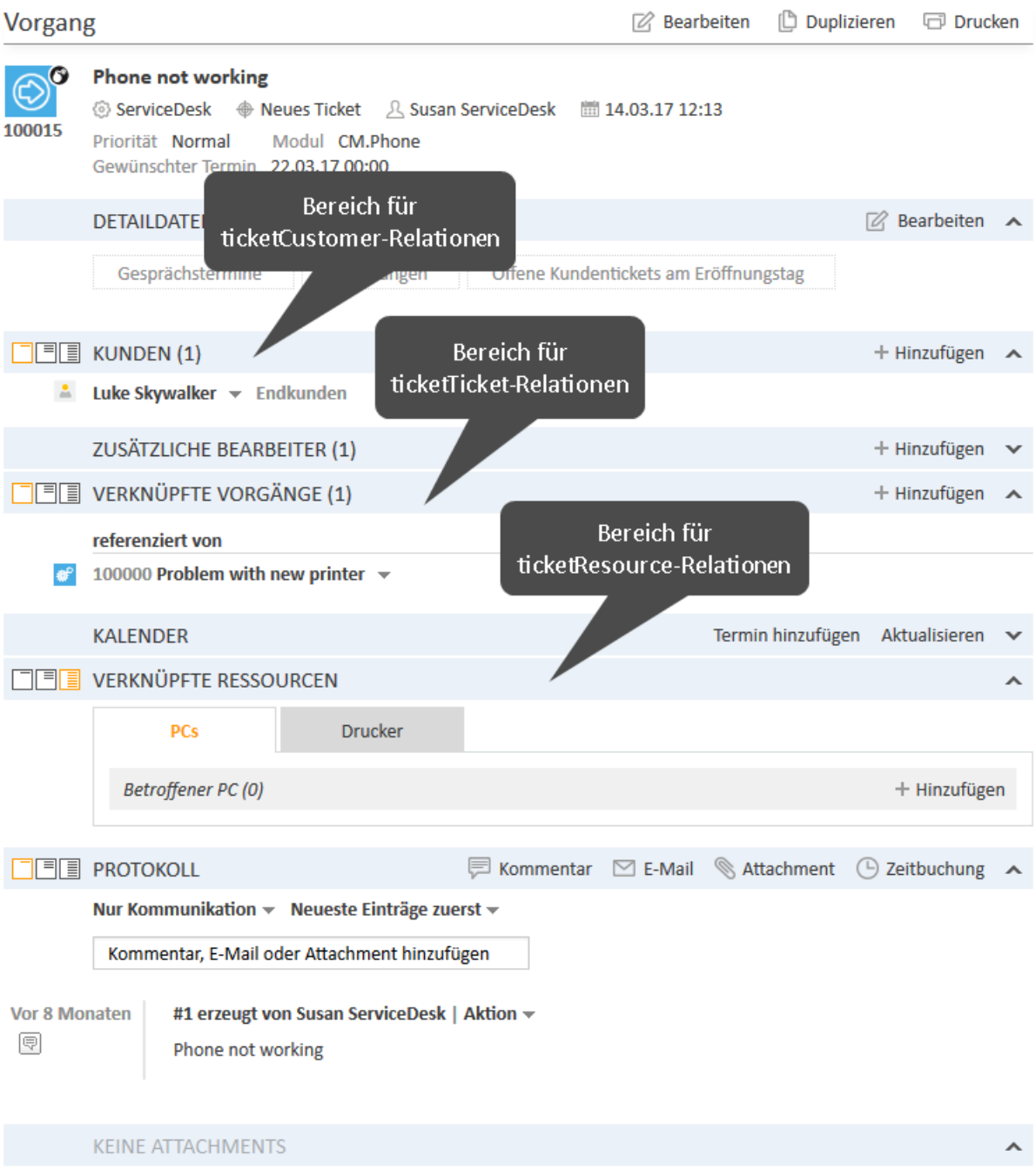


Abbildung 221: *ConSol CM Web Client - Ticketseite: Bereiche, in denen Relationen angezeigt werden*

Für jeden der Bereiche kann die Graphenansicht separat konfiguriert werden. Beginnen Sie zum Beispiel mit dem Kundenbereich. Melden Sie sich als Bearbeiter mit Administratorrechten an und klicken Sie im Hauptmenü auf *Seitenanpassung aktivieren*. Für den Kundenbereich werden mehrere Bereiche und Unterbereiche der Seitenanpassung angeboten. Konfigurieren Sie den Graph im bereichsspezifischen Unterbereich. Für den Kundenbereich ist dies der Unterbereich *contactSection*.

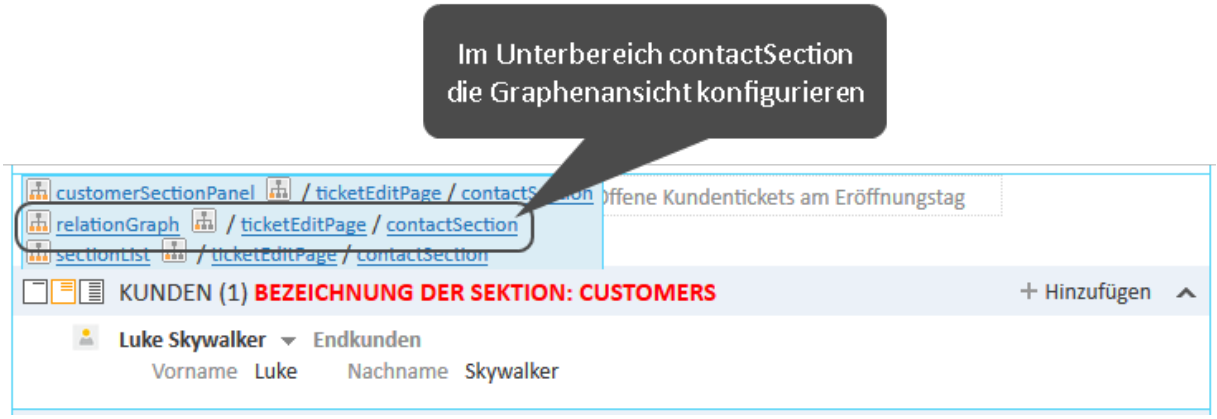


Abbildung 222: *ConSol CM Web Client - Auswahl des Bereichs der Seitenanpassung für relationGraph in contactSection*

Sie können den Unterbereich *contactSection* auch in der Baumstruktur der Seitenanpassung auswählen. In beiden Fällen wird die Konfigurationsseite *relationGraph* für den Kontaktbereich geöffnet (siehe folgende Abbildung).

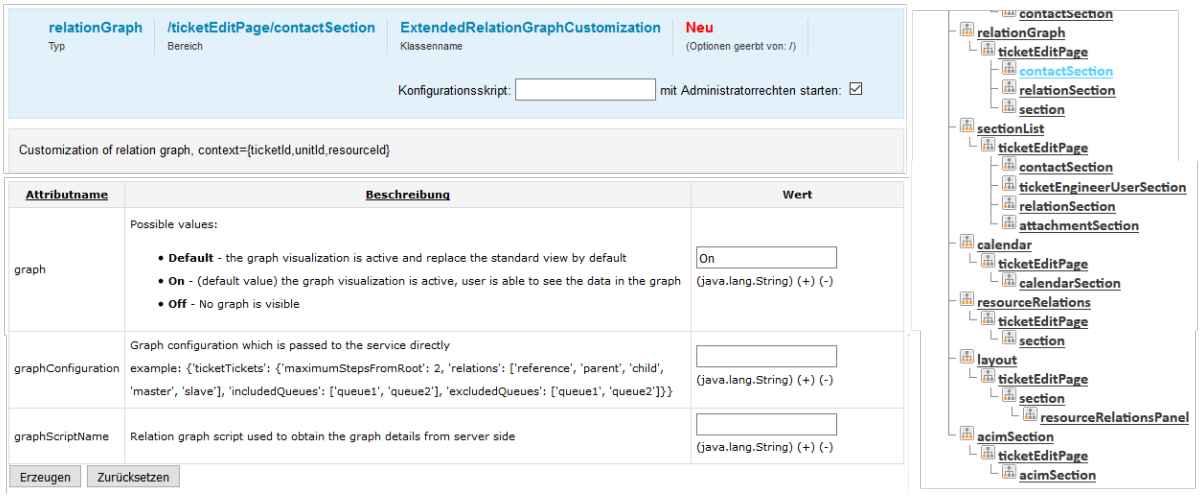


Abbildung 223: *Seitenanpassung des Kontaktbereichs auf ticketEditPage*

Attribute für die Seitenanpassung der Standardansicht des Relationsgraphen

Es sind folgende Attribute verfügbar:

- **graph** - String. Mögliche Werte sind:
  - „Default“ - Die Graphenvisualisierung ist aktiv und ersetzt die Standardansicht standardmäßig.
  - „On“ - (Standardwert in neuen Installationen) Die Graphenvisualisierung ist aktiv, der Benutzer kann die Daten im Graphen sehen.
  - „Off“ - Es wird kein Graph angezeigt.

- **graphConfiguration** - String, JSON-Ausdruck

  Mit diesem JSON-Ausdruck können Sie die Darstellung des Graphen konfigurieren. Im System wird implizit ein Standardwert definiert. Deshalb müssen Sie beim Einschalten der Graphenansicht keine *graphConfiguration* konfigurieren. Wenn die Standardkonfiguration nicht die gewünschte Ansicht enthält, können Sie dieses Attribut verwenden. Eine detaillierte Erklärung finden Sie im Abschnitt Konfigurieren der Graphenansicht von Relationen mit JSON-Ausdrücken.

- **graphScriptName** - String

  Bezieht sich auf ein Skript, das im Admin Tool im Bereich *Skripte und Templates* gespeichert ist. In dem Skript kann der Graph anhand komplexer Informationen gebaut werden. Es ist zum Beispiel möglich, Daten aus anderen CM-Objekten oder anderen Systemen, die für CM relevante Daten bereitstellen, einzubinden. Das Skript muss den Typ *Relationsgraph* haben und immer ein Objekt der Klasse `RelationGraph` zurückgeben. Eine detaillierte Erklärung dieses Skripttyps finden Sie im Abschnitt Konfigurieren der Graphenansicht von Relationen mithilfe von Skripten.

Geben Sie entweder *graphConfiguration* oder *graphScriptName* an. Wenn Sie beides angeben, wird das Skript verwendet und *graphConfiguration* nicht berücksichtigt.

Einfaches Beispiel mit *graphConfiguration*

Die folgende Beispielkonfiguration wird für eine erste Erklärung der Funktionsweise des Attributs *graphConfiguration* verwendet.

In der *contactSection* (siehe obige Abbildungen) werden folgende Werte für die Attribute gesetzt:

- *graph*: „on“
- *graphConfiguration*:

```
{"ticketCustomers" :
 {"maximumStepsFromRoot" : 2,
  "relations":["main","default","end customer"],
  "includedCustomerGroups":["Reseller","MyCustomerGroup"]}
 }
```

Damit wird der folgende Graph angezeigt:



Abbildung 224: *ConSol CM Web Client - Ticketseite: Graphenansicht der Kundenrelationen zum Ticket*

Wenn Sie *graphConfiguration* folgendermaßen ändern:

```
{"ticketCustomers" :
 {"maximumStepsFromRoot" : 2,
  "relations":["main","default","end customer"],
  "includedCustomerGroups":["DirectCustomers","MyCustomerGroup"]}
 }
```

Wird folgender Graph angezeigt:

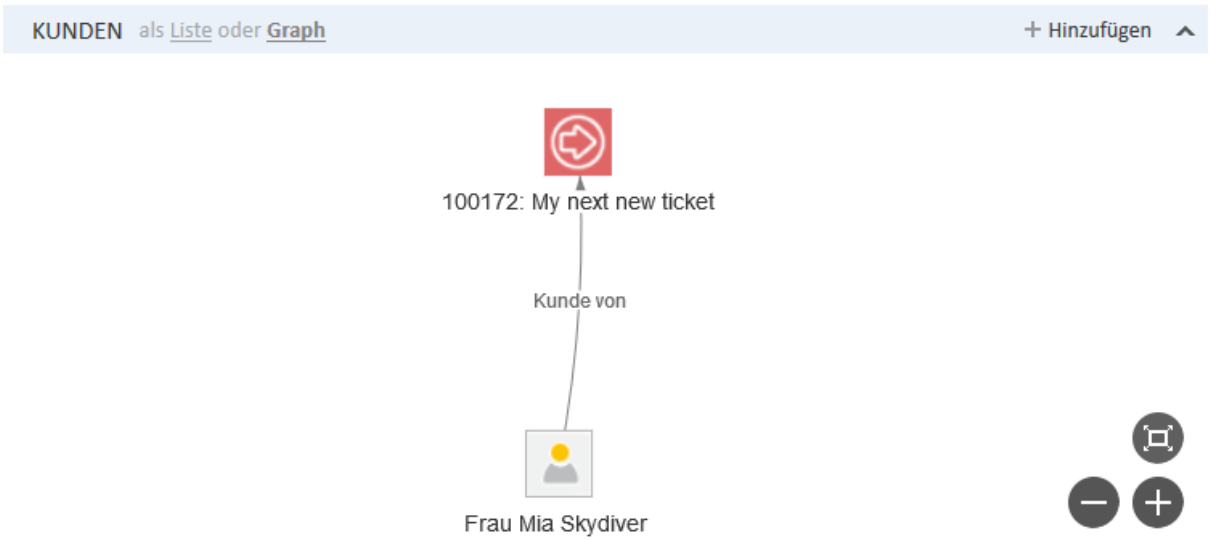


Abbildung 225: *ConSol CM Web Client - Ticketseite: Graphenansicht der Kundenrelationen zum Ticket, Variante*

Der Kunde (Kontakt) Luke Skywalker ist in der Kundengruppe *Reseller* und, wenn die Kundengruppe *Reseller* nicht explizit eingeschlossen wird, wird dieser Kunde nicht im Graphen angezeigt.

Dieses einfache Beispiel hilft Ihnen hoffentlich, das allgemeine Prinzip von *configurationGraph* zu verstehen. Eine detaillierte Erklärung der Parameter finden Sie im Abschnitt Konfigurieren der Graphenansicht von Relationen mit JSON-Ausdrücken.

Einfaches Beispiel mit *graphScript*

Ein *graphScript* muss immer ein Objekt der Klasse `RelationGraph` zurückgeben.

Das folgende Skript zeigt nur Relationen zwischen Ticket und Kunden und kann im Kundenbereich einer Ticketseite verwendet werden. In diesem Fall wäre es einfacher, mit *configurationGraph* zu arbeiten. Aber wir möchten Ihnen zeigen, wie das Skript funktioniert ...

```
import com.consol.cmas.common.model.relation.*

//create criteria using java api
def relationCriteria = new RelationCriteria()
   .withTicketCustomers(new RelationCriteria.TicketCustomers()
   .withMaximumStepsFromRoot(2));
def ticket = ticketService.getById(ticketId)
def relationGraph = relationService.load(ticket, relationCriteria);

return relationGraph;
```

Code-Beispiel 17: *Einfaches graphScript*

Es wird folgender Graph angezeigt:

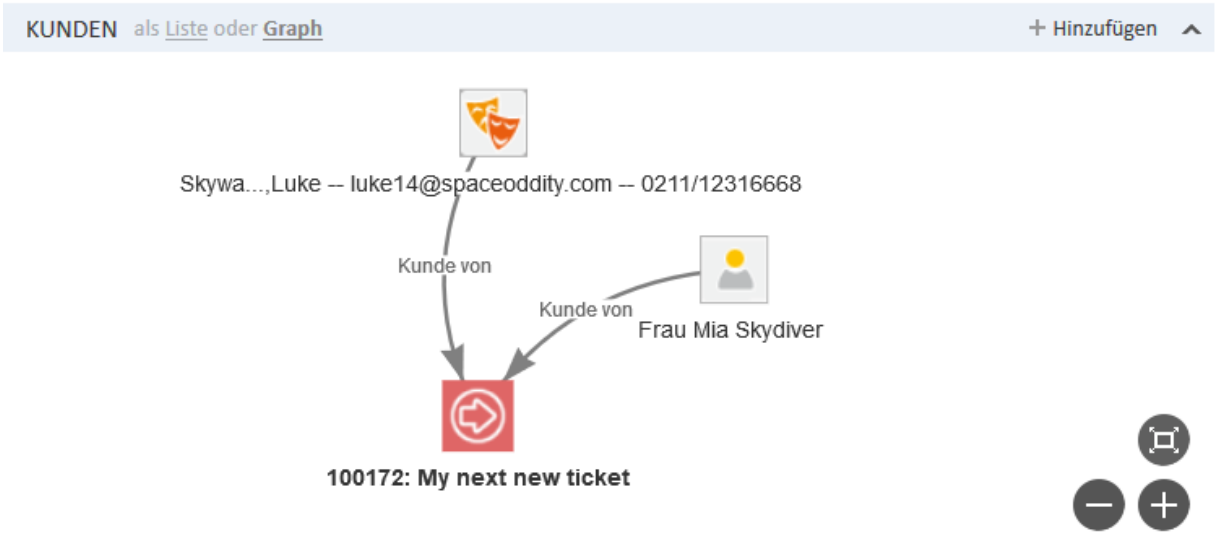


Abbildung 226: *ConSol CM Web Client - Ticketseite: Graphendarstellung im Kundenbereich basiert auf dem obigen Skript*

Wenn das Layout geändert werden soll, können Sie das Skript wie folgt ändern.

```
import com.consol.cmas.common.model.relation.*
//create criteria using java api
def relationCriteria = new RelationCriteria()
   .withTicketCustomers(new RelationCriteria.TicketCustomers()
   .withMaximumStepsFromRoot(2));
def ticket = ticketService.getById(ticketId)
def relationGraph = relationService.load(ticket, relationCriteria);
for (RelationNode node : relationGraph.getNodes()) {
   node.withProperty("font", [size:10,color:"blue"]) //change default property
      .withProperty("image", null) //remove default property
      .withProperty("shape", "diamond") //remove default property
   }
return relationGraph;
```

Code-Beispiel 18: *Einfaches graphScript mit einfacher Knotenlayoutdefinition*

Es wird folgender Graph angezeigt:

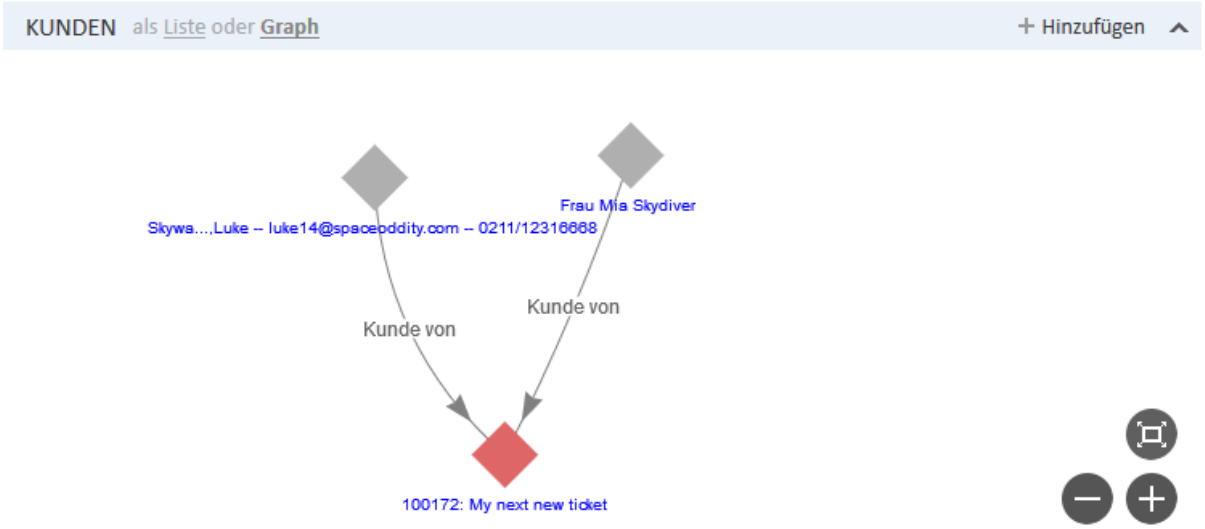


Abbildung 227: *ConSol CM Web Client - Ticketseite: Leicht veränderte Graphendarstellung im Kundenbereich, basiert auf dem obigen Skript*

Im obigen Skript wurde nur die Darstellung der Knoten geändert. Die Darstellung der Kanten kann ebenfalls verändert werden. Außerdem können zahlreiche Konfigurationsparameter gesetzt werden. Eine detaillierte Erklärung finden Sie im Abschnitt Konfigurieren der Graphenansicht von Relationen mithilfe von Skripten.

C.9.2 Seitenanpassung zur Visualisierung von Relationen, Experten

C.9.2.1 Einleitung

Die Standardkonfiguration der Graphenansichten für Relationen (siehe Abschnitt Seitenanpassung für die Visualisierung von Relationen, Standard) bietet die Möglichkeit, Graphenansichten in den Relationsbereichen von Ticket-, Kunden- und Ressourcenseiten anzuzeigen, d. h. sie erweitert die Funktionalität der Bereiche, die es auf den entsprechenden Seiten bereits gibt, und fügt die Option zur Anzeige der Graphenansicht hinzu.

Zusätzlich dazu kann ein ganz neuer Bereich zu

- Ticketseiten
- Kontaktseiten
- Firmenseiten
- Ressourcenseiten (wenn CM/Resource Pool aktiv ist)

hinzugefügt werden. Dieser Bereich ist in Standardinstallationen ausgeblendet und kann zur Anzeige eines individuellen, komplett systemspezifischen Relationsgraphen verwendet werden. Der Graph basiert auf einem Skript. Es gibt keine andere Möglichkeit, den Inhalt und die Darstellung des Graphen zu konfigurieren. Dies muss in einem Admin Tool-Skript geschehen.

> Beachten Sie, dass dies Folgendes bedeutet:
>
> Als Skriptentwickler steht es Ihnen frei, alle erforderlichen Objekte einzubinden, da ein Skript die gesamte ConSol CM Java/Groovy API verwenden kann.
>
> Als Skriptentwickler sind Sie alleine für den angezeigten Inhalt verantwortlich.

Das folgende Beispiel zeigt, wie man einen systemspezifischen Relationsgraphen auf der Ressourcenseite implementiert. Das gleiche Prinzip gilt auch für Ticket-, Kontakt- und Firmenseiten.

C.9.2.2 Beispiel der Seitenanpassung (Experten): neuer Bereich auf der Ressourcenseite

Die folgenden Schritte müssen durchgeführt werden, um einen systemspezifischen Relationsgraphen auf der Ressourcenseite zu implementieren:

- Schritt 1: Erforderlichen Bereich auf der Seite sichtbar machen
- Schritt 2: Attribute der Seitenanpassung für den neu angezeigten Bereich konfigurieren
- Schritt 3: Skript schreiben
- Schritt 4: Ergebnis überprüfen

Schritt 1: Erforderlichen Bereich auf der Seite sichtbar machen

Der Bereich, der angezeigt werden muss, heißt *customRelationGraphSection*. Da der Bereich noch nicht angezeigt wird, können Sie ihn im Seitenanpassungsmodus nicht direkt auswählen. Stattdessen müssen Sie die Baumstruktur der Seitenanpassung verwenden.

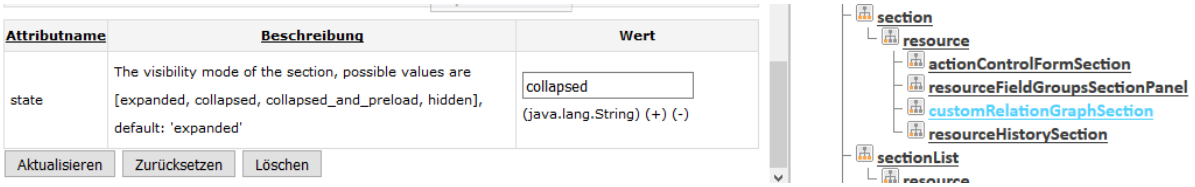


Abbildung 228: *Ein Satz an Attributen der Seitenanpassung für customRelationGraphSection*

Setzen Sie das Attribut *state* entweder auf „collapsed“ oder „expanded“. Der neue Bereich ist auf der Seite verfügbar, in diesem Beispiel auf der Ressourcenseite. Der Standardname / die Überschrift des neuen Bereichs ist *Custom relation graph*. Der Bereich wird über dem Protokollbereich angezeigt. Es wird noch kein Graph angezeigt. Wenn Sie den Bereich im Web Client anzeigen, wird eine Fehlermeldung angezeigt, da es noch kein Skript gibt, das den entsprechenden Graphen definiert.

Schritt 2: Attribute der Seitenanpassung für den neu angezeigten Bereich konfigurieren

Um den Inhalt und die Darstellung des Graphen zu konfigurieren, müssen Sie zuerst die Attribute der Seitenanpassung für den neuen Bereich setzen. Wählen Sie dazu den Bereich entweder auf der Seite oder in der Baumstruktur der Seitenanpassung aus:

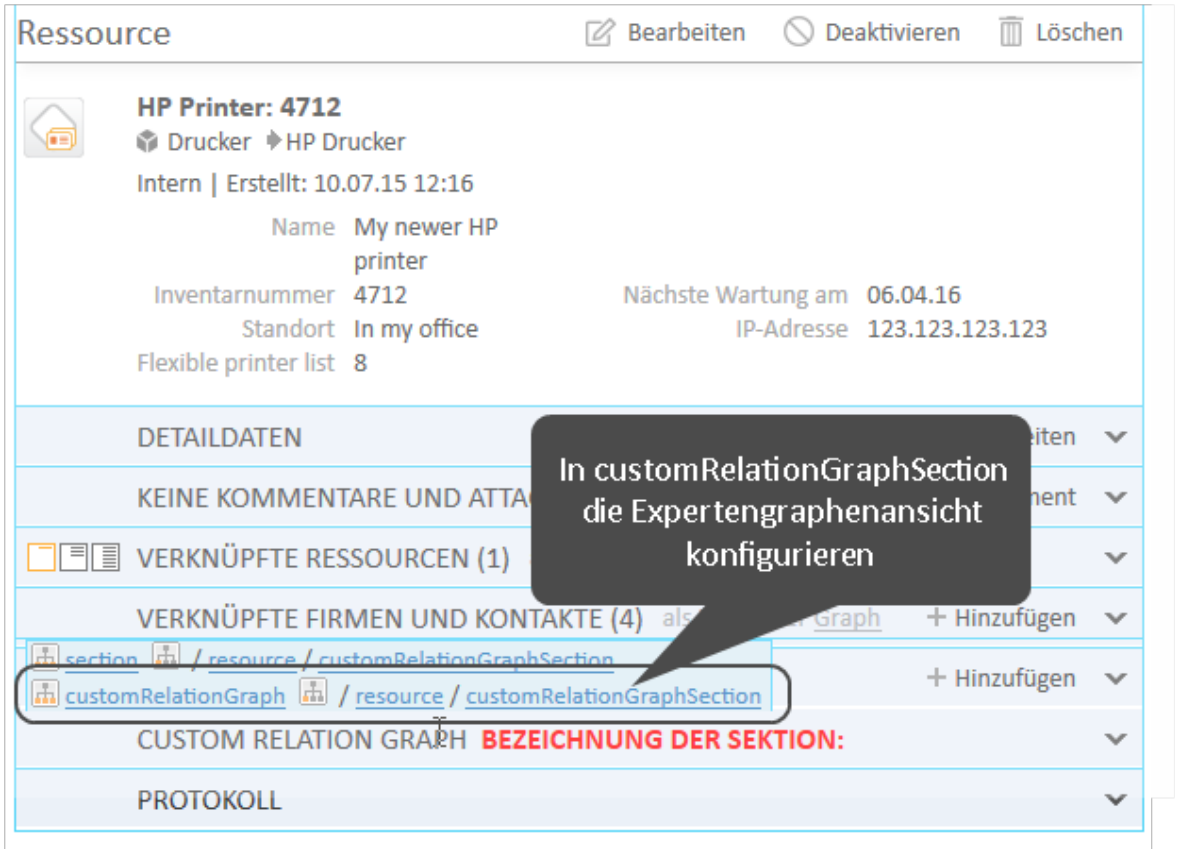


Abbildung 229: *ConSol CM Web Client - Auswahl des Bereichs der Seitenanpassung für den customRelationGraph auf der Ressourcenseite*

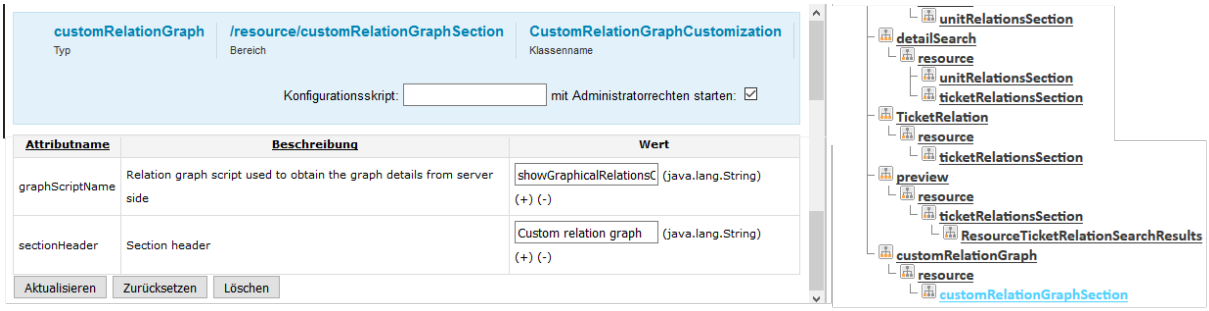


Abbildung 230: *Baumstruktur der Seitenanpassung und Attribute für customRelationGraphSection*

Es sind zwei Attribute verfügbar:

- **graphScriptName** - String. Bezieht sich auf ein Skript, das im Admin Tool im Bereich *Skripte und Templates* gespeichert ist. Im Skript kann der Relationsgraph anhand von komplexen Informationen gebaut werden. Zum Beispiel ist es möglich, Daten aus anderen CM-Objekten oder anderen Systemen, die für CM relevante Daten bereitstellen, einzubinden. Das Skript muss vom Typ *Relationsgraph* sein und immer ein Objekt der Klasse `RelationGraph` zurückgeben. Eine detaillierte Erklärung dieses Skripttyps finden Sie im Abschnitt Konfigurieren der Graphenansicht von Relationen mithilfe von Skripten.
- **sectionHeader** - String. Wird im Web Client als Überschrift des Bereichs angezeigt. Verwenden Sie einen beliebigen String.

Schritt 3: Skript schreiben

Der Wert von *graphScriptName* muss ein Skript des Typs *Relationsgraph* referenzieren, das im Bereich *Skripte und Templates* des Admin Tools gespeichert ist. Eine detaillierte Erklärung dieses Skripttyps finden Sie im Abschnitt Konfigurieren der Graphenansicht von Relationen mithilfe von Skripten.

Im aktuellen Beispiel werden die folgenden Werte für die Attribute gesetzt:

- sectionHeader = „This device within the entire infrastructure“
- graphScriptName = „showGraphicalRelationsOfResource2.groovy“

Es wird folgendes Skript verwendet:

```
import com.consol.cmas.common.model.relation.*
//create criteria using java api
def relationCriteria = new RelationCriteria()
   .withResourceTickets(new RelationCriteria.ResourceTickets()
   .withMaximumStepsFromRoot(2))
   .withTicketResources(new RelationCriteria.TicketResources()
   .withMaximumStepsFromRoot(2))
   .withResourceUnits(new RelationCriteria.ResourceUnits()
   .withMaximumStepsFromRoot(2))
   .withResourceResources(new RelationCriteria.ResourceResources()
   .withMaximumStepsFromRoot(2));


//variables available in script: ticketId, unitId, resourceId (depending on page)
def resource = resourceService.getById(resourceId)
def relationGraph = relationService.load(resource, relationCriteria);
//modify graph nodes, edges and visjs property
for (RelationNode node : relationGraph.getNodes()) {
   node.withProperty("font", [size:9,color:"black"]) //change default property
}
for (RelationEdge edge : relationGraph.getEdges()) {
   edge.withProperty("font", [size:12, color:"red"]); //change default property
}
relationGraph.withProperty("layout", [improvedLayout:true]) //change default
 property
return relationGraph;
```

Code-Beispiel 19: *Skript für customRelationGraph , showGraphicalRelationsOfResource2.groovy*

Schritt 4: Ergebnis überprüfen

Im Web Client wird der folgende Graph angezeigt:

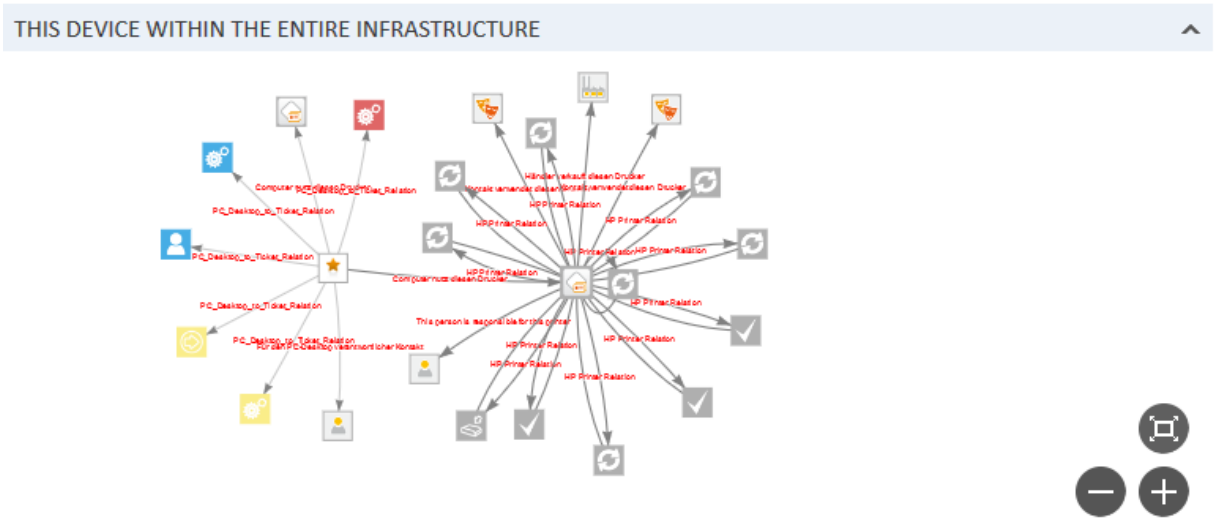


Abbildung 231: *ConSol CM Web Client - Anzeige eines Experten-Relationsgraphs auf der Ressourcenseite*

C.9.3 Konfiguration der Graphenansicht von Relationen

C.9.3.1 Einführung in die Konfiguration der Graphenansicht von Relationen

Die Graphenansicht von Relationen basiert auf vis.js, genauer gesagt auf der Bibliothek Network. Alle verfügbaren Konfigurationsparameter finden Sie in der detaillierten vis.js Network-Dokumentation. In den folgenden Abschnitten werden die wichtigsten Parameter anhand einiger Beispiele erklärt.

Wie im Abschnitt Attribute für die Seitenanpassung der Standardansicht des Relationsgraphen erklärt, kann zum Definieren des Layouts des Graphen entweder ein JSON-Ausdruck oder ein Skript verwendet werden.

C.9.3.2 Konfigurieren der Graphenansicht von Relationen mit JSON-Ausdrücken

Einleitung

Ein JSON-Ausdruck zur Konfiguration des Relationsgraphen kann auf zwei Arten angegeben werden:

- Als Wert eines Attributs der Seitenanpassung:
  Geben Sie den Ausdruck als Wert des Attributs der Seitenanpassung `graphConfiguration` an, siehe Abschnitt Attribute für die Seitenanpassung der Standardansicht des Relationsgraphen.
- In einem graphScript:
  Geben Sie den Ausdruck als JSON-String an, der dann von der Klassenmethode `RelationCriteria.parseJson(JSONstring)` verwendet wird.

Der erste Fall ist der häufigste Weg, um Standardrelationsgraphen zu konfigurieren. Der zweite Fall kann in Sonderfällen verwendet werden, normalerweise sind aber andere Methoden besser geeignet.

> Beachten Sie, dass nur der Umfang des Graphen mit einem JSON-Ausdruck definiert werden kann. Wenn Sie das Layout des Graphen ändern möchten (z. B. Farbe der Kanten, Schriftart), müssen Sie mit Skripten arbeiten. Siehe Abschnitt Konfigurieren der Graphenansicht von Relationen mithilfe von Skripten.

Syntax zur Konfiguration der Graphenansicht

Im JSON-Ausdruck kann der **Umfang** des Graphen definiert werden.

Für **Tickets** können folgende Parameter definiert werden:

- **Für Ticket-Ticket-Relationen:**

```
"ticketTickets" : {
 "maximumStepsFromRoot" : 2,
 "relations" : [ "reference", "parent", "child", "master", "slave" ],
 "includedQueues" : [ "queue1", "queue2" ],
 "excludedQueues" :[ "queue1", "queue2" ]
 }
```

Verwenden Sie entweder `includedQueues` oder `excludedQueues`!

Die folgenden Relationstypen sind möglich:

- „reference“
  Das Ticket referenziert das aktuelle Ticket.
- „parent“
  Das Ticket ist Parent-Ticket des aktuellen Tickets.
- „child“
  Das Ticket ist Child-Ticket des aktuellen Tickets.
- „master“
  Das Ticket ist Master-Ticket des aktuellen Tickets.
- „slave“
  Das Ticket ist Slave-Ticket des aktuellen Tickets.

- **Für Ticket-Kunden-Relationen:**

```
"ticketCustomers" : {
 "maximumStepsFromRoot" : 2,
 "relations" : [ "main", "default", "contactTicketRoleName" ],
 "includedCustomerGroups" : [ "customerGroup1", "customerGroup2" ],
 "excludedCustomerGroups" : [ "customerGroup1", "customerGroup2" ]
 }
```

Verwenden Sie entweder `includedCustomerGroups` oder `excludedCustomerGroups`!

Die folgenden Relationstypen sind möglich:

- „main“
  Der Kunde ist der Hauptkunde des Tickets.
- „default“
  Der Kunde ist ein Zusatzkunde ohne Rolle.
- Name der Kundenrolle

- **Für Ticket-Ressourcen-Relationen:**

```
"ticketResources" : {
 "maximumStepsFromRoot" : 2,
 "relations" : [ "related", "resourceRelationDefinition" ],
 "includedResourceTypes" : [ "resourceType1", "resourceType2" ],
 "excludedResourceTypes" : [ "resourceType1", "resourceType2" ]
 }
```

Für **Kunden = Units (d. h. Kontakte und Firmen)** können folgende Parameter definiert werden:

- **Für Kunden-Ticket-Relationen:**

```
"customerTickets" : {
 "maximumStepsFromRoot" : 2,
 "relations" : [ "main", "default", "contactTicketRoleName" ],
 "includedQueues" : [ "queue1", "queue2" ],
 "excludedQueues" : [ "queue1", "queue2" ]
 }
```

Folgende Relationstypen sind möglich:

- „main“
  Der Kunde ist der Hauptkunde des Tickets.
- „default“
  Der Kunde ist ein Zusatzkunde ohne Rolle.
- Name der Kundenrolle

- **Für Unit-Unit-Relationen:**

```
"unitUnits" : {
 "maximumStepsFromRoot" : 2,
 "relations" : [ "unitRelationDefinition1", "unitRelationDefinition2" ],
 "includedCustomerGroups" : [ "customerGroup1", "customerGroup2" ],
 "excludedCustomerGroups" : [ "customerGroup1", "customerGroup2" ]
 }
```

Verwenden Sie entweder `includedCustomerGroups` oder `excludedCustomerGroups`!

Bei gerichteten Unit-Unit-Relationen ist es möglich, die Richtung anzugeben, d. h. ob die im Graphen verknüpften Objekte die Quelle oder das Ziel der Relation sind. Die Syntax ist Folgende:

1. Die verknüpften Kunden sind die Quelle der Relation und der Wurzelknoten ist das Ziel:

```
"relations" : [ "unitRelationDefinition:source"]
```

2. Die verknüpften Kunden sind das Ziel der Relation und der Wurzelknoten ist die Quelle:

```
"relations" : ["unitRelationDefinition:target"]
```

- **Für Kontakt-Firmen-Relationen:**

```
"contactCompany" : {
 "maximumStepsFromRoot" : 2
 }
```

- **Für Firma-Kontakt-Relationen:**

```
"companyContacts" : {
 "maximumStepsFromRoot" : 2
 }
```

- **Für Unit-Ressourcen-Relationen:**

```
"unitResources" : {
 "maximumStepsFromRoot" : 2,
 "relations" : [ "related", "resourceRelationDefinition" ],
 "includedResourceTypes" : [ "resourceType1", "resourceType2" ],
 "excludedResourceTypes" : [ "resourceType1", "resourceType2" ]
 }
```

Verwenden Sie entweder `includedResourceTypes` oder `excludedResourceTypes`!

Für Ressourcen können folgende Parameter definiert werden:

- **Für Ressource-Ressourcen-Relationen:**

```
"resourceResources" : {
 "maximumStepsFromRoot" : 2,
 "relations" : [ "related", "resourceRelationDefinition" ],
 "includedResourceTypes" : [ "resourceType1","resourceType2" ],
 "excludedResourceTypes" : [ "resourceType1", "resourceType2" ]
 }
```

Verwenden Sie entweder `includedResourceTypes` oder `excludedResourceTypes`!

Bei gerichteten Ressource-Ressourcen-Relationen ist es möglich, die Richtung anzugeben, d. h. ob die im Graphen verknüpften Objekte die Quelle oder das Ziel der Relation sind. Die Syntax ist Folgende:

1. Die verknüpften Ressourcen sind die Quelle der Relation und der Wurzelknoten ist das Ziel:

```
"relations" : ["resourceRelationDefinition:source"]
```

2. Die verknüpften Ressourcen sind das Ziel der Relation und der Wurzelknoten ist die Quelle:

```
"relations" : ["resourceRelationDefinition:target"]
```

- **Für Ressource-Unit-Relationen:**

```
resourceUnits" : {
 "maximumStepsFromRoot" : 2,
 "relations" : [ "related", "resourceRelationDefinition" ],
 "includedCustomerGroups" : [ "customerGroup1", "customerGroup2" ],
 "excludedCustomerGroups" : [ "customerGroup1", "customerGroup2" ]
 }
```

Verwenden Sie entweder `includedCustomerGroups` oder `excludedCustomerGroups`!

C.9.3.3 Konfigurieren der Graphenansicht von Relationen mithilfe von Skripten

Skripte können an zwei Stellen verwendet werden, um die Graphenansicht der Relationen zu definieren:

- Im Standardansichtsmodus als Wert des Attributs `graphScriptName`
- Im angepassten Expertenansichtsmodus als Wert des Attributs `graphScriptName`

> Beachten Sie, dass in einem Skript jede Relation in die Anzeige aufgenommen werden kann. Daher ist die Person, die das Skript implementiert, für die Anzeige des Graphen verantwortlich. Stellen Sie sicher, dass die richtigen Relationen angezeigt werden, insbesondere in Skripten der Standardbereiche. Es macht zum Beispiel wenig Sinn, im Kundenbereich eines Tickets Relationen zwischen dem Ticket und Ressourcen darzustellen, auch wenn dies technisch möglich ist. Wenn komplexe Netze aus Relationen unterschiedlicher Typen angezeigt werden sollen, empfehlen wir, diese in einem angepassten Relationsgraphen darzustellen und dem Bereich eine sprechende Überschrift zu geben.

Ein Skript, das die Graphenansicht von Relationen definiert:

- Muss den Typ „Relationgraph“ haben.
- Muss ein Objekt der Klasse `RelationGraph` zurückgeben.
- Kann Folgendes definieren:
  - den Umfang des Graphen
  - das Layout des Graphen
- Kann folgende implizit im Skript vorhandene Objekte verwenden (je nach Ort des Skripts):
  - ticketId (auf Ticketseiten)
  - unitId (auf Kundenseiten)
  - resourceId (auf Ressourcenseiten)

Um den **Umfang** eines Relationsgraphen in einem Skript zu definieren, verwenden Sie die Klasse `RelationCriteria` und die statischen Klassen der unterschiedlichen Ausprägungen von `RelationCriteria`, z. B. `RelationCriteria.ResourceTickets` (siehe *ConSol CM Java/Groovy API-Dokumentation*).

Beispielskript:

```
import com.consol.cmas.common.model.relation.*
//create criteria using java api
def relationCriteria = new RelationCriteria()
   .withResourceTickets(new RelationCriteria.ResourceTickets()
   .withMaximumStepsFromRoot(2))
   .withResourceUnits(new RelationCriteria.ResourceUnits()
   .withMaximumStepsFromRoot(2))
   .withResourceResources(new RelationCriteria.ResourceResources()
   .withMaximumStepsFromRoot(2));
def resource = resourceService.getById(resourceId)
def relationGraph = relationService.load(resource, relationCriteria);
return relationGraph;
```

Anzeige:

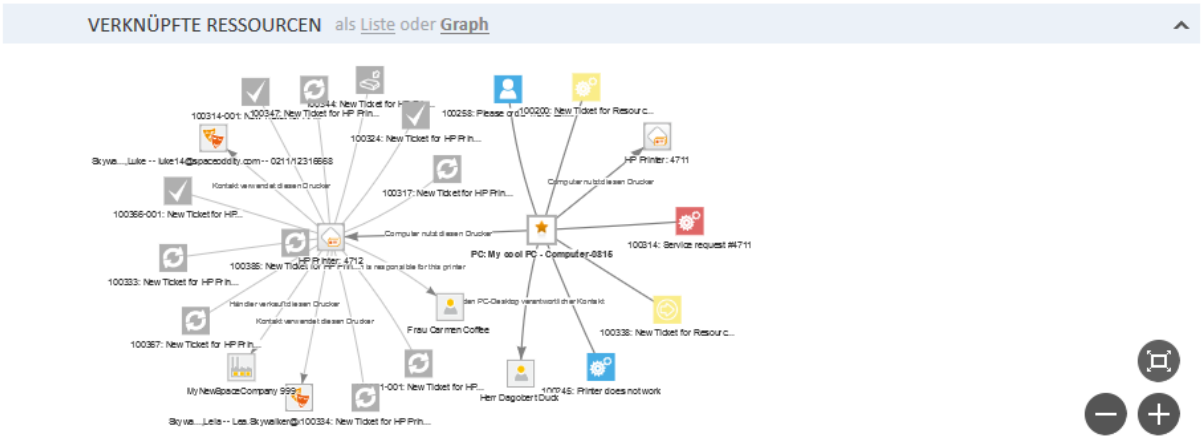


Abbildung 232: *ConSol CM Web Client - Ressourcenseite: Graph*

Zum Definieren des **Layouts** des Graphen verwenden Sie die Objekte `RelationNode` und `RelationEdge`.

Beispielskript (Skript wie oben, um die Layoutkonfiguration erweitert):

```
import com.consol.cmas.common.model.relation.*
//create criteria using java api
def relationCriteria = new RelationCriteria()
   .withResourceTickets(new RelationCriteria.ResourceTickets()
   .withMaximumStepsFromRoot(2))
   .withResourceUnits(new RelationCriteria.ResourceUnits()
   .withMaximumStepsFromRoot(2))
   .withResourceResources(new RelationCriteria.ResourceResources()
   .withMaximumStepsFromRoot(2));


def resource = resourceService.getById(resourceId)
def relationGraph = relationService.load(resource, relationCriteria);
//define layout:
for (RelationNode node : relationGraph.getNodes()) {
   node.withProperty("font", [size:12,color:"blue"]) //change default property
      .withProperty("image", null) //remove default property
      .withProperty("shape", "diamond") //remove default property
      .withProperty("margin", 5); //add property
}

for (RelationEdge edge : relationGraph.getEdges()) {
   edge.withProperty("arrows", null) //remove default property
      .withProperty("font", [size:16, color:"red"]); //change default property
}
relationGraph.withProperty("layout", [hierarchical:[direction:"UD"]])

return relationGraph;
```

Anzeige:

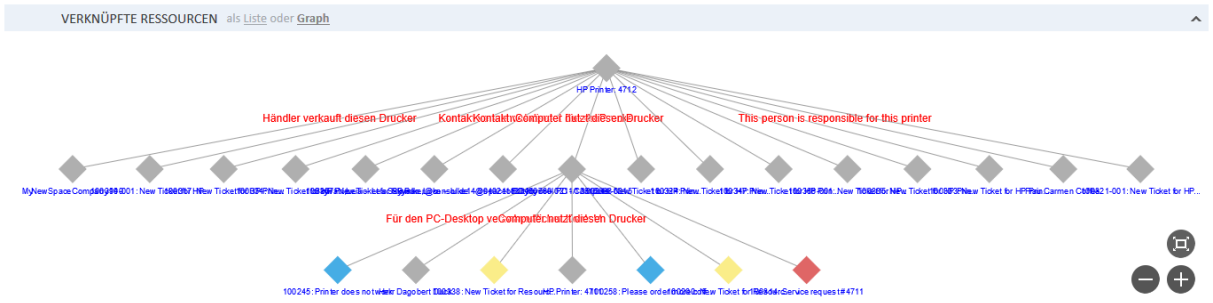


Abbildung 233: *ConSol CM Web Client - Ressourcenseite: Graph mit geänderten Layoutoptionen*

Die Schriftgröße wurde auf 12 pt gesetzt, die Schriftfarbe in Blau geändert, die Form wurde in Diamant geändert und das Layout wurde in die hierarchische Anzeige geändert.

Dieses Beispiel vermittelt einen Eindruck darüber, wie das Layout von Relationsgraphen verändert werden kann. Gehen Sie nach diesem Prinzip vor, um die Relationsgraphen in Ihrem CM-System an die Anforderungen Ihrer Firma anzupassen. Zusätzliche Informationen finden Sie in der vis.js Network-Dokumentation.

C.10 Design und Konfiguration von REST-basierten ConSol CM-Client-GUIs

C.10.1 Einleitung

Um eine bequeme Konfiguration der Benutzeroberfläche von REST-Clients zu ermöglichen, enthält das Admin Tool in CM-Versionen 6.10.7.0 und höher (in 6.11 seit Version 6.11.0.5) die Navigationsgruppe *Clients*.

Sie kann verwendet werden, um die Benutzeroberfläche von folgenden REST-Clients zu konfigurieren:

- CM/Track V2 (aktuell verfügbar)
- CM/Phone (verfügbar in zukünftigen Versionen)
- CM/Mobile (verfügbar in zukünftigen Versionen)
- kundenspezifische CM-REST-Clients

Konfiguration bedeutet in diesem Kontext Folgendes:

- Layout der Seite
- Lokalisierung für verschiedene Sprachen

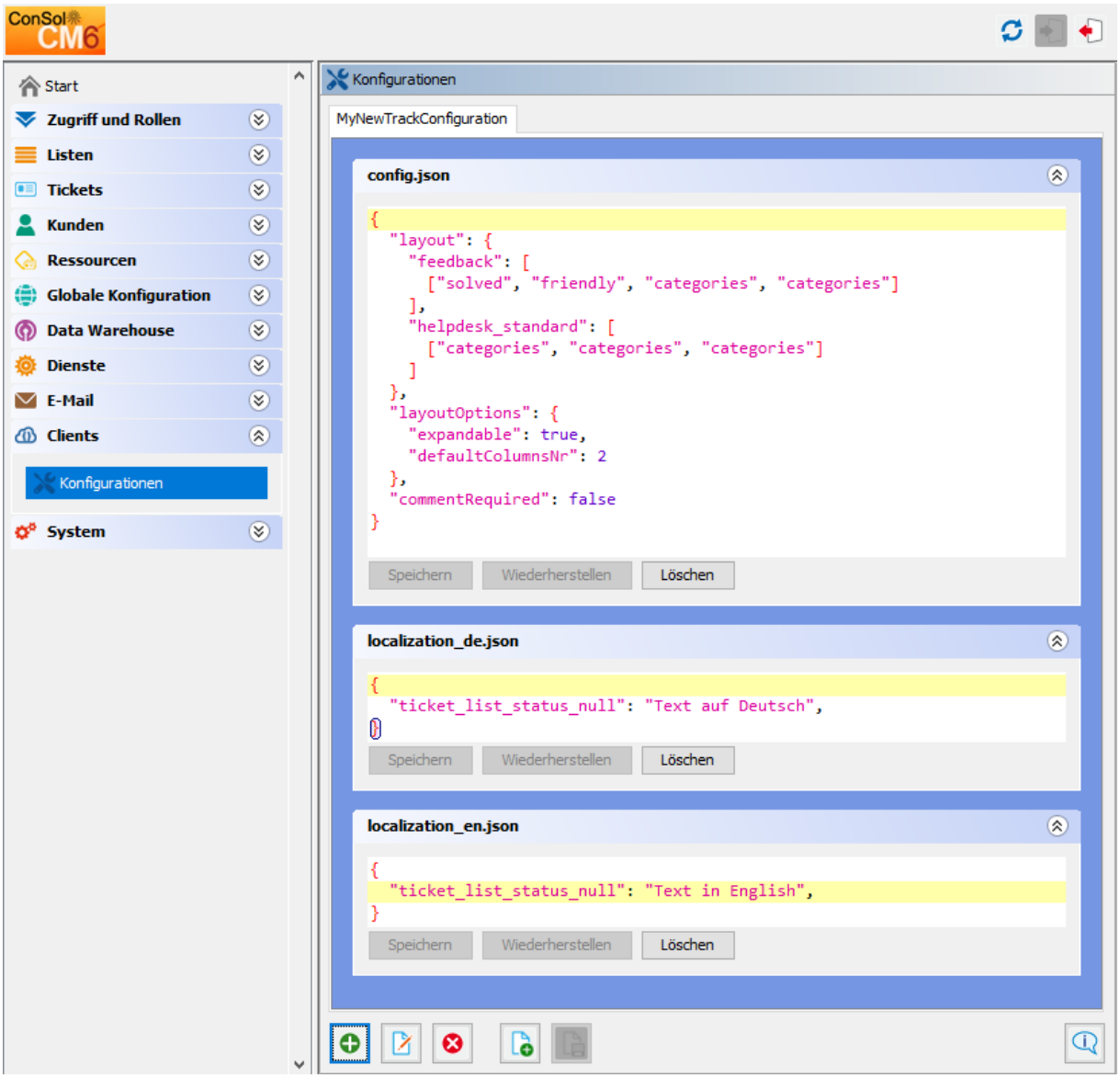


Abbildung 234: *ConSol CM Admin Tool - REST-Client-Konfiguration (Beispiel)*

Der Konfigurationsbereich hat für jede vorhandene Konfiguration einen Tab mit dem Namen der Konfiguration. Eine Konfiguration besteht aus mehreren Dateien, normalerweise im JSON-Format (JavaScript Object Notation), die auf Anfrage an den Client übermittelt werden. In diesem Tab werden die Dateien nacheinander zur Bearbeitung angezeigt. Unter dem Bereich befinden sich die Buttons zum Organisieren der Konfigurationen. Mit den Buttons können Sie folgende Aufgaben für eine Konfiguration als Ganzes ausführen:

- **Neue Konfiguration erstellen**
  Öffnet ein Dialogfenster zum Benennen der Konfiguration und Auswählen der Vorlage, auf der sie basieren soll.

- **Konfiguration bearbeiten**
  Öffnet ein Dialogfenster zum Umbenennen der ausgewählten Konfiguration und Auswählen der Vorlage.
- **Konfiguration löschen**
  Entfernt die gesamte Konfiguration, sodass der Tab nicht mehr vorhanden ist.
- **Neue Datei erstellen**
  Öffnet ein Dialogfenster zum Festlegen des Dateinamens und erweitert die Konfiguration um eine leere Datei mit diesem Namen.
- **Alle Dateien speichern**
  Speichert alle noch nicht gespeicherten Änderungen in allen Dateien der aktuell ausgewählten Konfiguration.
- **Beispielkonfigurationen**
  Öffnet ein Fenster mit Beispielen aus Vorlagen, wobei jede Vorlage in einem Tab angezeigt wird. Aktuell gibt es nur eine Vorlage, die *TrackV2* heißt und in den Dialogfenstern zum Erstellen und Bearbeiten von Client-Typen sowie im Fenster mit den Beispielen zur Auswahl steht. Das folgende Beispiel zeigt eine auf diesem Beispiel für eine CM/Track-Konfiguration basierende neu erstellte Konfiguration.

Jeder Dateiabschnitt hat dateiorientierte Funktionen. Die entsprechenden Buttons befinden sich unter dem Bereich zum Editieren des Dateiinhalts:

- **Speichern**
  Speichert die an der Datei vorgenommenen Änderungen, um sie an den Client zu übermitteln.
- **Wiederherstellen**
  Setzt den Dateiinhalt auf den Zustand nach dem letzten Speichern zurück.
- **Löschen**
  Entfernt die gesamte Datei aus der Konfiguration.

C.10.2 Konfigurationsprinzip

Die für den Client relevante Konfiguration hängt von der vom Client aufgerufenen URL ab. Die spezifische Konfiguration für den Client wird von der Angular-Applikation innerhalb von ConSol CM übermittelt.

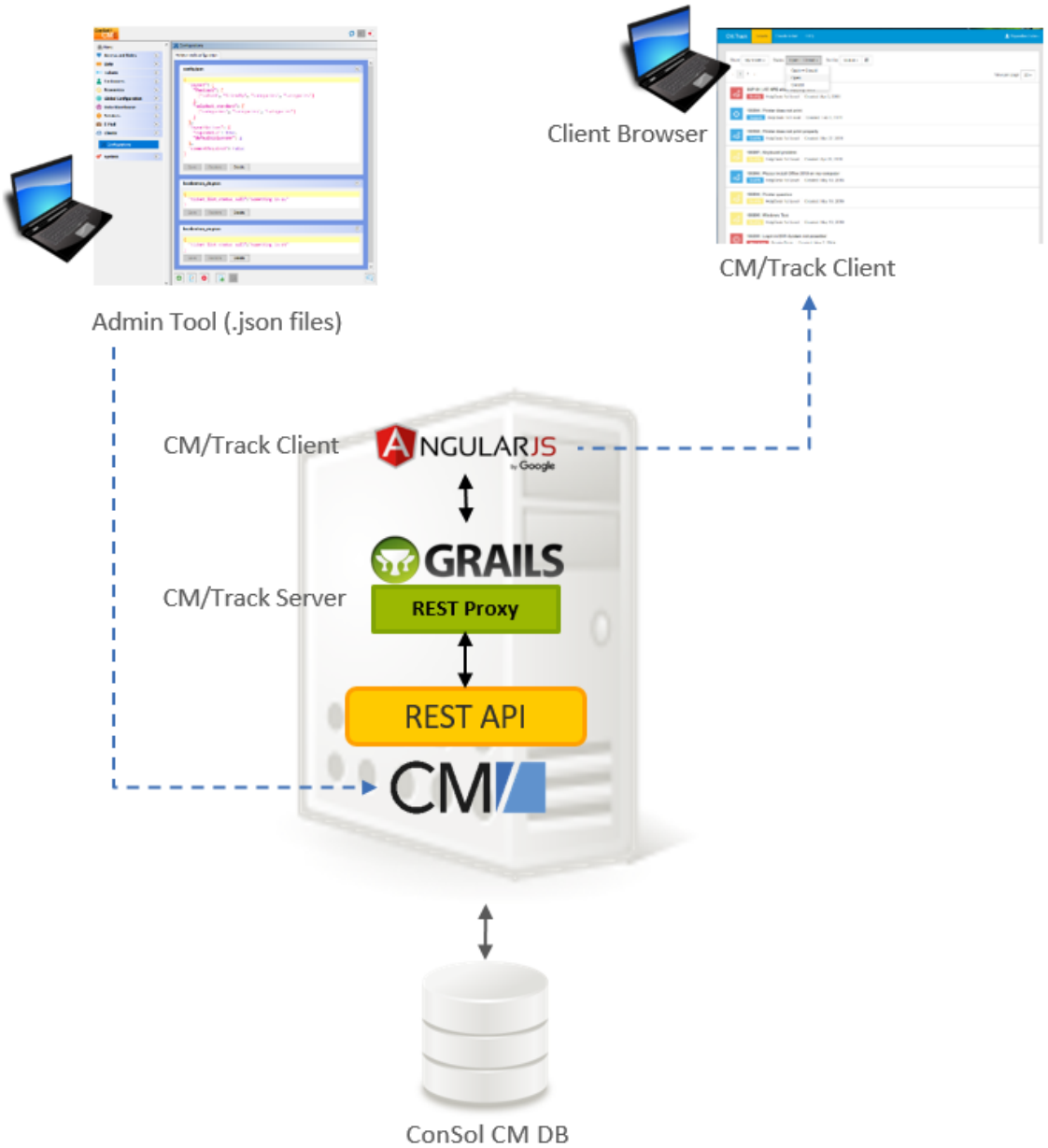


Abbildung 235: *Prinzip der Konfiguration der REST-Client-GUI mit dem Admin Tool*

Die Zuordnung zwischen der spezifischen Domäne und der erforderlichen Konfiguration wird über die folgende CM-System-Property konfiguriert:

`cmas-restapi-core, domain.map.for.client.config.<configName>`

Der Wert muss die Domäne und (sofern erforderlich) der Ziel-Port der entsprechenden CM/Track V2-Instanz sein.

Beispiel: `cmas-restapi-core, domain.map.for.client.config.trackConfig1` = „myserver:8080/track“

Wenn zwei CM/Track-Instanzen mit derselben Konfiguration arbeiten sollen, verwenden Sie folgende Syntax:

`cmas-restapi-core, domain.map.for.client.config.<configName>` = <Liste Domänen mit Ports>

Beispiel: `cmas-restapi-core, domain.map.for.client.config.trackConfig2` = „myOtherServer:8080/track , myNewServer:8180/track“

Die CM-System-Property muss manuell zur Systemkonfiguration hinzugefügt werden.

C.10.3 Konfiguration von speziellen Seiten in CM/Track

Die Konfiguration von speziellen Seiten in CM/Track erfolgt mithilfe von Konfigurations- und Lokalisierungsdateien.

In der Datei `config.json` werden Aspekte von CM/Track konfiguriert, die relevant sind, nachdem sich der Benutzer angemeldet hat. Sie enthält sowohl Layout-Einstellungen für die Seiten zum Erstellen und Anzeigen von Tickets als auch die Einstellungen für die Startseite. Die Lokalisierungen für die nicht-öffentlichen Seiten von CM/Track befinden sich in den sogenannten Lokalisierungsdateien, z. B. `localization_en.json` oder `localization_de.json`.

In der Datei `public.json` wird die Anmeldeseite konfiguriert. Sie enthält sowohl Einstellungen als auch Lokalisierungen.

C.10.3.1 Konfiguration der Seiten zum Erstellen und Anzeigen von Tickets

Die Konfiguration des Layouts der Seite zum Erstellen von einem Ticket und der Seite zum Anzeigen von einem Ticket basiert in CM/Track auf folgenden Dateien:

- config.json
- Lokalisierungsdatei (z. B. `localization_en` oder `localization_de`)

> Beachten Sie, dass jedes Ticketfeld, das in CM/Track angezeigt werden soll, mithilfe der Annotationen bezüglich der Sichtbarkeit für Kunden konfiguriert werden muss. Eine detaillierte Erklärung dazu finden Sie im Abschnitt CM/Track V2: Datenverfügbarkeit für Kunden.

config.json

In dieser Datei wird das Layout / die Reihenfolge der Ticketfelder definiert und die allgemeinen Layout- und Konfigurationsparameter werden gesetzt.

Das Objekt `layout` hat folgende Struktur:

```
"layout": {
   "<Ticket field group name 1>": [
      [<List of names of ticket fields in 1st line],
      [<List of names of ticket fields in 2nd line],

      [<List of names of ticket fields in n'th line],
   ],
   <Ticket field group name 2>": [
      [<List of names of ticket fields in 1st line],
      [<List of names of ticket fields in 2nd line],

      [<List of names of ticket fields in n'th line]
   ]
}
```

Regeln zum Schreiben eines JSON-Layoutobjekts:

- Jede Zeile (Array), die Ticketfelder enthält, muss die gleiche Anzahl an Elementen haben. Verwenden Sie Platzhalter für leere Positionen (siehe Infobox unten).
- Verwenden Sie eindeutige Namen für die Platzhalter (siehe Infobox unten).
- Verwenden Sie keine Kommentare innerhalb des JSON-Statements.

Info über Platzhalter:

Beachten Sie, dass der Platzhalter für eine leere Position eindeutig sein muss. Sie können entweder einen leeren String (nur einmal!) oder Platzhalter-Strings (z. B. "x1") verwenden. Ein Platzhalter wird wie ein echtes Datenfeld behandelt, d. h., wenn es zwei leere Positionen nebeneinander in einer Zeile gibt, können Sie diese zum Beispiel als "x1","x1" oder "x1","x2" schreiben. Der gleiche Platzhalter in unterschiedlichen Zeilen kann nicht verknüpft werden und funktioniert nicht. Null als Platzhalter funktioniert nicht.

Beispiel:

```
"layout": {
   "helpdesk_standard": [
      ["module", "categories","categories"],
      ["reactiontime","priority",""]
   ]
},
```

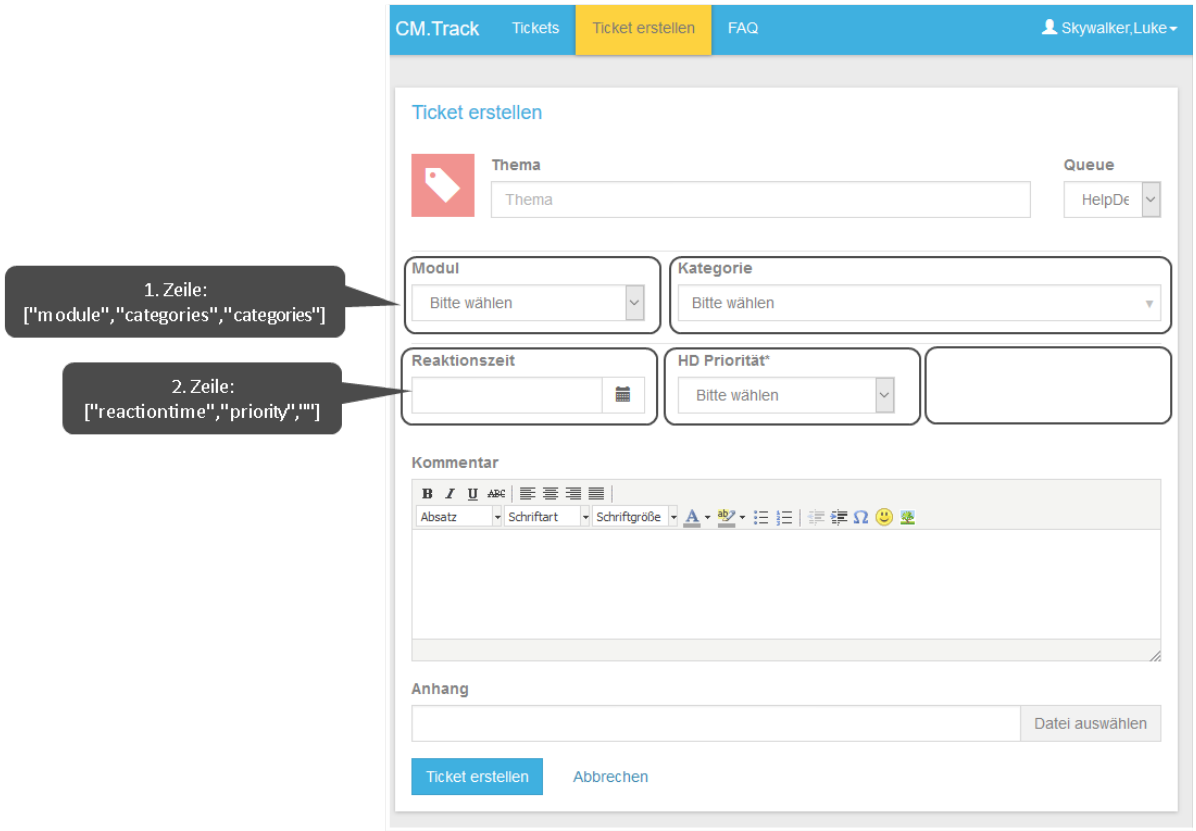


Abbildung 236: *CM/Track-Client - Anzeige von Ticketfeldern aus dem oben gezeigten Beispiellayout*

Die **layoutOptions** werden folgendermaßen gesetzt:

```
"layoutOptions": {
   "expandable": false,
   "defaultColumnsNr": 3
}
```

- `expandable` (siehe Beispiel unten)
  - „true“: Ticketfelder, die nicht explizit im JSON-Layoutobjekt erwähnt sind, aber als für den Kunden sichtbar annotiert sind, werden angezeigt.
  - „false“: Es werden nur die explizit im JSON-Layoutobjekt erwähnten Ticketfelder angezeigt.
- `defaultColumns`: Wird nur verwendet, wenn `expandable` = „true“ ist und es eine Ticketfeldgruppe gibt, deren Layout nicht explizit angegeben wurde. In diesem Fall wird `defaultColumns` für die Anzeige verwendet.

Wenn zur Erstellung eines Tickets ein **Kommentar** erforderlich ist, wird dies über folgendes Objekt definiert:

```
commentRequired (Boolean)
```

Beispiel:

Nur die ersten beiden Zeilen haben ein explizites Layout, aber es werden weitere Felder (z. B. *infotext*) angezeigt, da `expandable` = „true“ ist.

```
{
   "layout": {
      "helpdesk_standard": [
         ["module", "categories","categories"],
         ["reactiontime","priority",""]
      ]
   },
   "layoutOptions": {
      "expandable": true,
      "defaultColumnsNr": 2
   },
   "commentRequired": false
}
```

Code-Beispiel 20: *Beispiel für die Datei config.json*

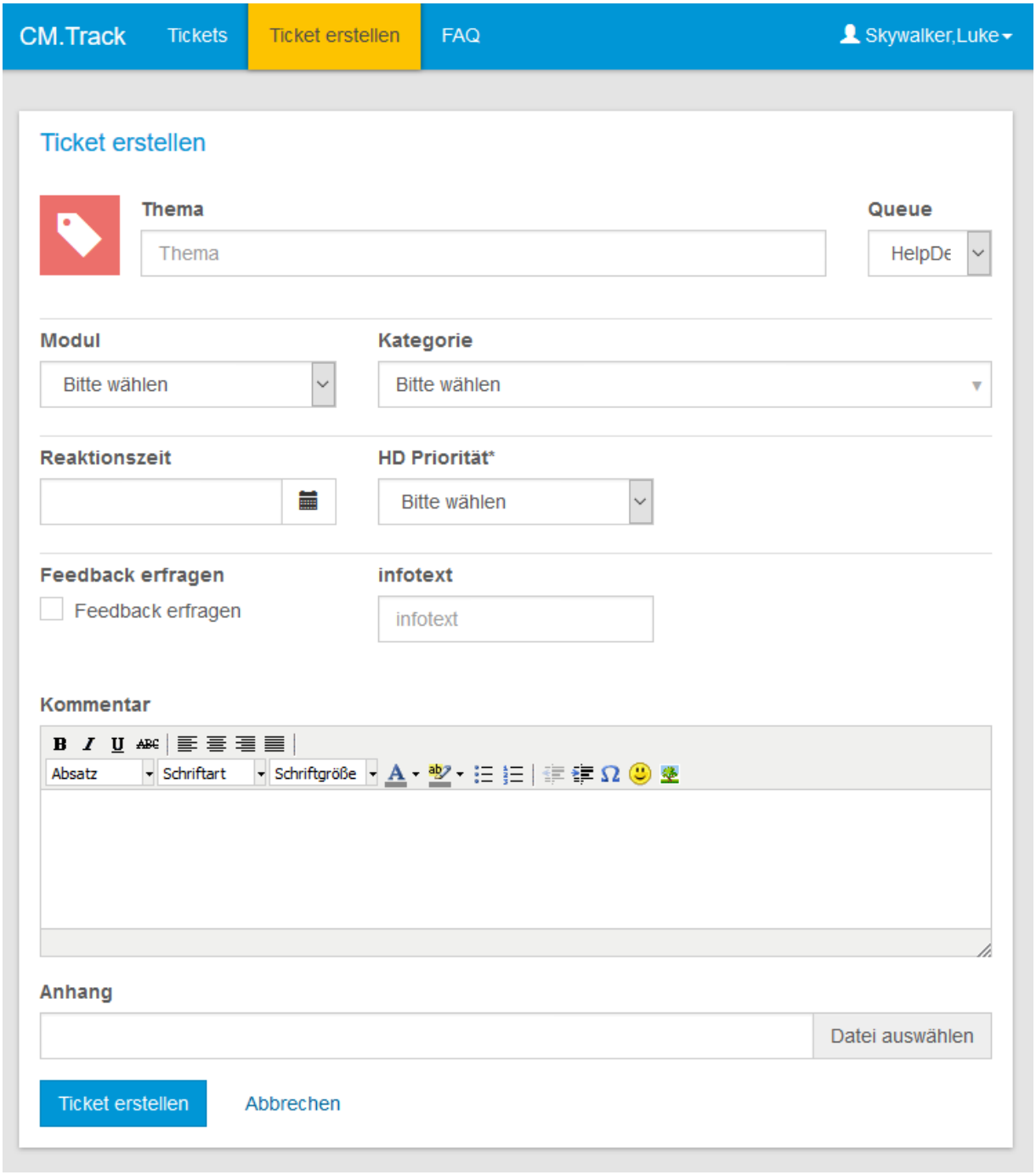


Abbildung 237: *CM/Track-Layout unter Verwendung des Layout-Codes aus dem obigen Beispiel*

Das gleiche Beispiel unter Verwendung von `expandable` = „false“: Es werden nur die explizit im Layout definierten Ticketfelder angezeigt.

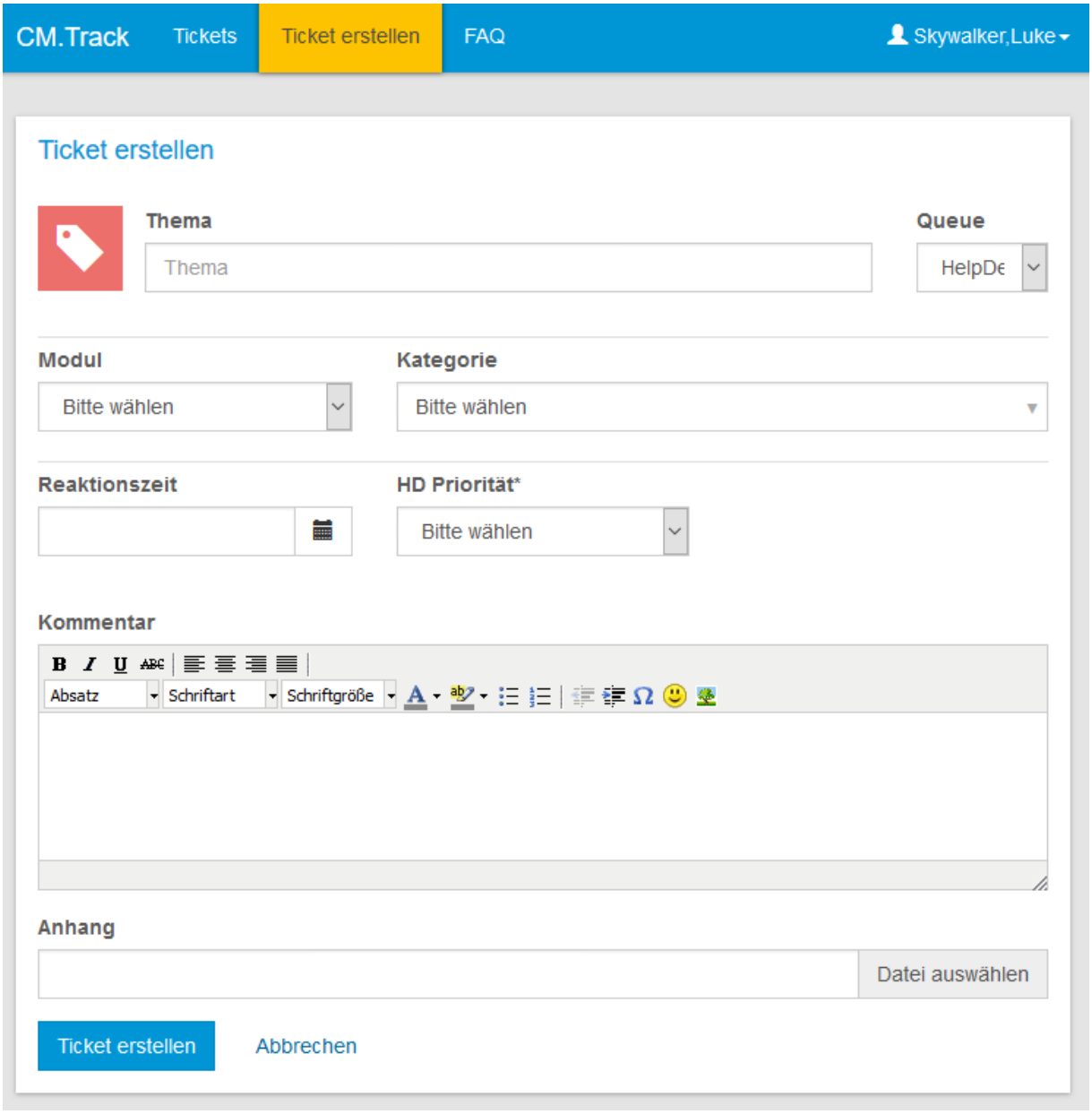


Abbildung 238: *CM/Track-Layout unter Verwendung des Layout-Codes aus dem obigen Beispiel, aber mit expandable = false*

Lokalisierungsdatei

Die Lokalisierungsdateien, z. B. `localization_en.json` und `localization_de.json`, enthalten Mappings von Properties auf Werte. Die Werte sind die Begriffe, die in der entsprechenden Sprache auf der GUI angezeigt werden sollen. Die folgenden beiden Beispiele zeigen einen Ausschnitt aus einer englischen und deutschen Lokalisierungsdatei.

```
{
```

```
"edit":"Edit",
"search":"Search",
"view":"View",
"delete":"Delete",
"filter":"Filter",
"add":"Add",
"save":"Save",
"cancel":"Cancel",
"true":"true",
"false":"false",
"previous":"Previous",
"next":"Next",
"validation_required" : "This field is required",
"validation_password" : "Please enter a valid password",
"validation_email" : "Please enter a valid e-mail",
"validation_pattern": "Invalid input format",
"validation_date" : "Please enter a valid date",
"validation_time" : "Please enter a valid time",
"validation_datetime" : "Please enter a valid date and time",
"validation_datetime_local" : "Please enter a valid date and time",
"validation_number" : "This field must be numeric",
"validation_color" : "Please enter a valid color",
"validation_range" : "Please enter a valid range",
"validation_month" : "Please enter a valid month",
"validation_url" : "Please enter a valid URL",
"validation_file" : "Invalid file",
"validation_minlength" : "Please use at least {{ minlength }} characters",
"validation_maxlength" : "Please do not exceed {{ maxlength }} characters",
"validation_min_time" : "The time provided should be after {{ min |date:
 \"HH:MM\" }}",
"validation_max_time" : "The time provided should be before {{ min |date:
 \"HH:MM\" }}",
"validation_ngFabMatch" : "The {{ type ===\"password\"? \"passwords\" :
 \"values\" }} should match",
"validation_min_date" : "The date provided should be after
 {{min|date:\"dd.MM.yy\"}}",
"validation_max_date" : "The date provided should be before {{max |date:
 \"dd.MM.yy\"}}",
"validation_min_number" : "The number provided should be at least {{min}}",
"validation_max_number" : "The number provided should be at max {{max}}",
"validation_fixedprecision" : "Illegal format. The integer part should have
 {{integer}} digits, the fraction part {{fractional}} digits",
"default_error_message" : "Sorry, it looks like something went wrong. An error
 has occurred. Please refresh your browser and try again later. If the error
 occurs another time, please contact your administrator.",
"cf_value_true" : "Yes",
"cf_value_false" : "No",
"session_expired" : "Session has expired",
"access_denied_error" : "You do not have permission to view this object. Please
 select a different object.",
"attachment_too_large" : "Sorry, the attachment size exceeds the configured
 limits for attachments. Please shrink the attachment.",
"datepicker_current" : "Today",
```

```
   "datepicker_clear" : "Clear",
   "datepicker_close" : "Done",
   "datepicker_Sunday": "Sunday"
}
```

Code-Beispiel 21: *Ausschnitt aus einer Datei localization_en.json*

```
{
   "edit":"Bearbeiten",
   "search":"Suchen",
   "view":"Anzeigen",
   "delete":"Löschen",
   "publish":"Veröffentlichen",
   "filter":"Filtern",
   "add":"Hinzufügen",
   "save":"Speichern",
   "cancel":"Abbrechen",
   "true":"Ja",
   "false":"Nein",
   "previous":"Vorheriges",
   "next":"Nächstes",
   "validation_required" : "Dieses Feld ist ein Pflichtfeld",
   "validation_password" : "Geben Sie ein gültiges Passwort ein",
   "validation_email" : "Geben Sie eine gültige E-Mail-Adresse ein",
   "validation_pattern": "Ungültiges Eingabeformat",
   "validation_date" : "Geben Sie ein gültiges Datum ein",
   "validation_time" : "Geben Sie eine gültige Zeit ein",
   "validation_datetime" : "Geben Sie ein gültiges Datum und eine gültige Zeit
    ein",
   "validation_datetime_local" : "Geben Sie ein gültiges Datum und eine gültige
    Zeit ein",
   "validation_number" : "Dieses Feld muss numerisch sein",
   "validation_color" : "Geben Sie eine gültige Farbe ein",
   "validation_range" : "Geben Sie einen gültigen Bereich ein",
   "validation_month" : "Geben Sie einen gültigen Monat ein",
   "validation_url" : "Geben Sie eine gültige URL ein",
   "validation_file" : "Ungültige Datei",
   "validation_minlength" : "Verwenden Sie mindestens {{ minlength }} Zeichen",
   "validation_maxlength" : "Verwenden Sie höchstens {{ maxlength }} Zeichen",
   "validation_min_time" : "Die angegebene Zeit sollte nach {{ min |date: \"HH:MM\"
    }} liegen",
   "validation_max_time" : "Die angegebene Zeit sollte vor {{ min |date: \"HH:MM\"
    }} liegen",
   "validation_ngFabMatch" : "Die {{ type ===\"password\"? \"passwords\" :
    \"values\" }} müssen übereinstimmen",
   "validation_min_date" : "Das angegebene Datum sollte nach dem
    {{min|date:\"dd.MM.yy\"}} liegen",
   "validation_max_date" : "Das angegebene Datum sollte vor dem {{max |date:
    \"dd.MM.yy\"}} liegen",
   "validation_min_number" : "Die angegebene Zahl sollte mindestens {{min}} sein",
   "validation_max_number" : "Die angegebene Zahl sollte höchstens {{max}} sein",
```

```
   "validation_fixedprecision" : "Ungültiges Format. Der ganzzahlige Teil sollte
    {{integer}} Stellen haben und der Dezimalteil {{fractional}} Stellen",
   "cf_value_true": "Ja",
   "cf_value_false": "Nein",
   "default_error_message" : "Entschuldigung, es scheint etwas schiefgelaufen zu
    sein. Ein Fehler ist aufgetreten. Bitte aktualisieren Sie Ihren Browser und
    versuchen es erneut. Sollte der Fehler nochmals auftreten, kontaktieren Sie
    bitte Ihren Administrator.",
   "session_expired" : "Die Sitzung ist abgelaufen",
   "access_denied_error" : "Sie haben keine Zugriffsrechte auf dieses Objekt. Bitte
    wählen Sie ein anderes Objekt aus.",
   "attachment_too_large" : "Leider überschreitet der Anhang die maximale Größe.
    Bitte verkleinern Sie die Datei.",
   "datepicker_current" : "Heute",
   "datepicker_clear" : "Löschen",
   "datepicker_close" : "Fertig",
   "datepicker_Sunday": "Sonntag"
}
```

Code-Beispiel 22: *Ausschnitt aus einer Datei localization_de.json*

C.10.3.2 Konfiguration der Startseite

Die Startseite wird angezeigt, nachdem sich der Benutzer in CM/Track angemeldet hat. Sie enthält folgende Elemente:

- Eine anpassbare Begrüßung
- Ein anpassbarer Begrüßungstext
- Ein Suchfeld
- Farbige Felder mit Links zum Erstellen von Tickets in den Queues, für die der Benutzer die erforderlichen Berechtigungen hat. Sollte es mehr als vier Queues geben, ist das vierte Feld ein Platzhalter mit einer Liste aller weiteren Queues. Jedes normale Feld enthält einen Button *Erstellen*, der direkt zur Seite zum Erstellen eines neuen Tickets in dieser Queue führt.

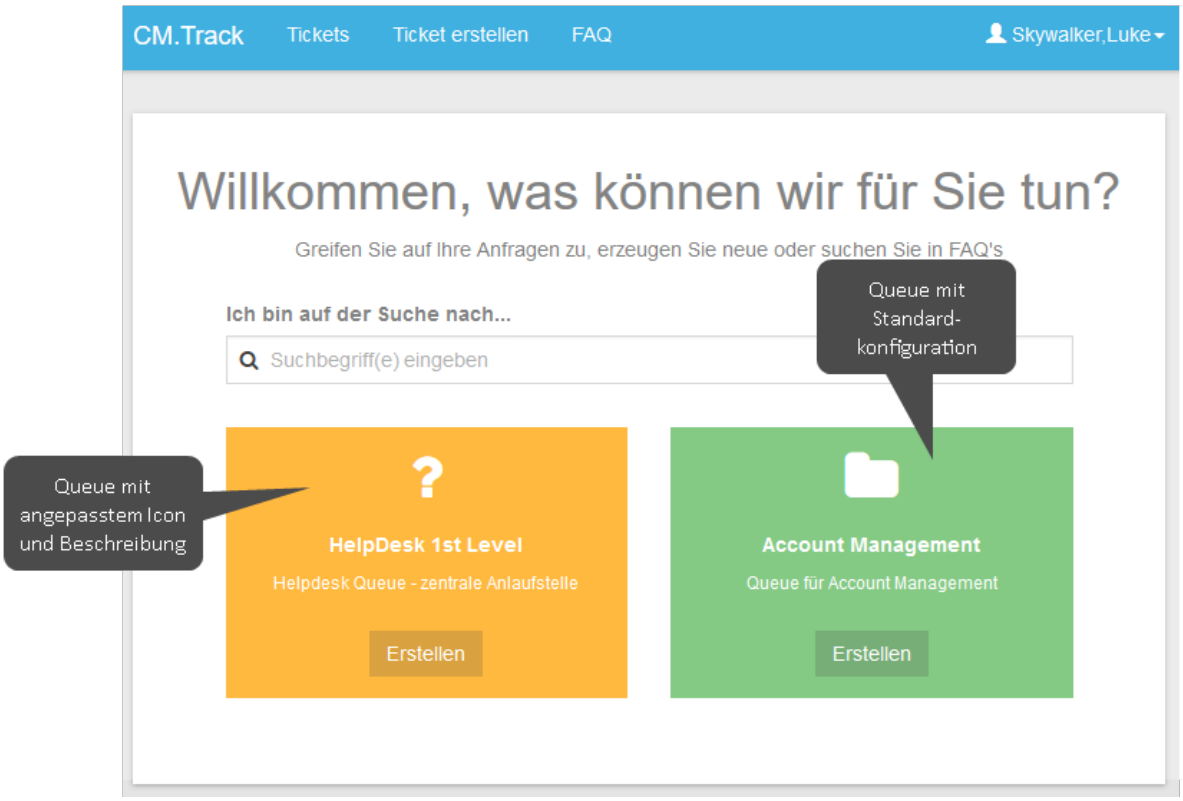


Abbildung 239: *Startseite in CM/Track*

Zum Konfigurieren der Startseite des Kundenportals, CM/Track, müssen Sie die Datei `config.json` anpassen.

Wenn Sie das CM-System von einer Version unter CM 6.11.1.1 aktualisiert haben, müssen Sie den Abschnitt für die Startseite manuell zur Datei `config.json` hinzufügen. In CM-Versionen 6.11.1.1 und höher ist ein beispielhafter Abschnitt für die Startseite in der Vorlage *TrackV2* enthalten.

Der folgende Code zeigt ein Beispiel für `config.json` mit einem Abschnitt für die Startseite.



Abbildung 240: *Beispiel für Datei config.json mit Abschnitt für die Startseite*

Es stehen folgende Parameter zur Verfügung:

- `@name`: technischer Name der Queue und somit Referenz auf die Queue
- `description`: Beschreibung der Queue, kann entweder ein festgelegter String ohne Lokalisierung oder ein Schlüssel aus der Datei `localization_en.json` oder einer anderen Lokalisierungsdatei sein. (Im obigen Beispiel ist *custom_HD1L_description* ein Schlüssel, der in den Lokalisierungsdateien vorhanden sein muss.)
- `icon`: Name eines Icons aus der Icon-Sammlung FontAwesome (siehe http://fontawesome.io/icons/). Verwenden Sie folgende Syntax: fa-<Name des Icons>.

Wenn keine explizite Konfiguration angegeben ist, werden die Standardwerte verwendet (Beschreibung der Queue im Admin Tool und Ordnersymbol). Alle Queues, für die der Benutzer die Berechtigung zum Erstellen von Tickets hat, werden angezeigt.

Die über die Queue-Konfiguration hinausgehende Lokalisierung der Startseite erfolgt über eine kleine Anzahl an Schlüsseln in den Lokalisierungsdateien für jede konfigurierte Sprache, z. B. `localization_en.json`. Diese Schlüssel müssen zu der vorhandenen Konfiguration hinzugefügt werden. Nur in neuen Konfigurationen (Stand 6.11.1.1), die auf der Vorlage *TrackV2* basieren, sind folgende Schlüssel standardmäßig enthalten.

```
"welcome_header" : "Hello, how can we help you?" ,
"welcome_subheader" : "Access your requests, submit a new one or search in FAQ's" ,
"welcome_search_label" : "Type search term(s)" ,
"searchbar_lbl" : "I am looking for..." ,
"queue_action" : "Create" ,
"jump_to_queues" : "More" ,
"custom_HD1_description": "The Helpdesk queue for general questions."
```

Code-Beispiel 23: *Ausschnitt aus localization_en.json mit den Standardwerten für den auf der Startseite angezeigten Text und der für die Queue-Beschreibung erstellten benutzerdefinierten Bezeichnung*

Die obige Konfiguration erzeugt die obige Startseite für einen Benutzer mit Zugriff auf zwei Queues. Beachten Sie, dass beide Queues angezeigt werden, obwohl nur eine Queue explizit konfiguriert wurde.

C.10.3.3 Konfigurieren der Anmeldeseite und anderer öffentlicher Seiten

Um eine öffentliche CM/Track-Seite (die Seite zum Anmelden, die Seiten zum Ändern des Passworts) anzupassen, müssen Sie eine Datei `public.json` zur Konfiguration Ihrer CM/Track-Clients hinzufügen. Die Datei `public.json` enthält Lokalisierungen und die Einstellung bezüglich der Funktion zum Zurücksetzen des Passworts. Die Lokalisierungen befinden sich alle im JSON-Objekt `signin`, `i18n` (Letzteres steht für Internationalisierung).

Beispiel:

```
{
   "signin": {
      "i18n" : {
         "en" : {
            "nav_track_brand":"CM/Track",
            "password_change_header": "Change password",
            "password_change_description": "You would like to change your password.
             Please fill the required fields.",
            "password_change_old": "Old password",
            "password_change_new": "New password",
            "password_change_confirm": "Confirm password",
            "password_change_submit": "Change password",
            "password_change_cancel": "Cancel",
            "password_change_changed": "Your password was successfully changed!",
```

```
        "password_old_not_match": "Password does not match the old password",
        "password_confirm_not_match": "Password does not match the confirmed
         password",
        "password_reset_header": "Change your password?",
        "password_reset_header_change": "Change password",
        "password_reset_hint": "Enter your user name and we\"ll help you reset
         your password. ",
        "password_reset_no_user": "User does not exist",
        "password_reset_mail_or_login": "User name",
        "password_reset_new": "New password",
        "password_reset_confirm": "Confirm password",
        "password_reset_set_new": "Set new password",
        "password_reset_submit": "Continue",
        "password_reset_cancel": "Cancel",
        "password_reset_sent": "An email was sent to your account. Please follow
         the instructions in the email.",
        "password_reset_changed": "Your password was successfully changed!",
        "password_confirm_not_match": "Password does not match the confirmed
         password",
        "reset_code_no_match": "Reset code does not match to any user",
        "reset_code_expired": "Reset code expired",
        "signin_header": "Please sign in",
        "signin_submit": "Sign in",
        "signin_forgotYourPassword": "Do not remember your password?",
        "signin_dontHaveAccountYet": "Don\"t have an account yet?",
        "signin_register_link": "Please register here",
        "signin_username_label": "Username",
        "signin_username_placeholder": "Username",
        "signin_password_label": "Password",
        "signin_password_placeholder": "Password",
        "signin_rememberme_label": "Remember me",
        "signin_error_signin_header": "Login failed",
        "signin_error_signin_text": "Please recheck your username and password
         and try again",
        "signin_error_signout_header": "Logout operation failed",
        "signin_error_signout_text": "Server responded with an error message
         while trying to signout. Please try again later.",
        "signin_error_forbidden_header": "The requested operation is forbidden",
        "signin_error_forbidden_text": "You\"re not allowed to perform the
         requested operation",
        "signin_error_serverfailed_header": "Unexpected server error",
        "signin_error_serverfailed_text": "An internal server error occurred and
         your request couldn’t be completed. Please try again",
        "signin_error_unauthorized_header": "Unauthorized access",
        "signin_error_unauthorized_text": "",
        "signin_success_signout_header": "You successfully signed out",
        "signin_success_signout_text": ""
      },
      "de" : {
        "nav_track_brand": "CM/Track",
        "password_change_header": "Passwort ändern",
        "password_change_description": "Sie möchten Ihr Passwort ändern. Füllen
         Sie dazu bitte die erforderlichen Felder aus.",
```

```
"password_change_old": "Altes Passwort",
"password_change_new": "Neues Passwort",
"password_change_confirm": "Passwort bestätigen",
"password_change_submit": "Passwort ändern",
"password_change_cancel": "Abbrechen",
"password_change_changed": "Ihr Passwort wurde erfolgreich geändert!",
"password_old_not_match": "Kennwort stimmt nicht mit altem Kennwort
 überein",
"password_confirm_not_match": "Passwortbestätigung stimmt nicht
 überein",
"password_reset_header": "Passwort vergessen?",
"password_reset_header_change": "Passwort ändern",
"password_reset_hint": "Geben Sie Ihren Benutzernamen ein, um Ihr
 Passwort zurückzusetzen. ",
"password_reset_no_user": "Benutzer nicht vorhanden",
"password_reset_mail_or_login": "Benutzername",
"password_reset_new": "Neues Passwort",
"password_reset_confirm": "Passwort bestätigen",
"password_reset_set_new": "Neues Passwort festlegen",
"password_reset_submit": "Weiter",
"password_reset_cancel": "Abbrechen",
"password_reset_sent": "Es wurde eine E-Mail an Ihr Konto gesendet.
 Bitte folgen Sie den Anweisungen in der E-Mail.",
"password_reset_changed": "Ihr Passwort wurde erfolgreich geändert!",
"password_confirm_not_match": "Passwortbestätigung stimmt nicht
 überein",
"reset_code_no_match": "Code zum Zurücksetzen passt zu keinem Benutzer",
"reset_code_expired": "Code zum Zurücksetzen abgelaufen",
"signin_header": "Bitte melden Sie sich an.",
"signin_submit": "Anmelden",
"signin_forgotYourPassword": "Sie haben Ihr Passwort vergessen?",
"signin_dontHaveAccountYet": "Sie haben noch keinen Zugang?",
"signin_register_link": "Bitte registrieren Sie sich hier.",
"signin_username_label": "Benutzername",
"signin_username_placeholder": "Benutzername",
"signin_password_label": "Passwort",
"signin_password_placeholder": "Passwort",
"signin_rememberme_label": "Anmeldedaten merken",
"signin_error_signin_header": "Anmeldung fehlgeschlagen",
"signin_error_signin_text": " Bitte überprüfen Sie Benutzername und
 Passwort und versuchen Sie es erneut.",
"signin_error_signout_header": "Abmeldung fehlgeschlagen",
"signin_error_signout_text": "Ihr Abmeldeversuch hat beim Server eine
 Fehlermeldung hervorgerufen. Bitte versuchen Sie es zu einem späteren
 Zeitpunkt erneut.",
"signin_error_forbidden_header": "Die angeforderte Operation ist nicht
 erlaubt",
"signin_error_forbidden_text": "Sie sind nicht berechtigt, die
 angeforderte Operation durchzuführen",
"signin_error_serverfailed_header": "Unerwarteter Serverfehler",
```

```
            "signin_error_serverfailed_text": "Es ist ein interner Serverfehler
             aufgetreten. Bitte versuchen Sie es zu einem späteren Zeitpunkt
             erneut",
            "signin_error_unauthorized_header": "Unberechtigter Zugriff",
            "signin_error_unauthorized_text": "",
            "signin_success_signout_header": "Sie haben sich erfolgreich
             abgemeldet",
            "signin_success_signout_text": ""
          }
       }
    }
}
```

Code-Beispiel 24: *Beispiel einer Datei public.json für die Konfiguration des CM-REST-Clients (dieses Beispiel ist auch im Admin Tool enthalten)*

Um Wörter und/oder Sätze zu ändern, die auf öffentlichen Seiten angezeigt werden, editieren Sie die Datei `public.json` im Admin Tool und speichern Sie sie. Die Änderungen sind direkt auf der GUI sichtbar. Siehe auch folgendes Beispiel.

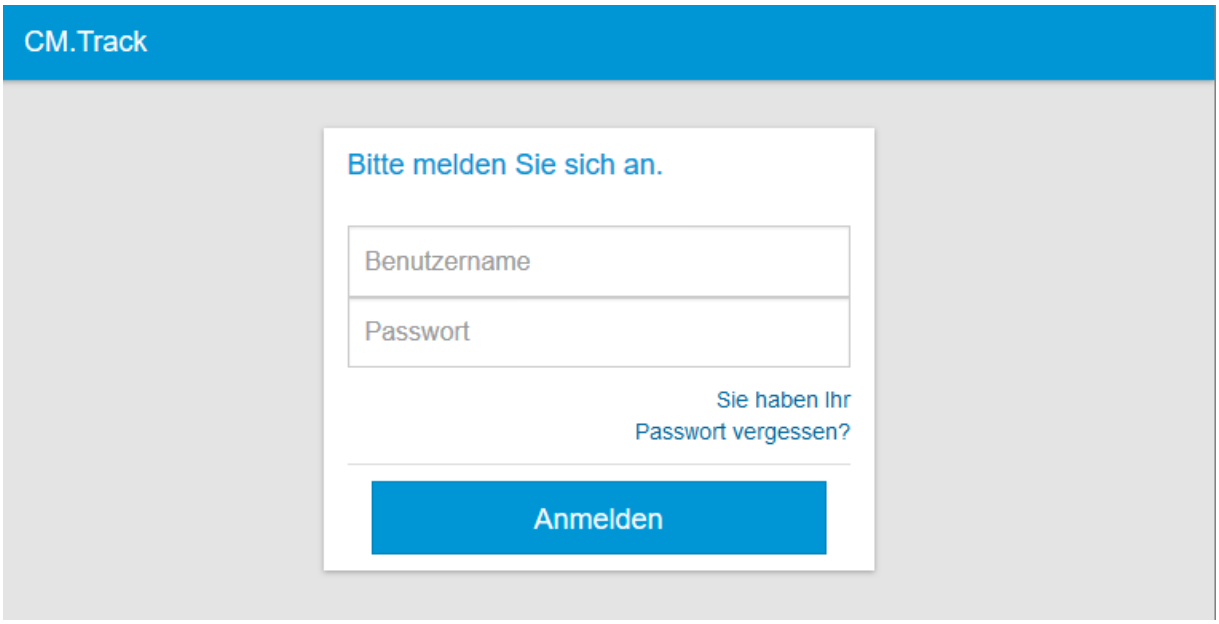


Abbildung 241: *Die CM/Track-Anmeldeseite im Standardformat*

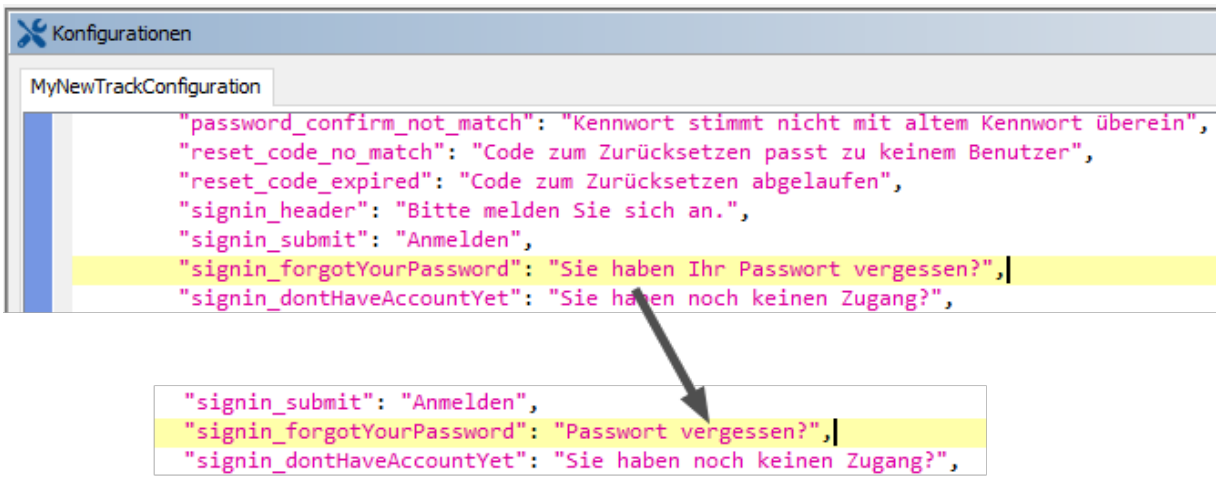


Abbildung 242: *Einfaches Editieren der Datei public.json*

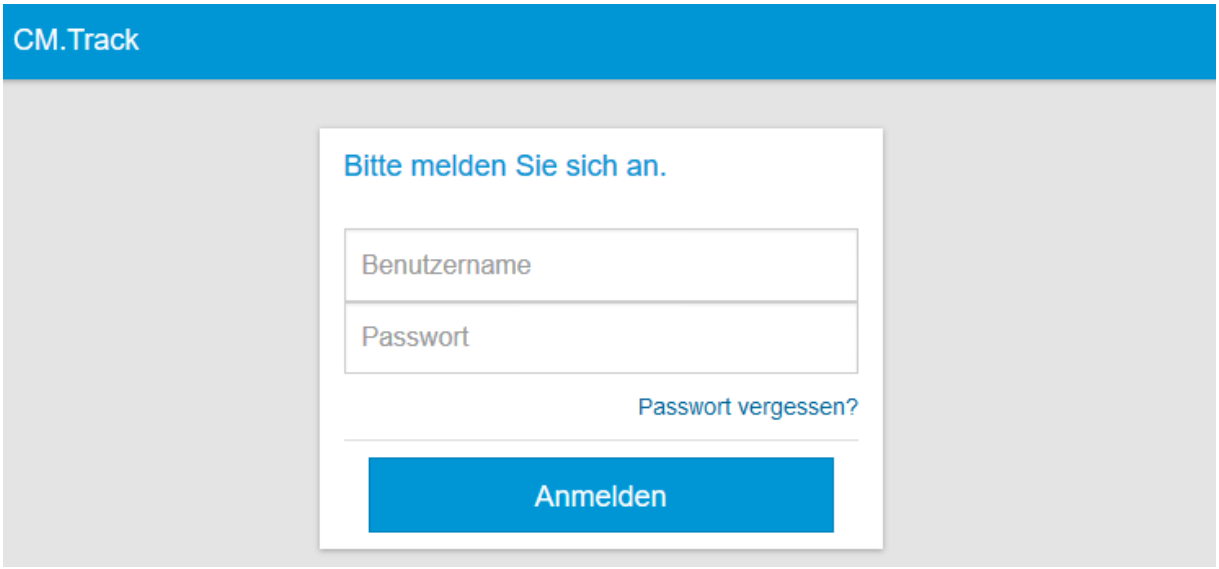


Abbildung 243: *Die CM/Track-Anmeldeseite leicht angepasst (ein Satz geändert)*

C.10.3.4 Konfiguration der Funktionen zum Zurücksetzen und Ändern des Passworts

Die Funktionen zum Zurücksetzen des Passworts auf der Anmeldeseite und Ändern des Passworts nach dem Anmelden können in den entsprechenden Konfigurationsdateien aktiviert bzw. deaktiviert werden.

config.json

In der Datei `config.json` können Sie festlegen, ob der Button *Password ändern* nach dem Anmelden in CM/Track angezeigt wird. Wenn `disablePasswordChange` auf „true“ gesetzt ist, kann der Benutzer das Passwort nicht ändern. Wenn „false“ gesetzt ist, ist das Ändern des Passworts möglich.

```
"disablePasswordChange": true,
```

Code-Beispiel 25: *Ausschnitt aus config.json zum Deaktivieren der Funktion zum Ändern des Passworts*

Standardmäßig, d. h. wenn der Parameter `disablePasswordChange` nicht explizit definiert wurde, ist das Menüelement *Passwort ändern* verfügbar.

public.json

In der Datei `public.json` können Sie festlegen, ob der Link *Password vergessen?* auf der Anmeldeseite angezeigt wird. Wenn `disablePasswordReset` auf „true“ gesetzt ist, kann der Benutzer das Passwort nicht zurücksetzen. Wenn „false“ gesetzt ist, ist das Zurücksetzen des Passworts möglich.

```
"disablePasswordReset": true,
```

Code-Beispiel 26: *Ausschnitt aus public.json zum Deaktivieren der Funktion zum Zurücksetzen des Passworts*

Standardmäßig, d. h. wenn der Parameter `disablePasswordReset` nicht explizit definiert wurde, wird der Link *Passwort vergessen?* angezeigt.

C.10.4 Erstellen von neuen Konfigurationsseiten

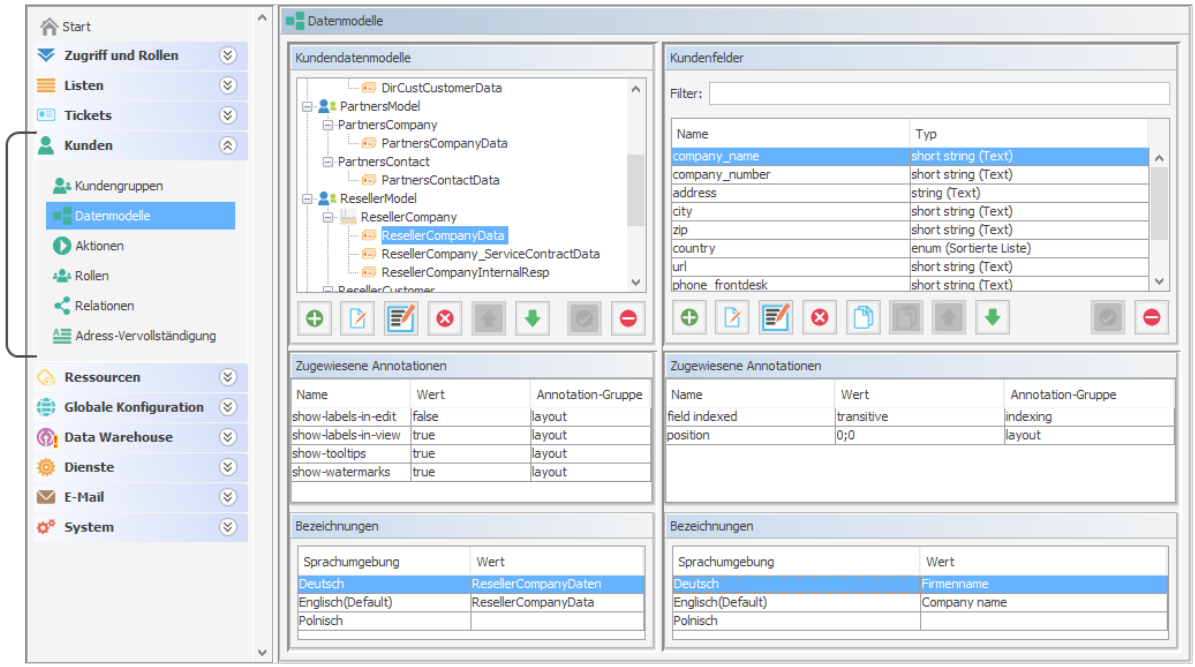
Mit dem Button *Neue Datei erstellen* können Sie eine neue JSON-Datei zur Konfiguration hinzufügen. Es gibt zwei Anwendungsfälle für diese Funktionalität:

- Sie möchten eine neue Lokalisierungsdatei hinzufügen, da Sie eine Sprache zur Systemkonfiguration hinzugefügt haben. Benennen Sie die Datei `localization_<Browser Locale>`. Sie können zum Beispiel für die französischen Übersetzungen eine Datei mit dem Namen `localization_fr` hinzufügen. Erstellen Sie die Datei basierend auf den vorhandenen Lokalisierungsdateien.
- Sie möchten eine neue Datei hinzufügen, die für ein kundenspezifisches Projekt erforderlich ist. Dies ist ein Anwendungsfall für erfahrene CM-Consultants und -Entwickler.

C.10.5 Verhalten bei Systeminstallationen und -aktualisierungen

Bei einer neuen Installation der ConSol CM-Version 6.11.0.5 oder bei einem Update auf diese Version werden die Vorlagen/Beispiele für die Konfiguration der REST-Client-GUI installiert. Alle Konfigurationen, einschließlich der Standardkonfiguration (*TrackV2*), müssen bei Bedarf manuell hinzugefügt werden.

D - Abschnitt Kundendatenmodell



Ab Version 6.9 bietet ConSol CM eine flexible und leistungsstarke Kundenverwaltung basierend auf FlexCDM, dem flexiblen Kundendatenmodell.

In diesem Abschnitt lernen Sie, wie Sie FlexCDM, das flexible ConSol CM-Kundendatenmodell, und alles, was dazu gehört, aufsetzen und pflegen. Es werden folgende Themen behandelt:

- Einführung in FlexCDM
- Kurze Einführung in die FlexCDM-spezifischen Funktionen des Web Clients
- Einrichten des Kundendatenmodells
- Verwaltung von Kundenfeldern und GUI-Design für Kundendaten
- Templates für Kundendaten
- Verwaltung von Kundengruppen
- Kundenrollen
- Kundenrelationen
- Action Framework - Kundenaktionen
- Adress-Vervollständigung

D.1 Einführung in FlexCDM

In diesem Kapitel werden folgende Themen behandelt:

D.1.1 FlexCDM auf einen Blick 369

D.1.2 Einführung in FlexCDM-Objekte 372

D.1.3 Verwaltung von FlexCDM-Objekten mit dem Admin Tool 374

Da das Kundendatenmodell ***FlexCDM***, das mit Version 6.9 von ConSol CM eingeführt wurde, sehr komplex und leistungsstark ist, gibt es ein eigenes Einleitungskapitel, in dem alle Details erklärt werden.

D.1.1 FlexCDM auf einen Blick

D.1.1.1 Flexibles Kundendatenmodell

Wie der Name *FlexCDM* nahe legt, bietet das ConSol CM-Kundendatenmodell ein hohes Maß an Flexibilität. Es können mehrere **Kundengruppen** definiert werden, die ihre eigenen Datenmodelle haben.

In ConSol CM sprechen wir von **Kunden**, um das allgemeine CM-Objekt zu beschreiben. Dies kann entweder eine Firma oder ein Kontakt sein. Eine **Firma** ist ein Objekt auf der Firmenstufe, bei dem es sich in den meisten Fällen um eine echte Firma, eine Filiale, eine Abteilung oder eine andere übergeordnete Organisationseinheit handelt. Es kann aber auch eine Sammlung von Produkten, Maschinen oder ein beliebiges anderes Objekt sein, das untergeordnete Objekte hat. Ein **Kontakt** ist ein Objekt auf der Kontaktstufe, d. h. auf der unteren Stufe des Kundenmodells. Ein Kontakt ist häufig eine echte Person; es kann sich aber auch um ein Produkt, eine Maschine oder ein anderes Objekt handeln.

Kontakte und Firmen können als Kunde für ein Ticket festgelegt werden.

Es gibt verschiedene Vorgehensweisen zum Konfigurieren von Datenmodellen für Kundengruppen. Innerhalb einer Kundengruppe kann es Folgendes geben...

- eine *Kontakt*- und eine *Firmenstufe*:
  In diesem Fall sprechen wir von einem **zweistufigen Kundendatenmodell** (in dem eine Firma mehrere Kontakte enthalten kann)
- nur eine *Kontaktstufe* oder nur eine *Firmenstufe*:
  Dann sprechen wir von einem **einstufigen Kundendatenmodell**



Abbildung 244: *Arten von Kundendatenmodellen in ConSol CM*

Zum Beispiel könnten Sie Ihre Kunden in zwei Kundengruppen einteilen:

1. **Händler**
   Mit einer Kontakt- und einer Firmenstufe.
2. **Endkunden**
   Nur mit einer Kontaktstufe.

Sie können so viele Kundendatenmodelle konfigurieren, wie erforderlich. Jedes Kundendatenmodell kann für eine oder mehrere Kundengruppen verwendet werden.

Ein Kundendatenmodell umfasst das allgemeine Modell, d. h. die Stufen (Kontakt und Firma oder nur Kontakt/Firma) und alle Datenfelder für alle Komponenten (z. B. Name, Adresse und Telefonnummer für eine Firma, Name, E-Mail und Raumnummer für einen Kontakt).

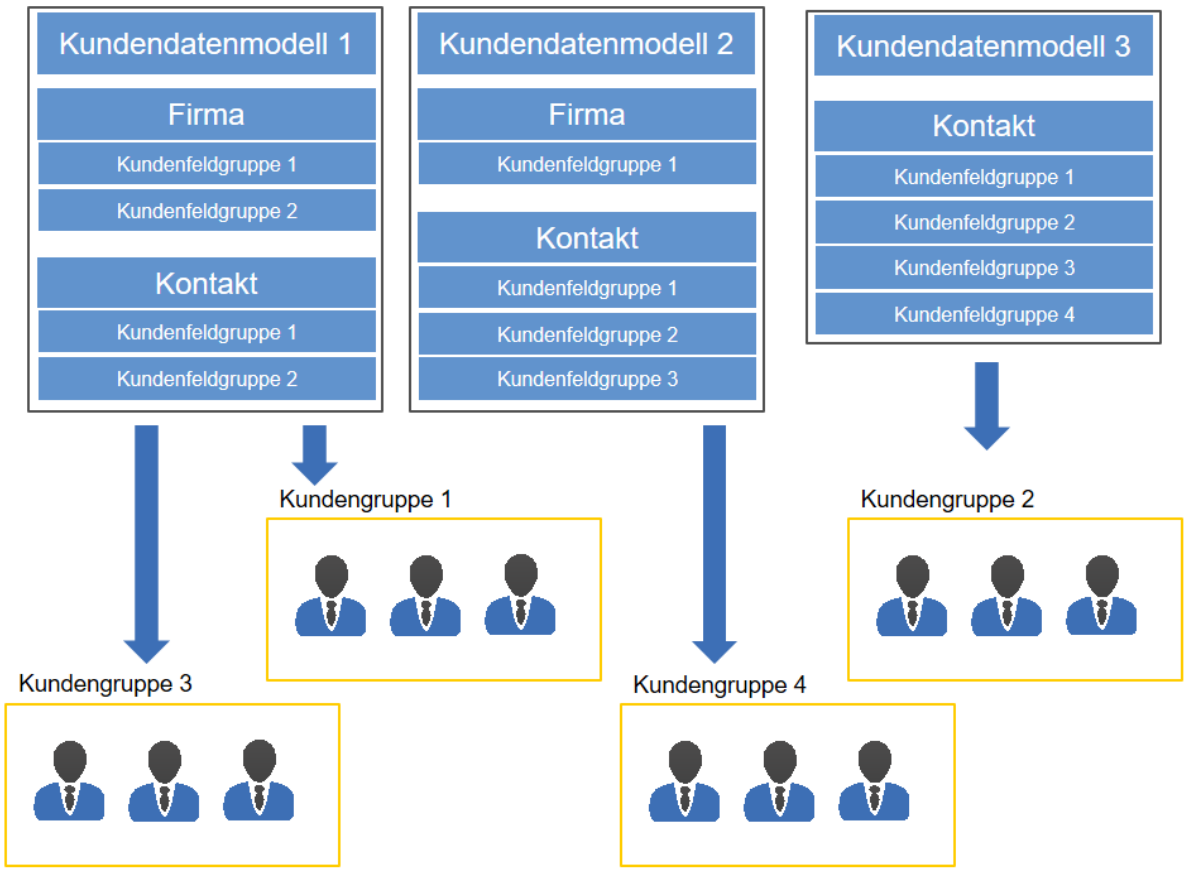


Abbildung 245: *ConSol CM FlexCDM - Grundprinzip*

Bei einem zweistufigen Kundendatenmodell:

Die Begriffe *Firma* und *Kontakt* werden verwendet, um die Stufe eines Objekts in der Hierarchie von FlexCDM zu beschreiben. Ein Objekt vom Typ *Firma* muss nicht unbedingt eine echte Firma darstellen, es kann auch ein Ort mit mehreren Maschinen (Kontakten), eine Organisation mit mehreren Filialen (Kontakten) oder sogar eine technische Einheit (z. B. ein Schiff) sein, in der es mehrere Kontakte gibt. Genauso muss ein Objekt des Typs *Kontakt* nicht unbedingt eine Person darstellen. Es kann sich auch um einen Ort, eine Maschine oder irgendetwas anderes handeln, das die Kontaktstufe darstellt.

Bei einem einstufigen Kundendatenmodell:

Die Kundenobjekte in einem einstufigen Kundenmodell sind entweder vom Typ *Kontakt* oder vom Typ *Firma*.

Die Stufen und die Namen aller Komponenten der in Ihrem ConSol CM-System verwalteten Kunden hängen einzig und alleine von der Konfiguration von FlexCDM ab.

Mit FlexCDM können Sie unterschiedliche Bereiche abbilden, die eine bestimmte Kundengruppe und die zugehörigen Daten und Prozesse umfassen.

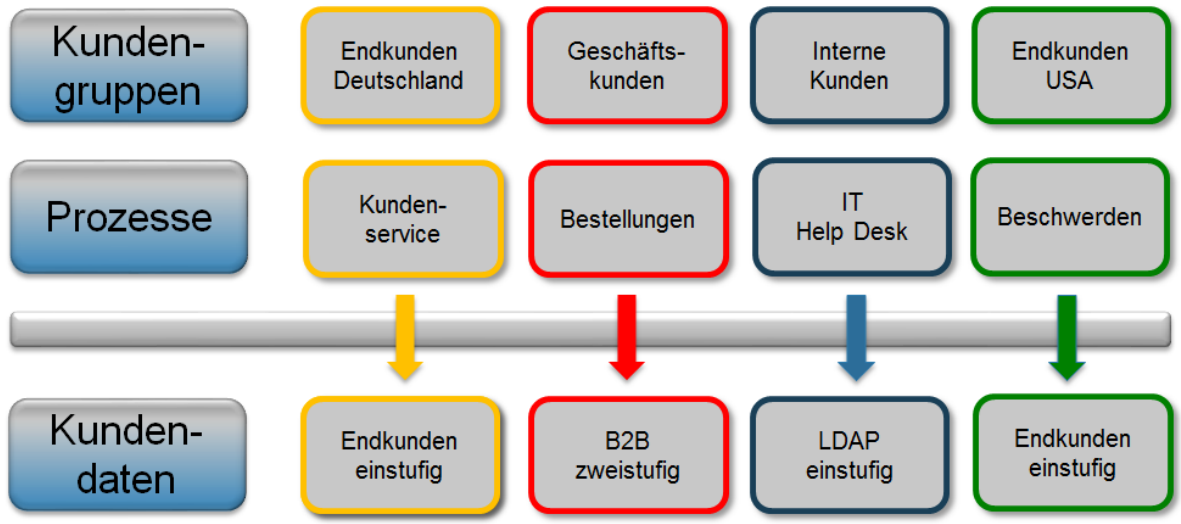


Abbildung 246: *ConSol CM FlexCDM - Kundendatenmodell*

Eine detaillierte Beschreibung der Kundenverwaltung finden Sie in Einrichten des Kundendatenmodells.

D.1.2 Einführung in FlexCDM-Objekte

Dieser Abschnitt enthält eine Übersicht über die Objekte, die für FlexCDM relevant sind.

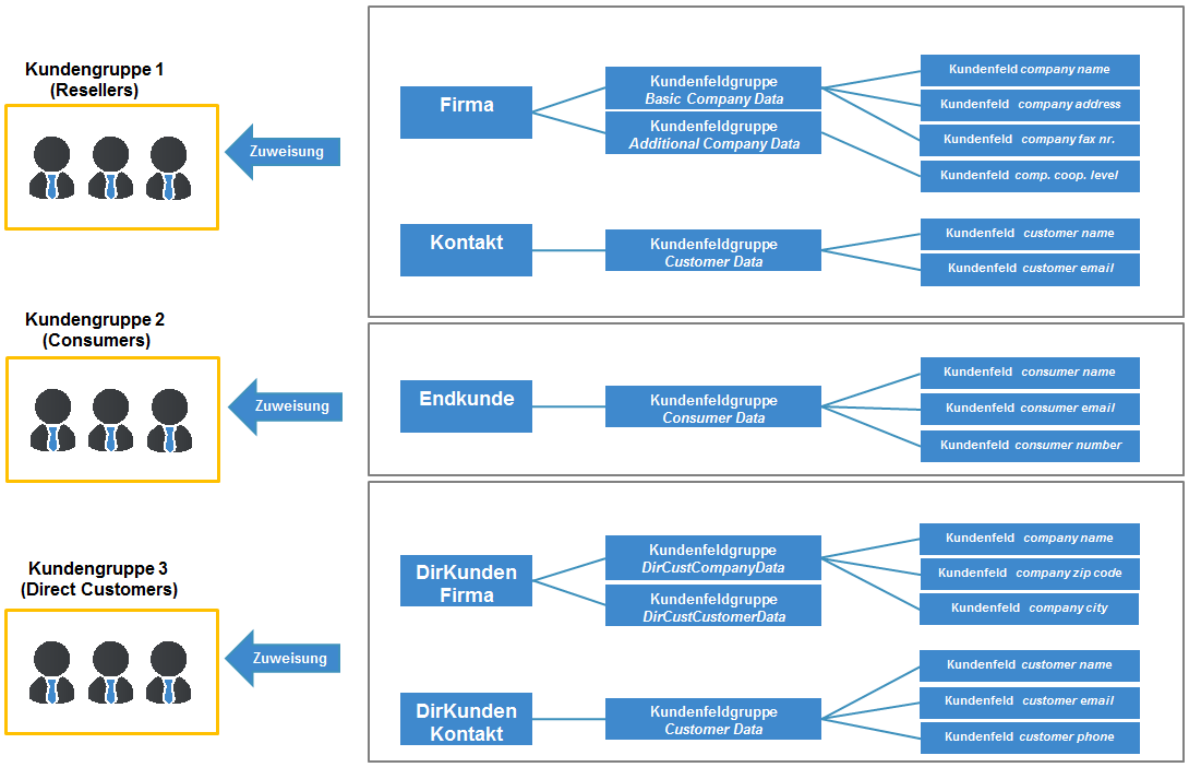


Abbildung 247: *ConSol CM FlexCDM - Beispielkonfiguration*

D.1.2.1 Wichtige Begriffe

Im Folgenden sind einige wichtige Begriffe von FlexCDM erklärt:

- **Kunde**
  Allgemeiner Begriff für Kundenobjekte, kann vom Typ *Kontakt* oder *Firma* sein.
- **Firma**
  Kundenobjekt vom Typ *Firma*, Firmenstufe.
- **Kontakt**
  Kundenobjekt vom Typ *Kontakt*, Kontaktstufe.
- **Kundengruppe**
  Eine Gruppe von Kunden mit einem bestimmten Kundendatenmodell. Die Bearbeiterberechtigungen (über Rollen) für die Kundenverwaltung werden für die einzelnen Kundengruppen vergeben.
- **Kundenobjekt**
  Ein Objekt im Kundendatenmodell. Der Objekttyp kann *Kontakt* (häufig eine Person) oder *Firma* sein. Das technische Äquivalent (Groovy) ist ein Objekt der Klasse `Unit`.

- **Kundenobjektdefinition**
  Alle Definitionen gehören zu dieser Unit. Bei einer Firma sind das z. B. alle Kundenfeldgruppen, alle Gruppenannotationen und die Zuweisung aller Templates (für die Anzeige von Kundendaten im Web Client, nicht zu verwechseln mit den anderen Templates in ConSol CM!)
- **Kundenfeldgruppe**
  Eine Gruppe besteht aus einem oder mehreren Datenfeldern (Kundenfeldern), analog zu den Ticketfeldern für die Ticketdaten. Eine Kundenfeldgruppe kann ein- oder ausgeblendet werden oder als Tab im Bereich für Detaildaten angezeigt werden.
- **Kundenfeld**
  Ein einzelnes Datenfeld (Datentypen wie bei Ticketfeldern), das Kundendaten enthalten kann, analog zu Ticketfeldern bei der Definition von Datenfeldern für Ticketdaten.
- **Kundendatenmodell**
  Das ganze Datenmodell kann einer Kundengruppe zugewiesen werden.
  Das Datenmodell kann:
  - nur eine Stufe haben (nur Kontakt oder nur Firma)
  - zwei Stufen haben (Firma und Kontakt)

  Das Kundendatenmodell enthält zudem die Definitionen von Kundenfeldgruppen und Kundenfeldern.
- **Kundenrelationen**
  Relationen zwischen einer Firma und Kontakten oder zwischen Firmen oder zwischen Kontakten. Alle Relationen gemeinsam bilden das Kundenrelationennetz.
- **Action Framework**
  Eine Reihe von CM-Modulen, die eine durch Events ausgelöste und vom Workflow unabhängige Programmierung ermöglichen. Die im Kontext von FlexCDM relevanten *Kundenaktionen* (auch *Unit-Aktionen* genannt) gehören zum Action Framework.

D.1.3 Verwaltung von FlexCDM-Objekten mit dem Admin Tool

Im Admin Tool erreichen Sie den Bildschirm mit den FlexCDM-Konfigurationsoptionen, indem Sie die Navigationsgruppe *Kunden* öffnen.

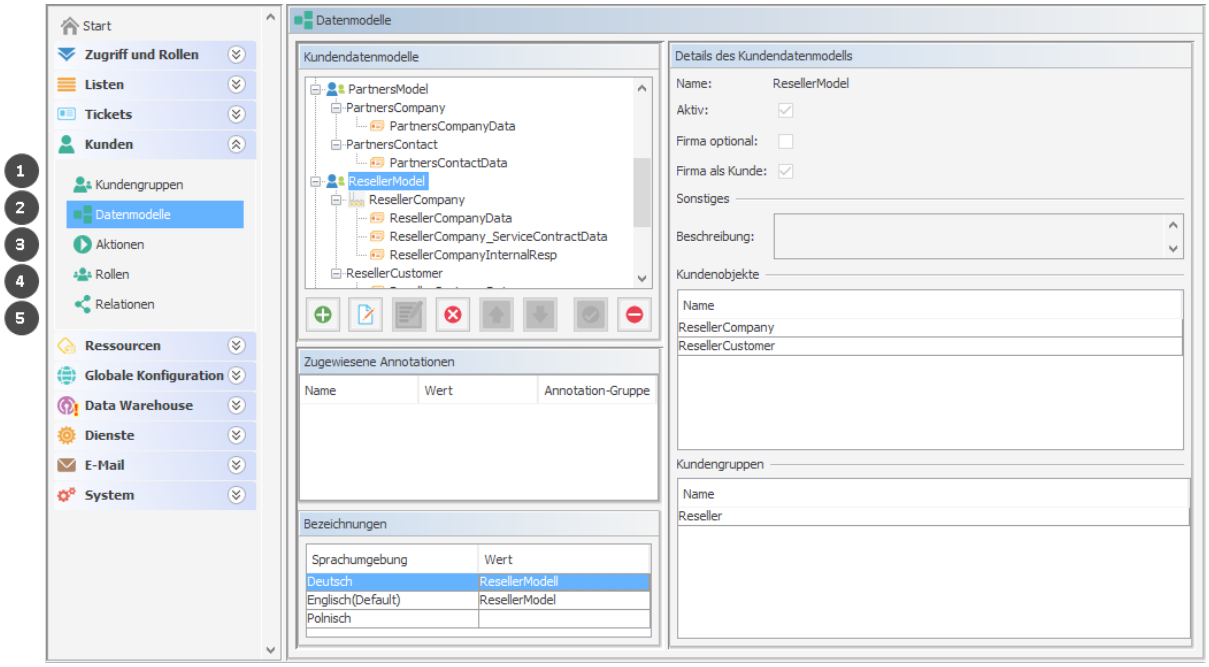


Abbildung 248: *ConSol CM Admin Tool - Für FlexCDM relevante Navigationselemente des Bereichs Kunden*

Folgende Navigationselemente werden für die Definition und Verwaltung des Kundendatenmodells verwendet:

- **Kundengruppen (1)**
  Zuweisung der folgenden Komponenten, die zuvor definiert wurden, zu einer Kundengruppe (siehe Abschnitt Verwaltung von Kundengruppen).
  - Kundendatenmodell
  - Kundenaktionen
  - Suche-Aktionen
  - CM/Phone-Konfiguration, wenn CM/Phone aktiv ist
- **Datenmodelle (2)**
  Definition der Datenmodelle, d. h. Definition der Datenfelder für Kunden (d. h. Kontakte und Firmen) und GUI-Design (d. h. Anordnung der Datenfelder auf der GUI des Web Clients). Siehe Abschnitte Einrichten des Kundendatenmodells und Verwaltung von Kundenfeldern und GUI-Design für Kundendaten.
- **Aktionen (3)**
  Definition von Kundenaktionen, d. h. Kontaktaktionen und Firmenaktionen (siehe Abschnitt Action Framework - Kundenaktionen.

- **Rollen (4)**
  Definition von Kundenrollen, siehe Abschnitt Kundenrollen.
- **Relationen (5)**
  Definition von Kundenrelationen, die Referenzen zwischen Kundenobjekten (d. h. Kontakt und Firma) darstellen, siehe Abschnitt Kundenrelationen.
  (Relationen von Ressourcen zu Kunden müssen im Datenmodell des Ressourcenpools definiert werden. Sie können hier nicht konfiguriert werden.)

D.2 Kurze Einführung in die FlexCDM-spezifischen Funktionen des Web Clients

In diesem Kapitel werden folgende Themen behandelt:

D.2.1 Einleitung 376

D.2.2 Arbeiten mit FlexCDM im ConSol CM Web Client 377

D.2.1 Einleitung

Sie als Administrator fragen sich vielleicht, wieso das Administratorhandbuch eine Einleitung in die GUI des Web Clients enthält. Dies hat den Hintergrund, dass Sie bei der Arbeit mit dem Kundendatenmodell ***FlexCDM*** alle Auswirkungen Ihrer Administrationstätigkeiten kennen sollten. Und diese betreffen selbstverständlich auch den Web Client. In diesem Abschnitt werden wir daher die Rolle eines Bearbeiters einnehmen und einige Beispiele für die Arbeit mit dem Kundendatenmodell zeigen.

Alle Konfigurationsdetails, die Sie kennen müssen, um das Systemverhalten zu verstehen, werden in den entsprechenden Abschnitten des Handbuchs erklärt.

D.2.2 Arbeiten mit FlexCDM im ConSol CM Web Client

D.2.2.1 Beispiel 1: Auswählen der Kundengruppe

Sofern die Bearbeiter Zugangsberechtigungen für mehr als eine Kundengruppe haben, können sie mit dem *Kundengruppenfilter*, einer Drop-down-Liste im Hauptmenü, die **Kundengruppe auswählen**, die für bestimmte Vorgänge verwendet werden soll. Angezeigt wird jeweils der lokalisierte Name der Kundengruppe.

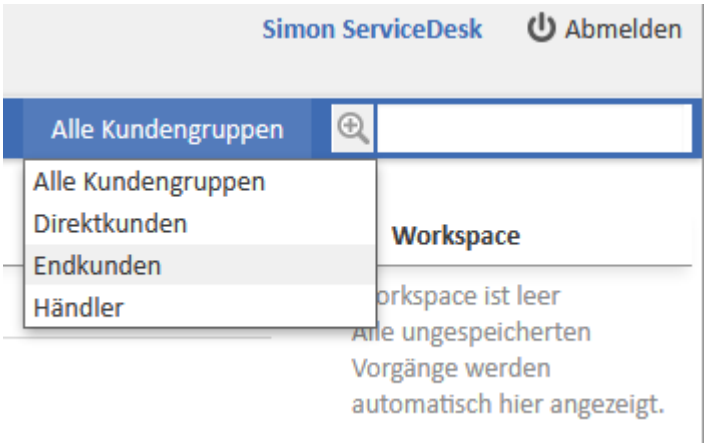


Abbildung 249: *ConSol CM Web Client - Auswählen einer Kundengruppe*

Die Auswahl wirkt sich auf folgende Aktionen aus:

- Die Schnellsuche sucht nur innerhalb der ausgewählten Kundengruppe.
- Bei der Detailsuche wird das Kriterium *Kundengruppe* nur angeboten, wenn im Drop-down-Menü *Alle Kundengruppen* ausgewählt ist. Andernfalls verwendet die Suche implizit nur die Daten der ausgewählten Kundengruppe.
- Bei der Detailsuche werden nur die Suchfelder der ausgewählten Kundengruppe angeboten.
- Bei der Erstellung eines Tickets wird die ausgewählte Kundengruppe (implizit) angeboten, wenn eine Firma und/oder ein Kontakt innerhalb des neuen Tickets erstellt werden soll.
- Ein Ticket kann nur in Queues erstellt werden, denen die ausgewählte Kundengruppe zugewiesen ist.
- In der Ticketliste sind nur Sichten verfügbar, die Tickets aus den Queues enthalten, denen die ausgewählte Kundengruppe zugewiesen ist.

Beachten Sie, dass die Liste der Kundengruppen im Kundengruppenfilter mit dem Seitenanpassungsattribut *hiddenCustomerGroups* beeinflusst werden kann. Details finden Sie im Abschnitt Seitenanpassung, customerGroupSelector.

D.2.2.2 Beispiel 2: Erstellen einer neuen Firma und eines neuen Kontakts

Wenn Bearbeiter Zugang zu mehreren Kundengruppen haben (und im Kundengruppenfilter *Alle Kundengruppen* ausgewählt haben, siehe Beispiel 1), können sie bei der Erstellung eines neuen Tickets die **Kundengruppe auswählen**, für die dabei ein Kontakt/eine Firma erstellt werden soll. Dies hängt zusätzlich von der ausgewählten Queue ab. Es sind nur Kundengruppen verfügbar, die der ausgewählten Queue zugewiesen sind. Wenn im Drop-down-Menü die Option *Alle Kundengruppen* ausgewählt ist, gibt es für jede Kundengruppe einen Tab mit den Kundendaten und der Bearbeiter kann die gewünschte Gruppe auswählen.

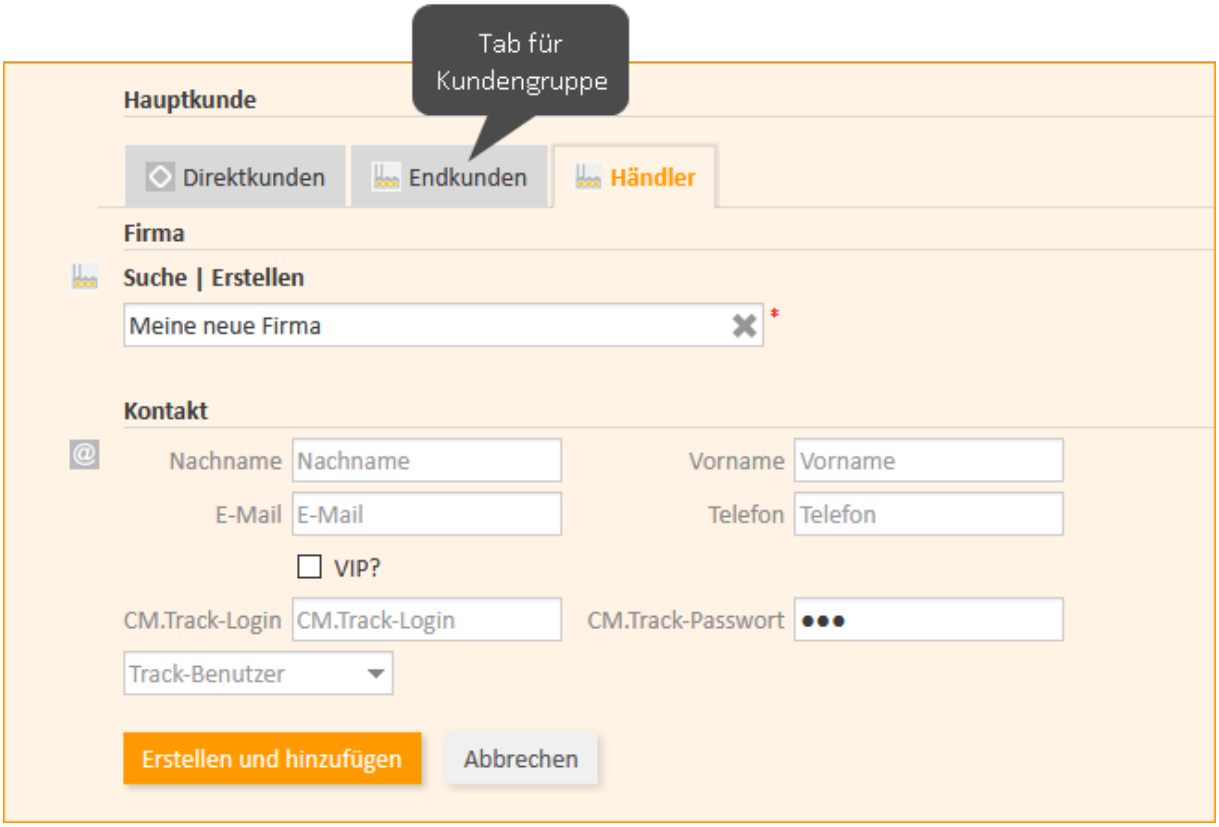


Abbildung 250: *ConSol CM Web Client - Erstellen einer neuen Firma innerhalb einer Kundengruppe*

> Wenn ein Bearbeiter hauptsächlich mit einer bestimmten Kundengruppe arbeitet, aber den Kundengruppenfilter nicht auf diese Gruppe setzen möchte, kann er in seinem Bearbeiterprofil die Standard-Kundengruppe setzen. Dann wird bei der Erstellung eines Tickets diese Kundengruppe geöffnet/angezeigt.

D.2.2.3 Beispiel 3: Verwenden der Firmen- und Kontaktseite

Bei zweistufigen Kundendatenmodellen (Firma und Kontakt) gibt es **separate Seiten für Firmen und Kontakte**. Auf beiden Seiten können Sie *Kommentare hinzufügen* und *Dateien anhängen*. Diese Vorgänge sind danach im Protokoll der Firmen- bzw. Kontaktseite sichtbar.

In jedem Kundendatenmodell können Icons für das Firmenobjekt und das Kontaktobjekt definiert werden, um die Benutzerfreundlichkeit zu erhöhen. Die Kopfzeile der Kontakt- und Firmenseite wird durch den lokalisierten Wert des entsprechenden Kundenobjekts definiert, siehe Abschnitt Einrichten des Kundendatenmodells, Name des Kundenobjekts.

Firmenseite

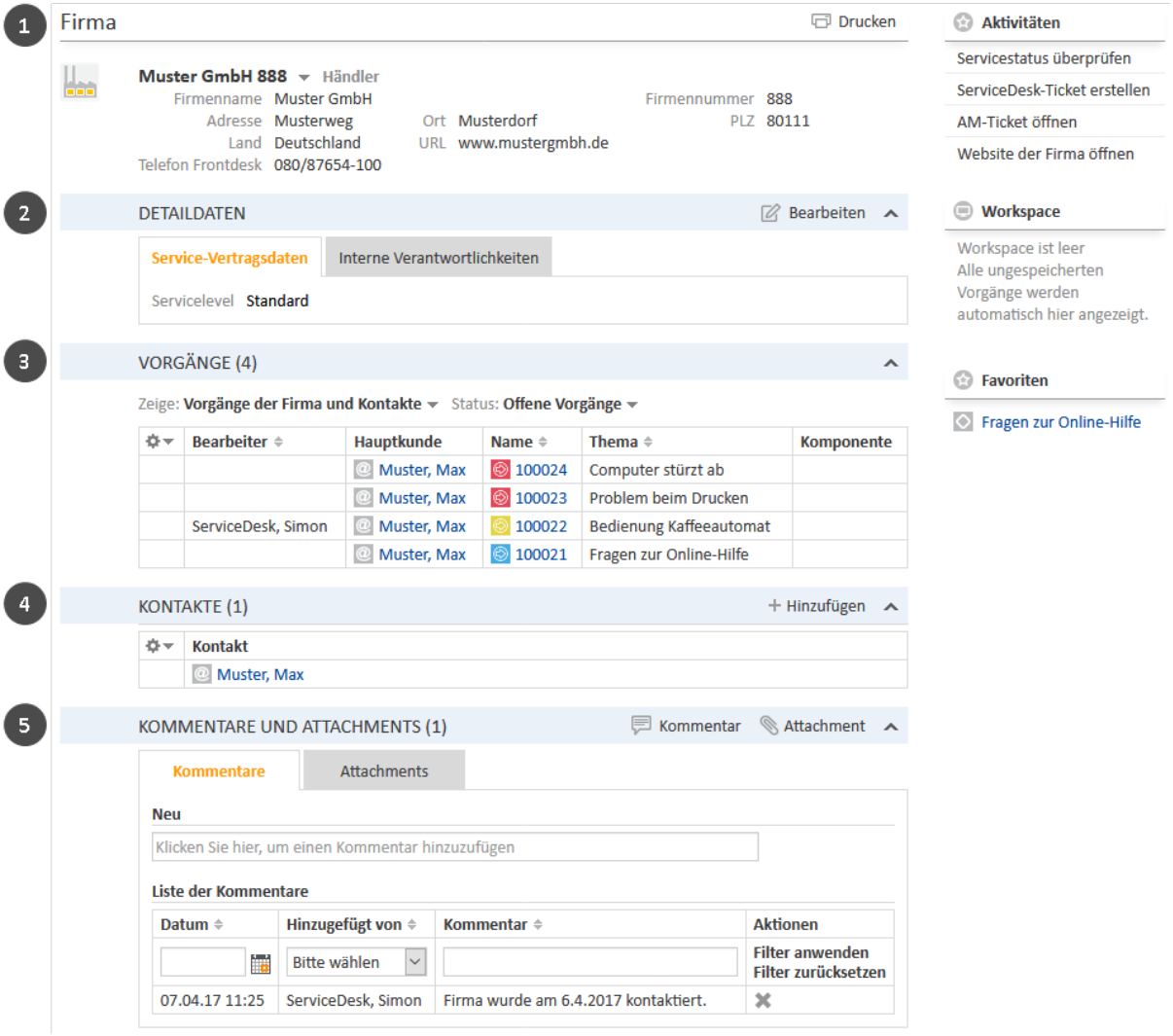


Abbildung 251: *ConSol CM Web Client - Oberer Abschnitt der Firmenseite*

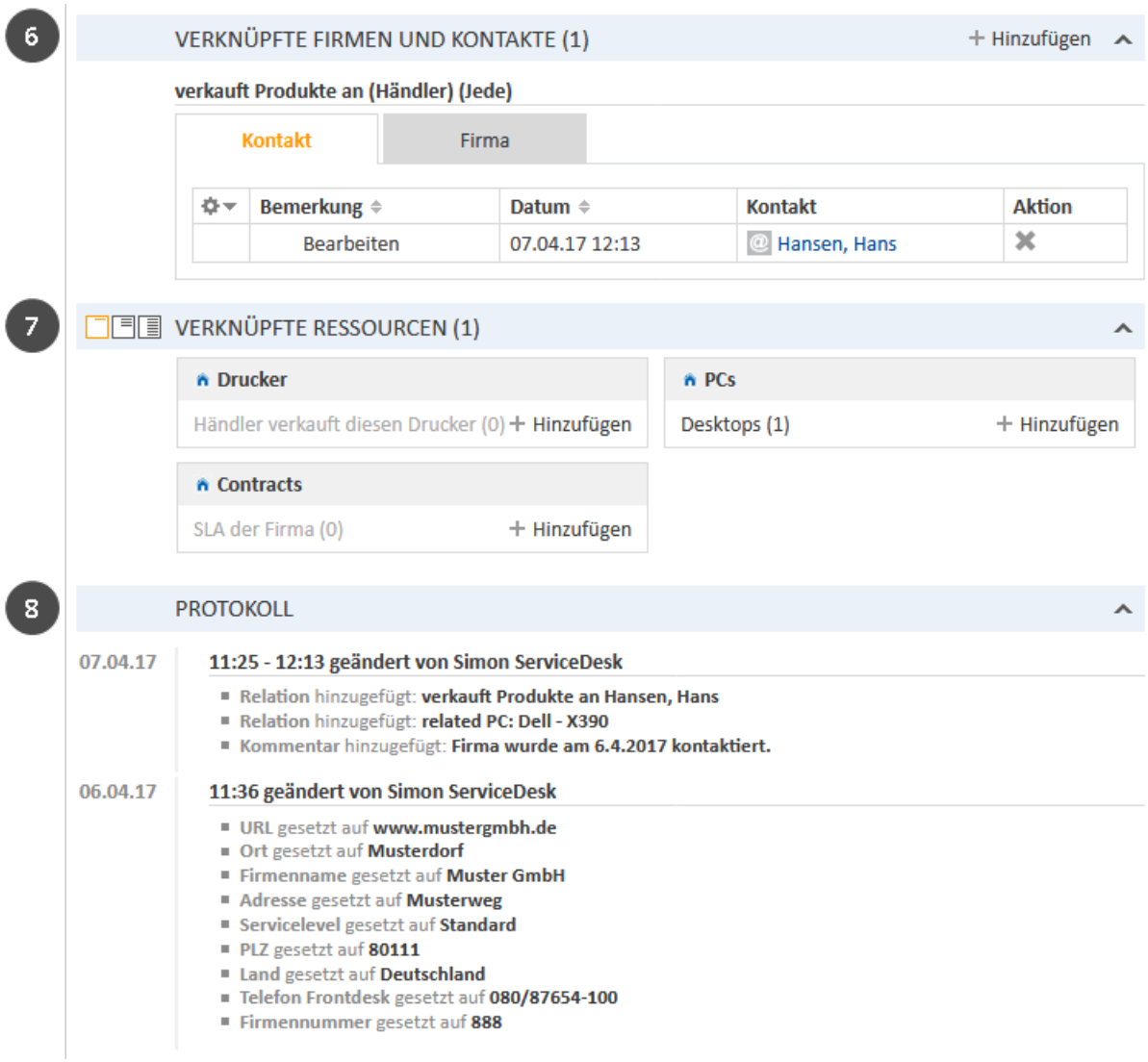


Abbildung 252: *ConSol CM Web Client - Unterer Abschnitt der Firmenseite*

Die *Firmenseite* hat folgende Bereiche:

- **Firmendaten (1, 2)**
  Firmendaten wie Adresse, Telefonnummer, Servicedaten (d. h. die Kundenfelder, die Sie beim Einrichten des Kundendatenmodells definiert haben). Die Firmendaten können entweder im Kopfbereich (1) stehen oder in einem oder mehreren Tabs im *Bereich für Detaildaten* (2).
  Der Firmenname wird mithilfe eines Kunden-Templates vom Typ *Ticket* dargestellt. Ist keines definiert, wird das Standard-Template verwendet. Details zu Kunden-Templates finden Sie im Abschnitt Templates für Kundendaten, speziell im Unterabschnitt Ticket.
- **Vorgänge (3)**
  In diesem Bereich sind alle Tickets der Firma und/oder ihrer Kontakte aufgeführt (siehe auch Beispiel 9: Verwenden der Ticketfilter auf der Firmen- oder Kontaktseite). Ab CM Version 6.9 ist es möglich, ein Ticket direkt einer Firma zuzuweisen. Im Kundendatenmodell muss die Option *Firma als Kunde* gesetzt werden, um diese Funktion zu aktivieren.

- **Kontakte (4)**
  In diesem Bereich werden alle Kontakte angezeigt, die zu dieser Firma gehören. Klicken Sie auf den Namen eines Kontakts, um die Kontaktseite zu öffnen.
- **Kommentare und Attachments (5)**
  Es gibt zwei Tabs:
  - **Kommentare**
    Hier werden alle Kommentare zu dieser Firma angezeigt.
  - **Attachments**
    Hier sind alle Attachments der Firma aufgeführt. Die Liste der Attachments kann nach Dateityp, Name, Beschreibung, Datum oder Bearbeiter gefiltert und sortiert werden.
- **Verknüpfte Firmen und Kontakte (6)**
  Hier sind alle Relationen aufgeführt, die zwischen der Firma und anderen Kunden (Firmen oder Kontakten) hergestellt wurden.
- **Verknüpfte Ressourcen (7)**
  Dieser Bereich wird nur angezeigt, wenn CM/Resource Pool im CM-System aktiviert ist und wenn mindestens eine Ressourcenrelation zu Firmen der entsprechenden Kundengruppe im Admin Tool konfiguriert ist. Hier sind alle mit dieser Firma verknüpften Ressourcen aufgeführt.
- **Protokoll (8)**
  In diesem Bereich sind alle mit diesem Firmenobjekt ausgeführten Aktionen aufgeführt, z. B. die Änderung eines Namens oder eines anderen Wertes in einem der Kundenfelder.

Eine detaillierte Erklärung über die Arbeit mit Firmen finden Sie im *ConSol CM Benutzerhandbuch*.

Kontaktseite

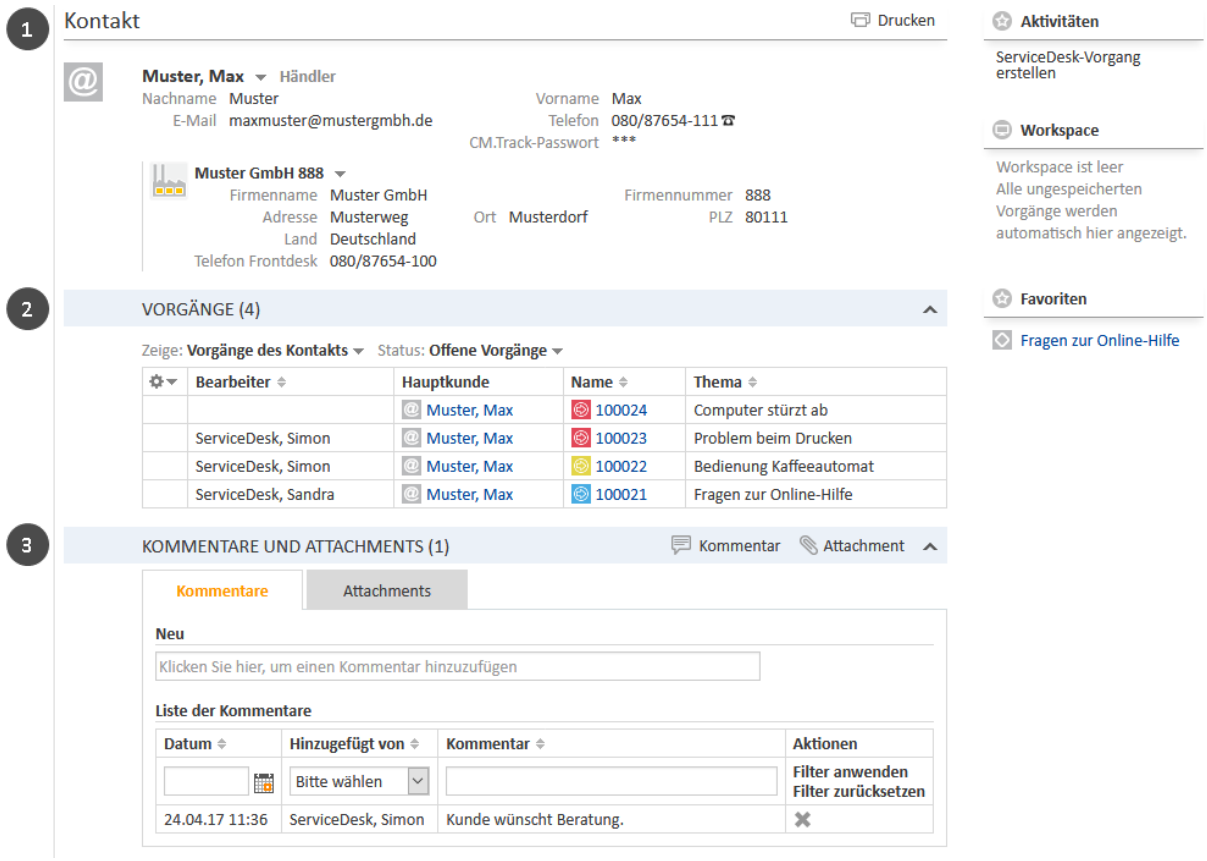


Abbildung 253: *ConSol CM Web Client - Oberer Abschnitt der Kontaktseite*



Abbildung 254: *ConSol CM Web Client - Unterer Abschnitt der Kontaktseite*

Die *Kontaktseite* hat folgende Bereiche:

- **Kontaktdaten (1)**
  Kontaktdaten wie Adresse, Telefonnummer, Servicedaten (d. h. die Kundenfelder, die Sie beim Einrichten des Kundendatenmodells definiert haben). Die Kontaktdaten können entweder im Kopfbereich stehen oder in einem oder mehreren Tabs im *Bereich für Detaildaten* (im Bild nicht dargestellt).
  Der Kontaktname wird mithilfe eines Kunden-Templates vom Typ *Ticket* dargestellt. Ist keines definiert, wird das Standard-Template verwendet. Details zu Kunden-Templates finden Sie im Abschnitt Templates für Kundendaten, speziell im Unterabschnitt Ticket.
- **Vorgänge (2)**
  In diesem Bereich sind alle Tickets des Kontakts und/oder seiner Firma aufgeführt (siehe auch Beispiel 9: Verwenden der Ticketfilter auf der Firmen- oder Kontaktseite).
- **Kommentare und Attachments (3)**
  Es gibt zwei Tabs:
  - **Kommentare**
    Hier werden alle Kommentare zu diesem Kontakt angezeigt.
  - **Attachments**
    Hier sind alle Attachments des Kontakts aufgeführt. Die Liste der Attachments kann nach Dateityp, Name, Beschreibung, Datum oder Bearbeiter gefiltert und sortiert werden.
- **Verknüpfte Firmen und Kontakte (4)**
  Hier sind alle Relationen aufgeführt, die zwischen dem Kontakt und anderen Kunden (Firmen oder Kontakten) hergestellt wurden.

- **Verknüpfte Ressourcen (5)**
  Dieser Bereich wird nur angezeigt, wenn CM/Resource Pool im CM-System aktiviert ist und wenn mindestens eine Ressourcenrelation zu Kontakten der entsprechenden Kundengruppe im Admin Tool konfiguriert ist. Hier sind alle mit diesem Kontakt verknüpften Ressourcen aufgeführt.
- **Protokoll (6)**
  In diesem Bereich sind alle mit diesem Kontaktobjekt ausgeführten Aktionen aufgeführt, z. B. die Änderung eines Namens oder eines anderen Wertes in einem der Kundenfelder, oder das Hinzufügen/Löschen von Relationen, Kommentaren oder Attachments.

Eine detaillierte Erklärung über die Arbeit mit Kontakten finden Sie im *ConSol CM Benutzerhandbuch*.

⚠ Beachten Sie, dass nur Bearbeiter, die mindestens eine Rolle mit den folgenden Zugangsberechtigungen für die entsprechende Kundengruppe haben, auf den *Bereich für Kommentare und Attachments* der Kundenseite zugreifen können!

- Details lesen
- Details hinzufügen
- Details löschen

Beachten Sie außerdem, dass nur Bearbeiter, die mindestens eine Rolle mit Schreibberechtigungen für die entsprechende Kundengruppe haben, auf der Kontaktseite den Link *Wechseln* im Kontextmenü der Firma verwenden können, um den Kontakt einer anderen Firma zuzuordnen.

D.2.2.4 Beispiel 4: Einrichten einer Firma als Hauptkunde eines Tickets

Wenn die Konfigurationsoption *Firma als Kunde* im Kundendatenmodell gesetzt wurde, kann eine **Firma als Hauptkunde** für ein Ticket verwendet werden.

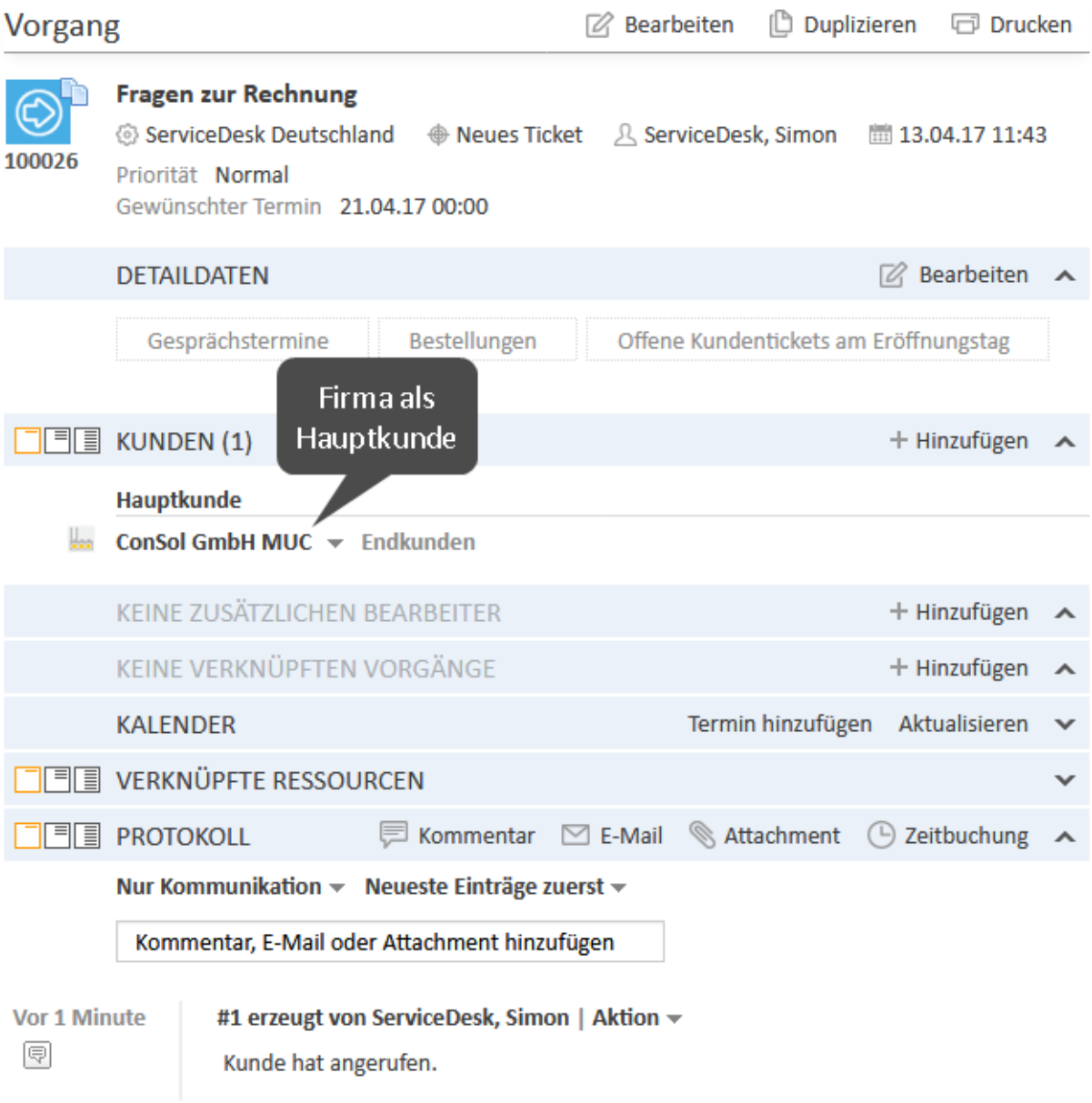


Abbildung 255: *ConSol CM Web Client - Verwendung einer Firma als Hauptkunde eines Tickets*

D.2.2.5 Beispiel 5: Verwendung von Firmen- und Kontaktaktionen

Für Firmen und Kontakte können manuelle und automatische Aktionen definiert werden.

Manuelle Aktionen werden über Links im Web Client angestoßen. Sie ähneln den Workflow-Aktionen (Aktivitäten) für Tickets. Auf diese Weise können Aktionen im Zusammenhang mit Firmen- oder Kontaktdaten ausgeführt werden, die sich nicht auf Ticketdaten beziehen. Ein Bearbeiter kann beispielsweise die KPIs des letzten Monats für die Firma laden, einen neuen Kontakt innerhalb der Firma erstellen (siehe folgende Abbildung), die Kontaktdaten aus einer anderen Datenbank laden, oder ein Ticket für den Kontakt erstellen (siehe Abbildung danach).

Automatische Aktionen können durchgeführt werden, wenn eine Systemaktion ausgeführt wird (Erstellen/Aktualisieren/Löschen eines Kunden). Die Firmen- und Kontaktaktionen sind Bestandteile des Action Frameworks.

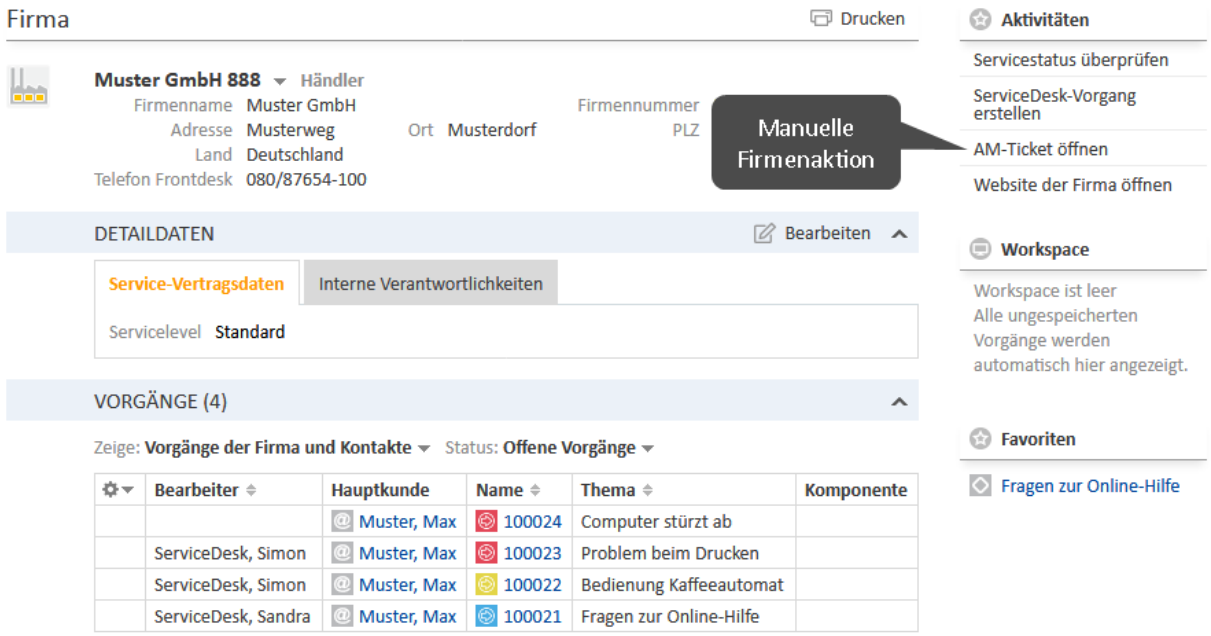


Abbildung 256: *ConSol CM Web Client - Manuelle Firmenaktionen*

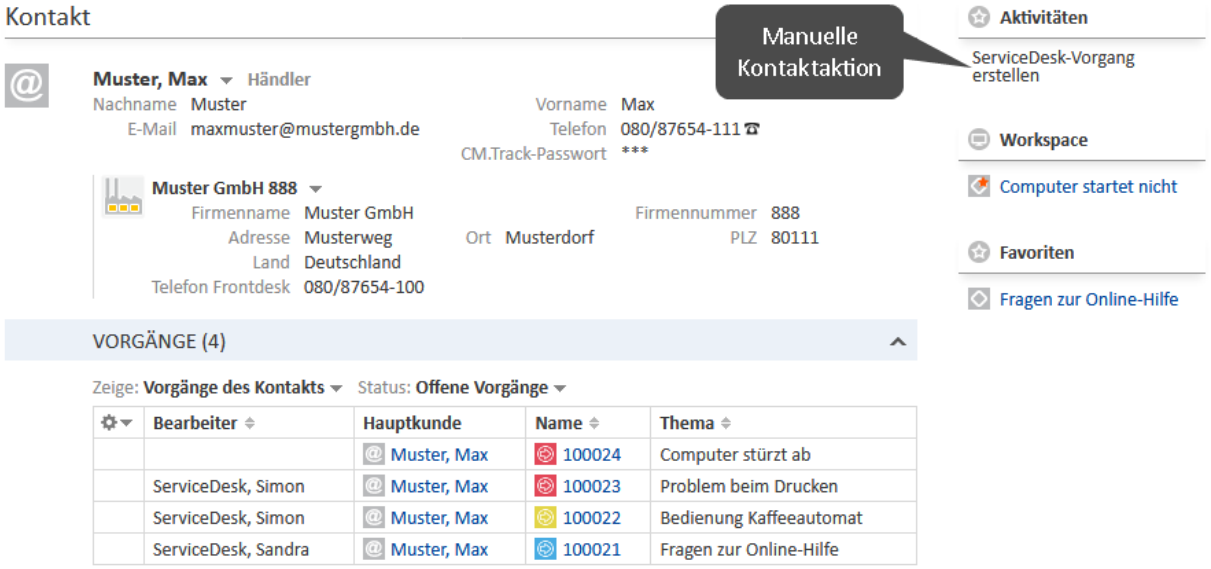


Abbildung 257: *ConSol CM Web Client - Manuelle Kontaktaktionen*

Beachten Sie, dass nur Bearbeiter, die mindestens eine Rolle mit den folgenden Zugangsberechtigungen für die entsprechende Kundengruppe haben, die Kundenaktionen verwenden dürfen, d. h. nur bei ihnen werden die Aktivitäten im Web Client angezeigt!

- Ausführen

D.2.2.6 Beispiel 6: Erstellen von Relationen zwischen Kontakten und Firmen

Besonders wenn Sie mit mehreren Kundengruppen arbeiten, kann es wichtig sein, **Relationen zwischen Kontakten und/oder Firmen** herzustellen. Ihr ConSol CM-System kann dann zum Beispiel eine Referenz *verkauft Produkte an...* zwischen einer Firma und einem Kontakt darstellen, oder eine Relation ist *Vorgesetzter von...* zwischen zwei Kontakten. Auf diese Weise können Sie ein Netzwerk mit Ihren Firmen und Kontakten erzeugen und CM für Customer Relationship Management (CRM) nutzen.

Im Web Client werden Relationen zwischen Firmen und/oder Kontakten ähnlich wie Ticketrelationen erzeugt und angezeigt. In diesem Beispiel verkauft *Muster GmbH 888* Produkte an *Hans Hansen*.

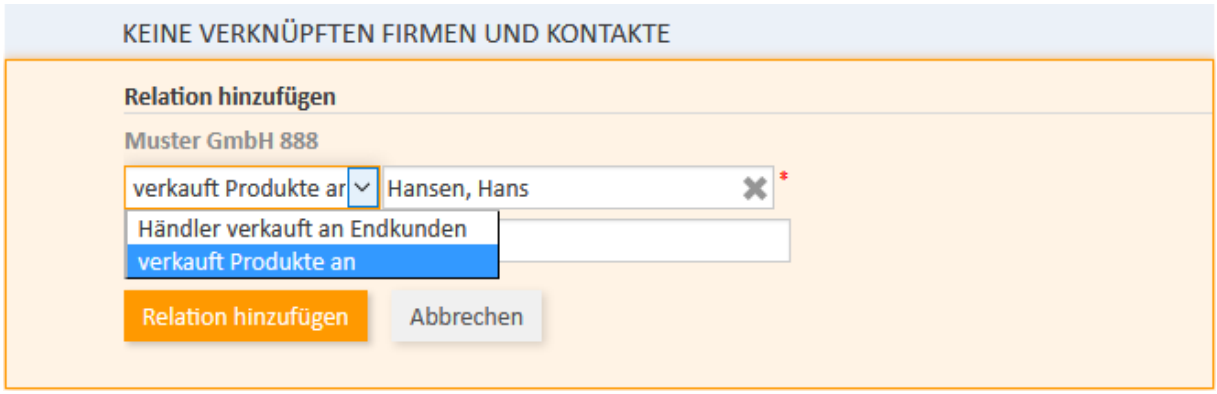


Abbildung 258: *ConSol CM Web Client - Erzeugen einer Relation Firma-Kontakt*

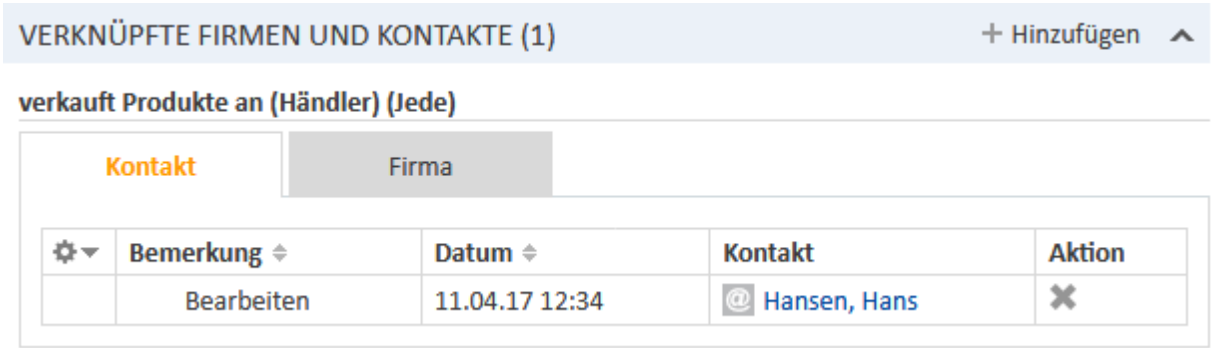


Abbildung 259: *ConSol CM Web Client - Anzeigen einer Relation Firma-Kontakt*

D.2.2.7 Beispiel 7: Deaktivieren eines Kunden (d. h. einer Firma oder eines Kontakts)

Ein Kunde, d. h. **eine Firma oder ein Kontakt**, **kann deaktiviert werden**. Diese Funktion kann praktisch sein, wenn ein Vertrag mit einer Firma ausgelaufen ist oder wenn ein Mitarbeiter (= Kontakt) die Firma verlassen hat. Auf diese Weise können Sie die Tickets behalten und unter dem **alten** Name des Kontakts/der Firma abrufen, aber Sie können keine neuen Tickets für diesen Kunden mehr erstellen. Wenn der Kunde gelöscht werden muss, müssen alle seine Tickets (offene und geschlossene) verschoben werden. In diesem Fall ist die frühere Relation Kontakt-Ticket oder Firma-Ticket nicht mehr leicht zu finden.

Der Kontakt bzw. die Firma kann nur deaktiviert werden, wenn ihm bzw. ihr keine **offenen** Tickets zugewiesen sind.

⚠ Beachten Sie, dass nur Bearbeiter, die mindestens eine Rolle mit den folgenden Zugangsberechtigungen für die entsprechende Kundengruppe haben, zum Deaktivieren (und Reaktivieren) von Firmen und Kontakten berechtigt sind, d. h. nur bei ihnen werden die Menüelemente Deaktivieren/Aktivieren im Web Client angezeigt!

- Deaktivieren/Aktivieren

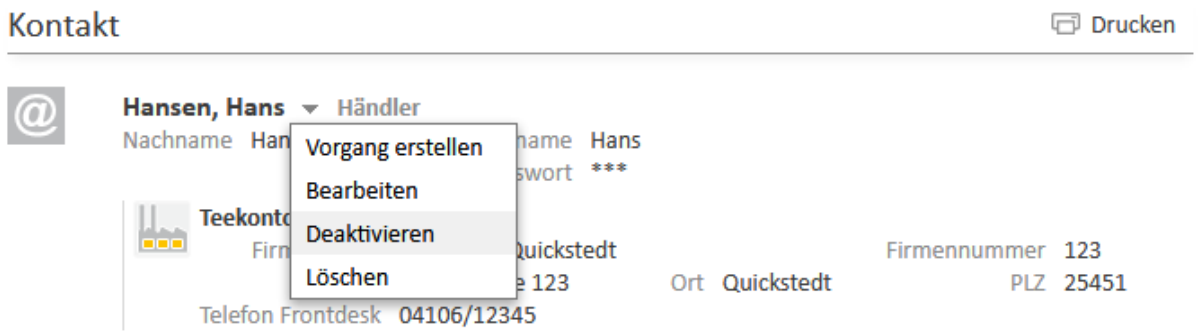


Abbildung 260: *ConSol CM Web Client - Deaktivieren eines Kontakts*

Die folgenden Aktionen **können** für einen deaktivierten Kunden ausgeführt werden:

- Kundendaten editieren (z. B. Name, Adresse, Telefonnummer).
- Kunden löschen.
- Geschlossene Tickets an einen anderen Kunden übertragen.
- Sofern die Checkbox *Deaktivierte einschließen* markiert ist: mit der Detailsuche im System nach dem Kunden suchen.

Die folgenden Aktionen **können nicht** für einen deaktivierten Kunden ausgeführt werden:

- Neues Ticket erstellen.
- Dem Kunden ein Ticket zuweisen.
- Deaktivierten Kontakt einer anderen Firma zuweisen.
- Kontakte einer deaktivierten Firma zuweisen.

- Nach dem Kunden suchen (Deaktivierte Kontakte und Firmen werden in den Suchergebnissen nicht angezeigt. Die Ausnahme bildet die Detailsuche, wenn der Bearbeiter explizit die Checkbox *Deaktivierte einschließen* markiert.).

⚠ **Deaktivierung in einem zweistufigen Kundendatenmodell**
Wenn eine Firma (oder allgemeiner ausgedrückt, ein Objekt auf Firmenstufe) deaktiviert wird, werden alle zugewiesenen Kontakte automatisch ebenfalls deaktiviert.

Es gibt hierbei zwei Anwendungsfälle:

1. **Alle** Kontakte der Firma **können** deaktiviert werden (ihnen sind keine offenen Tickets zugewiesen).
   In diesem Fall werden die Firma **und** alle zugewiesenen Kontakte deaktiviert. Danach wird die Firmenseite neu geladen und die Firmen- und Kontaktdaten werden als deaktiviert markiert.
2. Die Firma hat noch Kontakte, die **nicht** deaktiviert werden können, da sie offene Tickets haben.
   In diesem Fall es ist **nicht** möglich, die Firma zu deaktivieren. Die Option *Deaktivieren* kann nicht ausgewählt werden.

Reaktivierung in einem zweistufigen Kundendatenmodell
Wenn eine Firma wieder aktiviert wird, werden die zugewiesenen Kontakte **nicht automatisch** reaktiviert. Sie können **manuell** reaktiviert werden.

D.2.2.8 Beispiel 8: Löschen von Kundendaten

Es gibt drei Möglichkeiten, einen Kunden bzw. Kundendaten aus dem CM-System zu löschen:

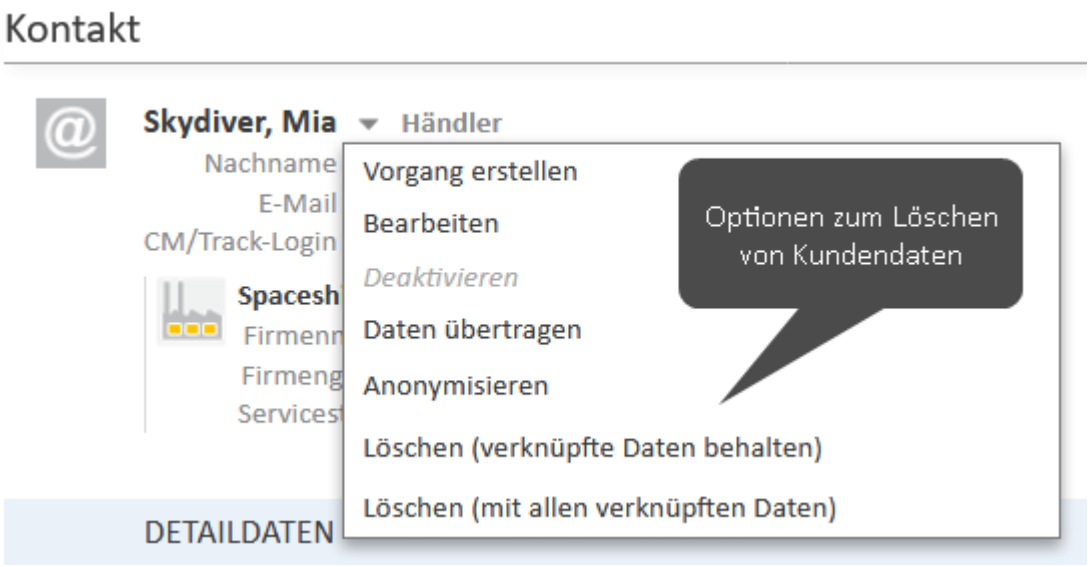


Abbildung 261: *Kontextmenü mit Löschoptionen auf Kontaktseite*

- **Löschen (verknüpfte Daten behalten) / Löschen**

  Der Kunde wird gelöscht, aber einige Referenzen auf den Kunden können im System in Protokolleinträgen anderer Objekte verbleiben. Bevor Sie einen Kunden löschen können, müssen alle seine Tickets auf einen anderen Kunden übertragen werden.

  > Die Option heißt für Kontakte *Löschen (verknüpfte Daten behalten)* und für Firmen *Löschen*.

- **Löschen (mit allen verknüpften Daten)**

  Diese Option ist nur für Kontakte verfügbar. Der Kontakt und alle Tickets, bei denen der Kontakt Hauptkunde ist, werden gelöscht. Alle Referenzen auf den Kontakt in Protokolleinträgen anderer Objekte werden anonymisiert.

  > Diese Option kann verwendet werden, um Kontaktdaten gemäß DSGVO zu löschen, wenn der Kontakt und seine Tickets nicht mehr für Reports benötigt werden.

- **Anonymisieren**

  Die persönlichen Daten im Kontakt und in Tickets, bei denen der Kontakt Hauptkunde ist, werden gelöscht. Wie persönliche Daten definiert werden, erfahren Sie in ... ein Feld für persönliche Daten definieren. Alle vorhandenen Protokolleinträge, Kommentare und Attachments des Kontakts werden gelöscht, und es wird ein Eintrag über die Anonymisierung des Kontakts zum Protokoll hinzugefügt. Das gesamte Protokoll der Tickets des Kontakts wird gelöscht und ein Eintrag über die Anonymisierung der Daten wird zum Protokoll hinzugefügt. Relationen zu anderen Objekten werden entfernt und der Kontakt wird im Protokoll dieser Objekte anonymisiert.

  > Diese Option kann verwendet werden, um Kontaktdaten gemäß DSGVO zu löschen, wenn der Kontakt und seine Tickets noch für Reports benötigt werden.

D.2.2.9 Beispiel 9: Verwenden der Ticketfilter auf der Firmen- oder Kontaktseite

Auf den Firmen- und Kontaktseiten gibt es Ticketfilter, d. h. es können **Filteroptionen** zum Anzeigen von ausgewählten Tickets für eine Firma oder einen Kontakt verwendet werden.

Ticketfilter auf Firmenseite

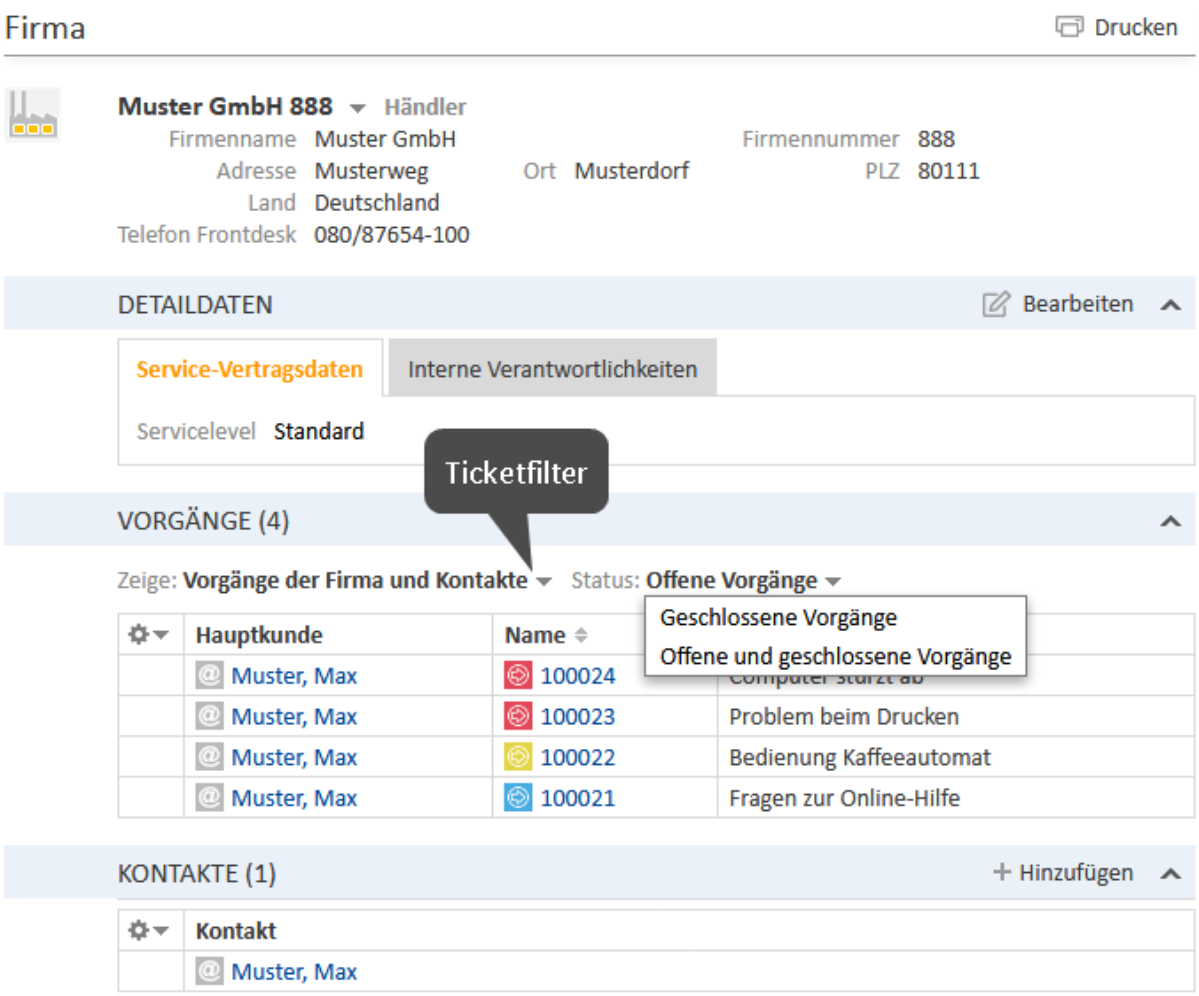


Abbildung 262: *ConSol CM Web Client - Ticketfilter auf Firmenseite*

Welche Tickets angezeigt werden, hängt von der Auswahl in den beiden Ticketfiltern ab:

- **Zeige**
  - **Vorgänge der Firma:** Tickets, bei denen die Firma entweder der Hauptkunde oder ein Zusatzkunde ist.
  - **Vorgänge der Firma (nur als Hauptkunde):** Tickets, bei denen die Firma selbst der Hauptkunde ist. (Dies ist nur bei Kundengruppen möglich, die ein Kundendatenmodell verwenden, bei dem die Option *Firma als Kunde* aktiviert ist.)
  - **Vorgänge der Kontakte:** Tickets, bei denen ein Kontakt der Firma entweder der Hauptkunde oder ein Zusatzkunde ist.
  - **Vorgänge der Firma und Kontakte:** Tickets, bei denen die Firma oder ein Kontakt der Firma entweder der Hauptkunde oder ein Zusatzkunde ist.

- **Status**
  - Offene Vorgänge
  - Geschlossene Vorgänge
  - Offene und geschlossene Vorgänge

Ticketfilter auf Kontaktseite

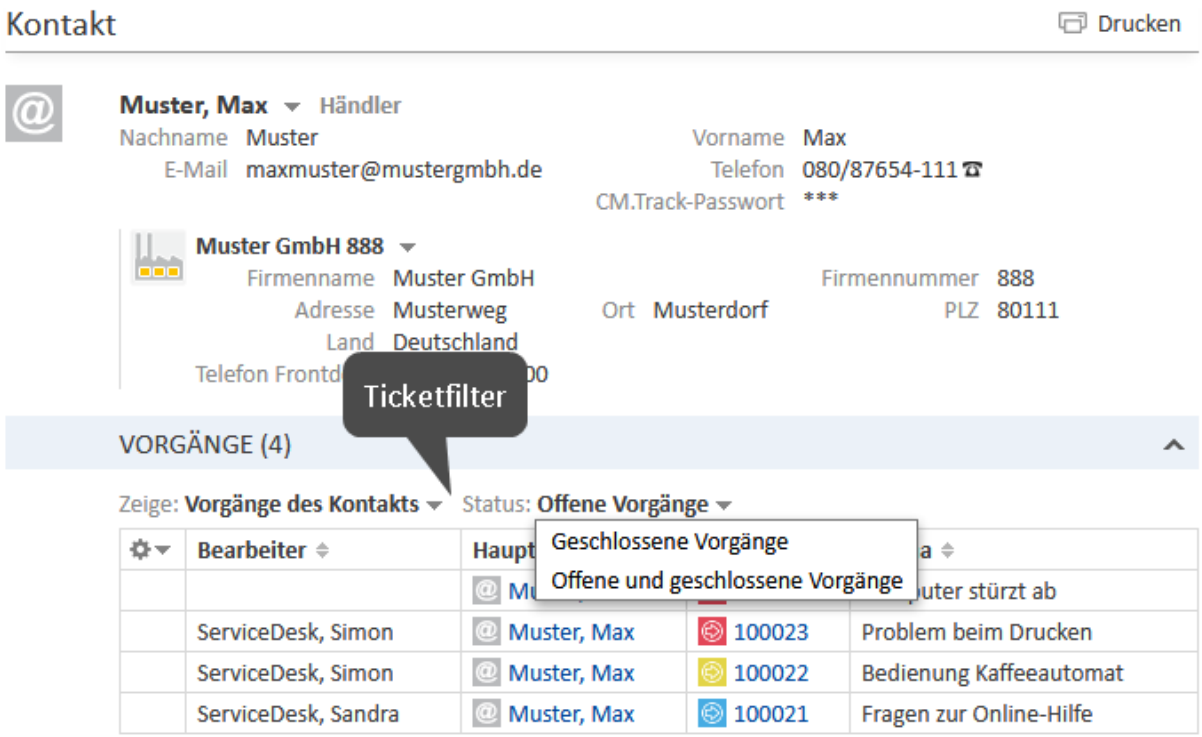


Abbildung 263: *ConSol CM Web Client - Ticketfilter auf Kontaktseite*

Welche Tickets angezeigt werden, hängt von der Auswahl in den beiden Ticketfiltern ab:

- **Zeige**
  - **Vorgänge des Kontakts:** Tickets, bei denen der Kontakt entweder der Hauptkunde oder ein Zusatzkunde ist.
  - **Vorgänge des Kontakts (nur als Hauptkunde):** Tickets, bei denen der Kontakt der Hauptkunde ist.
  - **Alle Vorgänge der Firma:** Tickets, bei denen die Firma des Kontakts, der Kontakt selbst oder ein weiterer Kontakt der Firma entweder der Hauptkunde oder ein Zusatzkunde ist.
- **Status**
  - Offene Vorgänge
  - Geschlossene Vorgänge
  - Offene und geschlossene Vorgänge

D.2.2.10 Beispiel 10: In der Schnellsuche angezeigte Kundengruppen

Die **Kundengruppe wird für alle Suchergebnisse in der Liste** angezeigt. Dafür wird folgende Notation verwendet:

```
<Lokalisierter Name der Kundenfeldgruppe> (<Lokalisierter Name der Kundengruppe>)
```

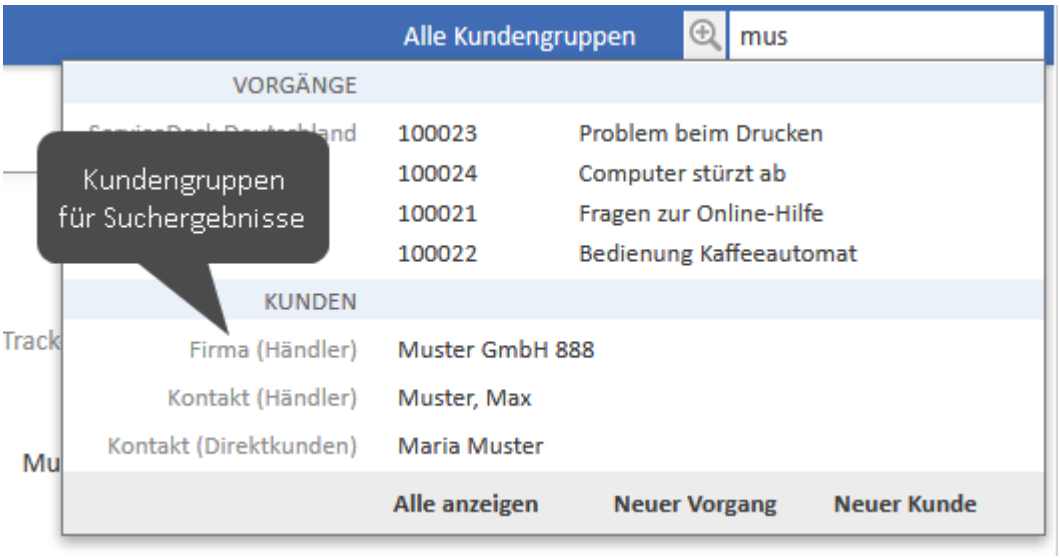


Abbildung 264: *ConSol CM Web Client - Suchergebnisse der Schnellsuche*

D.2.2.11 Beispiel 11: Verwendung von Suche-Aktionen für Firmen- oder Kontaktdaten

Siehe Abschnitt Action Framework - Action Framework - Suche-Aktionen.

D.3 Einrichten des Kundendatenmodells

In diesem Kapitel werden folgende Themen behandelt:

D.3.1 Einführung in die Einrichtung des Kundendatenmodells auf Basis von FlexCDM 394

D.3.2 Verwaltung von Kontakten und Firmen mit dem Admin Tool 395

D.3.1 Einführung in die Einrichtung des Kundendatenmodells auf Basis von FlexCDM

Mit **FlexCDM** können verschiedene Kundendatenmodelle implementiert werden. Eine detaillierte Einführung in FlexCDM finden Sie im Abschnitt Einführung in FlexCDM. Um mit einem neuen Kundendatenmodell innerhalb einer bestimmten Kundengruppe (oder auch in mehreren Kundengruppen) arbeiten zu können, müssen folgende Schritte durchgeführt werden:

1. Erstellen eines Kundendatenmodells.
   (Dafür müssen Sie zuerst entscheiden, ob es sich um ein einstufiges oder um ein zweistufiges Datenmodell handeln soll. In diesem Beispiel erstellen wir ein zweistufiges Modell.)
2. Erstellen einer neuen Kundengruppe.
3. Zuweisen des Kundendatenmodells zur Gruppe.

Ein Kundendatenmodell enthält Objekte auf drei Modellebenen:

1. **Die Definition des Kundendatenmodells**
2. **Die Kundenobjekte innerhalb dieses Modells**
   Ein Kundenobjekt kann von einem der folgenden beiden Typen sein:
   a. **Firma**
      z. B. eine Organisation, es kann sich aber auch um eine Maschine, ein Schiff oder irgendetwas anderes auf der Firmenstufe handeln.
   b. **Kontakt**
      z. B. eine Person, es kann sich aber auch um eine Maschine, ein Hardwaregerät, ein Produkt oder irgendetwas anderes auf der Kontaktstufe handeln.
      Wenn es eine Firmenstufe gibt, ist der Kontakt die der Firma untergeordnete Stufe. Für ein einfaches Kundendatenmodell können Sie nur das Kontaktobjekt oder nur das Firmenobjekt verwenden.
3. **Die Kundenfelder**
   Dies sind Datenfelder für Kundenobjekte, d. h. entweder Kundenfelder für Firmendaten (z. B. Postleitzahl, Telefon) oder Kundenfelder für Kontaktdaten (z. B. Name, Vorname, E-Mail-Adresse).

D.3.2 Verwaltung von Kontakten und Firmen mit dem Admin Tool

Die Komponenten des Kundendatenmodells werden in der Navigationsgruppe *Kunden* im Admin Tool verwaltet. Öffnen Sie das Navigationselement *Datenmodelle*, um ein neues Datenmodell einzurichten oder ein vorhandenes Modell zu editieren.

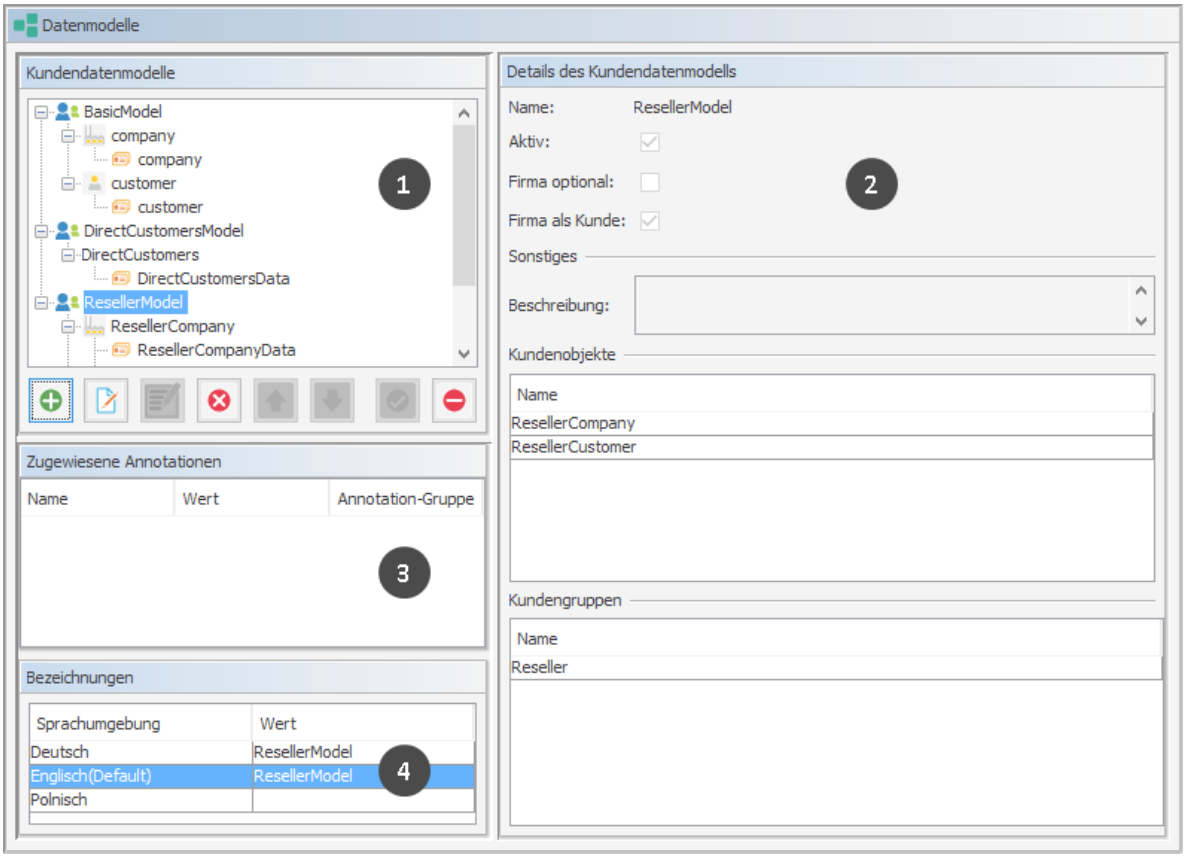


Abbildung 265: *ConSol CM Admin Tool - Kunden, Datenmodelle: Definition des Kundendatenmodells*

Die Abbildung zeigt die folgenden Bereiche:

- Baumstruktur aller Komponenten aller Kundendatenmodelle (1)
- Details der ausgewählten Komponente (2)
- Annotationen für die ausgewählte Kundenfeldgruppe eines Kundendatenmodells, leer für Kundendatenmodelle oder wenn keine Kundenfeldgruppe ausgewählt ist (3)
- Bezeichnungen für die ausgewählte Komponente (4)

Die Arbeit mit dem Kundendatenmodell wird anhand eines Beispiels in den folgenden Abschnitten erklärt.

D.3.2.1 Erstellen eines neuen zweistufigen Kundendatenmodells

Um ein neues Kundendatenmodell zu erstellen, müssen Sie die Objekte auf allen Ebenen des Datenmodells erstellen. Im folgenden Beispiel richten wir ein Kundendatenmodell für Händlerdaten ein. Wir erstellen ein Kundendatenmodell mit einem Firmen- und einem Kontaktobjekt, d. h. wir brauchen folgende Objekte:

- das Kundendatenmodell selber
- das Firmenobjekt (1. Stufe)
- die Kundenfelder für die Firma
- das Kontaktobjekt (2. Stufe)
- die Kundenfelder für den Kontakt

Nach der Definition eines Objekts können (oder besser gesagt sollten) die Objektparameter konfiguriert werden.

Schritt 1: Erstellen des Kundendatenmodells mit dem ersten Kundenobjekt

Wenn Sie ein neues Kundendatenmodell erstellen, müssen Sie ein Kundenobjekt und die entsprechenden Kundenfelder in einem Schritt hinzufügen.

Um ein neues Kundendatenmodell zu erstellen, markieren Sie ein anderes Kundendatenmodell (auf diese Weise wählen Sie die Ebene, auf der Sie arbeiten möchten) und klicken Sie auf den Button *Hinzufügen*, um das Pop-up-Fenster zu öffnen.

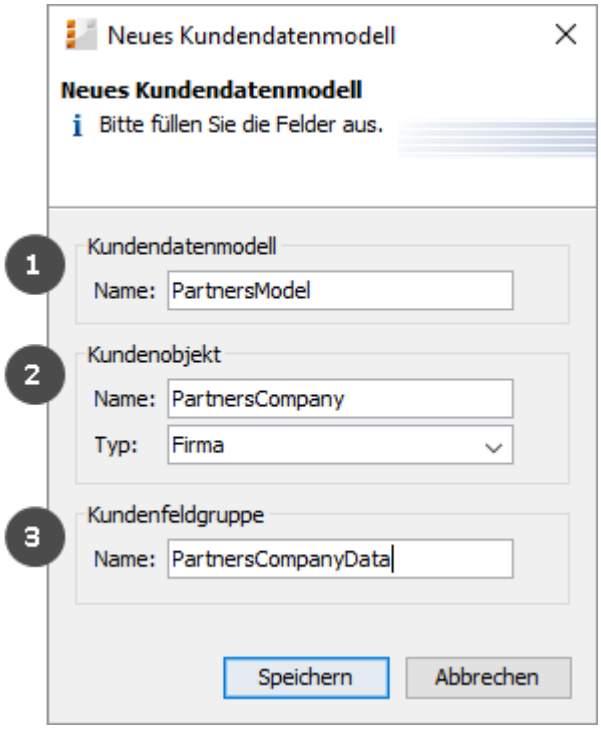


Abbildung 266: *ConSol CM Admin Tool - Kunden, Datenmodelle: Erstellen eines neuen Kundendatenmodells*

Sie müssen folgende Felder ausfüllen:

- **Kundendatenmodell**
  - **Name (1)**
    Der Name des neuen Kundendatenmodells. Die lokalisierten Namen für die Datenmodelle werden im Abschnitt *Bezeichnungen* des Hauptfensters *Datenmodelle* festgelegt. Details dazu finden Sie im Abschnitt Lokalisierung von Datenfeldern.
- **Kundenobjekt (2)**
  - **Name**
    Der eindeutige technische Name des Firmen-/Kontaktobjekts. Die lokalisierten Namen für das Kundenobjekt werden im Abschnitt *Bezeichnungen* des Hauptfensters *Datenmodelle* festgelegt. Details dazu finden Sie im Abschnitt Lokalisierung von Datenfeldern.
  - **Typ**
    Wählen Sie *Kontakt*oder *Firma*. Es kann in einem Kundendatenmodell nur ein Kontaktobjekt und ein Firmenobjekt geben.
- **Kundenfeldgruppe (3)**
  - **Name**
    Der eindeutige technische Name der ersten Kundenfeldgruppe für Firmendaten im definierten Kundenobjekt. Weitere Kundenfeldgruppen können später hinzugefügt werden. Die lokalisierten Namen für die Kundenfeldgruppe werden im Abschnitt *Bezeichnungen* des Hauptfensters *Datenmodelle* festgelegt. Details dazu finden Sie im Abschnitt Lokalisierung von Datenfeldern.

Schritt 2: Erstellen eines weiteren Kundenobjekts

Im nächsten Schritt fügen Sie das Kontaktobjekt hinzu. Wählen Sie das Objekt *PartnersCompany* (um die korrekte Ebene für den darauf folgenden Vorgang *Hinzufügen* zu bestimmen) und klicken Sie auf den Button *Hinzufügen*, um das Pop-up-Fenster zu öffnen.

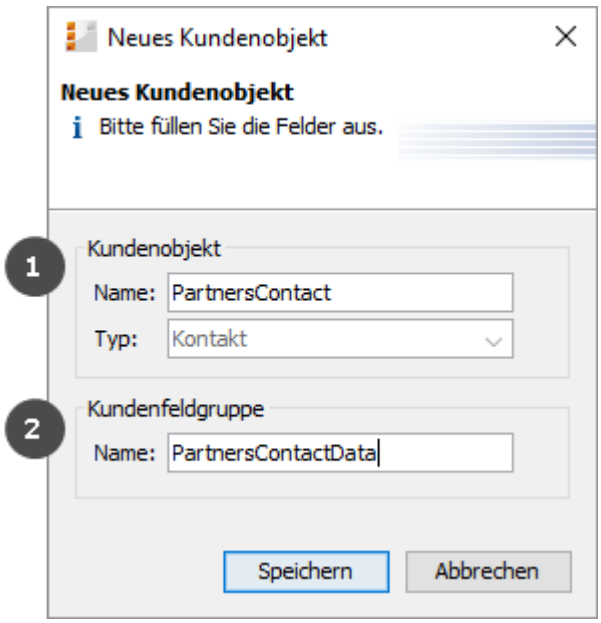


Abbildung 267: *ConSol CM Admin Tool - Kunden, Datenmodelle: Hinzufügen eines neuen Kundenobjekts*

Sie müssen folgende Felder ausfüllen:

- **Kundenobjekt (1)**
  - **Name**
    Der eindeutige technische Name des Kontaktobjekts. Die lokalisierten Namen für das Kundenobjekt werden im Abschnitt *Bezeichnungen* des Hauptfensters *Datenmodelle* festgelegt. Details dazu finden Sie im Abschnitt Lokalisierung von Datenfeldern. Der lokalisierte Name wird auch als Kopfzeile der Kundenseite (d. h. Kontakt- oder Firmenseite) verwendet, siehe Abschnitt Beispiel 3: Verwenden der Firmen- und Kontaktseite.
  - **Typ**
    Hier ist *Kontakt* schon ausgewählt und kann nicht geändert werden, da das Kundendatenmodell bereits ein Firmenobjekt enthält.
- **Kundenfeldgruppe (2)**
  - **Name**
    Der eindeutige technische Name der ersten Kundenfeldgruppe für Kontaktdaten im definierten Kundenobjekt. Weitere Kundenfeldgruppen können später hinzugefügt werden. Die lokalisierten Namen für die Kundenfeldgruppe werden im Abschnitt *Bezeichnungen* des Hauptfensters *Datenmodelle* festgelegt. Details dazu finden Sie im Abschnitt Lokalisierung von Datenfeldern.

Schritt 3: Konfigurieren der Parameter für die definierten Kundenobjekte

Parameter für das Kundendatenmodell

Doppelklicken Sie auf den Namen des Kundendatenmodells (in unserem Beispiel *PartnersModel*) oder markieren Sie das Kundendatenmodell in der Liste und klicken Sie auf den Button *Bearbeiten*, um das Pop-up-Fenster zu öffnen, in dem Sie die Parameter für das Modell definieren können.

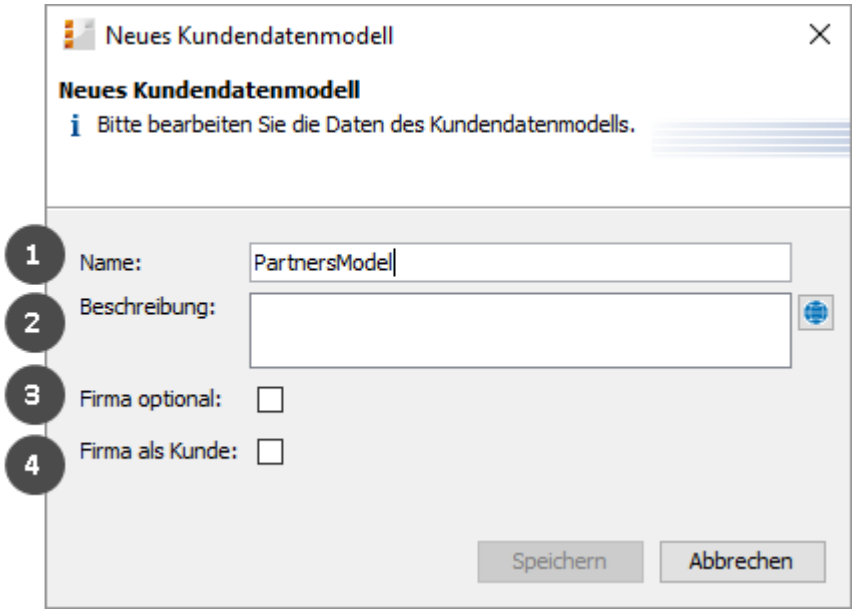


Abbildung 268: *ConSol CM Admin Tool - Kunden, Datenmodelle: Parameter für ein Kundendatenmodell*

Sie können folgende Felder ausfüllen:

- **Name (1)**
  Überprüfen oder ändern Sie den vorhandenen Namen des Modells.
- **Beschreibung (2)**
  Optionale Beschreibung. Kann lokalisiert werden. Details dazu finden Sie im Abschnitt Lokalisierung von Objekten allgemein, Typ 1.
- **Firma optional (3)**
  Wenn diese Checkbox markiert ist, ist es möglich, einen Kontakt, der nicht zu einer Firma gehört, zu einem Ticket hinzuzufügen. Die Firma kann später gesetzt werden, ist aber nicht unbedingt erforderlich. Auf diese Weise können Sie das System so einrichten, dass es auch innerhalb eines zweistufigen Kundendatenmodells mit Einzelkontakten arbeiten kann.
- **Firma als Kunde (4)**
  Markieren Sie diese Checkbox, wenn es in diesem Modell möglich sein soll, Tickets nicht nur für Kontakte sondern auch für Firmen zu erstellen.

Parameter für Kundenobjekte

Doppelklicken Sie auf den Namen des Kundenobjekts, z. B. *PartnersCompany*, oder wählen Sie das Objekt und klicken Sie auf den Button *Editieren*, um das Pop-up-Fenster zu öffnen, in dem Sie die Parameter für das Objekt konfigurieren können.

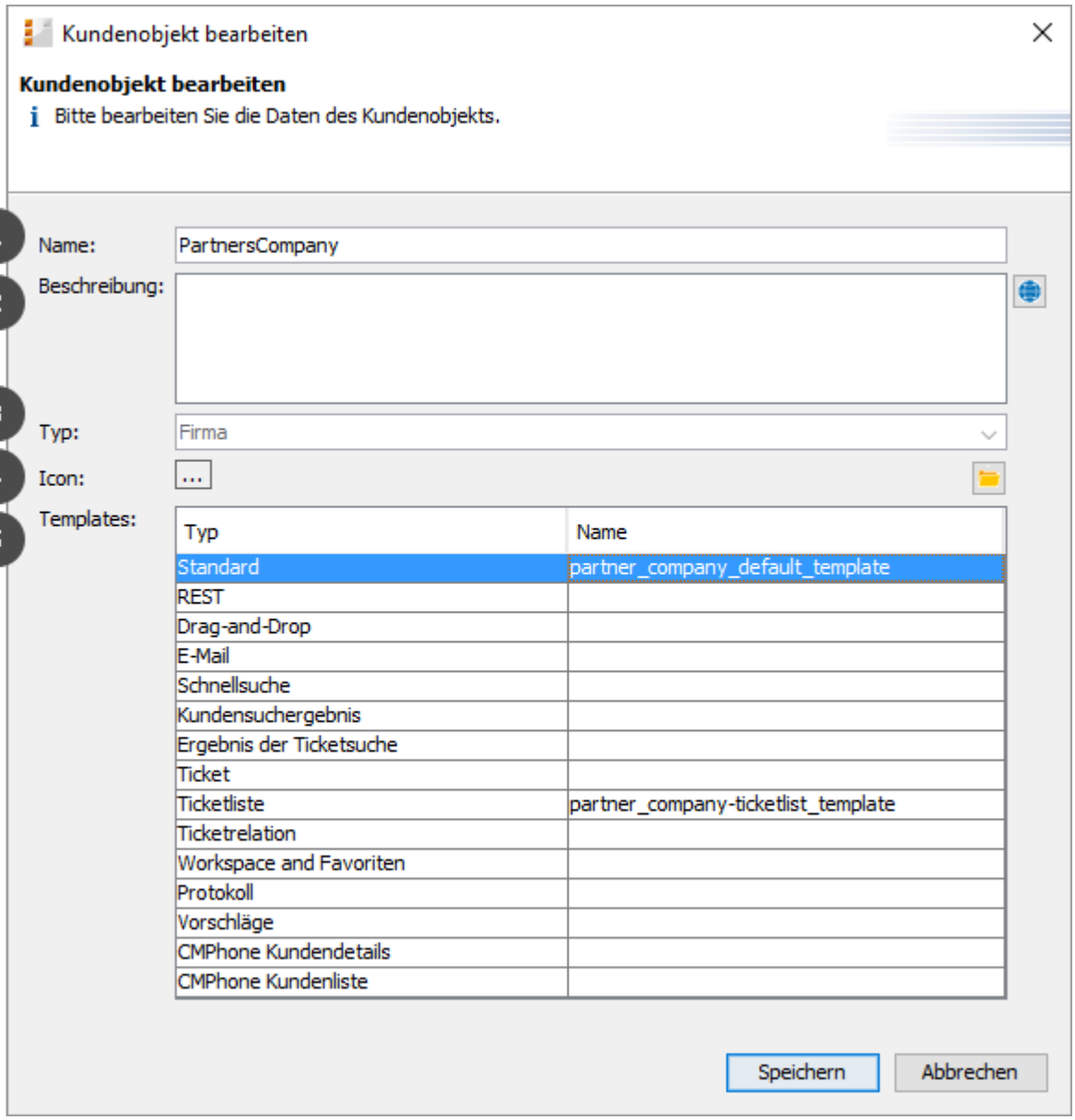


Abbildung 269: *ConSol CM Admin Tool - Kunden, Datenmodelle: Parameter für ein Kundenobjekt*

Sie können folgende Felder ausfüllen:

- **Name (1)**
  Der eindeutige technische Name des Kundenobjekts und die lokalisierten Namen. Der lokalisierte Name wird auch als Kopfzeile der Kundenseite (d. h. Kontakt- oder Firmenseite)

verwendet, siehe Abschnitt Beispiel 3: Verwenden der Firmen- und Kontaktseite.

- **Beschreibung (2)**
  Die Beschreibung des Kundenobjekts. Wird in zukünftigen Versionen von ConSol CM verwendet. Kann lokalisiert werden. Details dazu finden Sie im Abschnitt Lokalisierung von Objekten allgemein, Typ 1.
- **Typ (3)**
  Ein schreibgeschütztes Feld mit dem Typ (*Kontakt* oder *Firma*) des Kundenobjekts.
- **Icon (4)**
  Das Icon für alle Firmen dieses Modells. Es wird im Web Client angezeigt. Sie können entweder eines der Standard-Icons von CM benutzen (klicken Sie dazu auf den Button mit den drei Punkten) oder ein Icon aus dem Dateisystem hochladen (klicken Sie dazu auf den Button *Ordner*).
- **Templates (5)**
  Die Templates, mit denen die Daten des Kundenobjekts angezeigt werden, d. h. die Templates, mit denen die Datenfelder definiert werden, die im Web Client für Objekte dieses Typs angezeigt werden. Es gibt mehrere Stellen auf der GUI, für die das Layout der Firmen- oder Kontaktdaten definiert werden kann. Die Templates sind im Menü *Skripte und Templates* des Admin Tools gespeichert. Eine detaillierte Erklärung dazu finden Sie im Abschnitt Templates für Kundendaten.

Parameter für die Kundenfeldgruppe

Doppelklicken Sie auf den Namen der Kundenfeldgruppe, um den technischen Namen zu ändern (nur möglich, wenn die entsprechenden Felder in den Kundendatensätzen leer sind) oder um der Gruppe Skripte für abhängige sortierte Listen zuzuweisen bzw. diese zu entfernen.

Mit den GUI-Elementen auf der rechten Seite können Sie die Kundenfelder definieren. Die folgende Abbildung zeigt ein Beispiel für die Definition von Kundenfeldern für *PartnersCompany*. Es gibt mehrere Kundenfeldgruppen, ausgewählt ist in diesem Beispiel *PartnersCompanyData*. Sie können so viele Kundenfeldgruppen in einem Kundenobjekt verwenden, wie es für Ihr System passend ist.

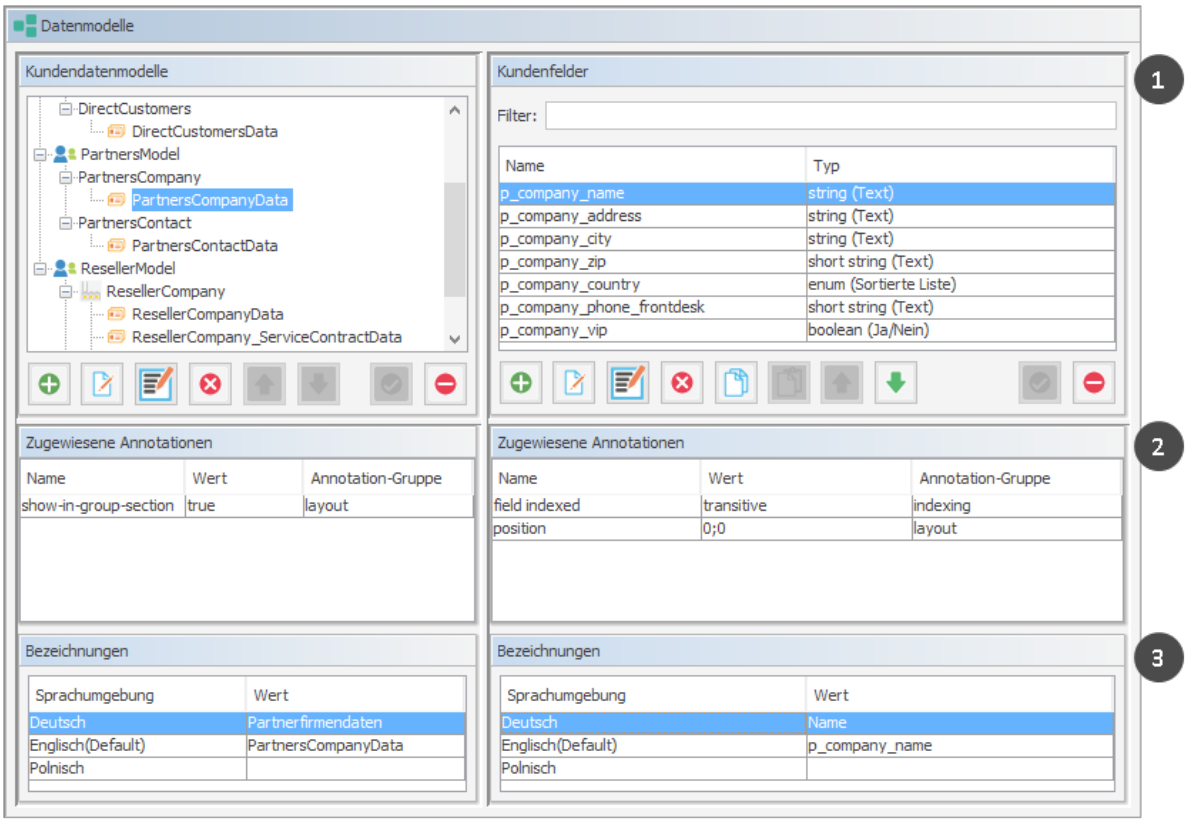


Abbildung 270: *ConSol CM Admin Tool - Kunden, Datenmodelle: Parameter für die Kundenfeldgruppe*

Die Abbildung zeigt die folgenden Bereiche:

- Liste der Kundenfelder in der ausgewählten Kundenfeldgruppe (1)
- Feldannotationen für das ausgewählte Kundenfeld (2)
- Bezeichnungen für das ausgewählte Kundenfeld (3), d. h., der lokalisierte Wert für den Kundenfeldnamen

Die Definition der Kundenfelder innerhalb der Kundendatenmodelle basiert auf den gleichen Prinzipien wie die Definition der Ticketfelder für Ticketdaten. Eine detaillierte Einführung in die Definition und Verwaltung von Ticketfeldern finden Sie in den Abschnitten Verwaltung von Ticketfeldern (Einrichten des Ticketdatenmodells) und Verwaltung von Kundenfeldern und GUI-Design für Kundendaten.

Die für Kundenfelder verfügbaren Annotationen sind im Abschnitt Annotationen aufgeführt. Die Annotation `unit is a contact` wird nicht mehr verwendet, da die Stufe der Unit (d. h. Firma oder Kontakt) jetzt durch den Typ der Unit (*Firma* oder *Kontakt*) bestimmt wird.

> ⚠ Stellen Sie sicher, dass die Annotation `field indexed` für alle Felder gesetzt ist, die durchsuchbar sein sollen. Dies betrifft die Schnellsuche, die Detailsuche und alle Autocomplete-Vorgänge! Siehe auch Abschnitt Konfiguration der Suche.

Herzlichen Glückwunsch! Wenn Sie alle in den obigen Abschnitten beschriebenen Schritte ausgeführt haben, haben Sie ein neues Kundendatenmodell für ConSol CM erstellt. Jetzt können Sie das Modell einer oder mehreren Kundengruppen zuweisen.

D.3.2.2 Erstellen einer neuen Kundengruppe mit dem neuen Kundendatenmodell

Wenn das Kundendatenmodell definiert wurde, kann es einer oder mehreren Kundengruppen zugewiesen werden. In diesem Beispiel erstellen wir die neue Kundengruppe *OurPartnerCompanies*, die das neue *PartnersModel* verwendet.

Im Navigationselement *Kundengruppen* der Navigationsgruppe *Kunden* des Admin Tools können Sie eine neue Kundengruppe erstellen und dem gewünschten Kundendatenmodell zuweisen.

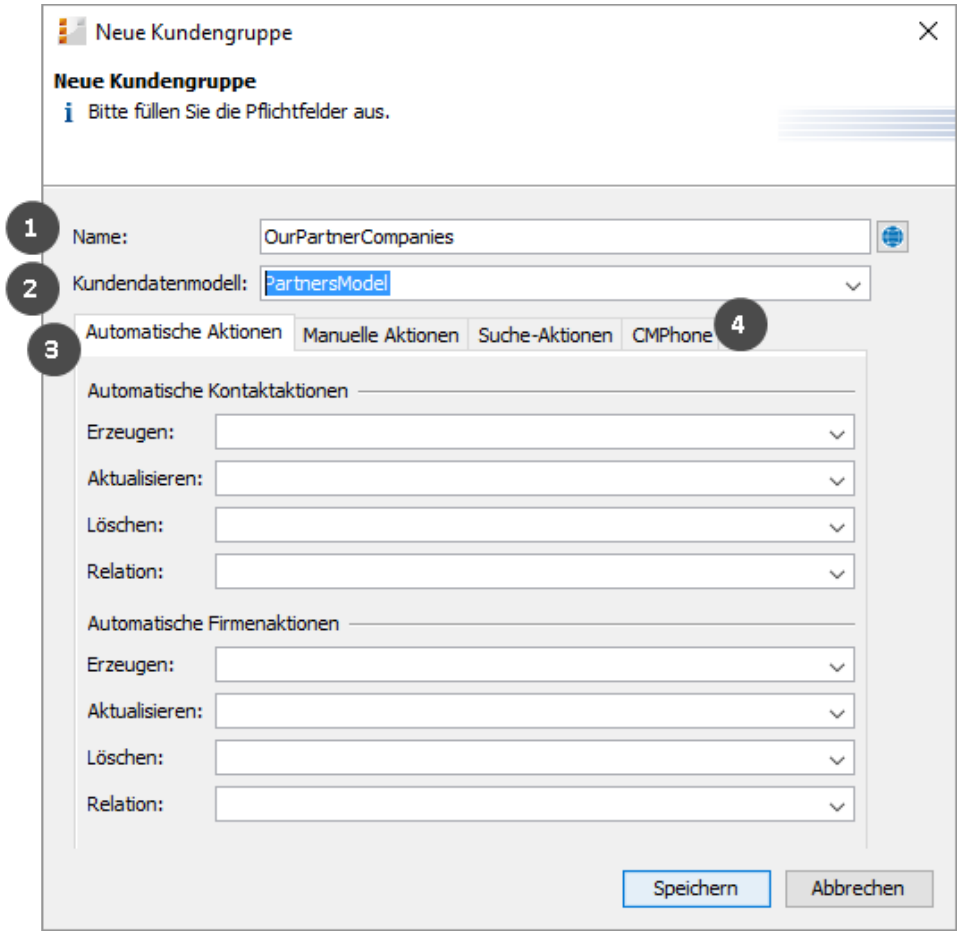


Abbildung 271: *ConSol CM Admin Tool - Kunden, Kundengruppen: Definition einer neuen Kundengruppe*

Sie können folgende Felder ausfüllen:

- **Name (1)**
  Der eindeutige technische Name (und der lokalisierte Name) der neuen Kundengruppe. Kann lokalisiert werden. Details dazu finden Sie im Abschnitt Lokalisierung von Objekten allgemein, Typ 1.
- **Kundendatenmodell (2)**
  Wählen Sie das gewünschte Datenmodell aus dem Drop-down-Menü aus.
- **Automatische/Manuelle/Suche-Aktionen (3)**
  In diesen Tabs können Sie die Kundenaktionen definieren. Dies wird im Abschnitt Action Framework - Kundenaktionen näher erklärt.
- **CMPhone (4)**
  Dieser Tab wird nur angezeigt, wenn CM/Phone installiert ist (siehe Abschnitt CM/Phone: CTI mit ConSol CM).

Für einen Bearbeiter, der Zugangsberechtigungen für vier Kundengruppen hat, sieht der Web Client folgendermaßen aus.

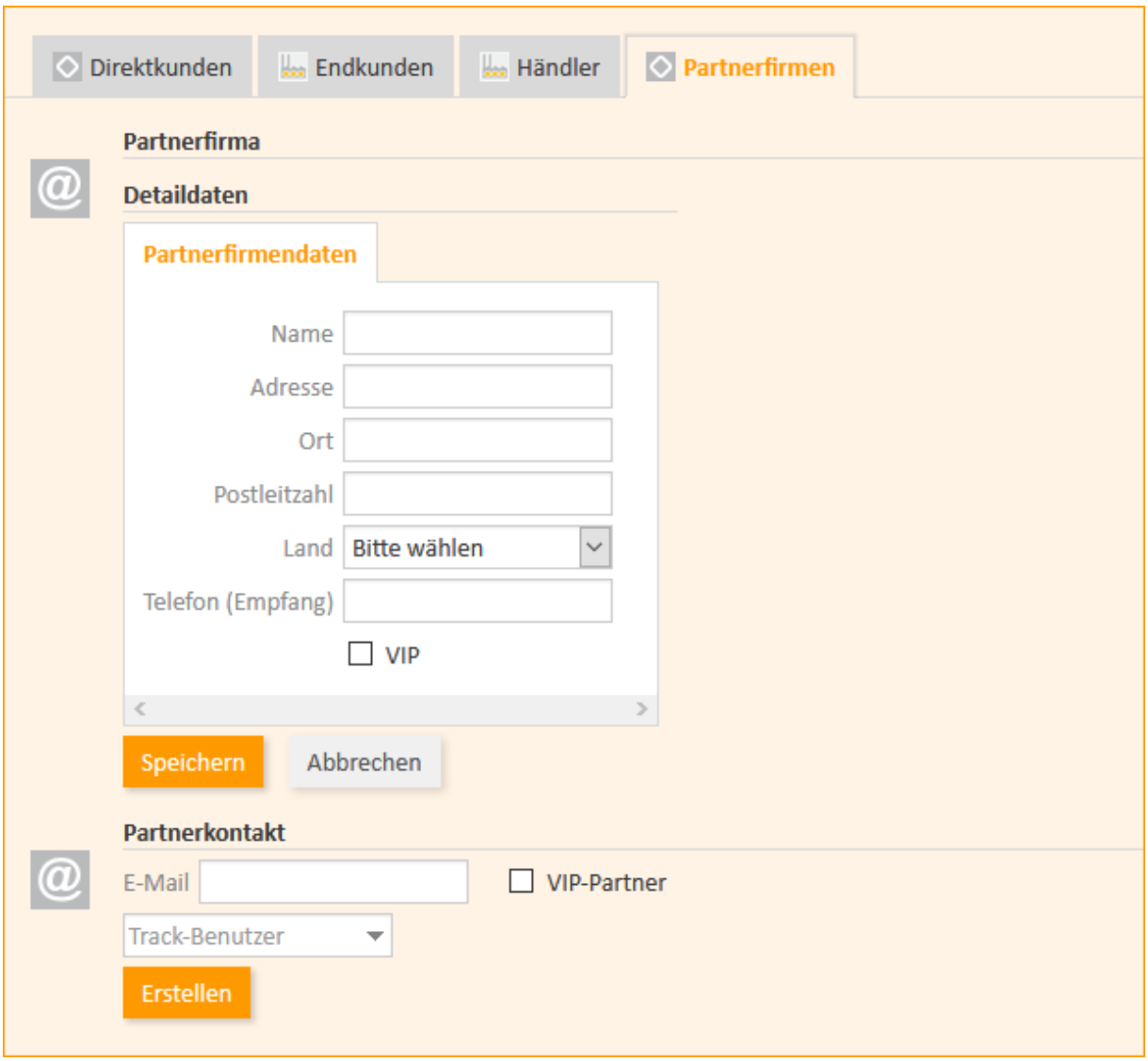


Abbildung 272: *ConSol CM Web Client - Erstellen einer neuen Firma und eines neuen Kontakts*

D.3.2.3 Zuweisen von Zugangsberechtigungen für Kundengruppen zu Rollen mit dem neuen Modell

Damit die Bearbeiter mit den Kundendaten der neuen Kundengruppe arbeiten können, d. h. neue Datensätze für Händler erstellen oder vorhandene Datensätze ändern, müssen Sie einer oder mehreren Rollen Zugangsberechtigungen für die Kundengruppen erteilen.

Siehe Abschnitt Tab Kundengruppen-Berechtigungen.

D.3.2.4 Zuweisen der neuen Kundengruppen zu Queues

Denken Sie daran, dass Sie die neue Kundengruppe allen Queues zuweisen müssen, in denen Tickets für Kunden dieser Gruppe erzeugt werden sollen. Details finden Sie im Abschnitt Queue-Verwaltung.

D.3.2.5 Deaktivieren von Objekten im Kundendatenmodell

Die folgenden Objekte innerhalb eines Kundendatenmodells können deaktiviert werden (die Umsetzung dieser Funktion ist in den aktuellen CM-Versionen nur sehr rudimentär):

- das gesamte Kundendatenmodell
- ein Objekt auf Firmenebene - hat momentan (CM-Version 6.11) keine Auswirkungen
- ein Objekt auf Kontaktebene - hat momentan (CM-Version 6.11) keine Auswirkungen
- eine Kundenfeldgruppe - hat momentan (CM-Version 6.11) keine Auswirkungen

Um ein bestimmtes Objekt zu deaktivieren, wählen Sie das Objekt auf der gewünschten Stufe aus (achten Sie darauf, zwischen dem gesamten Kundendatenmodell, der Kontakt- oder Firmenstufe und den Kundenfeldgruppen zu unterscheiden).

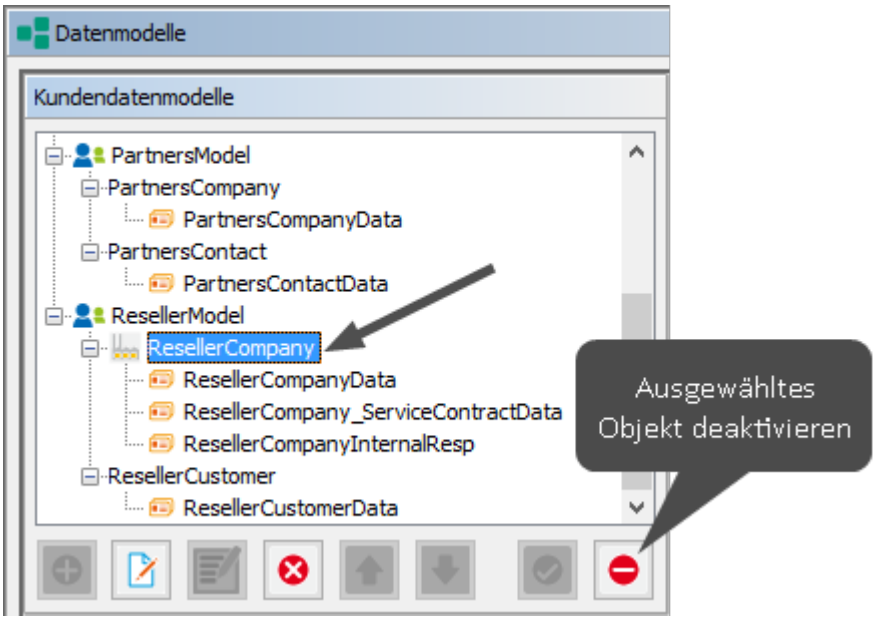


Abbildung 273: *ConSol CM Admin Tool - Kunden, Datenmodelle: Deaktivieren eines FlexCDM-Objekts (hier: Firmenobjekt im ResellerModel)*

D.4 Verwaltung von Kundenfeldern und GUI-Design für Kundendaten

In diesem Kapitel werden folgende Themen behandelt:

D.4.1 Einleitung 408

D.4.2 Definieren von Kundenfeldern für Kundendaten mit dem Admin Tool 409

D.4.3 Verwendung von Objekten aus FlexCDM in Skripten 423

D.4.4 Verwenden von skriptbasierter Feldvisualisierung für Kundenfelder 427

D.4.1 Einleitung

Eine der Stärken der Versionen 6.9 und höher von ConSol CM ist die große Flexibilität des Kundendatenmodells (FlexCDM) und GUI-Designs. Als Administrator können Sie alle benötigten Datenfelder definieren und an der gewünschten Stelle der Benutzeroberfläche platzieren. Das Grundprinzip ist jetzt das gleiche, das Sie bereits von den Ticketfeldern für Ticketdaten kennen: völlige Flexibilität.

Die Verwaltung des Ticketdatenmodells und das GUI-Design werden im Abschnitt Verwaltung von Ticketfeldern (Einrichten des Ticketdatenmodells) erklärt. Die Verwaltung von Objekten innerhalb des Kundendatenmodells wird im Abschnitt Einrichten des Kundendatenmodells detailliert erklärt. Bitte lesen Sie dort die entsprechenden Abschnitte. In diesem Kapitel gehen wir davon aus, dass Sie mit diesen Themen bereits vertraut sind.

D.4.2 Definieren von Kundenfeldern für Kundendaten mit dem Admin Tool

D.4.2.1 GUI des Admin Tools

Die Definition von Kundendatenfeldern ist Bestandteil der Definition des gesamten Kundendatenmodells, siehe Abschnitt Einrichten des Kundendatenmodells. Das Datenmodell wird im Admin Tool im Navigationselement *Datenmodelle* der Navigationsgruppe *Kunden* definiert.

Die Datenfelder für Kundenobjekte innerhalb des Kundendatenmodells heißen *Kundenfelder*. Die Arbeit mit Kundenfeldern basiert auf den gleichen Prinzipien wie die Arbeit mit Ticketdatenfeldern (Ticketfeldern): Die Datenfelder werden in Gruppen verwaltet und sowohl die Gruppen als auch die einzelnen Felder können annotiert werden.

Sie gelangen auf diese Seite, indem Sie das Navigationselement *Datenmodelle* in der Navigationsgruppe *Kunden* öffnen.

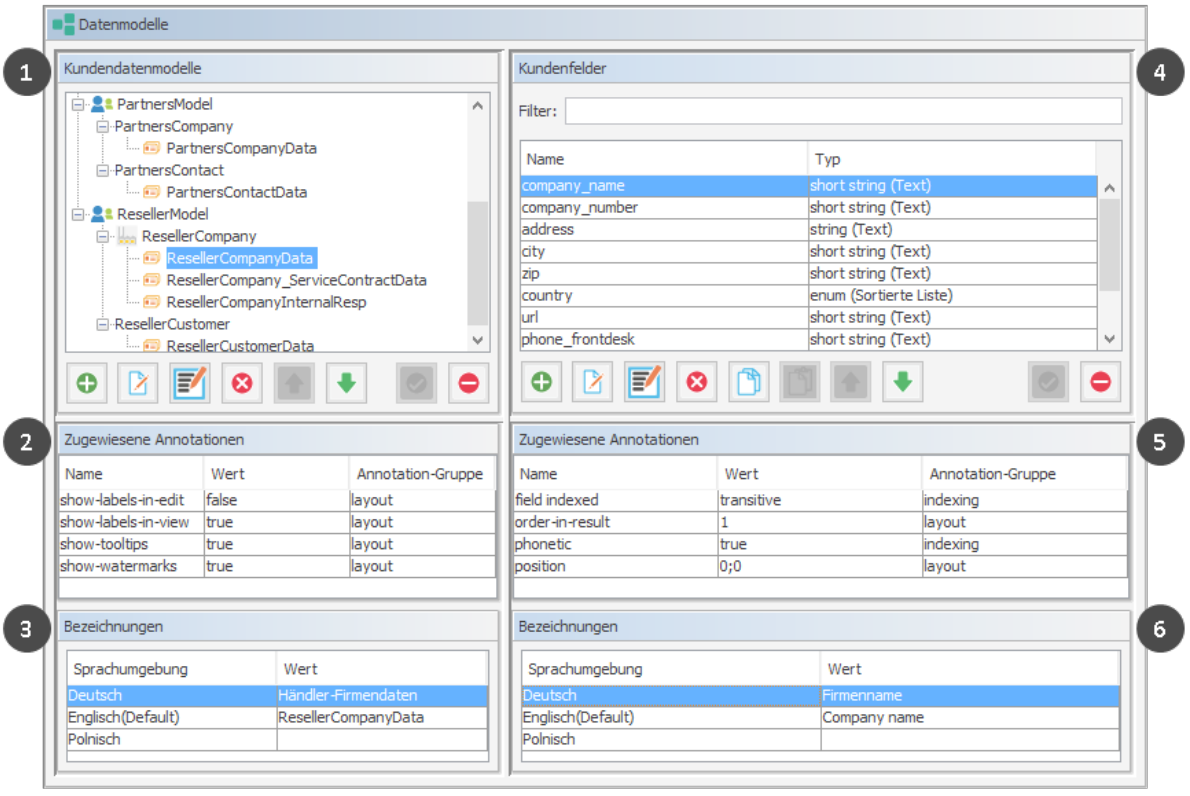


Abbildung 274: *ConSol CM Admin Tool - Kunden, Datenmodelle: Definition von Kundenfeldern*

Die Seite *Datenmodelle* besteht aus sechs Bereichen:

- Kundendatenmodelle (1)
- Annotationen für die ausgewählte Kundenfeldgruppe (2)

- Bezeichnungen für die ausgewählte Kundenfeldgruppe (3), d. h. die lokalisierten Werte für den Namen der Kundenfeldgruppe
- Kundenfelder für die ausgewählte Kundenfeldgruppe (4)
- Annotationen für das ausgewählte Kundenfeld (5)
- Bezeichnungen für das ausgewählte Kundenfeld (6), d. h. die lokalisierten Werte für den Kundenfeldnamen

D.4.2.2 Kundenfeldgruppen

Wie die Ticketfelder, die Sie bereits aus früheren CM-Versionen kennen, werden auch die Kundenfelder in Gruppen angelegt, den *Kundenfeldgruppen*. Jedes Kundenobjekt eines Kundendatenmodells kann so viele Kundenfeldgruppen haben, wie nötig sind. So kann zum Beispiel eine Firma im Händlerdatenmodell eine Kundenfeldgruppe für allgemeine Daten haben, eine für Vertragsdaten und eine für die Personen, die für diesen Händler zuständig sind. Für die Kontakte innerhalb des Händlerdatenmodells wird eine Kundenfeldgruppe mit allgemeinen Daten definiert.

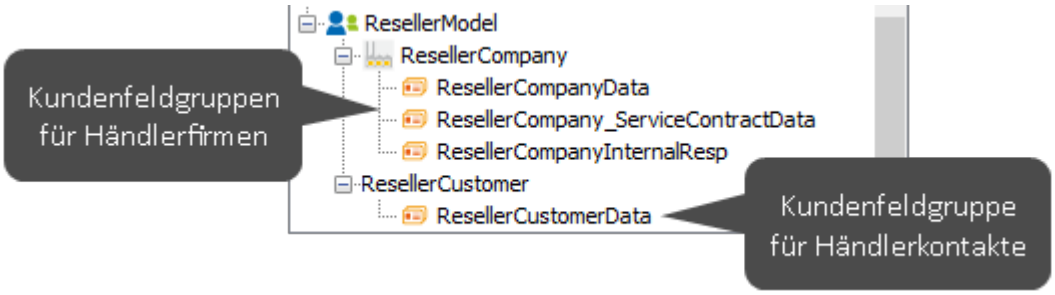


Abbildung 275: *ConSol CM Admin Tool - Kunden, Datenmodelle: Kundendatenmodell mit mehreren Kundenfeldgruppen*

Die Organisation der Datenfelder in Gruppen hat mehrere Auswirkungen. Stellen Sie sicher, dass das Design des Datenmodells den Bedürfnissen der Benutzer entspricht.

Eine Kundenfeldgruppe...

- kann auf der GUI während des Prozesses ein- und ausgeblendet werden; nur die ganze Gruppe, nicht die einzelnen Felder (Kundenfelder).
- kann als Tab oder im Kundendatenbereich angezeigt werden. Der Titel (und das Mouseover) des Tabs ist der (lokalisierte) Name der Kundenfeldgruppe. Details über die Lokalisierung von Kundenfeldgruppen finden Sie im Abschnitt Lokalisierung von Datenfeldern.
- wird mittels Gruppenannotationen konfiguriert.
- wird auf der GUI entsprechend der Position in der Liste der Kundenfeldgruppen angeordnet, die z. B. die Reihenfolge der Tabs festlegt.

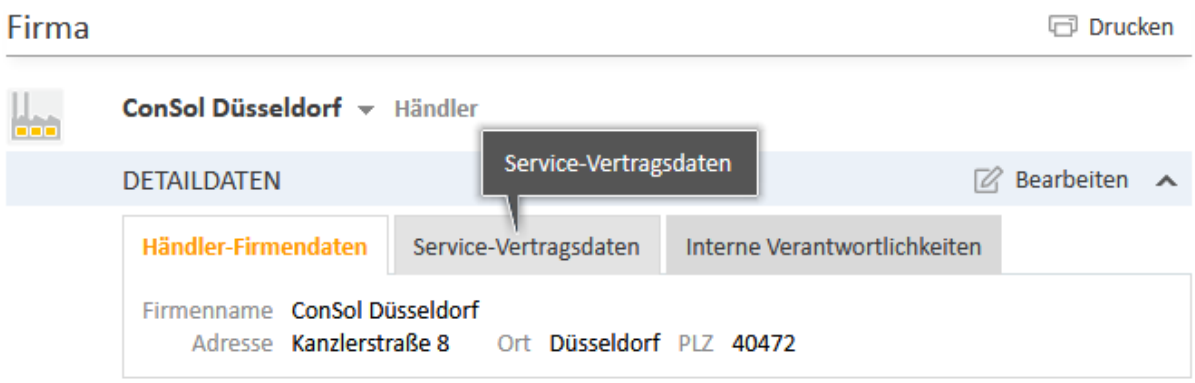


Abbildung 276: *ConSol CM Web Client - Firmendaten in Tabs (basierend auf Kundenfeldgruppen)*

D.4.2.3 Definieren von Kundenfeldern

Die Definition von Kundenfeldern (d. h. Datenfeldern wie *Name*, *Adresse* oder *Telefonnummer*) basiert auf dem bewährten Prinzip, das seit der ersten CM6-Version bereits für die Ticketfelder verwendet wurde.

Ein Kundenfeld...

- wird durch einen Datentyp definiert.
- wird mittels Feldannotationen konfiguriert (z. B. `position` oder `field-indexed`), siehe Annotationen.

> Im Gegensatz zu den Ticketfeldern und dem Layout von Tickets, wo für die Datenfelder nur drei Spalten zur Verfügung stehen, können Sie für die Definition von Kundenfeldern so viele Spalten verwenden, wie Sie möchten.
>
> Beachten Sie bitte, dass bis einschließlich CM-Version 6.10.4 das Business Card Feature greift, d. h. um Kundendaten immer bündig anzuordnen, werden alle Felder einer Zeile immer linksbündig angeordnet (d. h. beginnend mit Spalte 0), auch wenn das erste Feld in der Zeile erst in Spalte 2 (oder einer anderen) positioniert ist (Annotation `position`).

Die folgenden Datentypen sind für Ticketfelder, Kundenfelder und Ressourcenfelder verfügbar:

- **autocomplete (Autovervollständigung)**
  Ein Datenfeld, das eine skriptbasierte Autocomplete-Liste enthält. Das ist eine dynamische Liste, die auf einem Skript des Typs *Text-Autovervollständigung* basiert. Eine detaillierte Beschreibung von skriptbasierten Autocomplete-Listen finden Sie im Abschnitt Skriptbasierte Autocomplete-Listen.
- **boolean (Ja/Nein)**
  Werte: true/false. Je nach Einstellung in der Annotation `boolean-type` wird der Wert als Checkbox, Radio-Button oder Drop-down-Liste angezeigt.

Wenn Sie in CM-Workflows oder im Admin Tool mit Skripten arbeiten, sollten Sie wissen, dass das Verhalten von Boolean-Feldern, die als Checkboxen dargestellt werden, d. h. mit der Annotation `boolean-type` = „checkbox “(Standardwert) versehen sind, anders ist als in früheren CM-Versionen!

- **In CM-Versionen vor 6.9.4.0:**
  Wenn ein Boolean-Feld noch nicht angefasst wurde, ist der Wert „false“. Wenn es aktiviert wird, ist der Wert „true“, und wenn es danach wieder deaktiviert wird, ist der Wert wieder „false“.
- **In Version 6.9.4.0 und höher:**
  Wenn ein Boolean-Feld noch nicht angefasst wurde, ist der Wert „NULL“. Wenn es aktiviert wird, ist der Wert „true“, und wenn es danach wieder deaktiviert wird, ist der Wert „false“.

Felder, die in der Datenbank bereits mit Werten gefüllt sind, werden während des Updates von einer Version vor 6.9.4.0 auf Version 6.9.4.0 und höher nicht verändert.

Boolean-Felder, die als Radio-Button (Annotation `boolean-type` = „radio“) oder Drop-down-Liste (Annotation `boolean-type` = „select“) dargestellt sind, hatten schon immer das Verhalten, das für Version 6.9.4.0 oder höher beschrieben ist, d. h. sie haben den Wert „NULL“, wenn sie noch nicht angefasst wurden.

- **date (Datum)**
  Format und Genauigkeit können über Annotationen festgelegt werden.
- **enum (Sortierte Liste)**
  Für Sortierte Listen. Die Bearbeiter können im Web Client zwischen einem der Listenwerte wählen. Die Sortierten Listen und ihre Werte müssen zuvor in der Verwaltung von Sortierten Listen erstellt worden sein. Wählen Sie den gewünschten *Listentyp* und die *Listengruppe* in den Feldern darunter.
- **list (Liste)**
  Ein Feld dieses Datentyps ist der erste Schritt zur Erstellung einer Liste (eine Spalte) oder einer Tabelle (mehrere Spalten) von Eingabefeldern im Web Client.
  - Für eine **Tabelle** wird im nächsten Schritt ein anderes Feld des Typs *struct* erstellt (siehe unten), das später die Eingaben der einzelnen Listenfelder enthält (die die Spalten der Tabelle sind). Wenn Sie also eine Tabelle erstellen möchten, müssen Sie zuerst ein Feld des Typs *struct* erzeugen (siehe unten), bevor Sie die Felder für die Tabellenspalten hinzufügen können.
  - Für eine **einfache Liste** ist der nächste Schritt die Erstellung der Felder, die zu der Liste gehören. In diesem Fall ist kein *struct* erforderlich.

  Für alle Felder, die zu einer Liste oder Tabelle gehören, müssen Sie die Abhängigkeiten im Feld *Gehört zu* angeben (siehe unten). Ein Tabellenfeld (ein normales Datenfeld), gehört zum Beispiel immer zu einem *struct* und ein *struct* gehört immer zu einer *list*.

- **struct (Struktur)**
  Mit einem Datenfeld dieses Typs wird eine Datenstruktur (Zeile einer Tabelle) definiert, die ein oder mehrere Felder umfasst. Das ist der zweite Schritt zur Erstellung einer Tabelle, nachdem Sie zuerst ein Feld des Typs *list* erstellt haben. Im nächsten Schritt fügen Sie die Felder für die Spalten der Tabelle hinzu. Die Abhängigkeiten müssen für jedes Datenfeld im Feld *Gehört zu* (siehe unten) gesetzt werden, d. h. ein *struct* gehört immer zu einer *list*.

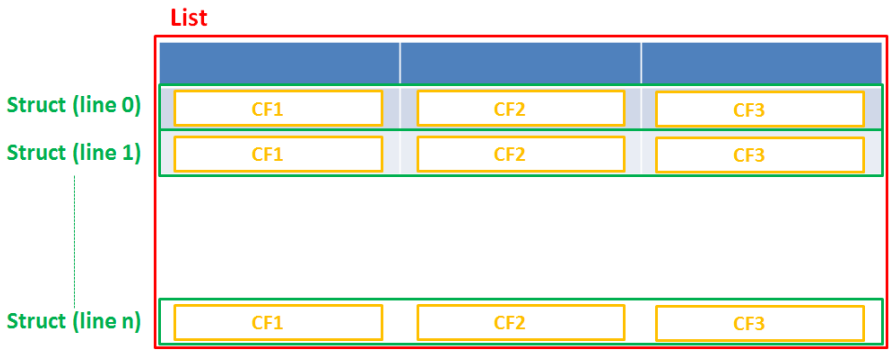


Abbildung 277: *Schema: List of Structs*

Technisch gesehen ist eine Liste ein Array, der in jedem Feld eine Map (= Paare von Schlüssel:Wert) enthält.

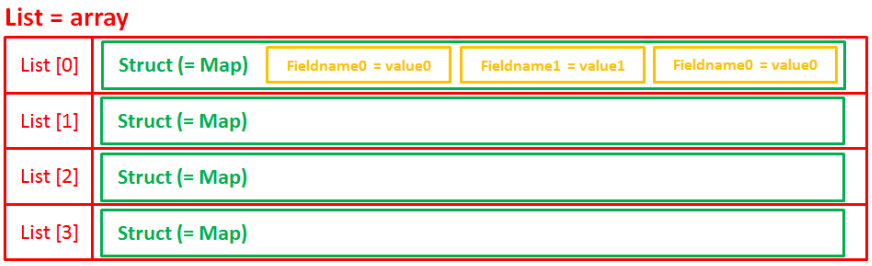


Abbildung 278: *List of structs, technisches Prinzip*

- **number (Zahl)**
  Für Ganzzahlen (Integer).
- **fixed point number (Festkommazahl)**
  Für Festkommazahlen, z. B. Währungen. Sie müssen die Gesamtanzahl der Ziffern (*Genauigkeit*) und die Anzahl der Ziffern nach dem Komma (*Skalierung*) in den entsprechenden Feldern eingeben.
- **string (Text)**
  Für bis zu 4000 alphanumerische Zeichen.

> ⚠ Beschränkung bei der Verwendung einer Oracle-Datenbank: Es können höchstens 4000 Bytes in UTF-Codierung gespeichert werden. Ab Oracle12c.

- **long string (Text)**
  Für große Objekte

  > ⚠ Beim Datentyp *long string (Text)* hängt die Grenze von dem für ConSol CM verwendeten Datenbanksystem ab: MS SQL Server: 2 GByte; MySQL: 4 GByte; Oracle: 16 - 64 GByte (je nach Page-Größe des Tablespace).

- **short string (Text)**
  Für bis zu 255 alphanumerische Zeichen.

  > ⓘ Bei Feldern des Datentyps *string (Text)* können Sie die Felddefinition mithilfe von Annotationen präzisieren. Ein Feld des Datentyps *string* kann zum Beispiel so definiert werden, dass es eine URL enthält und automatisch als Hyperlink angezeigt wird. Lesen Sie dazu den folgenden Abschnitt.

- **contact data reference (Referenz auf ein Kontaktdatenfeld)**
  Besonderer Datentyp, der intern als Referenz auf die mit dem Ticket verknüpften Kontakte verwendet wird. In FlexCDM (d. h. ab CM-Version 6.9.0), wird dieser Datentyp nicht mehr angezeigt, sondern nur noch intern im CM-System verwendet.
- **MLA field (Baum sortierter Listen)**
  Dieser Datentyp wird für Felder verwendet, die hierarchische Listen mit einer Baumstruktur enthalten, das sogenannte *MLA* (Multi Level Attributes). Der Name dieses Feldes ist der Name des neuen MLA, das in der MLA-Verwaltung angelegt werden muss. Die Gruppe des Feldes muss angegeben werden, nachdem das MLA erstellt wurde.

> ⚠ Der Datentyp, den Sie beim Erstellen eines Datenfeldes verwenden, kann danach nicht mehr verändert werden!

String-Felder werden häufig für Kunden-, Ticket- und Ressourcendaten verwendet und in Strings kann unterschiedlicher Inhalt gespeichert werden, zum Beispiel ein Textfeld mit einem Kommentar, ein einfaches Eingabefeld mit nur 20 Zeichen, eine URL oder ein Passwort. Das Anpassen von String-Feldern erfolgt mithilfe von speziellen Annotationen, die auf der Seite Annotationen des Anhangs aufgeführt sind. Da die Arbeit mit diesen Annotationen zu den alltäglichen Aufgaben eines CM-Administrators gehört, sind die wichtigsten und am häufigsten verwendeten Annotationen zusätzlich an dieser Stelle erklärt.

Wie kann ich ...

... statt einer einzelnen Zeile ein **Textfeld** einfügen?

Wert für Annotation `text-type`: „textarea“

Die Größe des Textfeldes kann angepasst werden. Es wird je nach Webbrowser als Standardtextfeld angezeigt. Verwenden Sie die Annotation `field-size`, wenn das Textfeld eine bestimmte Größe haben soll.

... die Eingabe des Feldes für **Passwörter** ausblenden?

Wert für Annotation `text-type`: „password“

Es werden nur Punkte angezeigt. Diese Annotation legt **nicht** fest, dass das Feld ein Passwort enthält! Es definiert nur den Anzeigemodus! Verwenden Sie die Annotation `password`, um ein String-Feld zu definieren, das das CM/Track-Passwort enthält.

... einen **Hyperlink** anzeigen, den Namen statt dem Link anzeigen?

Wert für Annotation `text-type`: „url“

Die Eingabe wird im Ansichtsmodus als Hyperlink angezeigt. Der String muss einem bestimmten URL-Muster entsprechen:

- "^((?:mailto\:|(?:(?:ht|f)tps?)\://)1\S+)(?: (?:\|)?(.*))?$"

Der erste Teil des Strings ist der Link (URL) und der zweite Teil ist der Name, der angezeigt werden soll.

Beispiel: "http://consol.de ConSol"

... einen **Dateilink** anzeigen?

Wert für Annotation `text-type`: „file-url“

Die Eingabe wird als Link auf eine Datei im Dateisystem angezeigt. Der Webbrowser muss solche Links zulassen/unterstützen!

Beispiel: Aktivieren von file://-URLs in einem Firefox-Browser

Fügen Sie die folgenden Zeilen entweder zur Konfigurationsdatei `prefs.js` oder zu `user.js` im Benutzerprofil hinzu. Auf einem Windows-System befindet sich diese normalerweise in einem Ordner wie `C:\Benutzer\<BENUTZERNAME>\AppData\Roaming\Mozilla\Firefox\Profiles\uvubg4fj.default`

- user_pref("capability.policy.localfilelinks.checkloaduri.enabled", "allAccess");
- user_pref("capability.policy.localfilelinks.sites", "http://cm-server.domain.com:8080");
- user_pref("capability.policy.policynames", "localfilelinks");

Alternativ kann ein Add-on des Firefox-Browsers wie ***Local Filesystem Links*** installiert werden, um einfacher auf die referenzierten Dateien und Ordner zugreifen zu können.

Dieser Link wird auch als Tooltip angezeigt.

Die URL ist korrekt formatiert, wenn folgende Bedingungen erfüllt sind:

- Sie beginnt mit `file:` gefolgt von normalen Schrägstrichen:
  - drei Schrägstriche "///" für Dateien, die sich auf demselben Computer befinden wie der Browser (alternativ "//localhost/") oder
  - zwei Schrägstriche gefolgt vom Servernamen und einem weiteren Schrägstrich für Dateien auf Dateiservern, die vom Rechner, auf dem der Webbrowser läuft, erreichbar sind.
- Danach folgt der vollständige Dateipfad, der mit dem Dateinamen endet.

- Auf Microsoft Windows-Systemen wird der Pfad ebenfalls mit normalen Schrägstrichen anstelle von umgekehrten Schrägstrichen geschrieben.
- Der Laufwerksbuchstabe eines lokalen Pfads auf Microsoft Windows-Systemen wird wie üblich verwendet, zum Beispiel `C:`.
- Pfade mit Leerzeichen und Sonderzeichen wie "{, }, ^, #, ?" müssen auf Microsoft Windows-Systemen mit Prozentzeichen kodiert werden (z. B. mit "%20" für ein Leerzeichen).

Beispiel-URLs:

- file://file-server/path/to/my/file.ext
- file:///linux/local/file.pdf
- file:///C:/Users/myuser/localfile.doc

Siehe auch die Erklärung von `file-url` im Abschnitt Liste der Feldannotationen

… ein **Label** definieren?

Wert für Annotation `text-type`: „label“

Dies ist ein schreibgeschütztes Feld, das grau angezeigt wird. Verwenden Sie die Annotation `label-group`, um das Label und die dazugehörigen Eingabefelder zu verknüpfen. Berücksichtigen Sie die Annotationen für Label (`show-label-in-edit`, `show-label-in-view`), bevor Sie spezielle Label-Felder implementieren!

… ein Feld für **gültige E-Mail-Adressen** definieren?

Wert für Annotation `email`: „true“

Das Feld darf nur gültige E-Mail-Adressen enthalten. Die Eingabe wird gemäß des Standardformats für E-Mail-Adressen validiert <name>@<domain>.

… eine skriptbasierte Autocomplete-Liste definieren?

Wert für die Annotation `text-type` = „autocomplete“

Optional: Wert für die Annotation `autocomplete-script` = <Name des entsprechenden Skripts>

Eine skriptbasierte Autocomplete-Liste wird verwendet, um ein Drop-down-Menü bereitzustellen, das anhand der Eingabe, die der Bearbeiter bereits vorgenommen hat, dynamisch gefüllt wird. Wenn der Benutzer zum Beispiel "Mei" eingibt, werden die möglichen Werte "Meier", "Meister" und "Meinert" als Liste angezeigt und der Bearbeiter kann den für das Feld erforderliche Wert auswählen. Sie kennen dieses Verhalten von anderen Autocomplete-Feldern, z. B. der Suche nach Bearbeitern für ein Ticket oder die Suche nach Kunden bei der Ticketerstellung. In diesen Fällen erzeugt CM die Liste allerdings automatisch und das Verhalten kann nicht beeinflusst oder angepasst werden. Im Gegensatz dazu können skriptbasierte Autocomplete-Listen vom CM-Administrator implementiert werden. Die Werte basieren auf einem Ergebnissatz, der dynamisch erstellt wird. Der Ergebnissatu kann Strings, Bearbeiter, Kunden (Units) und Ressourcen enthalten.

Eine detaillierte Beschreibung von skriptbasierten Autocomplete-Listen finden Sie im Abschnitt Skriptbasierte Autocomplete-Listen.

… ein Feld für das **CM/Track-Login** definieren?

Wert für Annotation `username`: „true“

Wird für die Authentifizierung beim CM/Track-Server verwendet. Nur für Kundenfelder in einem Kontaktobjekt.

… ein Feld für das **CM/Track-Passwort** definieren?

Wert für Annotation `password`: „true“

Wird (im Datenbankmodus) für die Authentifizierung beim CM/Track-Server verwendet. Nur für Kundenfelder in einem Kontaktobjekt.

… ein Feld für persönliche Daten definieren?

Wert für Annotation `personal-data`: „true“

Diese Annotation kann Ticket- und Kontaktfeldern zugewiesen werden. Kontaktfelder mit dieser Annotation werden gelöscht, wenn ein Kontakt anonymisiert wird. Ticketfelder mit dieser Annotation werden gelöscht, wenn der Hauptkunde des Tickets anonymisiert wird. Siehe Beispiel 8: Löschen von Kundendaten für Informationen über das Anonymisieren von Kontakten.

> ⚠ Beachten Sie bei der Definition von Feldern für persönliche Daten, dass das Löschen des Feldes während des Anonymisierungsprozesses wie eine normale Aktualisierung behandelt wird. Das heißt, Business-Event-Trigger, die auf Änderungen an Ticketfeldern reagieren, feuern und das Aktionsskript zur Kontaktaktualisierung wird ausgeführt.
>
> Dies kann zu unerwünschten Nebeneffekten führen.

Labels für Kundenfelder

In CM-Versionen vor 6.9.4 wurden die Labels von Kundenfeldern (z. B. Name, Adresse) im Web Client standardmäßig als Wasserzeichen angezeigt. Wenn ein anderes Label benötigt wird, muss ein eigenes Kundenfeld des Typs STRING mit der Annotation `text-type` = „label“ verwendet werden, wie in den folgenden Abbildungen dargestellt ist.

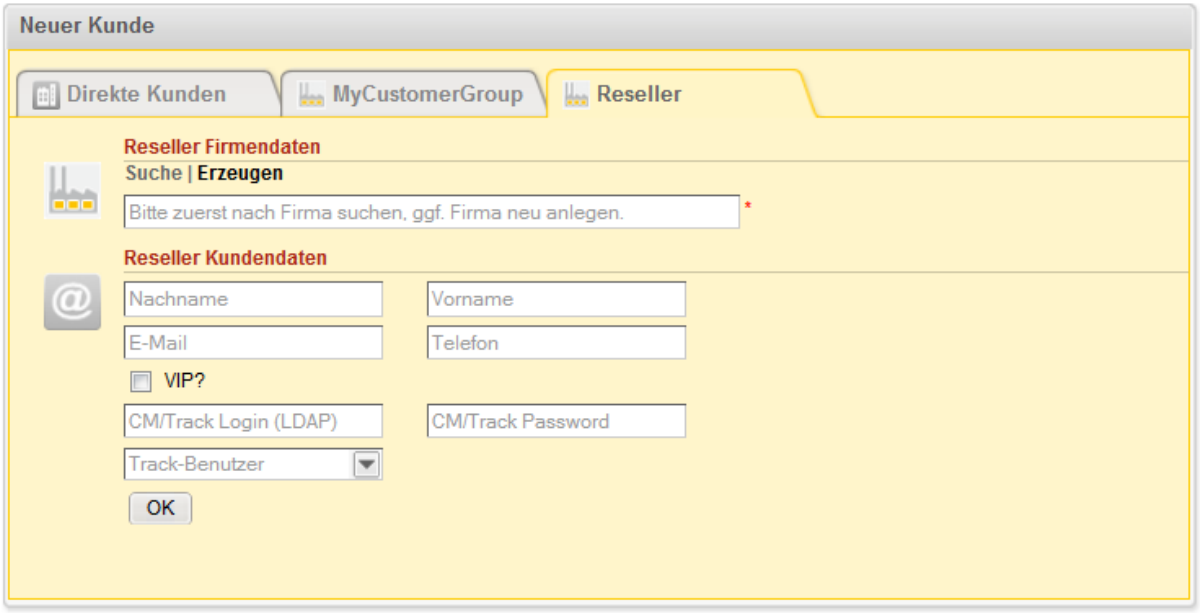


Abbildung 279: *ConSol CM Web Client - Standardanzeige der Labels von Kundenfeldern in Versionen vor 6.9.4*

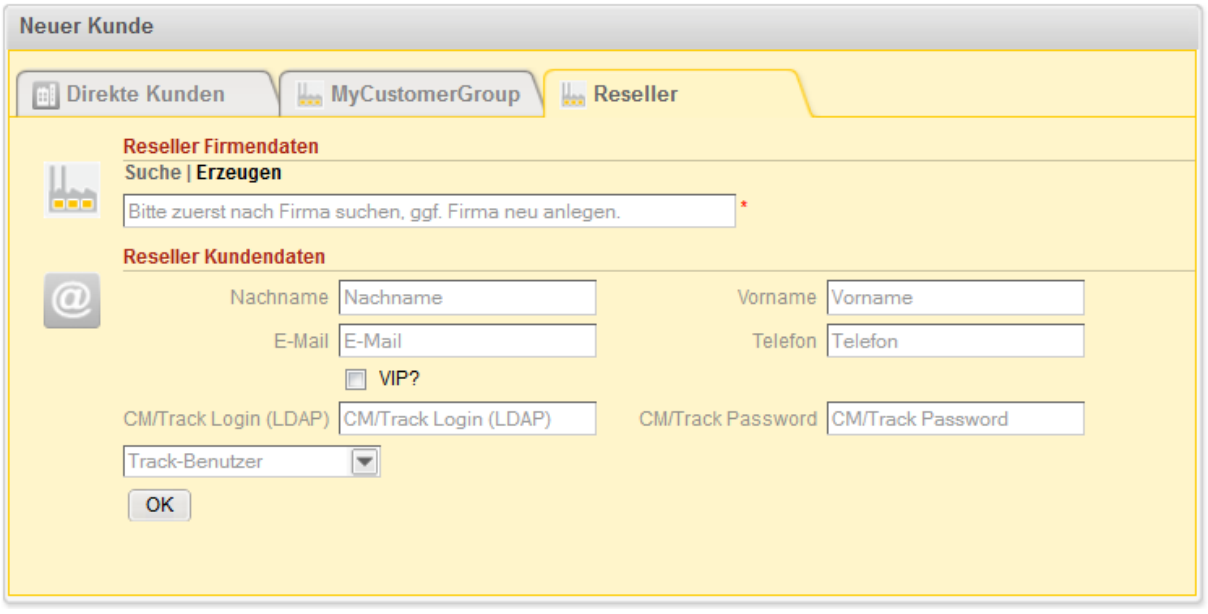


Abbildung 280: *ConSol CM Web Client - Label als eigenständige Kundenfelder in Versionen vor 6.9.4*

Ab CM-Version 6.9.4 können Sie als Administrator entscheiden, ob Wasserzeichen oder Labels (oder beides) verwendet werden sollen. Labels sind jetzt auf eine ähnliche Weise implementiert wie die Labels von Ticketfeldern (d. h. Datenfeldern für Ticketdaten). Außerdem können Tooltips hinzugefügt werden. Der Anzeigemodus der Labels von Kundenfeldern wird über Annotationen gesteuert. Annotationen können für Kundenfeldgruppen oder einzelne Kundenfelder gesetzt werden. Die Feldan-

notationen überschreiben dabei die Gruppenannotationen. Auf diese Weise können Sie ein Verhalten für die Anzeige der Gruppe definieren und es für ein oder mehrere Einzelfelder ändern. Dies ist im folgenden Beispiel erklärt.

Annotationen für Kundenfeldgruppen:

- layout:show-labels-in-view („true“, wenn nicht gesetzt)
- layout:show-labels-in-edit („true“, wenn nicht gesetzt)
- layout:show-watermarks („false“, wenn nicht gesetzt)
- layout:show-tooltips („true“, wenn nicht gesetzt)

Annotationen für Kundenfelder:

- layout:show-label-in-view
- layout:show-label-in-edit
- layout:show-watermark
- layout:show-tooltip

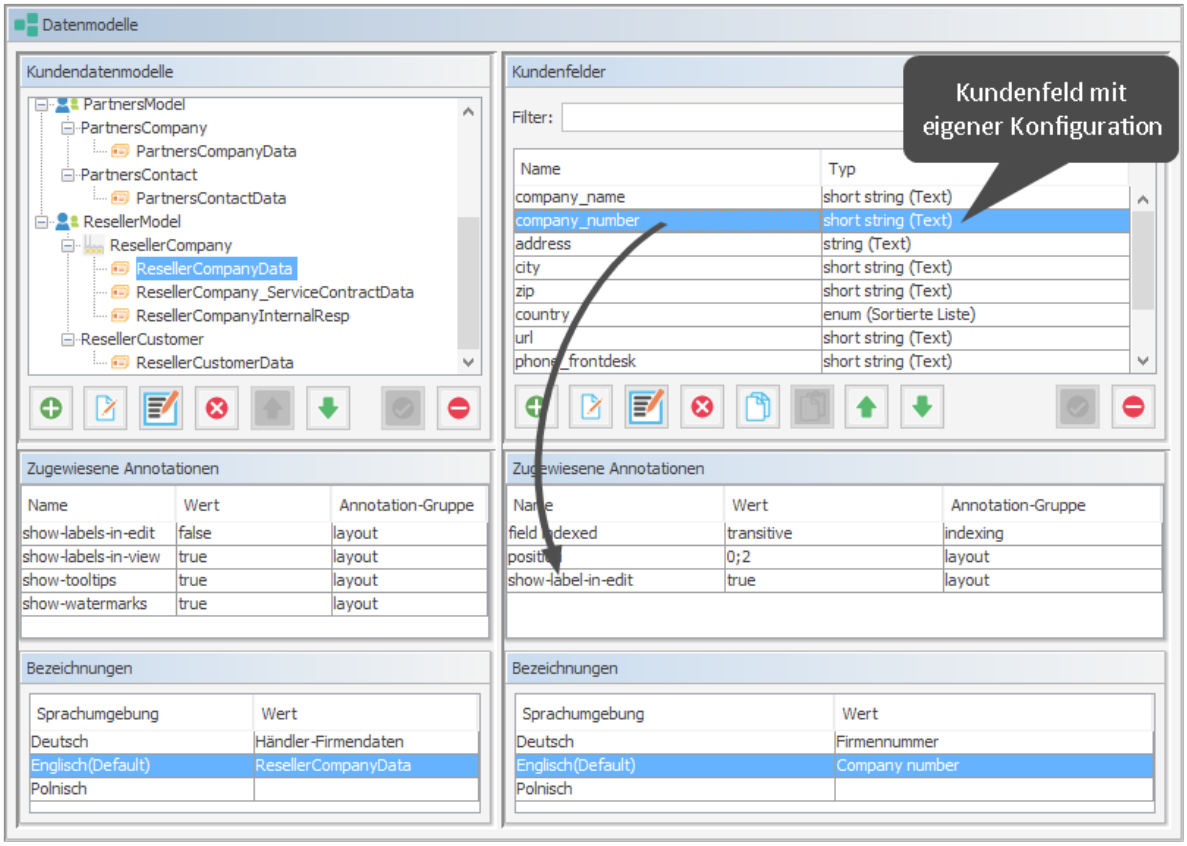


Abbildung 281: *ConSol CM Admin Tool - Kunden, Datenmodelle: Konfiguration einer Händler-Kundenfeldgruppe, ein Feld mit eigener Konfiguration*

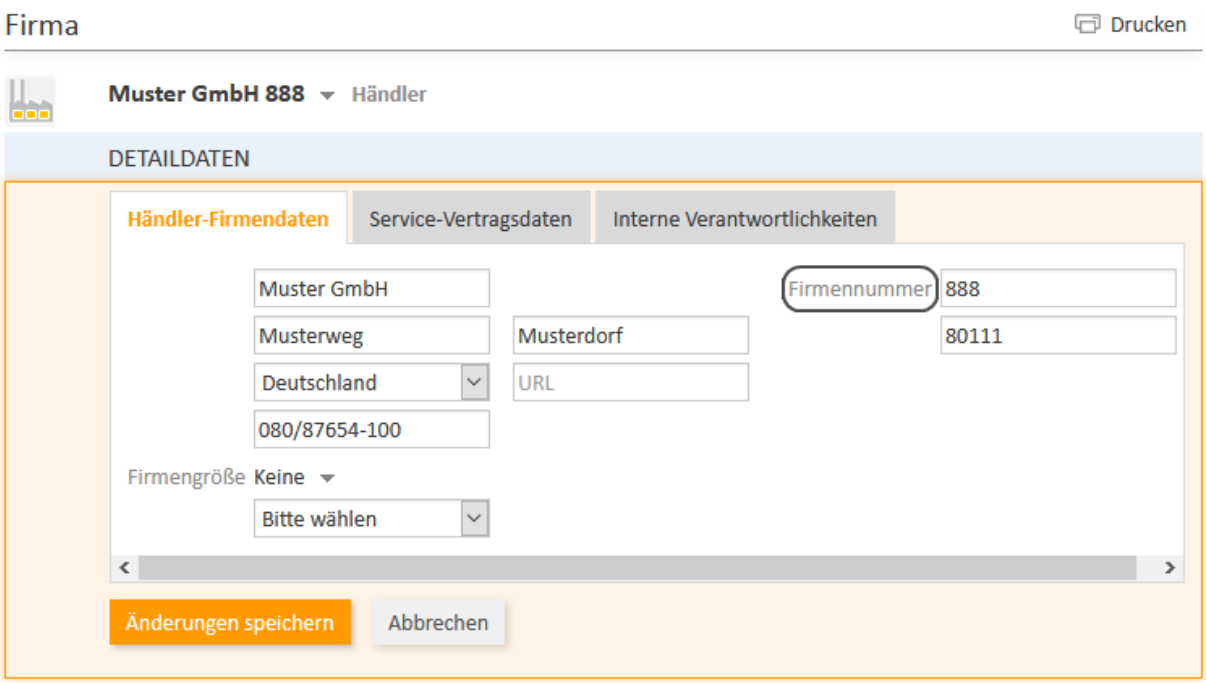


Abbildung 282: *ConSol CM Web Client - Ansicht der Kundenfeldgruppe Händler, ein Feld (company_number) mit eigener Annotation (Bearbeitungsmodus)*

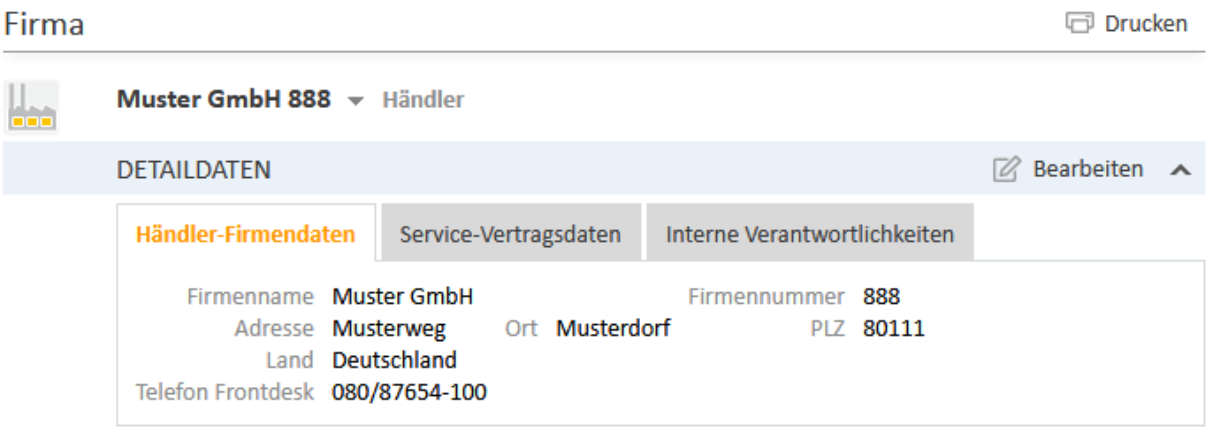


Abbildung 283: *ConSol CM Web Client - Ansicht der Kundenfeldgruppe Händler, ein Feld (company_number) mit eigener Annotation (Beispiel: keine Auswirkungen auf Ansichtsmodus)*

Die Labels werden automatisch eingefügt, sodass es am Ende bis zu sechs Spalten geben kann. Im folgenden Beispiel werden zwei Spalten verwendet.

- Feld_1: position 0;0
- Feld_2: position 0;1
- Feld_3: position 1;0
- Feld_4: position 1;1
- Feld_5: position 2;0
- Feld_6: position 2;1

Label_1	Feld_1	Label_2	Feld_2
Label_3	Feld_3	Label_4	Feld_4
Label_5	Feld_5	Label_6	Feld_6

Abbildung 284: *Beispiel für die Position von Kundenfeldern im Raster*

Label-Modus nach einem Update von einer CM-Version vor 6.9.4 auf Version 6.9.4 und höher

Das Layout des Kundenmodells wird beim Update beibehalten! Alle Annotationen werden entsprechend gesetzt und können später geändert werden.

Für alle vorhandenen Kundenfeldgruppen wird die layout-Annotation: `show-labels-in-edit` = „false“ (mit der die Standard-Labels im Bearbeitungsmodus ausgeblendet werden) beim Update gesetzt.

Standardmodus für neue Kundenfeldgruppen und Kundenfelder

Alle neuen Kundenfeldgruppen werden entsprechend der neuen Standardkonfiguration angezeigt. Das bedeutet:

- Standard-Labels
- Wasserzeichen deaktiviert
- Tooltip aktiviert

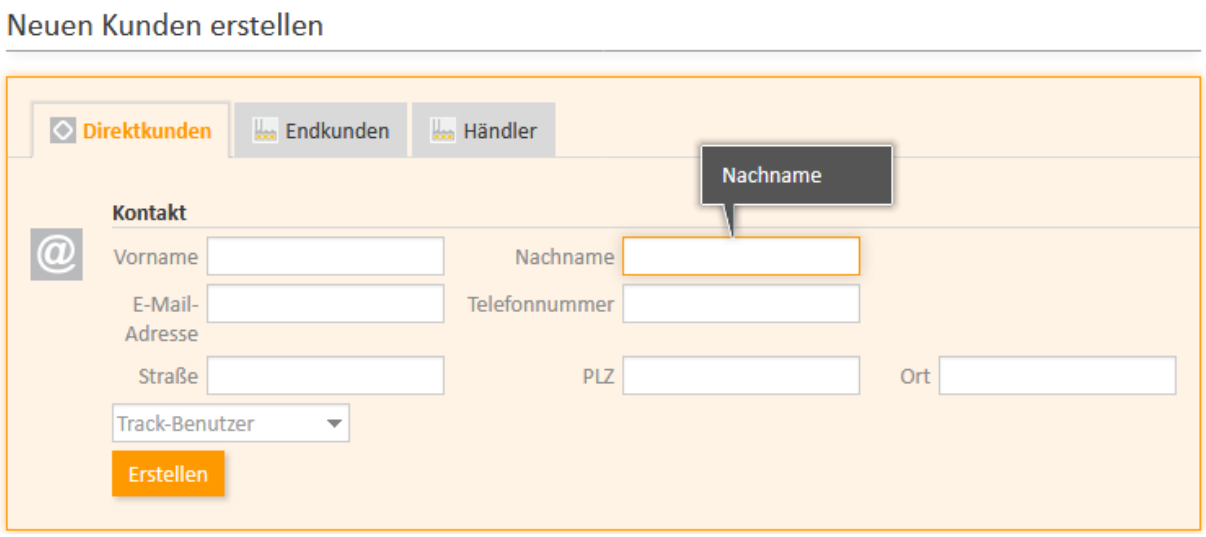


Abbildung 285: *ConSol CM Web Client - Standardanzeigemodus für Kundenfelder*

D.4.3 Verwendung von Objekten aus FlexCDM in Skripten



In diesem Handbuch werden die Begriffe *Kunde* und *Kundendefinition* verwendet. In früheren ConSol CM-Versionen wurde für Kunden der Begriff *Datenobjekt* verwendet. Die Namen der entsprechenden Java-Klassen sind *Unit* und *UnitDefinition*. Alle anderen Java-Klassen, die Kundenobjekte verarbeiten, heißen ebenfalls noch *Unit*... Beachten Sie dies bei Ihrer Arbeit als ConSol CM-Administrator und -Programmierer. Details finden Sie in der *ConSol CM Java API Doc*.

D.4.3.1 Neue Skriptfunktionen ab ConSol CM-Version 6.9

Bis ConSol CM-Version 6.8 konnte eine *Unit* nur eine Ticketfeldgruppe haben. Der Ausdruck `unit.get("name")` war immer gültig, weil ein Ticketfeld mit einem bestimmten Namen nur einmal vorkommen konnte, zum Beispiel *group1.name*.

Ab ConSol CM-Version 6.9 kann ein Kundenobjekt (*Unit*) ein oder mehrere Kundenfelder mit dem gleichen Namen haben. So kann sich z. B. *"name"* auf *"group1.name"* oder *"group2.name"* beziehen. In solchen Fällen ist der Ausdruck `Unit.get("name")` nicht gültig und führt zu einer Exception.



Für die Rückwärtskompatibilität funktioniert der Code `unit.get("name")`, solange das Kundenfeld *name* eindeutig ist. Wenn ein weiteres Kundenfeld mit dem gleichen Namen hinzugefügt wird, funktioniert der Code `unit.get("name")` nicht mehr!

Verwenden Sie folgende Notation, um Daten aus Unit-Feldern (Kundenfeldern) abzurufen:

- **Für ein Feld:**
  unit.get("group1:name")
- **Für mehrere Felder:**
  unit.get("group1:field1.group2:field2")

```
unit.get("contactFields:companyReference.companyFields:name")
```

Code-Beispiel 27: *Beispiel für den Abruf des Firmennamens eines Kontakts*

D.4.3.2 Erweiterung der Custom Field Expression Language (CFEL)

Ab ConSol CM-Version 6.9 wurde die Custom Field Expression Language (CFEL) verbessert, um einfachen Zugriff auf viele Objekte zu ermöglichen. Dies betrifft Skripte für Tickets und andere Objekte sowie Objekte aus dem Kundendatenmodell, d. h. Units (Kundenobjekte: Objekte auf Kontakt- oder Firmenstufe). Im Folgenden werden die Verbesserungen bezüglich Units (Kundenobjekten) erklärt. Eine detaillierte Beschreibung über das Schreiben von Java- oder Groovy-Code, der Objekte aus dem Kundendatenmodell verwendet, finden Sie im *ConSol CM Process Designer Handbuch*.

Sie können aus unterschiedlichen Skripten auf Objekte des Kundendatenmodells zugreifen:

- **Workflow:**
  - Skripte in Workflow-Aktivitäten
  - Skripte in Workflow-Bedingungen
- **Admin-Tool-Skripte des Typs:**
  - Abhängige sortierte Listen
  - E-Mail
  - Duplizieren
  - Standardwerte
  - Kundenaktion
  - Kundenbedingung
  - Workflow

D.4.3.3 Convenience-Methoden

Beispiele für Convenience-Methoden:

Object.Method	Erklärung
`def contacts = unit.get ("contacts()")` `List contacts = company.getContacts()`	Mit CFEL `("contacts()")` wird eine Liste aller Kontakt aus der Firma (Unit) abgerufen.
`Unit company = mainContact.getCompany()`	Für einen Kontakt kann die Firma einfach abgerufen werden.
`newContact.set("company ()", newCompany)`	Für einen (neuen) Kontakt wird der Firma der CFEL-Ausdruck `"company()"` zugewiesen, der einen einfachen Zugriff auf das Firmenobjekt ermöglicht.
`List tickets = company.get ("tickets()")`	Es werden alle Tickets für eine Firma abgerufen.

`Ticket ticket = getTicket ();`
`Unit mainContact = ticket.getMainContact()`
`List tickets = mainContact.get("tickets ()")`
	Es werden alle Tickets für einen Kontakt abgerufen.

Object.Method	Erklärung
`Integer count = contact.get("company().contacts()[0].tickets()[count]");`	Ein Kettenausdruck, mit dem die Anzahl der Tickets für einen bestimmten Kontakt abgerufen wird.

```
TicketCriteria ticketCriteria = new TicketCriteria();
Unit patternContact = new Unit("contact", customerGroup);
mdcmCriteriaBuilder.setReferencedContactCriteria(ticketCriteria, patternContact);
```

Code-Beispiel 28: *Beispiel 1: Suche nach den Tickets eines Kontakts oder einer Firma*

```
TicketCriteria ticketCriteria = new TicketCriteria();
Unit contactPattern = new Unit("contact", customerGroup);
mdcmCriteriaBuilder.setReferencedContactCriteria(ticketCriteria, contactPattern);
Unit companyPattern = new Unit("company", customerGroup);
companyPattern.setFieldValue("name", „ConSol");
mdcmCriteriaBuilder.setReferencedCompanyCriteria(contactPattern, companyPattern);
```

Code-Beispiel 29: *Suche nach den Tickets eines Kontakts, der zu einer bestimmten Firma gehört*

```
UnitCriteria unitCriteria = new UnitCriteria();
Unit companyPattern = new Unit("company", customerGroup);
mdcmCriteriaBuilder.setReferencedCompanyCriteria(unitCriteria, companyPattern);
```

Code-Beispiel 30: *Suche nach Kontakten einer bestimmten Firma*

Detaillierte Informationen über die Methoden einschließlich der Eingabeparameter (Methodensignaturen) und der Ausgabedatentypen finden Sie in der *ConSol CM Java API Doc*.

D.4.3.4 Wichtige Objekte

Welche Objekte im Skript verfügbar sind, hängt natürlich vom Kontext des Skripts ab. Die folgenden Beispiele zeigen einige mögliche Anwendungsfälle:

Ausgangspunkt	Skript	Objekte	Beispiel
Firmenseite	Kundenaktionsskript	unit steht für die Firma	`def contacts = unit.get("contacts()")`

Ausgangspunkt	Skript	Objekte	Beispiel
Kontaktseite	Kundenaktionsskript	unit steht für den Kontakt	`List tickets = unit.get("tickets()")`
Workflow-Aktivität	Skript für Workflow-Aktion oder -Bedingung	ticket	`def id = ticket.getId()`
Workflow-Aktivität mit Admin-Tool-Skript	Skript für Workflow-Aktion oder -Bedingung	ticket ist nicht implizit vorhanden!	`import com.consol.cmas.common.model.ticket.Ticket` `def id = ticket.getId()`

D.4.4 Verwenden von skriptbasierter Feldvisualisierung für Kundenfelder

Mit der skriptbasierten Feldvisualisierung können Sie die Anzeige von Daten in Kundenfeldern verbessern. Details dazu finden Sie im Abschnitt Skripte des Typs Feld-Visualisierung.

D.5 Templates für Kundendaten

In diesem Kapitel werden folgende Themen behandelt:

D.5.1 Einführung in die Verwendung von Templates für die Anzeige von Kundendaten 428

D.5.2 Coding von Templates 430

D.5.3 Lokalisieren von Werten von Sortierten Listen in Kunden-Templates 431

D.5.4 Abkürzen von Werten in Kunden-Templates 432

D.5.5 Template-Typen 434

D.5.1 Einführung in die Verwendung von Templates für die Anzeige von Kundendaten

Im ConSol CM Web Client werden Kundendatensätze an mehreren Stellen in der auf den Templates basierenden Kurzform angezeigt. In der Ticketliste wird möglicherweise der Kontaktname und der Firmenname benötigt, wohingegen im Kundendatenbereich des Tickets der Name, Vorname und die Telefonnummer eines Kontakts gebraucht werden. In diesem Abschnitt wird erklärt, wo die Kurzformen verwendet werden, und wie Sie die zugehörigen Templates mit dem Admin Tool konfigurieren können.

Die Konfiguration basiert auf folgendem Prinzip:

- Im Navigationselement *Datenmodelle* der Navigationsgruppe *Kunden* im Admin Tool werden die Templates einem Kundenobjekt, d. h. einer Kontakt- oder Firmendefinition zugewiesen. Diese Templates steuern die Anzeige an bestimmten Stellen des Web Clients. Wie die folgende Abbildung zeigt, können verschiedene Template-Typen definiert werden. Ein Template vom Typ *Standard* ist immer erforderlich, da es als Ersatz für alle Template-Typen dient, außer REST. Wenn die REST-API verwendet wird (entweder direkt oder über CM/Track), muss auch ein REST-Template definiert werden.
- Die referenzierten Templates müssen in der Navigationsgruppe *System*, Navigationselement *Skripte und Templates*, Tab *Templates* des Admin Tools gespeichert sein. Den Namen eines Templates kann der Administrator selbst festlegen.

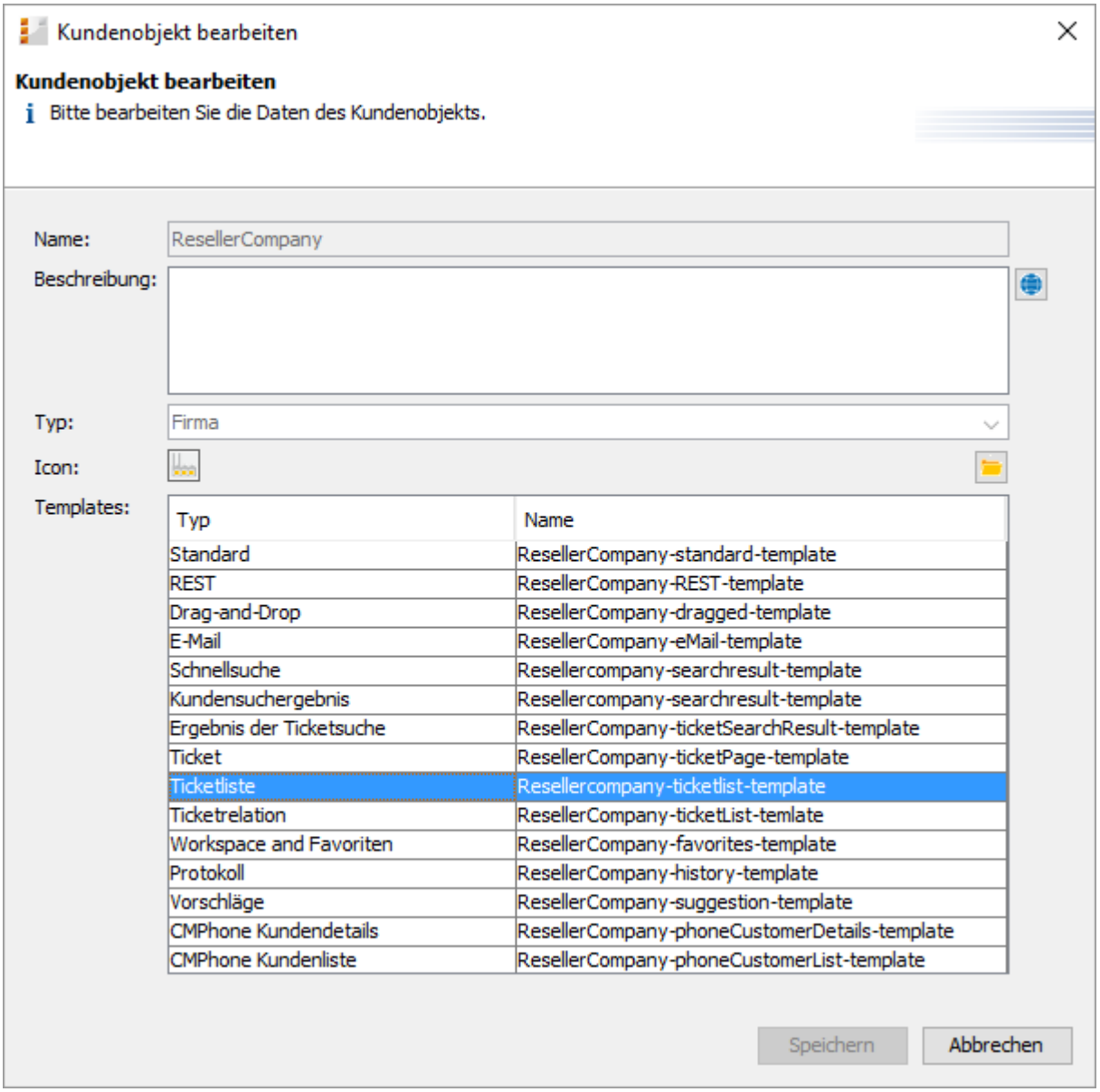


Abbildung 286: *ConSol CM Admin Tool - Kunden, Datenmodelle: Template-Zuweisungen für Kundenobjekt (hier: Firma)*

In den folgenden Absätzen werden die Syntax und das Coding für Templates sowie alle möglichen Template-Typen erklärt.

D.5.2 Coding von Templates

Die Templates werden in der *FreeMarker-Notation* geschrieben. Detaillierte Informationen dazu finden Sie auf der FreeMarker-Website.

Innerhalb der Templates arbeiten Sie mit drei Objekttypen:

1. **Name des Kundenobjekts**
   d. h. der Name des Firmen- oder Kontaktobjekts
2. **Name der Kundenfeldgruppe**
   innerhalb des Kundenobjekts (eine Kundenfeldgruppe ist ähnlich wie eine Ticketfeldgruppe für Ticketdaten)
3. **Name des Kundenfeldes**
   innerhalb der Kundenfeldgruppe

Die folgende Abbildung zeigt ein Beispiel dafür:

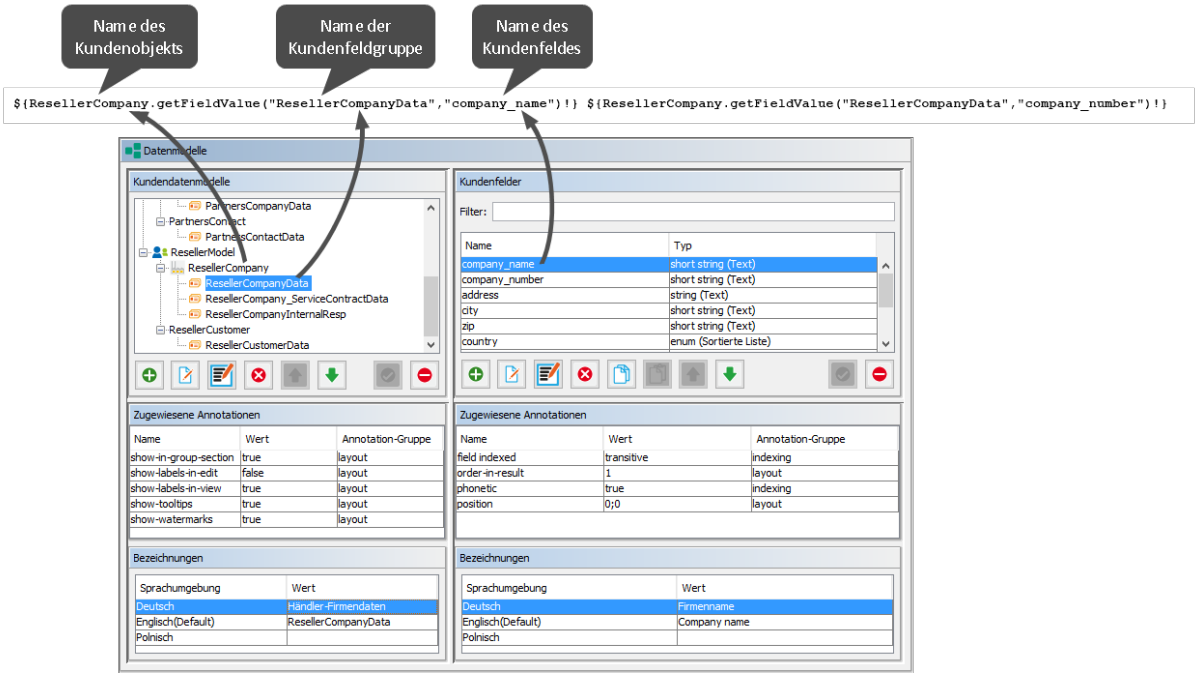


Abbildung 287: *ConSol CM Admin Tool - Kunden, Datenmodelle: Schreiben von Kunden-Templates*

⚠ Die Kunden-Templates dürfen nur eine Zeile haben! Zeilenumbrüche sind nicht zulässig!

D.5.2.1 Beispiel für Templates

Hier sind einige Beispiele für Templates:

```
<#if ResellerCustomer.getFieldValue("ResellerCustomerData","customer_name")?has_
 content &&
 ResellerCustomer.getFieldValue("ResellerCustomerData","firstname")?has_content>
 ${ResellerCustomer.getFieldValue("ResellerCustomerData","customer_name")!},
 ${ResellerCustomer.getFieldValue("ResellerCustomerData","firstname")!}
 <#else> ${ResellerCustomer.getFieldValue("ResellerCustomerData","customer_
 name")!}</#if>
```

Code-Beispiel 31: *Beispiel für ein Kunden-Template mit dem Kundenname (muss in eine Zeile geschrieben werden!)*

```
${company.getFieldValue("company", "name1")!}${company.getFieldValue("company",
 "name2")!}
 ${company.getFieldValue("company", "mainaddr_city")!},${customer.getFieldValue
 ("customer", "firstname")!}
 ${customer.getFieldValue("customer", "name")!}
```

Code-Beispiel 32: *Beispiel für ein Template mit dem Firmenkontakt (muss in eine Zeile geschrieben werden!)*

```
<#if company??>${company.getFieldValue("company", "name1")!}${company.getFieldValue
 ("company", "name2")!}
 ${company.getFieldValue("company", "mainaddr_city")!},
 </#if>${customer.getFieldValue("customer", "firstname")!}${customer.getFieldValue
 ("customer", "name")!}
```

Code-Beispiel 33: *Beispiel für ein Kundensuche-Template (muss in einer Zeile geschrieben werden!)*

```
<#setting number_format="#"/>${customerModelCompany.getFieldValue("groupName",
 "numberValueField")!}
```

Code-Beispiel 34: *Einstellen des Zahlenformats: Entfernen von "." in der Anzeige von Zahlen*

D.5.3 Lokalisieren von Werten von Sortierten Listen in Kunden-Templates

Ab ConSol CM-Version 6.10.5.4 können Werte von Sortierten Listen (Enums) lokalisiert werden, d. h. Sie können die lokalisierten Werte im Web Client anzeigen lassen. Die Sortierte Liste *salutation* enthält

die technischen Werte „mr“ und „mrs“, aber im Web Client soll die richtige Anrede in der im Browser eingestellten Sprache angezeigt werden. Wenn im Browser keine Sprache eingestellt ist, wird die Standardsprache von CM verwendet.

Das folgende Beispiel zeigt dies anhand der Sortierten Liste *salutation*.

```
${localize(customer.getFieldValue("customer", "salutation"))!}
 ${customer.getFieldValue("customer", "firstname")!} ${customer.getFieldValue
 ("customer", "name")!}
```

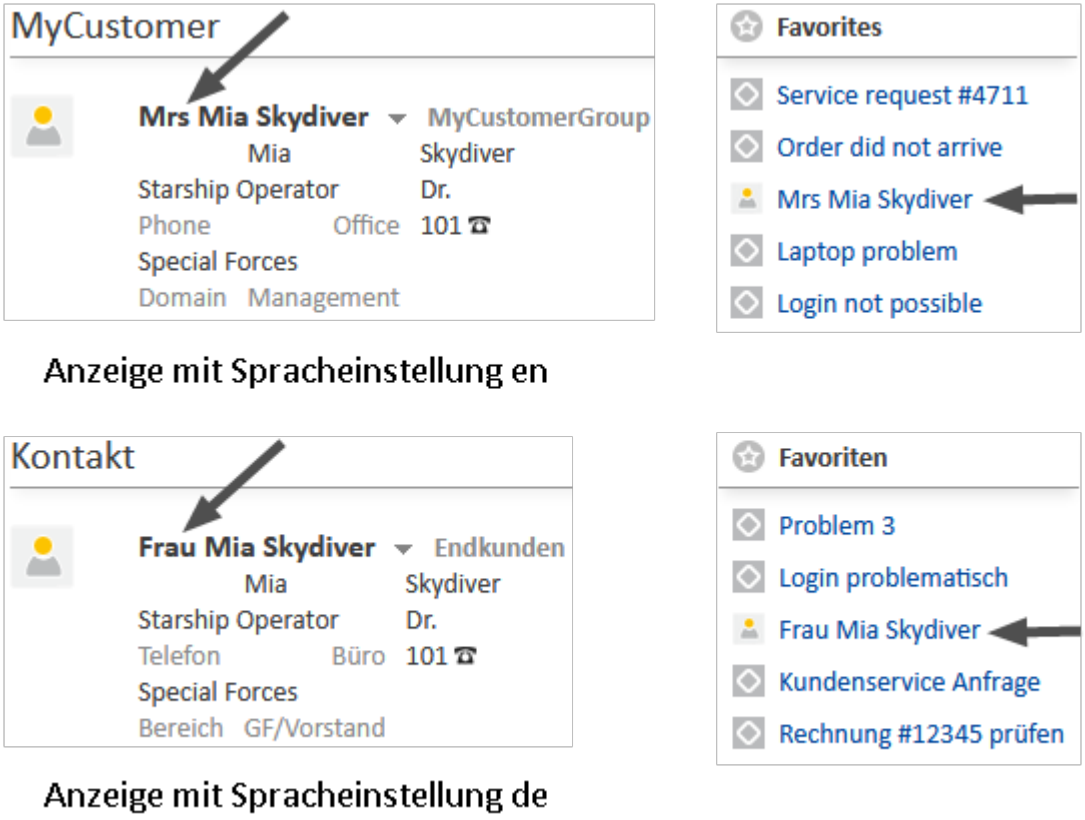
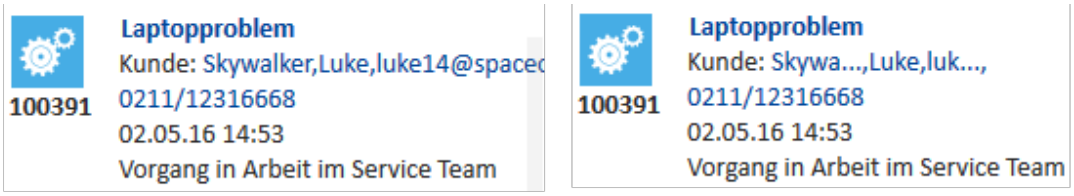


Abbildung 288: *ConSol CM Web Client: Anzeige eines Wertes einer Sortierten Liste mittels Kunden-Template*

D.5.4 Abkürzen von Werten in Kunden-Templates

Um zu vermeiden, dass Werte aus Templates nicht angezeigt werden, da auf der GUI nicht ausreichend Platz vorhanden ist, kann jeder Wert mit der Methode `abbreviate()` abgekürzt werden. Auf diese Weise können alle Felder des Templates auch dann angezeigt werden, wenn die ersten Felder des Templates sehr lange Werte enthalten. Dies ist anhand des folgenden Beispiels genauer erklärt.



Nicht abgekürzt

Mit dem unten gezeigten Template abgekürzt

```
${abbreviate(ResellerCustomer.getFieldValue
("ResellerCustomerData","customer_name"),8)!},${ResellerCustom
er.getFieldValue
("ResellerCustomerData","forename")!},${abbreviate
(ResellerCustomer.getFieldValue
("ResellerCustomerData","email"),6)!},
```

Die Methode `abbreviate()` benötigt zwei Parameter:

- Wert, der abgekürzt werden soll.
- Anzahl der anzuzeigenden Zeichen. Beachten Sie, dass dies die Anzahl aller anzuzeigenden Zeichen ist, d. h. einschließlich der drei Punkte. Wenn also ein Buchstabe und drei Punkte angezeigt werden sollen, muss „4“ als Parameter angegeben werden, 4 ist auch der Mindestwert.

D.5.5 Template-Typen

D.5.5.1 Standard

Dieses Template, bzw. Format, wird an allen folgenden Stellen verwendet, für die keine speziellen Templates definiert wurden, d. h. alle anderen speziellen Templates können weggelassen werden, wenn ein Standard-Template definiert ist.

> ⚠ Wenn für eine bestimmte Stelle im Web Client kein Template definiert ist und es auch kein Standard-Template gibt, wird eine Fehlermeldung in die Log-Datei geschrieben und im Web Client wird an dieser Stelle -- *unknown* -- angezeigt.
>
> Stellen Sie also sicher, dass mindestens ein Standard-Template definiert wird. Bei einem zweistufigen Kundendatenmodell muss dies für die Firmenstufe und für die Kontaktstufe erfolgen!

D.5.5.2 REST

Dieses Template konfiguriert die Darstellung der Kundendaten über die REST-API. In der Standardkonfiguration werden keine Kundendaten im CM-Portal CM/Track angezeigt, das auf der REST-API basiert. Dieses Template gilt nur dann, wenn die REST-API direkt angesprochen wird, z. B. bei der Programmierung von CM-Schnittstellen. Die folgenden Beispiele zeigen eine Abfrage von Kundendaten durch einen REST-Client und die entsprechende Antwort des CM-Servers über die REST-API. Der Wert im Tag <mark> der XML-Ausgabe ist die Kundeninformation, die mit dem REST-Template formatiert wurde. In diesem Beispiel umfasst das Template den Firmennamen (*Spaceoddity*) und die Firmennummer (*42*).

```
${ResellerCompany.getFieldValue("ResellerCompanyData","company_name")!}
 ${ResellerCompany.getFieldValue("ResellerCompanyData","company_number")!}
```

Code-Beispiel 35: *REST-Template*

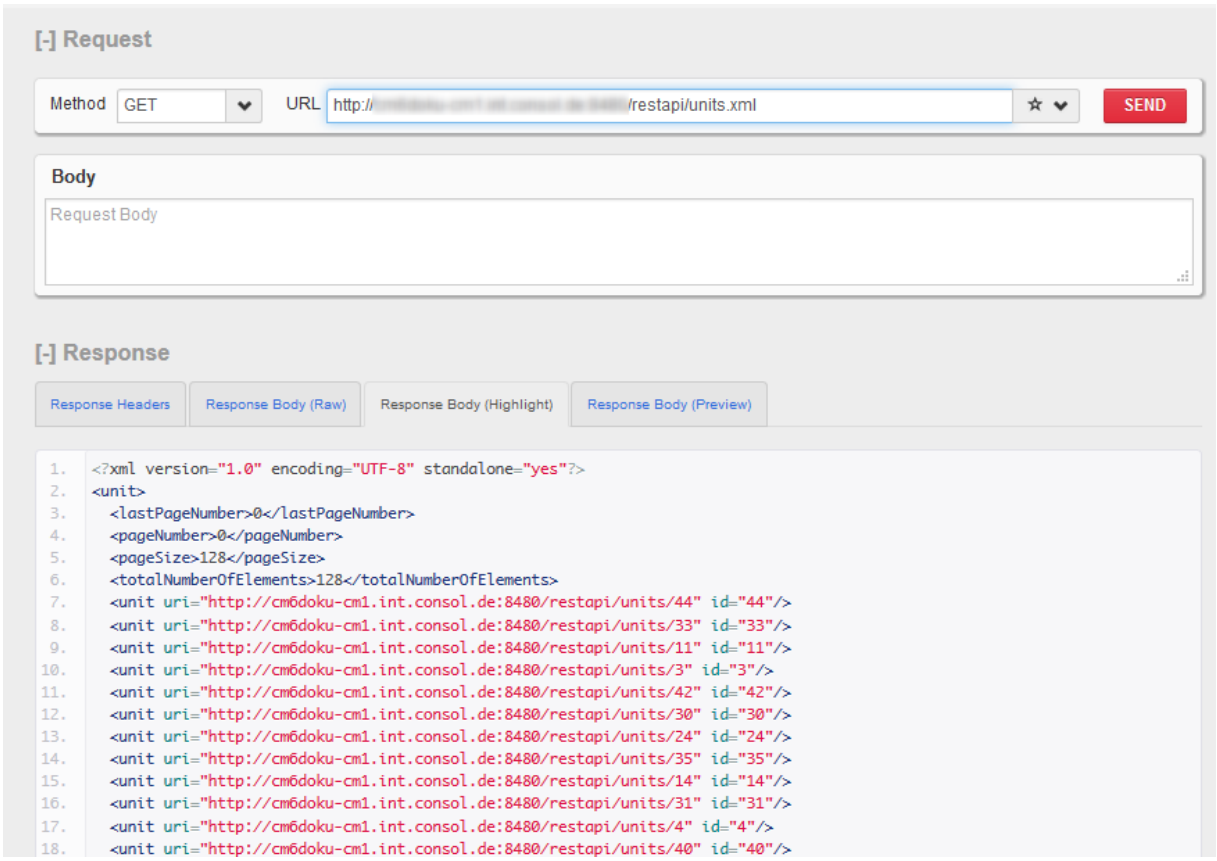


Abbildung 289: *REST API - Abfrage aller Units (die Abbildung zeigt nur einen Teil der Liste)*

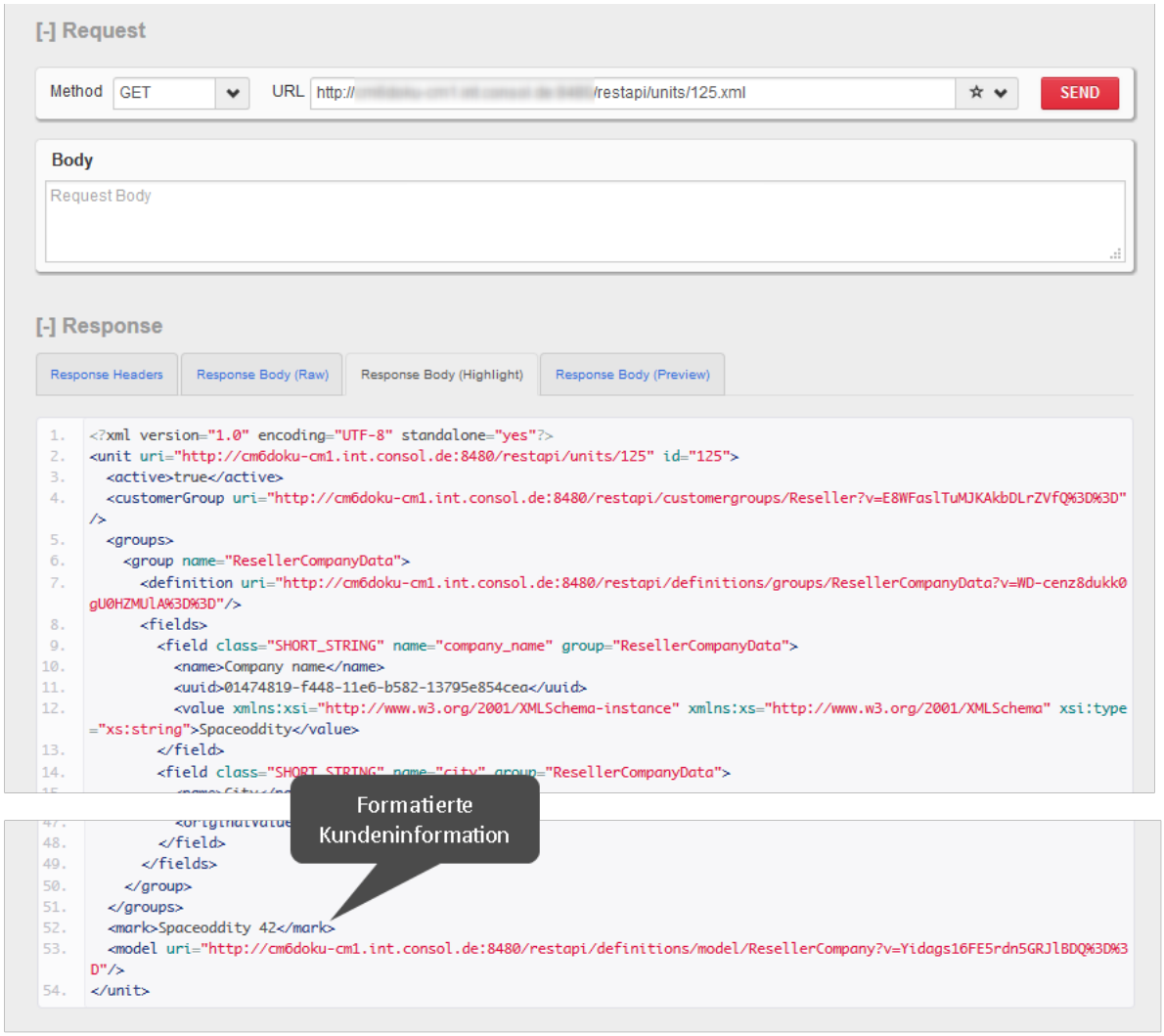


Abbildung 290: *REST API - Abfrage einer Unit (ID 125) als XML-Ausgabe*

> Ab CM Version 6.10.3 unterstützt die ConSol CM REST API zahlreiche Request-Parameter bezüglich Objekten aus CM/Resource Pool. Wenn möchten Sie diese Objekte via REST API ansprechen möchten, stellen Sie bitte sicher, dass für jeden Ressourcentyp ein REST-Template oder wenigstens ein Standard-Template zur Verfügung steht.
>
> Beachten Sie: CM/Track nutzt ebenfalls die REST API und benötigt somit REST-Templates.

D.5.5.3 Drag-and-drop

Mit diesem Template wird das Format der Kundendaten definiert, das beim Ziehen von Datensätzen verwendet wird, zum Beispiel vom Bereich *Kunden* in die *Favoriten*.

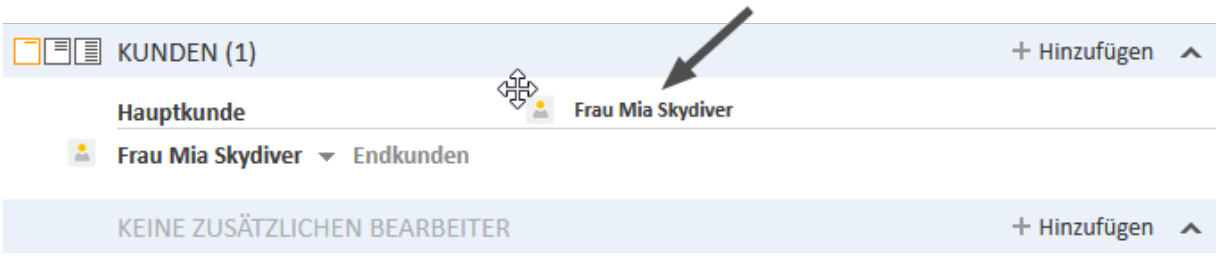


Abbildung 291: *ConSol CM Web Client - Kunden-Template vom Typ Drag-and-drop*

D.5.5.4 E-Mail

Der Ticket-E-Mail-Editor liefert automatische Suchergebnisse in Form einer Drop-down-Liste. Das Format dieser Suchergebnisse kann mit diesem Template konfiguriert werden.

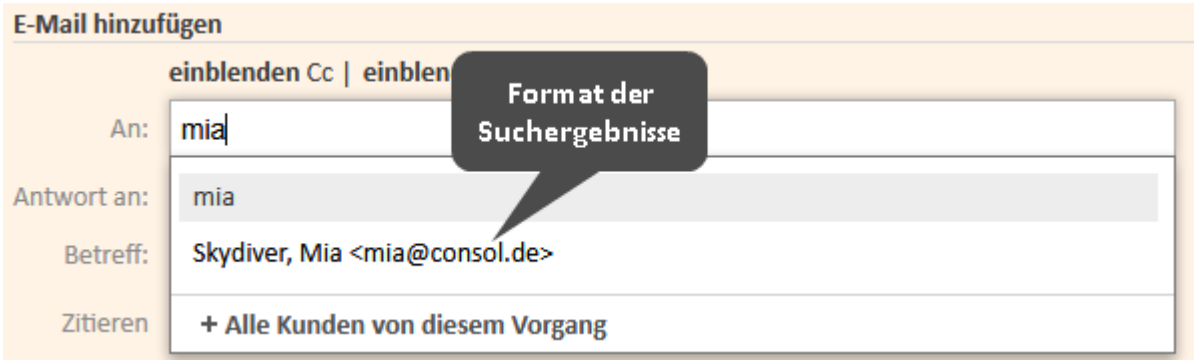


Abbildung 292: *ConSol CM Web Client - Kunden-Template vom Typ E-Mail*

```
<#if customer.getFieldValue("customer","name")?has_content
 && customer.getFieldValue("customer","firstname")?has_content
 && customer.getFieldValue("customer","division")?has_content>
 ${customer.getFieldValue("customer","name")!},
 ${customer.getFieldValue("customer","firstname")!},
 ${customer.getFieldValue("customer","division")!}
 <#else> ${customer.getFieldValue("customer","name")!},
 ${customer.getFieldValue("customer","division")!}</#if>
```

Code-Beispiel 36: *E-Mail-Template (muss in eine Zeile geschrieben werden!)*

D.5.5.5 Schnellsuche

Mit diesem Template wird das Format der Suchergebnisse für Kontakte und Firmen in der Schnellsuche definiert. Das Template muss kurz und einzeilig sein.

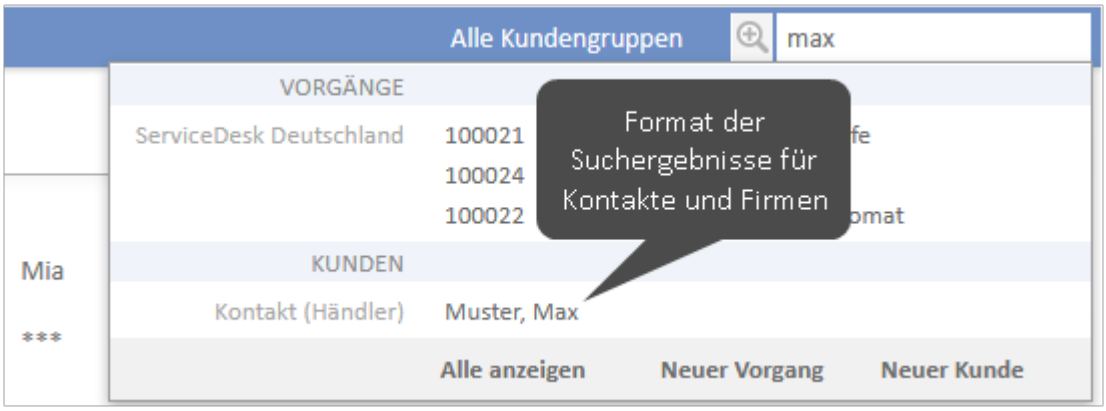


Abbildung 293: *ConSol CM Web Client - Kunden-Template vom Typ Schnellsuche*

D.5.5.6 Kundensuchergebnis

Mit diesem Template werden die Suchergebnisse für automatische Suchen in Autovervollständigen-Feldern definiert.

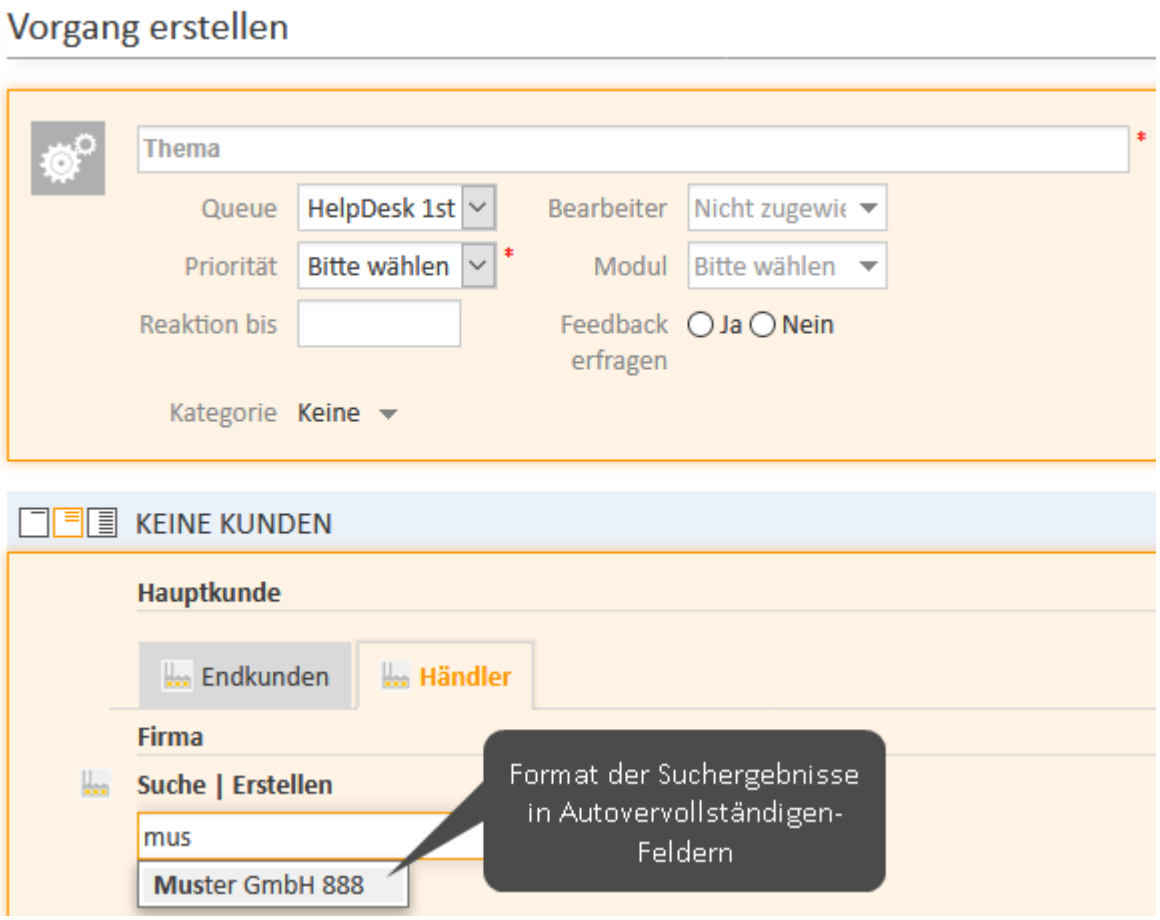


Abbildung 294: *ConSol CM Web Client - Kunden-Template vom Typ Kundensuchergebnis*

Folgendes Template wird angewendet:

```
${ResellerCompany.getFieldValue("ResellerCompanyData","company_name")!}
 ${ResellerCompany.getFieldValue("ResellerCompanyData","company_number")!}
```

Code-Beispiel 37: *Suchergebnis-Template von ResellerCompany (muss in eine Zeile geschrieben werden!)*

D.5.5.7 Ergebnis der Ticketsuche

Auf der Ergebnisseite der Detailsuche werden die gefundenen Tickets in einer Liste angezeigt. Eine Spalte dieser Liste enthält den Hauptkunden des Tickets. Das Template *Ergebnis der Ticketsuche* definiert das Layout der Kundendaten in dieser Spalte.

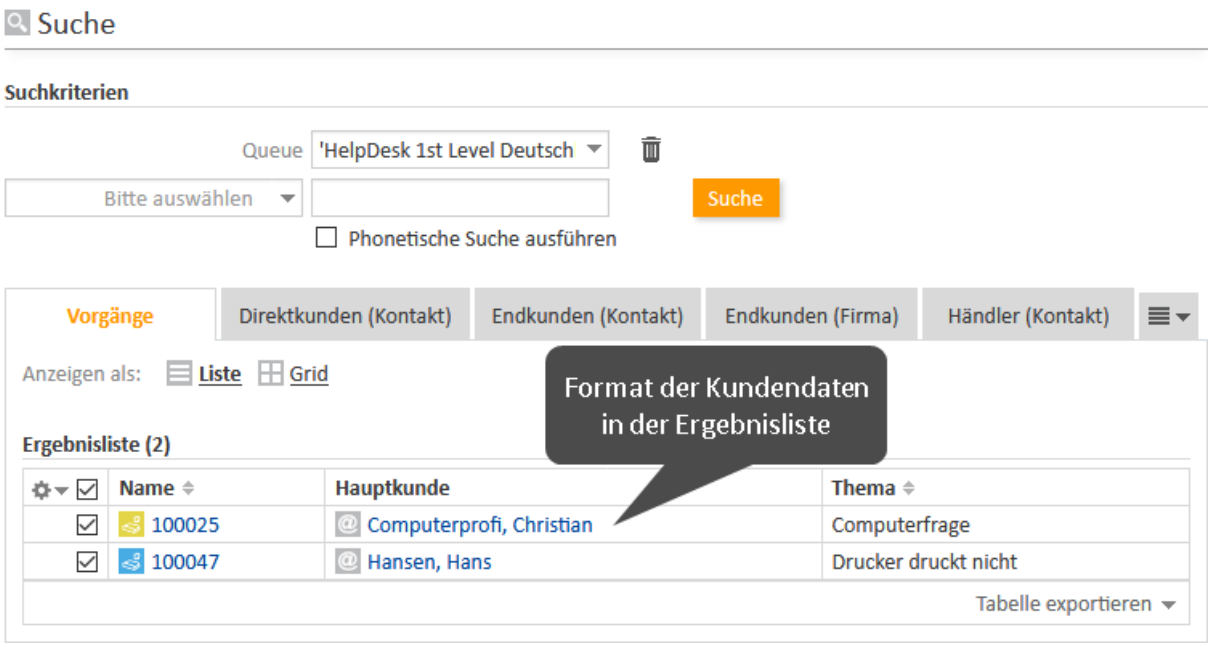


Abbildung 295: *ConSol CM Web Client - Kunden-Template vom Typ Ergebnis der Ticketsuche*

D.5.5.8 Ticket

Mit diesem Template wird die Darstellung der Kundendaten im *Kundenbereich* eines Tickets definiert. Die Auswirkung dieses Templates hängt von der ConSol CM-Version ab:

- In CM-Versionen 6.10 und niedriger steuert das Template die Darstellung der Kundendaten nur auf dem kleinsten Sichtbarkeitslevel. Im mittleren und großen Sichtbarkeitslevel wird die erste Zeile (Positionen 0;x) der Definition der Kundenfelder angezeigt. Dies greift nur, wenn die Annotation `position` tatsächlich gesetzt ist. Wenn keine Annotation `position` gesetzt ist und der Web Client die Anordnung der Ticketfelder automatisch rendert, funktioniert dieser Mechanismus nicht und die Datenfelder der ersten Zeile werden nicht als Template verwendet!
- In CM-Versionen 6.11.0 und höher wird das Template auf allen drei Levels verwendet.

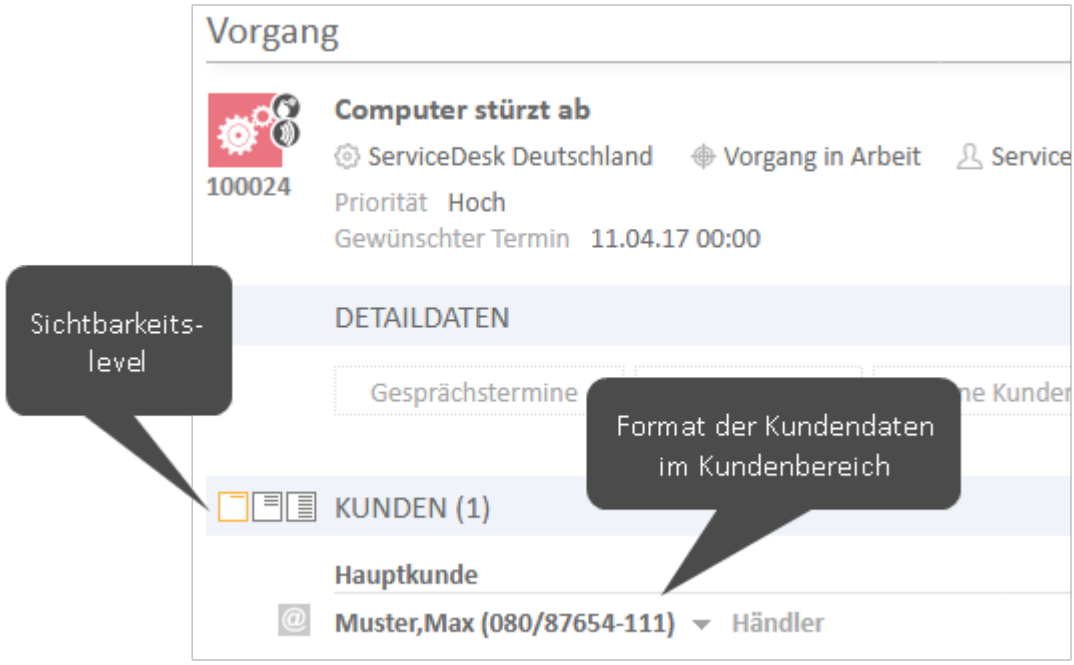


Abbildung 296: *ConSol CM Web Client - Kunden-Template vom Typ Ticket*

In ConSol CM-Versionen 6.11 und höher wird das Template *Ticket* auch zur Darstellung der Kundendaten auf den Kundenseiten verwendet, d. h. auf Kontakt- und Firmenseiten. Die folgende Abbildung zeigt dies am Beispiel einer Kundenseite.

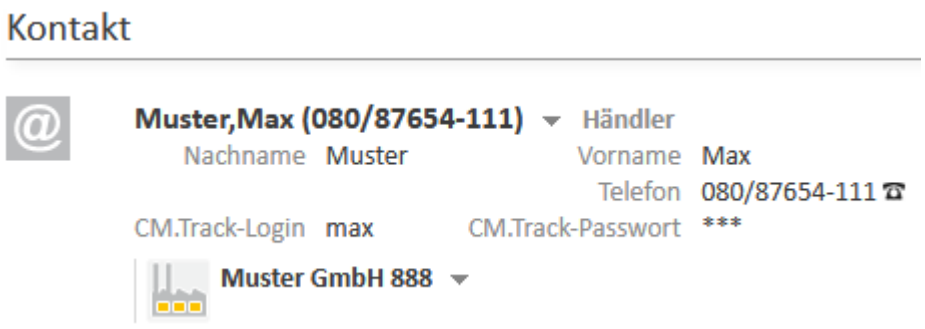


Abbildung 297: *ConSol CM Web Client - Kopfbereich einer Kontaktseite, Darstellung des Kontaktnamens mit dem Template Ticket*

Das folgende Template wird verwendet:

```
<#if ResellerCustomer.getFieldValue("ResellerCustomerData","customer_name")?has_
 content>${ResellerCustomer.getFieldValue("ResellerCustomerData","customer_
 name")!},${ResellerCustomer.getFieldValue
 ("ResellerCustomerData","forename")!}<#else> ${ResellerCustomer.getFieldValue
 ("ResellerCustomerData","customer_name")!}</#if><#if
 ResellerCustomer.getFieldValue("ResellerCustomerData","phone")?has_content>
 (${ResellerCustomer.getFieldValue("ResellerCustomerData","phone")!})</#if>
```

Beachten Sie, dass Sie nach einem Update von einer CM-Version vor 6.11 auf Version 6.11 und höher möglicherweise die Template-Konfiguration anpassen müssen. In früheren Versionen wurde die erste Zeile mit Kundenfeldern (früher *Datenobjektgruppenfelder*), d. h. Felder mit der Position (0,x), dazu verwendet, die grundlegenden Kundendaten im Kopfbereich der Kundenseite anzuzeigen. Falls Sie die Anzeige angepasst und die Sichtbarkeit einiger Felder auf *edit* gesetzt haben, können Sie diese Annotation gegebenenfalls entfernen.

D.5.5.9 Ticketliste

Mit diesem Template wird die Darstellung der Kundendaten in der Ticketliste definiert.

Wenn Sie mit diesem Template-Typ arbeiten möchten, muss der Seitenanpassungsparameter *accordionTicketList.mainCustomerDescriptionVisible* auf „true“ gesetzt sein. Andernfalls können die Kundendaten nicht in der Ticketliste angezeigt werden.



Abbildung 298: *ConSol CM Web Client - Kunden-Template vom Typ Ticketliste*

D.5.5.10 Ticketrelation

Mit diesem Template wird die Darstellung der Kundendaten in Ticketreferenzen im Bereich *Verknüpfte Vorgänge* eines Tickets definiert. Denken Sie daran, dass die Kundendaten von referenzierten Tickets nur im Sichtbarkeitslevel *Erweitert* angezeigt werden.

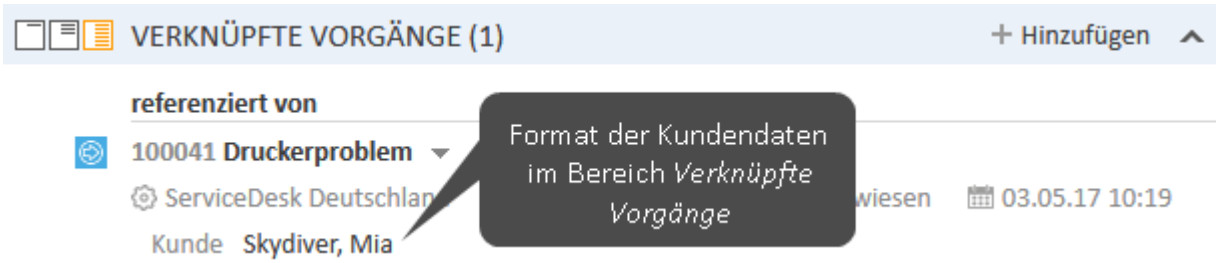


Abbildung 299: *ConSol CM Web Client - Kunden-Template vom Typ Ticketrelation*

D.5.5.11 Workspace und Favoriten

Mit diesem Template wird die Darstellung der Kundendaten in den *Favoriten* eines Tickets definiert.

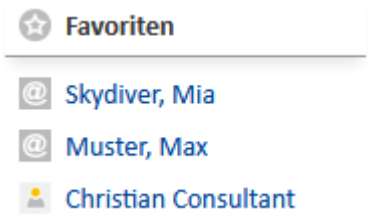


Abbildung 300: *ConSol CM Web Client - Kunden-Template vom Typ Favoriten*

D.5.5.12 Protokoll

Mit diesem Template wird die Darstellung der Kundendaten im Ticketprotokoll, d. h. dem Protokollbereich eines Tickets definiert.

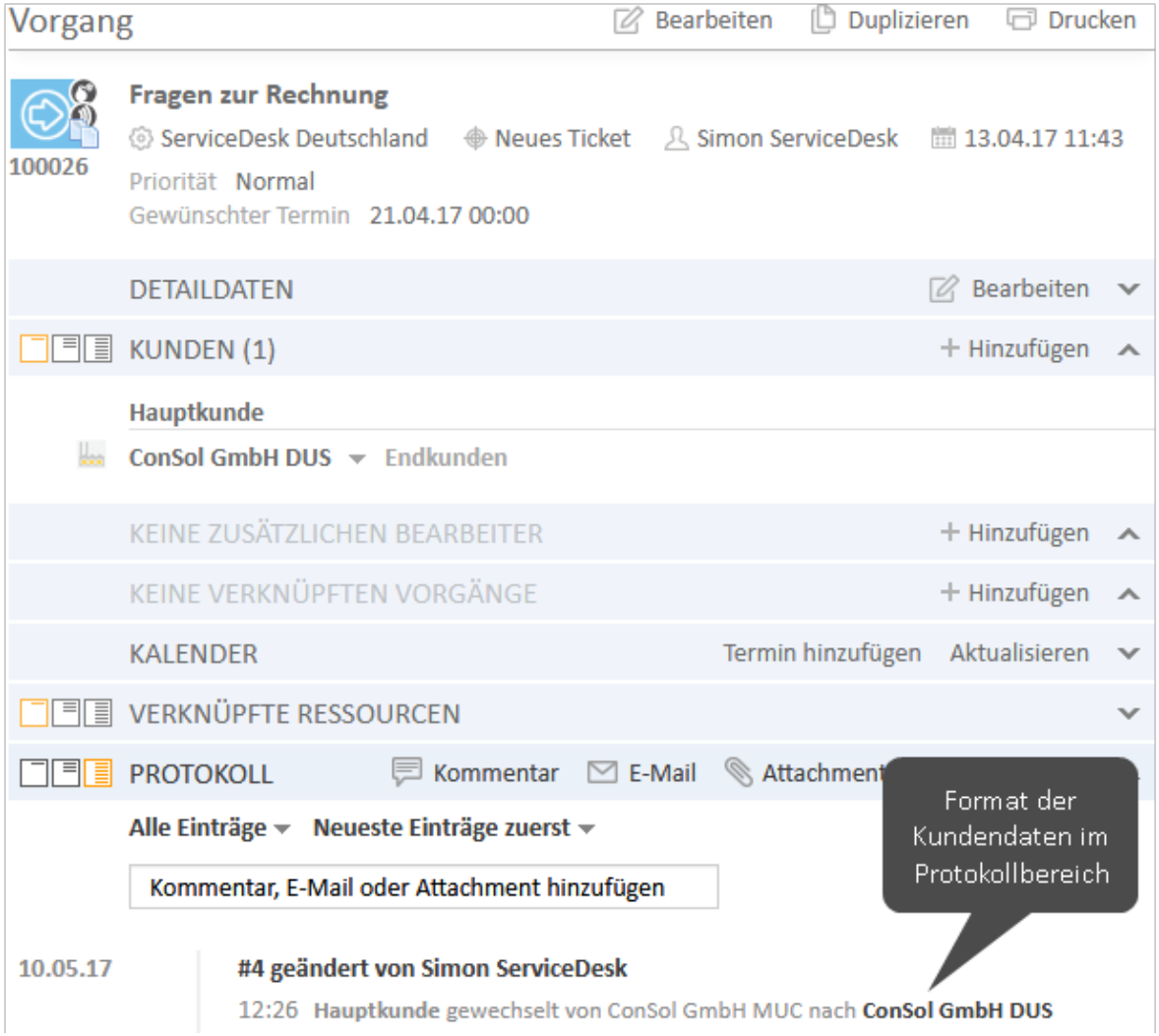


Abbildung 301: *ConSol CM Web Client - Kunden-Template vom Typ Protokoll*

D.5.5.13 Vorschläge

Mit diesem Template wird die Darstellung der Kundendaten in den Vorschlägen definiert, die bei der Erstellung eines Tickets angezeigt werden.

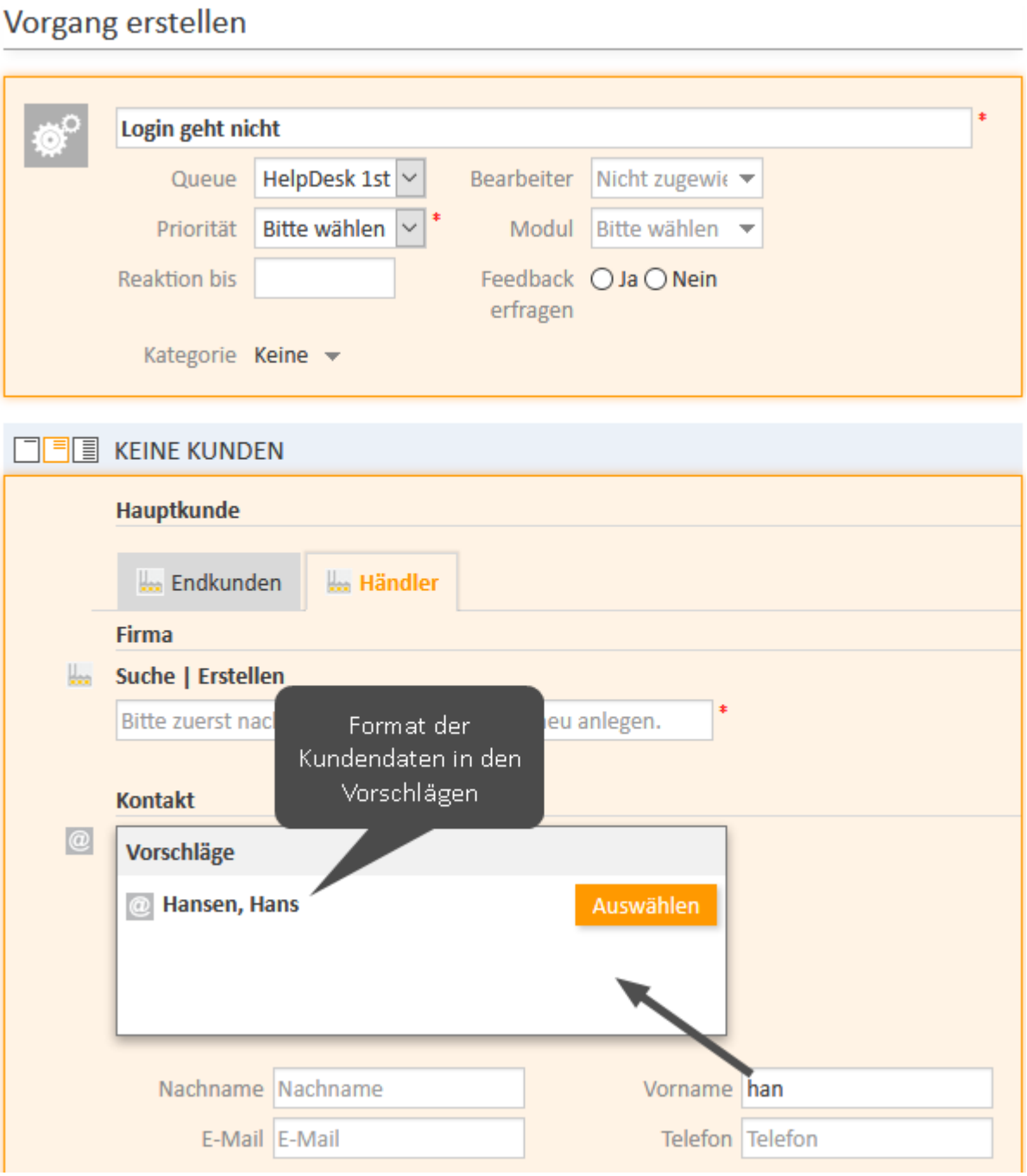


Abbildung 302: *ConSol CM Web Client - Kunden-Template vom Typ Vorschläge*

D.5.5.14 CM/Phone Kundendetails

Siehe Abschnitt CM/Phone: CTI mit ConSol CM.

D.5.5.15 CM/Phone Kundenliste

Siehe Abschnitt CM/Phone: CTI mit ConSol CM.

D.6 Verwaltung von Kundengruppen

In diesem Kapitel werden folgende Themen behandelt:

D.6.1 Grundprinzip von Kundendatenmodellen und Kundengruppen 445

D.6.2 Verwaltung von Kundengruppen mit dem Admin Tool 446

D.6.3 Zuweisen von Zugangsberechtigungen für Kundengruppen 451

D.6.1 Grundprinzip von Kundendatenmodellen und Kundengruppen

In einem ConSol CM-System können mehrere Kundengruppen verwendet werden.

> ⚠ **Prinzipien:**
>
> Es kann eine beliebige Anzahl an Kundengruppen und Kundendatenmodellen geben.
>
> Jede Kundengruppe hat genau ein Kundendatenmodell.
>
> Jedes Kundendatenmodell kann einer beliebigen Anzahl an Kundengruppen zugewiesen werden.

Im folgenden Beispiel enthält das System vier Kundengruppen, die jeweils ihr eigenes Kundendatenmodell haben.

D.6.2 Verwaltung von Kundengruppen mit dem Admin Tool

Im Admin Tool werden Kundengruppen im Navigationselement *Kundengruppen* der Navigationsgruppe *Kunden* verwaltet.

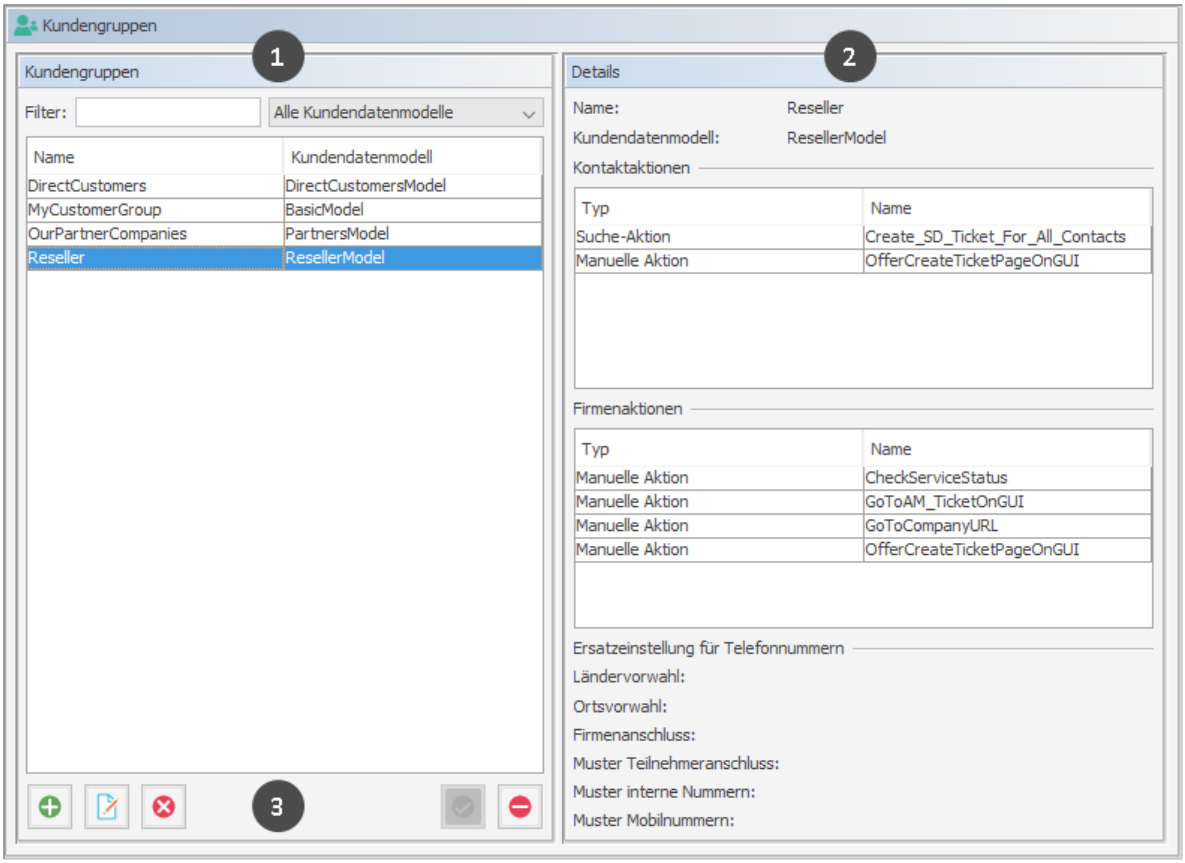


Abbildung 303: *ConSol CM Admin Tool - Kunden, Kundengruppen: Verwaltung von Kundengruppen*

Die Seite *Kundengruppen* besteht aus den folgenden Bereichen:

- Liste aller Kundengruppen (1). Kann gefiltert und durch Klicken auf die Kopfzeile der Spalte sortiert werden.
- Details der ausgewählten Kundengruppe (2)
- Buttons zum Hinzufügen, Bearbeiten, Löschen, Aktivieren oder Deaktivieren einer Kundengruppe (3)

D.6.2.1 Liste der Kundengruppen

Auf der linken Seite sind alle Kundengruppen aufgeführt:

- **Name**
  Der technische Name der Kundengruppe.
- **Kundendatenmodell**
  Der Name des Kundendatenmodells, das der Kundengruppe zugewiesen wurde.

Sie können zwei Arten von Filtern anwenden:

- **Filter nach Namen**
  Geben Sie einen Text oder einige Zeichen im Feld *Filter* ein. In der Liste werden dann nur die Kundengruppen angezeigt, deren Name den Text bzw. die Zeichen enthält.
- **Filter nach Kundendatenmodell**
  Wählen Sie ein Kundendatenmodell aus der Drop-down-Liste aus. In der Liste werden dann nur die Kundengruppen angezeigt, die das ausgewählte Kundendatenmodell verwenden.

D.6.2.2 Details der Kundengruppe

Auf der rechten Seite werden die Details der ausgewählten Kundengruppe angezeigt. Im folgenden Abschnitt werden alle Parameter erklärt.

D.6.2.3 Erstellen einer neuen Kundengruppe

Sie können eine neue Kundengruppe erstellen, indem Sie unter der Liste der Kundengruppen auf den Button *Hinzufügen* klicken. Es wird ein Pop-up-Fenster geöffnet, in dem Sie die Parameter der Kundengruppe eingeben müssen.

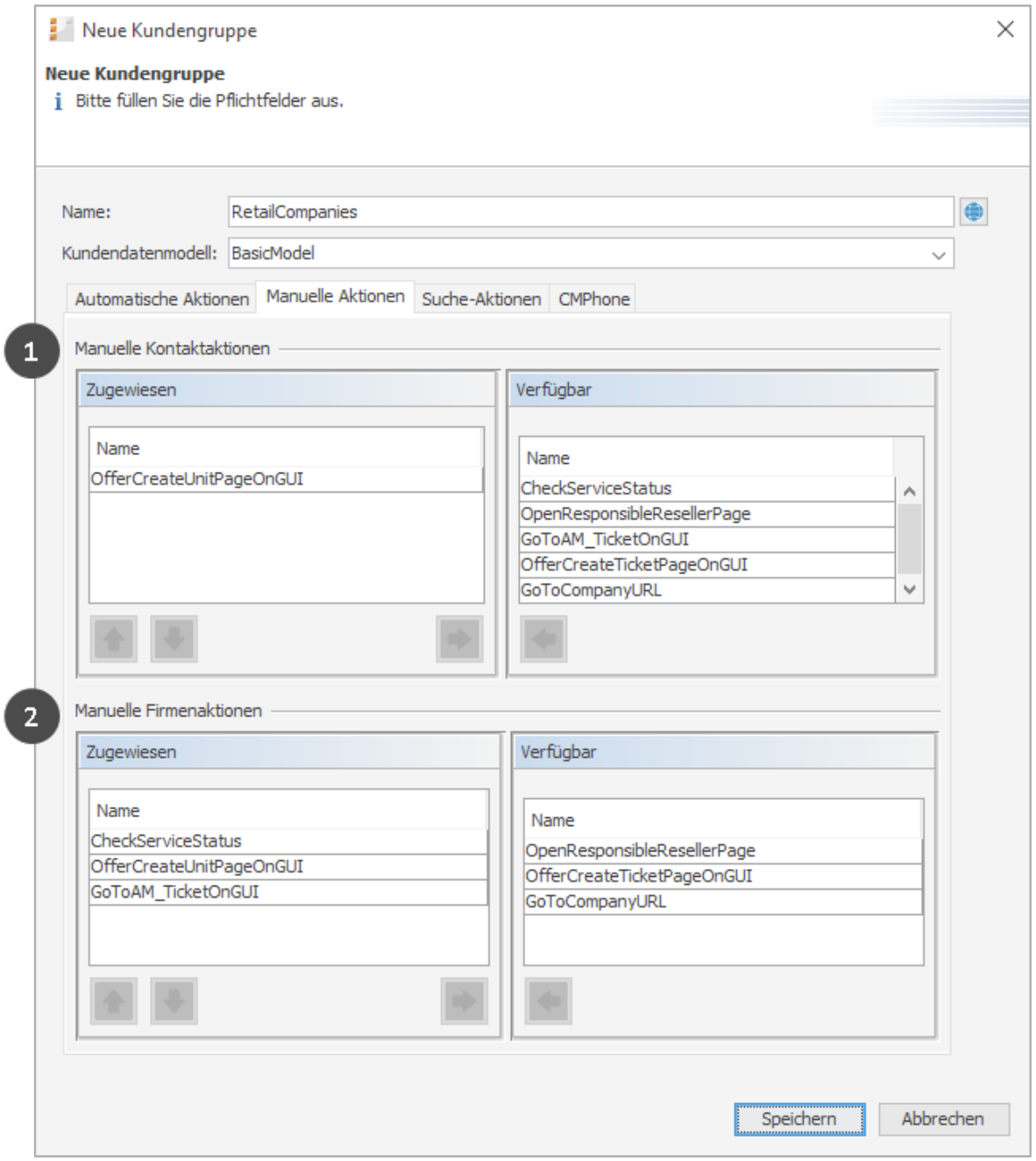


Abbildung 304: *ConSol CM Admin Tool - Kunden, Kundengruppen: Parameter für eine Kundengruppe*

- **Name**
  Der eindeutige technische Name der Kundengruppe. Kann über den Button *Lokalisieren* lokalisiert werden. Details dazu finden Sie im Abschnitt Lokalisierung von Objekten allgemein, Typ 1.

- **Kundendatenmodell**
  Wählen Sie das Kundendatenmodell aus der Drop-down-Liste aus. Zur Verfügung stehen alle Kundendatenmodelle, die definiert wurden (siehe Abschnitt Einrichten des Kundendatenmodells).
- **Automatische Aktionen**
  Hier können der Kundengruppe automatische Kundenaktionen zugewiesen werden. In den Listen werden alle bereits definierten Kundenaktionen angeboten. Sie können diese Aktionen den Kontakten und/oder Firmen der Kundengruppe zuweisen. Automatische Aktionen werden beim Erstellen, Editieren oder Löschen eines Kunden ausgeführt. Die gesamte Konfiguration wird detailliert im Abschnitt Action Framework - Kundenaktionen.
- **Manuelle Aktionen**
  Manuelle Aktionen können ebenfalls den Kontakten und/oder Firmen zugewiesen werden. In den Listen werden alle bereits definierten Kundenaktionen angeboten. Die Aktionen werden als Aktivitäten auf der Kontakt- oder Firmenseite angeboten und müssen, ähnlich wie Workflow-Aktivitäten für Tickets, von einem Bearbeiter manuell angestoßen werden. Mit den Pfeilsymbolen unter der Liste der zugewiesenen Aktionen können Sie die Reihenfolge der angebotenen Aktionen im Web Client definieren. Details über manuelle Kundenaktionen sind im Abschnitt Action Framework - Kundenaktionen erklärt.
- **Suche-Aktionen**
  Suche-Aktion werden als Aktivitäten für die Ergebnisliste von Detailsuchen angeboten, und müssen den Kontakten und/oder Firmen einer Kundengruppe zugewiesen werden. Eine Suche-Aktion für Firmen für die Kundengruppe *Reseller* wird zum Beispiel immer angeboten, wenn ein Suchergebnis eine Liste mit Reseller-Firmen enthält. Alle bereits definierten Suche-Aktionen für Kunden werden in den Listen angeboten. Suche-Aktionen sind Teil des Action Frameworks und werden im Abschnitt Action Framework - Suche-Aktionen detailliert beschrieben.
- **CMPhone**
  Tab für alle CM/Phone-Parameter. Nur verfügbar, wenn CM/Phone aktiv ist, siehe Abschnitt CM/Phone: CTI mit ConSol CM

Aktionen können Kontakten (1) und/oder Firmen (2) einer Kundengruppe zugewiesen werden.

D.6.2.4 Editieren einer Kundengruppe

Wenn Sie eine Kundengruppe editieren möchten, wählen Sie sie in der Liste aus und klicken Sie auf den Button *Bearbeiten*, oder doppelklicken Sie einfach auf den Namen der Kundengruppe. Ändern Sie die Parameter der Kundengruppe und klicken Sie auf *Speichern*, um die Änderungen zu speichern.

D.6.2.5 Löschen einer Kundengruppe

Wählen Sie die Kundengruppe, die Sie löschen möchten, in der Liste aus und klicken Sie auf den Button *Löschen*. Wenn Sie das daraufhin angezeigte Fenster mit *Ja* bestätigen, wird die Kundengruppe gelöscht und ist im System nicht mehr verfügbar. Eine Kundengruppe kann nur gelöscht werden, wenn sie keiner Queue zugewiesen ist und wenn es keine Tickets für Kunden dieser Gruppe gibt. In einem System, das seit einiger Zeit in Betrieb ist, ist es normalerweise nicht möglich, eine Kundengruppe zu löschen.

D.6.2.6 Deaktivieren und Reaktivieren einer Kundengruppe

Um eine Kundengruppe zu deaktivieren, wählen Sie sie in der Liste aus und klicken Sie auf den Button *Deaktivieren*. Der entsprechende Listeneintrag wird jetzt kursiv angezeigt. Klicken Sie unten auf der Seite einfach auf den Button *Aktivieren*, um die Kundengruppe wieder zu aktivieren. Wenn eine Kundengruppe deaktiviert ist, ist es nicht möglich, neue Tickets für Firmen oder Kontakte dieser Gruppe zu erstellen. Die Tickets der Gruppe sind aber noch sichtbar.

D.6.3 Zuweisen von Zugangsberechtigungen für Kundengruppen

Damit die Bearbeiter mit den Kundendaten einer Kundengruppe arbeiten können, z. B. um neue Datensätze für Händler zu erstellen oder vorhandene Datensätze zu ändern, müssen Sie einer oder mehreren Rollen Zugangsberechtigungen für die Kundengruppen erteilen.

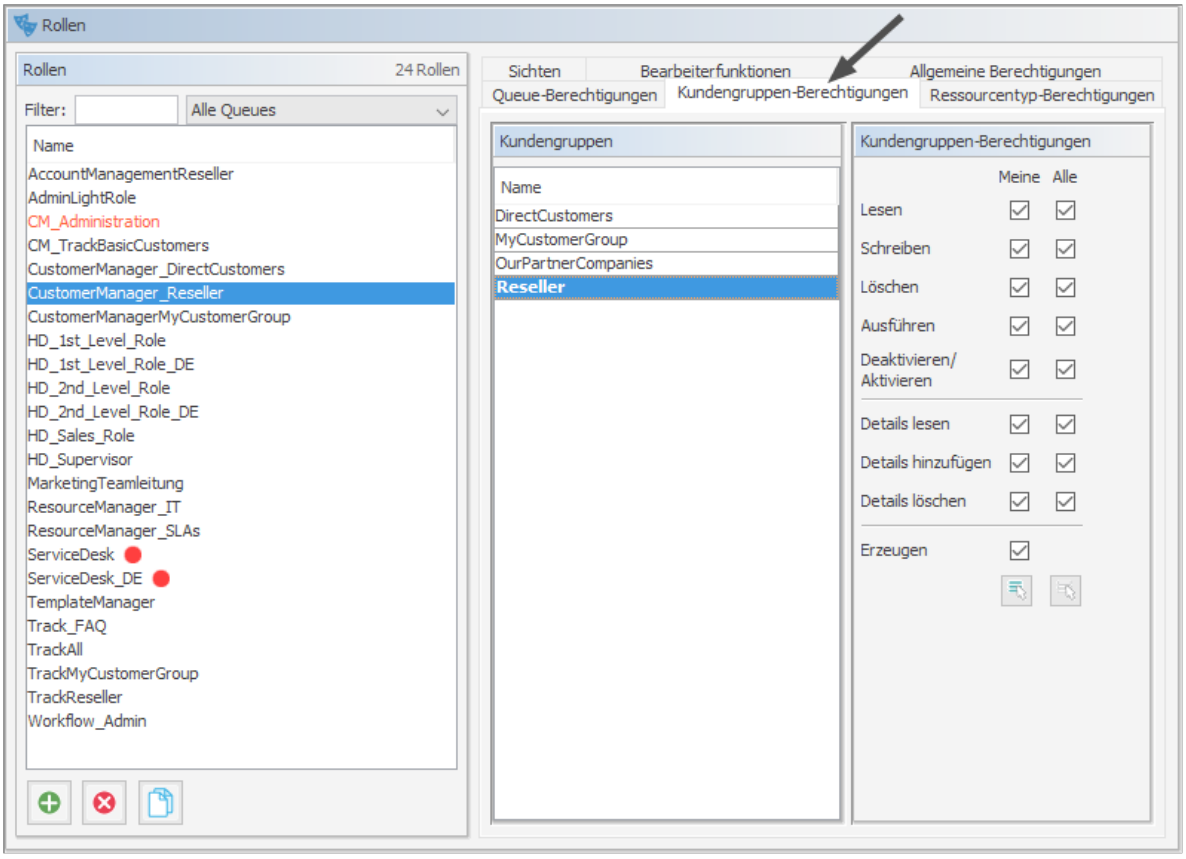


Abbildung 305: *ConSol CM Admin Tool - Zugriff und Rollen, Rollen: Zuweisen von Berechtigungen für Kundengruppen zu einer Rolle*

Die Zugangsberechtigungen, die für Kundendaten erteilt werden können, umfassen auch die Zugriffsberechtigungen für den Bereich für Kommentare und Attachments auf der Kundenseite, d. h. auf der Firmenseite und/oder der Kontaktseite.

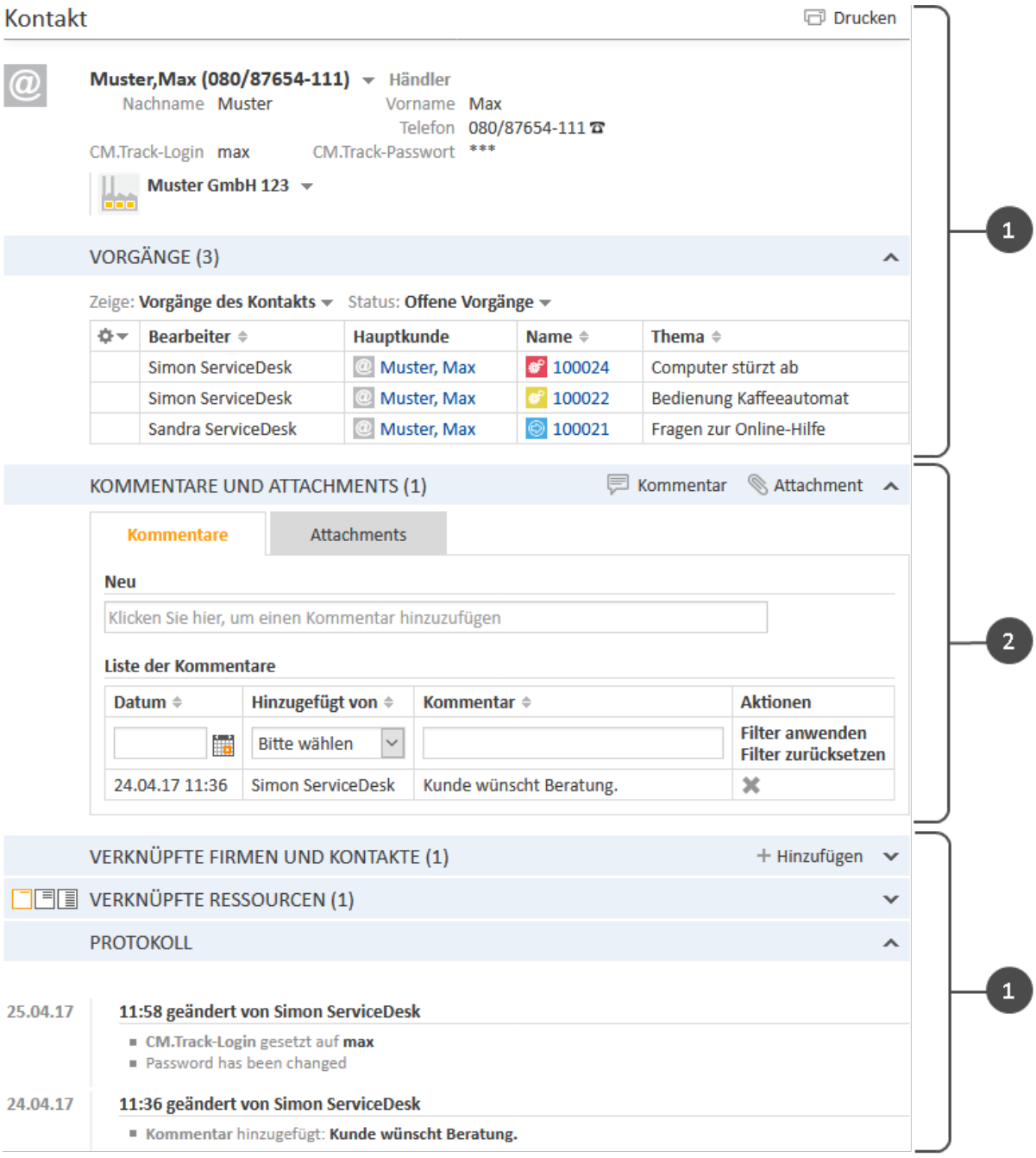


Abbildung 306: *ConSol CM Web Client - Kontaktseite: Detailbereich*

Die folgenden Zugangsberechtigungen können erteilt werden:

- **Art des Kunden**
  Bezieht sich auf die Tickets des Kunden.

    - **Meine**
      Alle Kunden (Hauptkunden und Zusatzkunden) der Tickets, die dem Bearbeiter aktuell zugewiesen sind oder bei denen der Bearbeiter als zusätzlicher Bearbeiter angegeben ist.
    - **Alle**
      Alle Kunden.
- **Allgemeine Bereiche (1)**
    - **Lesen**
      Kundendaten lesen.
    - **Schreiben**
      Kundendaten schreiben/ändern.
    - **Löschen**
      Kundendatensatz löschen.
    - **Ausführen**
      Aktionen für diesen Kunden ausführen (Details über Kundenaktionen finden Sie im Abschnitt Action Framework - Kundenaktionen).
    - **Deaktivieren/Aktivieren**
      Kunden deaktivieren und reaktivieren. Für einen deaktivierten Kunden können keine Tickets erstellt werden.
- **Detailbereich (2)**
    - **Details lesen**
      Kundendaten im Bereich für Kommentare und Attachments lesen.
    - **Details hinzufügen**
      Kundendaten im Bereich für Kommentare und Attachments hinzufügen/ändern.
    - **Details löschen**
      Kundendaten im Bereich für Kommentare und Attachments löschen.
- **Allgemein**
    - **Erzeugen**
      Kundendatensatz erstellen. Bei einem zweistufigen Kundendatenmodell gilt diese Berechtigung sowohl für Kontaktdatensätze als auch für Firmendatensätze.

⚠ Beachten Sie, dass ein Bearbeiter mindestens Lese-Berechtigungen für eine Kundengruppe haben muss, um Tickets für Kunden aus dieser Gruppe öffnen bzw. erstellen zu können!

D.7 Kundenrollen

In diesem Kapitel werden folgende Themen behandelt:

D.7.1 Einleitung 454

D.7.2 Definieren von Kundenrollen mit dem Admin Tool 455

D.7.1 Einleitung

Kundenrollen helfen Ihnen dabei, die verschiedenen mit einem Ticket verknüpften Kunden auseinanderzuhalten. Es kann zum Beispiel für einen Bearbeiter sehr hilfreich sein, wenn er beim Öffnen eines Vorfalltickets direkt sieht, welcher der Kunden der Manager ist, welcher der technische Ansprechpartner und welcher der Ansprechpartner für die Rechnungsstellung.

Sie können eine beliebige Anzahl an Kundenrollen definieren. Technisch gesehen handelt es sich um eine einfache Liste. Der Bearbeiter kann im Web Client allen Zusatzkunden eines Tickets eine Kundenrolle zuweisen, muss es aber nicht tun. Ein Zusatzkunde kann entweder eine oder keine Kundenrolle haben, nicht aber mehrere Kundenrollen.

Neben den Vorteilen, die Kundenrollen den Bearbeitern bei ihrer täglichen Arbeit bieten, sind sie auch für die Gestaltung von Prozessen und das Schreiben von Skripten nützlich, z. B. kann eine E-Mail an alle Manager aller offenen Service-Desk-Tickets einer Firma geschrieben werden, um diese über ein bestimmtes Thema zu informieren.

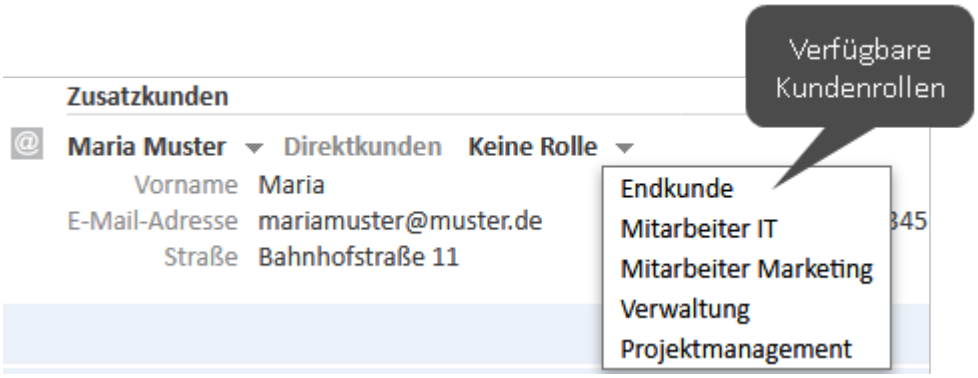


Abbildung 307: *ConSol CM Web Client - Kundenrollen für einen Zusatzkunden*

D.7.2 Definieren von Kundenrollen mit dem Admin Tool

Öffnen Sie das Navigationselement *Rollen* in der Navigationsgruppe *Kunden*, um Kundenrollen hinzuzufügen, zu editieren oder zu löschen.

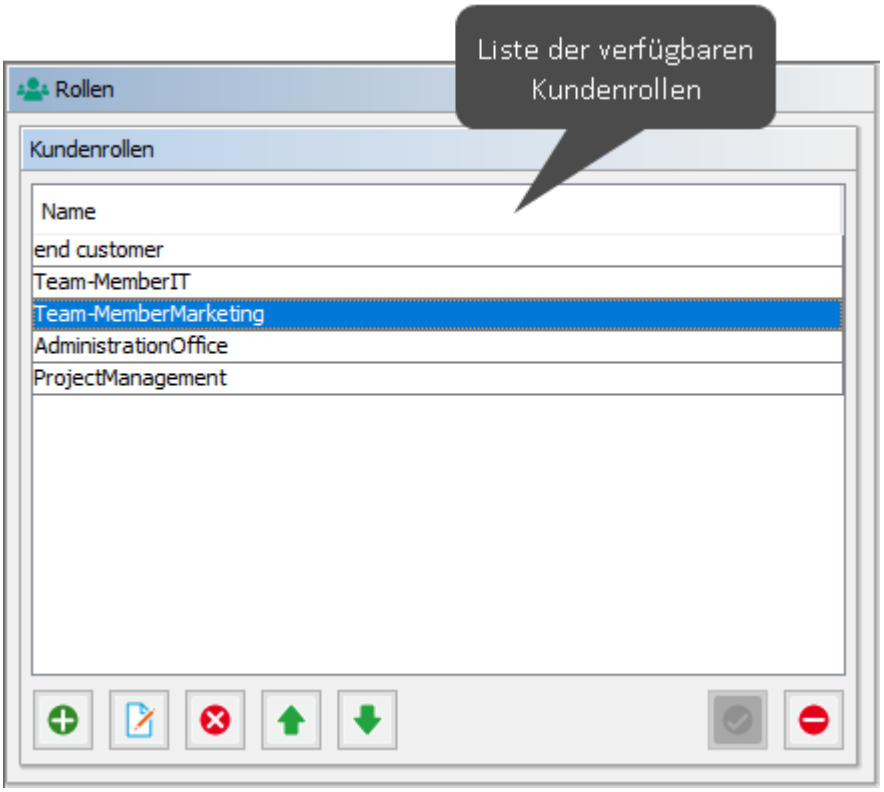


Abbildung 308: *ConSol CM Admin Tool - Kunden, Rollen: Kundenrollen*

Im Web Client können diese Rollen Zusatzkunden eines Tickets zugewiesen werden, um die Funktion dieser Kunden zu zeigen, z. B. Projektmanager oder Endkunde.

Die Zuweisung von Kundenrollen zu Zusatzkunden hat zwei Folgen:

1. Im Web Client wird diese Information angezeigt (z. B. soll der Bearbeiter eine Log-Datei nicht an einen *Manager*, sondern an den *Mitarbeiter IT* schicken).
2. Die Kundenrolle kann bei der Workflow-Programmierung zur Prozesssteuerung verwendet werden (z. B. Senden einer E-Mail an alle *Mitarbeiter IT* aber nicht an die Kontakte mit anderen Rollen).

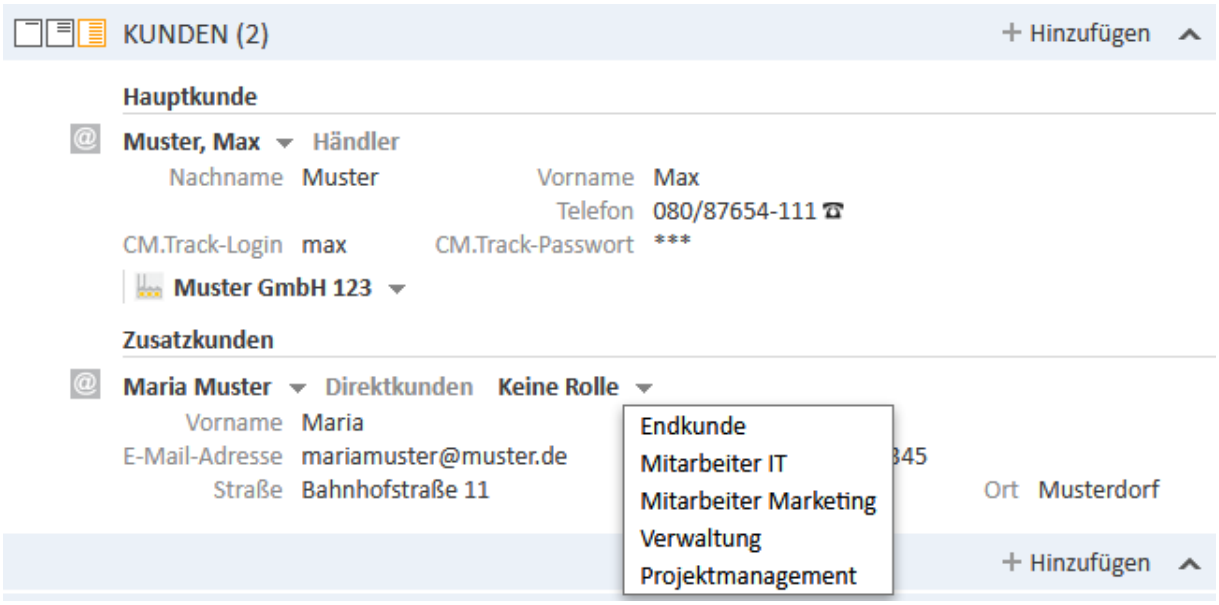


Abbildung 309: *ConSol CM Web Client - Setzen einer Kundenrolle für einen Zusatzkunden*

D.7.2.1 Erstellen oder Editieren einer Kundenrolle

Eine Kundenrolle wird durch ihren Namen definiert. Wenn Sie auf den Button *Hinzufügen* klicken, öffnet sich ein Pop-up-Fenster, in dem Sie den Namen eingeben können. Danach können Sie den Namen über den Button *Lokalisieren* neben dem Namensfeld lokalisieren (siehe unten). Details über die Lokalisierung finden Sie im Abschnitt Lokalisierung von Objekten allgemein, Typ 1. Die Checkbox *Aktiviert* ist bereits ausgewählt, damit die Kundenrolle im System aktiv ist (siehe auch Deaktivieren und Aktivieren einer Kundenrolle). Wenn Sie zum Editieren einer Kundenrolle auf den Button *Bearbeiten* klicken, wird das gleiche Fenster angezeigt.

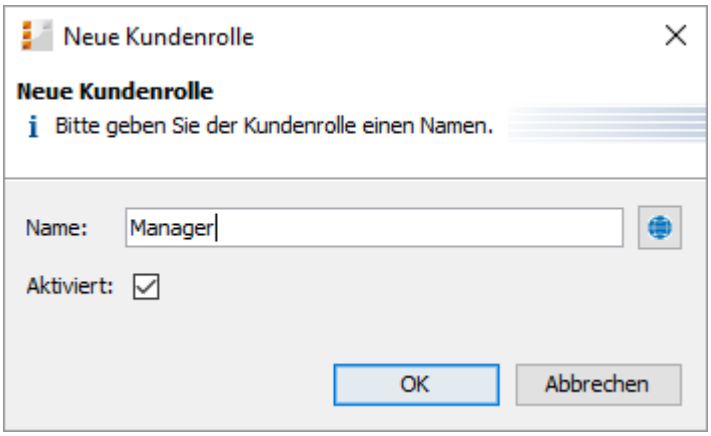


Abbildung 310: *ConSol CM Admin Tool - Kunden, Rollen: Erstellen oder Editieren einer Kundenrolle*

D.7.2.2 Löschen einer Kundenrolle

Eine Kundenrolle kann nur gelöscht werden, wenn sie keinem Kunden zugewiesen ist. Andernfalls wird eine Warnung angezeigt und Sie können diese Kundenrolle nur deaktivieren (siehe unten).

Um eine Kundenrolle zu löschen, wählen Sie sie in der Liste aus und klicken Sie auf den Button *Löschen*. Wenn Sie im daraufhin angezeigten Bestätigungsfenster auf *Ja* klicken, wird die Kundenrolle aus der Liste und aus dem System gelöscht.

D.7.2.3 Deaktivieren und Aktivieren einer Kundenrolle

Wenn eine Kundenrolle noch Kunden zugewiesen ist, aber nicht mehr benötigt wird, können Sie sie deaktivieren. Wählen Sie dazu die Kundenrolle aus und klicken Sie auf den Button *Deaktivieren*. Der entsprechende Listeneintrag wird danach kursiv angezeigt. Die Kundenrolle kann nicht mehr zugewiesen werden. Klicken Sie unten auf der Seite einfach auf den Button *Aktivieren*, um die Rolle wieder zu aktivieren.

Sie können eine Kundenrolle im Fenster zum Editieren von Kundenrollen aktivieren oder deaktivieren, indem Sie die Checkbox *Aktiviert* markieren oder leer lassen. Wenn Sie eine neue Kundenrolle erstellen, ist die Checkbox automatisch markiert.

D.7.2.4 Lokalisieren einer Kundenrolle

Klicken Sie im Fenster zum Erstellen oder Editieren einer Kundenrolle auf den Button *Lokalisieren*, um den lokalisierten Namen einer Kundenrolle einzugeben. Details über die Lokalisierung finden Sie im Abschnitt Lokalisierung von Objekten allgemein, Typ 1.

D.8 Kundenrelationen

In diesem Kapitel werden folgende Themen behandelt:

D.8.1 Einleitung 459
D.8.2 Verwaltung von Kundenrelationen mit dem Admin Tool 461
D.8.3 Erstellen von Kundenrelationen im Web Client 464
D.8.4 Verwendung von Relationen in Skripten 465
D.8.5 Kundenrelationen zu Ressourcen 465

D.8.1 Einleitung

Kundenrelationen sind Relationen zwischen Kunden (Units), d. h. Firmen und Kontakten. Eine Relation kann einem der folgenden beiden Typen angehören:

- **Gerichtet** (unterschiedliche Hierarchiestufen)
- **Referenz** (gleiche Stufe, keine Hierarchie)

Eine Relation hat einen der folgenden Typen:

- **Firma - Firma**
  z. B. ... *hat eine Kooperation mit* ... (Firma X arbeitet mit Firma Y zusammen)
  - Die Firmen können zur gleichen oder zu unterschiedlichen Kundengruppen gehören.
  - Die betroffenen Kundengruppen können das gleiche oder unterschiedliche Kundendatenmodelle haben.
- **Firma - Kontakt**
  z. B. ... *ist Kunde von* ... (Kontakt X ist Kunde von Firma Y)
  - Die Firma und der Kontakt können zur gleichen oder zu unterschiedlichen Kundengruppen gehören.
  - Die betroffenen Kundengruppen können das gleiche oder unterschiedliche Kundendatenmodelle haben.
- **Kontakt - Kontakt**
  z. B. ... *wird betreut von* ... (Kontakt X von Firma X wird durch Kontakt Y von Firma Y betreut)
  - Die Firmen und Kontakte können zur gleichen oder zu unterschiedlichen Kundengruppen gehören.
  - Die betroffenen Kundengruppen können das gleiche oder unterschiedliche Kundendatenmodelle haben.

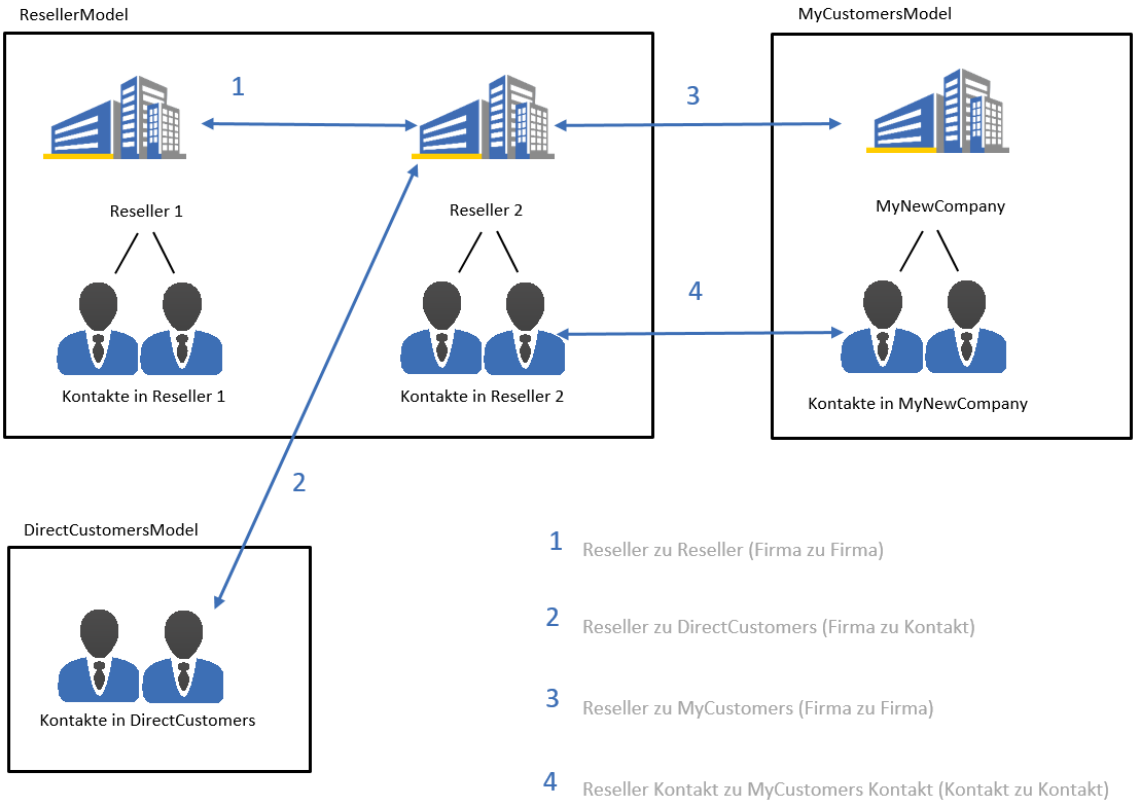


Abbildung 311: *ConSol CM FlexCDM - Beispiele für Kundenrelationen*

D.8.2 Verwaltung von Kundenrelationen mit dem Admin Tool

Damit die Bearbeiter mit Kundenrelationen arbeiten können, müssen die Relationen zuerst im Admin Tool definiert werden. Öffnen Sie das Navigationselement *Relationen* in der Navigationsgruppe *Kunden*. Alle Relationen sind darin aufgeführt. Es können neue Relationen hinzugefügt und alte Relationen gelöscht werden.

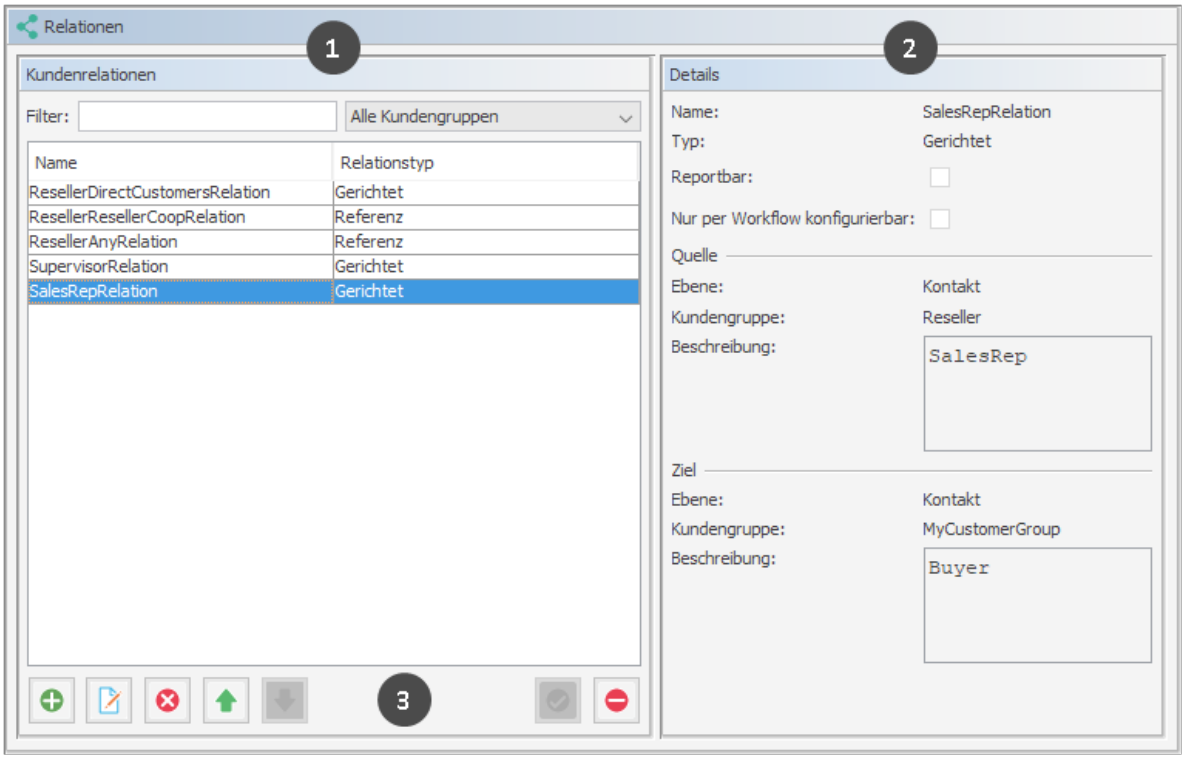


Abbildung 312: *ConSol CM Admin Tool - Kunden, Relationen: Verwaltung von Kundenrelationen*

Es stehen folgende Elemente zur Verfügung:

- **Liste der Relationen (1)**
- **Filter**
  - Nach einem Ausdruck filtern, der im Feld *Filter* eingegeben wurde. Sie können das Sternchen als Platzhalter für ein beliebiges Zeichen verwenden.
  - Im Drop-down-Menü nach Kundengruppen filtern.
- **Details der Relation (2)**
- **Buttons (3)**
  - **Button *Hinzufügen***
    Eine neue Relation hinzufügen. Das Pop-up-Fenster *Neue Kundenrelation* mit den Detailfeldern (siehe nächster Abschnitt) wird geöffnet.

- **Button *Bearbeiten***
  Die Parameter einer Relation ändern. Das Pop-up-Fenster *Kundenrelation bearbeiten* mit den Detailfeldern (siehe nächster Abschnitt) wird geöffnet.
- **Button *Löschen***
  Eine vorhandene Relation löschen. Dies ist nur möglich, wenn (im Web Client) keine Relationen dieses Typs gesetzt wurden.
- **Reihenfolge ändern (Pfeil *Nach oben* und *Nach unten*)**
  Eine Relation an einer bestimmten Stelle der Liste platzieren. Damit wird die Reihenfolge der manuellen Relationen im Web Client festgelegt.
- **Relationen aktivieren / deaktivieren**
  Eine deaktivierte Relation ist im Web Client nicht verfügbar, d. h. es kann dort keine Relation dieses Typs erstellt werden. Vorhandene Relationen dieses Typs werden dadurch nicht verändert und werden weiterhin im Web Client angezeigt.

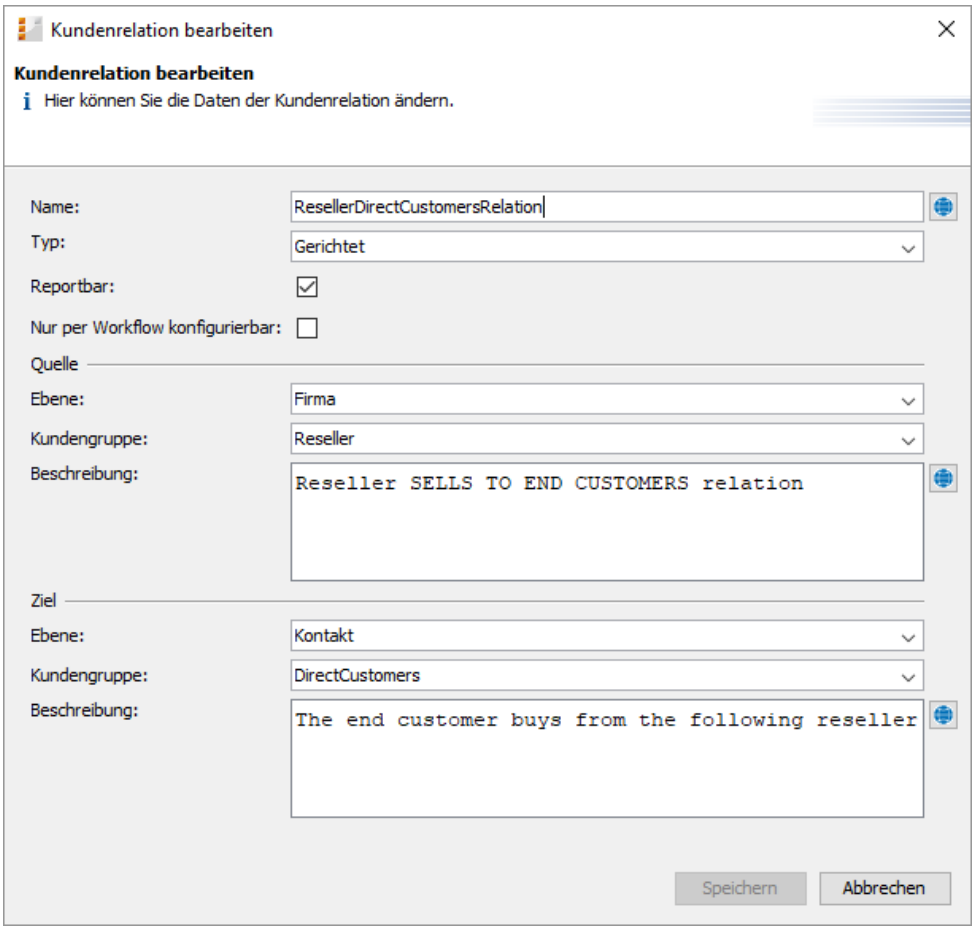


Abbildung 313: *ConSol CM Admin Tool - Kunden, Relationen: Details einer Kundenrelation*

Um eine neue Relation zu erstellen, klicken Sie auf den Button *Hinzufügen*. Klicken Sie auf den Button *Bearbeiten*, um eine Relation zu editieren. In beiden Fällen wird das Detailfenster für eine Relation geöffnet. Dort können Sie folgende Felder editieren:

- **Name**
  Name der Relation. Der technische Name für den internen Gebrauch (Skripte). Wie bei den meisten Feldern in ConSol CM wird im Web Client der lokalisierte Name angezeigt. Details zur Lokalisierung finden Sie im Abschnitt Lokalisierung von Objekten allgemein, Typ 1.
- **Typ**
  Wählen Sie einen der folgenden Typen:
  - **Gerichtet**
    Eine gerichtete Relation hat eine festgelegte Quelle und ein festgelegtes Ziel. Ein Kunde kann für verschiedene Relationstypen gleichzeitig Quelle und Ziel sein. Ein Beispiel für eine gerichtete Relation ist eine Relation zwischen einem Händler (Firma) und einem Endkunden (Kontakt): *verkauft Produkte an*. Eine Firma (Händler) verkauft Produkte an einen Kontakt (Endkunde). Oder eine Relation zwischen zwei Kontakten einer Firma: *ist Vorgesetzter von*. Die andere Richtung *arbeitet für* kann ebenfalls verwendet werden. Es sollte allerdings eine konsistente Struktur für das gesamte System entworfen werden, um Missverständnissen vorzubeugen.
  - **Referenz**
    Eine Referenz ist eine ungerichtete Relation, die keine Hierarchie impliziert, z. B. *hat eine Kooperation mit*.
- **Reportbar**
  Definiert, ob Relationen dieses Typs an das Data Warehouse übertragen werden sollen.
- **Nur per Workflow konfigurierbar**
  Wenn diese Checkbox markiert ist, steht die Relation im Web Client nicht zur Verfügung und kann nur über Workflow-Skripte erstellt werden. Solche Relationen können daher nicht manuell verändert werden.
- Wählen Sie für eine gerichtete Relation:
  - **Quelle**
    - **Ebene**
      Stufe der Relationsquelle, d. h. *Firma* oder *Kontakt* oder *Jede* (wählen Sie Letzteres, wenn die Quelle entweder eine Firma oder ein Kontakt sein kann).
    - **Kundengruppe**
      Die Kundengruppe des Quellkundenobjekts.
    - **Beschreibung**
      Wird im Web Client als Beschreibung der Relation auf der Quellseite angezeigt.
  - **Ziel**
    - **Ebene**
      Stufe des Relationsziels, d. h. *Firma* oder *Kontakt* oder *Jede* (wählen Sie Letzteres, wenn das Ziel entweder eine Firma oder ein Kontakt sein kann).
    - **Kundengruppe**
      Die Kundengruppe des Zielkundenobjekts.
    - **Beschreibung**
      Wird im Web Client als Beschreibung der Relation auf der Zielseite angezeigt.

D.8.3 Erstellen von Kundenrelationen im Web Client

Obwohl es sich um ein Administratorhandbuch handelt, zeigen wir Ihnen, wie Relationen im Web Client gesetzt werden, da Sie als Administrator alle Auswirkungen Ihrer administrativen Änderungen kennen sollten.

Als Bearbeiter, der Zugangsberechtigungen für die Quell- und die Zielkundengruppe hat, können Sie im Bereich *Verknüpfte Firmen und Kontakte* eine Relation von einem Kundenobjekt zu einem anderen Kundenobjekt hinzufügen. Sie müssen mindestens eine Rolle mit der Zugangsberechtigung *Schreiben* für die Quellkundengruppe und die Zielkundengruppe haben, um diesen Vorgang durchführen zu können. Sie können die Relationen auf der Seite des entsprechenden Kundenobjekts setzen, d. h. Sie öffnen die Firmenseite oder die Kontaktseite des potentiellen Quellobjekts.

Sie können zum Beispiel eine Relation *Händler verkauft an Endkunden* von der Firma in der Kundengruppe *Händler* zu einem Kontakt in der Kundengruppe *Endkunden* erstellen. Diese Relation muss natürlich zuerst im Admin Tool definiert werden. Klicken Sie im Bereich *Verknüpfte Firmen und Kontakte* auf den Link *Hinzufügen* und wählen Sie die Relation im Drop-down-Menü. Geben Sie den Namen des Ziels (z. B. den Kontaktnamen) im Autovervollständigen-Feld ein. Sie können auch eine Bemerkung hinzufügen. Klicken Sie dann auf *OK*. Die Relation kann später über die jeweiligen Links (*Bearbeiten*, *Entfernen*) editiert oder gelöscht werden.

Eine detaillierte Beschreibung der Arbeit mit Kundenrelationen finden Sie im *ConSol CM Benutzerhandbuch*.

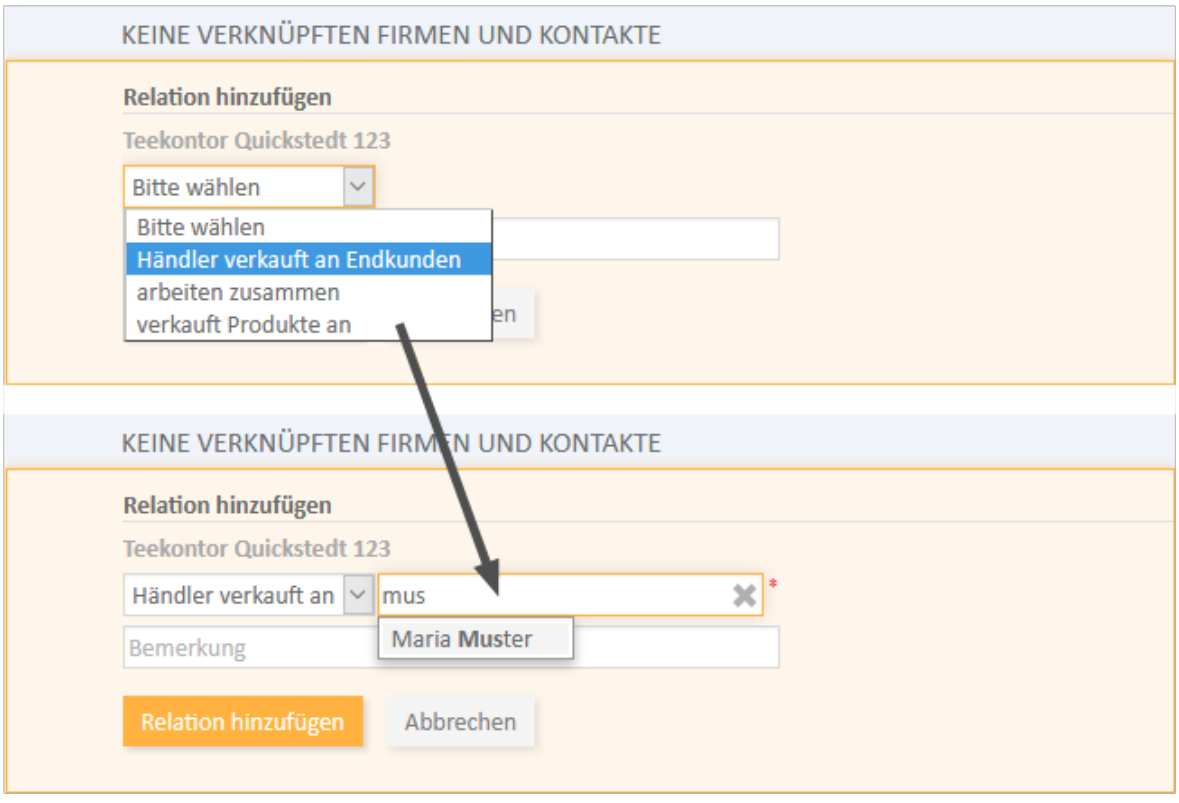


Abbildung 314: *ConSol CM Web Client - Setzen einer Relation*

D.8.4 Verwendung von Relationen in Skripten

Es gibt eine neue Klasse `UnitRelationService`. Details finden Sie in der *ConSol CM API Java Doc*.



In diesem Handbuch werden die Begriffe *Kunde* und *Kundendefinition* verwendet. In früheren ConSol CM-Versionen wurde für Kunden der Begriff *Datenobjekt* verwendet. Die Namen der entsprechenden Java-Klassen sind *Unit* und *UnitDefinition*. Alle anderen Java-Klassen, die Kundenobjekte verarbeiten, heißen ebenfalls noch *Unit*... Beachten Sie dies bei Ihrer Arbeit als ConSol CM-Administrator und -Programmierer. Details finden Sie in der *ConSol CM Java API Doc*.

```
// Creates the unit relation
UnitRelation create(UnitRelation pUnitRelation)
// Deletes the unit relation
void delete(UnitRelation pUnitRelation)
// Get a set of relations by criteria
PageResult<UnitRelation> getByCriteria(UnitRelationCriteria pCritieria)
// Gets unit relations by source and target units
Set<UnitRelation> getBySourceAndTarget(Unit pSourceUnit, Unit pTargetUnit)
// Gets unit relations by source or target units
Set<UnitRelation> getByUnits(Collection<Unit> pUnits)
// Updates the unit relation
void update(UnitRelation pUnitRelation)
```

Eine detaillierte Beschreibung über das Schreiben von Skripten mit Kundenrelationen finden Sie im *ConSol CM Process Designer Handbuch*.

D.8.5 Kundenrelationen zu Ressourcen

Die Relationen von Ressourcen zu Kontakten oder Firmen werden immer als Ressourcenrelationen definiert, d. h. sie werden für die Ressourcen und nicht für die Kundenobjekte definiert. Die Firmen- oder Kontaktobjekte werden bei der Definition der Ressourcenrelation als mögliche Ziele zugewiesen. Ressourcenrelationen sind im Abschnitt CM/Resource Pool - Ressourcenrelationen beschrieben.

D.9 Action Framework - Kundenaktionen

In diesem Kapitel werden folgende Themen behandelt:

D.9.1 Einleitung 467
D.9.2 Verwaltung von Kundenaktionen mit dem Admin Tool 469
D.9.3 Verwendung von Kundenaktionen als Bearbeiter (Benutzer) 475
D.9.4 Beispiele für Kundenaktionsskripte 475
D.9.5 Skripte für das Action Framework: Programmierung von Kundenaktionen 480

D.9.1 Einleitung

Kundenaktionen (auch *Unit-Aktionen* genannt, früher *Datenobjektaktionen* genannt) sind Teil des ConSol CM Action Frameworks. Sie können für einen Kunden, d. h. einen Kontakt oder eine Firma, ausgeführt werden. Die Aktionen können automatisch vom System durchgeführt werden oder manuell von einem Bearbeiter mit den entsprechenden Berechtigungen angestoßen werden. Mögliche Anwendungsfälle für Kundenaktionen sind beispielsweise:

- Laden zusätzlicher Daten in den Datensatz einer Firma.
- Erstellen eines automatischen Reports über die firmenspezifischen KPIs.
- Übertragen der ConSol CM-Daten in ein anderes System (z. B. ERP-System).
- Erstellen/Aktualisieren eines Google Maps-Links (siehe Marken) in den Adressdaten.

Sie können folgende Arten von Kundenaktionen verwenden:

- **Automatische** Aktionen, die vom System nach einer der folgenden Kundenoperationen durchgeführt werden:
  - Erzeugen
  - Aktualisieren
  - Löschen
  - Relation
  - Suche
- **Manuelle** Aktionen, die von einem Bearbeiter mithilfe der Aktivitätslinks auf der Kundenseite (Firmen- oder Kontaktseite) des Web Clients durchgeführt werden (ähnlich wie *Workflow-Aktivitäten* für Tickets). Welche Aktionen für die manuelle Ausführung verfügbar sind, hängt vom aktuell angezeigten Kunden ab, d. h. wenn die Firmenseite offen ist, werden Firmenaktionen angeboten, und wenn die Kontaktseite offen ist, werden Kontaktaktionen angeboten.

> ⚠ Beachten Sie, dass nur Bearbeiter, die mindestens eine Rolle mit den folgenden Zugangsberechtigungen für die entsprechende Kundengruppe haben, die Kundenaktionen verwenden dürfen, d. h. nur bei ihnen werden die Aktivitäten im Web Client angezeigt!
>
> - Ausführen

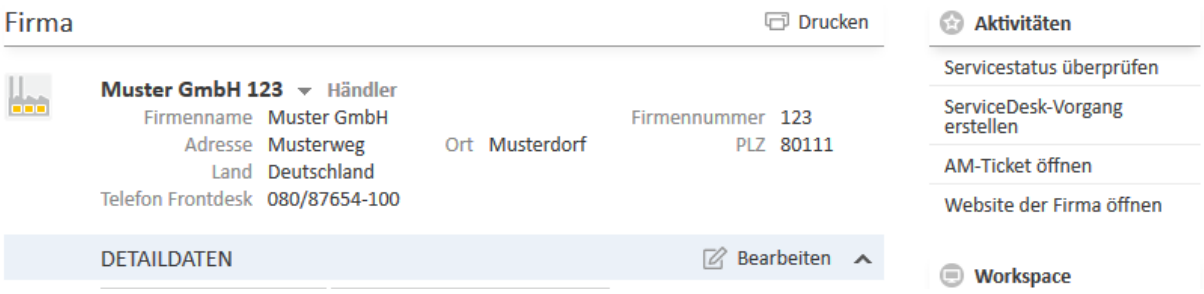


Abbildung 315: *ConSol CM Web Client - Beispiel für manuelle Kundenaktivitäten*

Kundenaktionen werden als Groovy-Skripte definiert, die im Abschnitt *Skripte und Templates* des Admin Tools gespeichert werden.

Die Ausführung von Kundenaktionen kann mit Bedingungsskripten gesteuert werden, d. h. Sie können ein Kundenbedingungsskript implementieren, das vor der Kundenaktionen selber ausgeführt wird. Das Ausführungsscript wird nur ausgeführt, wenn das Bedingungsskript „true“ zurückgibt.

Es gibt also zwei Arten von Skripten, mit denen Sie im ConSol CM Action Framework arbeiten:

- **Kundenaktionsskripte**
  Definiert die Aktion, die ausgeführt werden soll.
- **Kundenbedingungsskripte**
  Definiert eine oder mehrere Bedingungen für die Anzeige der Aktion im Web Client. Muss „true“ oder „false“ zurückgeben. Wenn das Skript „false“ zurückgibt, wird die Aktion nicht auf der GUI angezeigt und kann daher nicht ausgeführt werden.

Führen Sie zur Implementierung einer Kundenaktion folgende drei Schritte durch:

1. Erstellen Sie ein Skript für die Kundenaktion (entweder nur ein Aktionsskript oder ein Aktionsskript und ein Bedingungsskript).
2. Erstellen Sie die Kundenaktionen, die die Skripte verwenden.
3. Weisen Sie die Kundenaktionen den Kundengruppen zu, in denen sie verfügbar sein sollen. Sie können diese Aktionen den Kontakten und/oder Firmen einer Kundengruppe zuweisen.

In den folgenden Abschnitten sind diese drei Schritte detailliert beschrieben.

D.9.2 Verwaltung von Kundenaktionen mit dem Admin Tool



In diesem Handbuch werden die Begriffe *Kunde* und *Kundendefinition* verwendet. In früheren ConSol CM-Versionen wurde für Kunden der Begriff *Datenobjekt* verwendet. Die Namen der entsprechenden Java-Klassen sind *Unit* und *UnitDefinition*. Alle anderen Java-Klassen, die Kundenobjekte verarbeiten, heißen ebenfalls noch *Unit*... Beachten Sie dies bei Ihrer Arbeit als ConSol CM-Administrator und -Programmierer. Details finden Sie in der *ConSol CM Java API Doc*.

D.9.2.1 Schritt 1: Schreiben des Kundenaktionsskripts

Erstellen Sie ein neues Admin-Tool-Skript des Typs *Kundenaktion*. Falls nötig, erstellen Sie ein weiteres Skript des Typs *Kundenbedingung*.

Eine detaillierte Beschreibung der Admin-Tool-Skripte finden Sie im Abschnitt Admin-Tool-Skripte. Eine Einführung in die Admin-Tool-Skripte für Kundenaktionen finden Sie im Abschnitt Skripte für das Action Framework: Programmierung von Kundenaktionen dieses Kapitels.

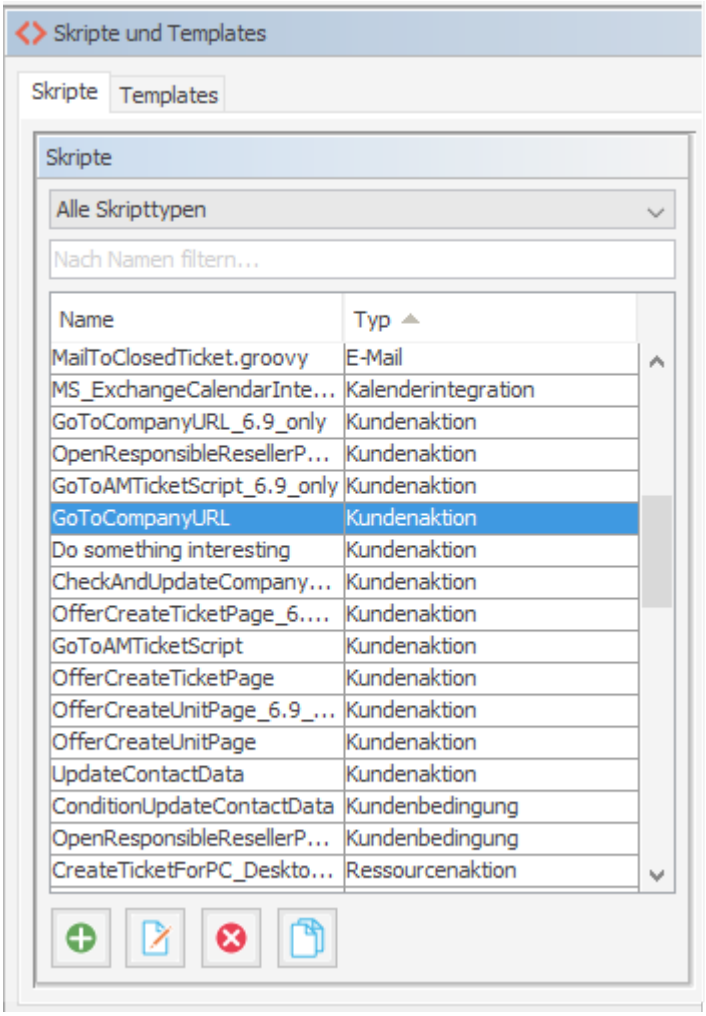


Abbildung 316: *ConSol CM Admin Tool - System, Skripte und Templates: Skripte für Kundenaktionen*

```
//create and return action result that will tell the web to create a new ticket
 with unit as
//a main customer
def queueId = queueService.getByName("Helpdesk").getId();
Map<String, Object> valuesMap = new HashMap<String, Object>
valuesMap.put(PostActionParameter.UNIT_ID, unit.getId())
valuesMap.put(PostActionParameter.QUEUE_ID, queueId)
return unitActionScriptResultFactory.getPostAction(PostActionType.CREATE_TICKET,
 valuesMap)
```

Code-Beispiel 38: *Kundenaktionsskript für CM-Version 6.9.4*

```
// offer Create Ticket page for a new Service Desk ticket
import com.consol.cmas.core.server.service.action.PostActionType
import com.consol.cmas.common.model.ticket.Ticket
def newtic = new Ticket()
def qu = queueService.getByName("ServiceDesk")
newtic.setQueue(qu)
return actionScriptResultFactory.getPostAction(PostActionType.CREATE_TICKET,
 newtic, unit)
```

Code-Beispiel 39: *Kundenaktionsskript für CM-Version 6.10*

D.9.2.2 Schritt 2: Erstellen der Kundenaktionen, die das Skript verwenden

Öffnen Sie das Navigationselement *Aktionen* der Navigationsgruppe *Kunden* im Admin Tool und fügen Sie ein neues Aktionsobjekt hinzu, indem Sie auf den Button *Hinzufügen* klicken. Für das Erstellen und das Editieren einer Kundenaktion wird das gleiche Pop-up-Fenster verwendet.

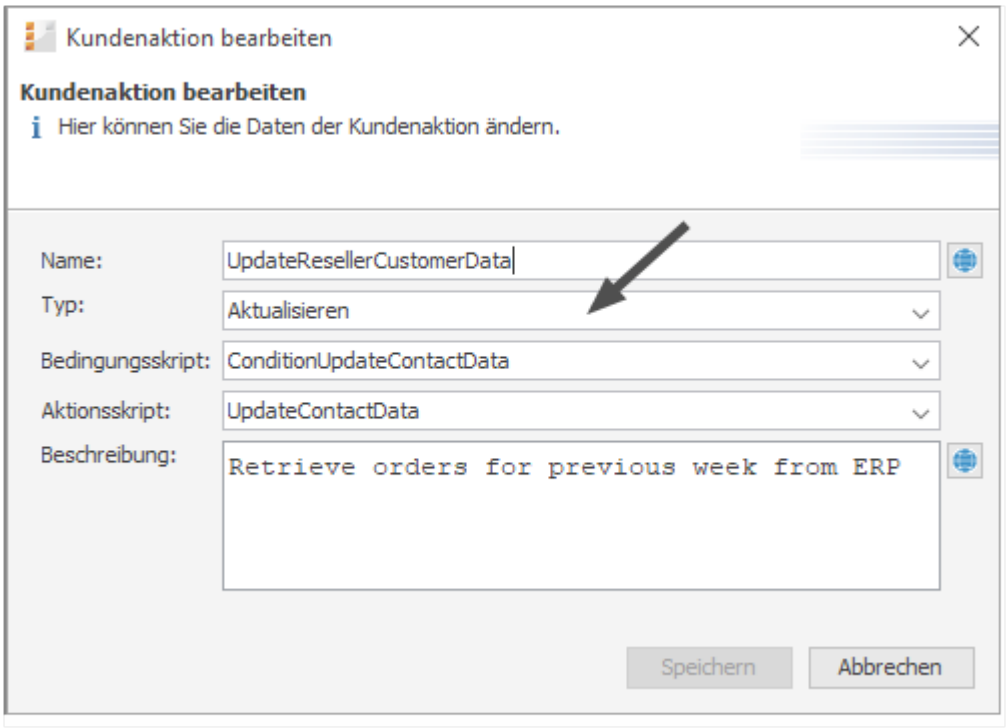


Abbildung 317: *ConSol CM Admin Tool - Kunden, Aktionen: Editieren einer Kundenaktion*

Im Pop-up-Fenster müssen die Parameter für die neue Aktion definiert werden:

- **Name**
  Der eindeutige technische Name der Aktion. Er kann mit dem Button *Lokalisieren* lokalisiert werden. Dies ist für manuelle Aktionen erforderlich, da der lokalisierte Name im Web Client unter *Aktivitäten* angezeigt wird. Details über die Lokalisierung finden Sie im Abschnitt Lokalisierung

von Objekten allgemein, Typ 1.

- **Typ**
  Mit dem Aktionstyp wird definiert, wann diese ausgeführt werden soll. Wählen Sie einen der folgenden Typen:
  - **Erzeugen**
    Das Skript wird automatisch ausgeführt, wenn der Kontakt bzw. die Firma erstellt wird.
  - **Aktualisieren**
    Das Skript wird automatisch ausgeführt, wenn der Kontakt bzw. die Firma aktualisiert wird, d. h. wenn die Daten (entweder manuell oder automatisch) verändert und gespeichert wurden. Ab CM-Version 6.10.5.4 können die Änderungen, die während der Aktion *Aktualisieren* durchgeführt wurden, mit dem Objekt `changes` überwacht werden. Dies ist detailliert im Abschnitt Arbeiten mit dem Changes-Objekt in Kundenaktionen des Typs Aktualisieren erklärt.
  - **Löschen**
    Das Skript wird automatisch ausgeführt, wenn der Kontakt bzw. die Firma gelöscht wird.
  - **Relation**
    Dieses Skript wird automatisch ausgeführt, wenn eine Relation von oder zu einem Kunden dieser Kundengruppe
    - erstellt
    - gelöscht

    wird. (Das Skript wird nicht ausgeführt, wenn der Kommentar einer Relation geändert wird.)
  - **Manuell**
    Das Skript wird als manuelle Aktivität auf der Kontakt-/Firmenseite angeboten, vorausgesetzt der Bearbeiter hat die nötigen Berechtigungen (Berechtigung *Ausführen* für diese Kundengruppe).
- **Bedingungsskript**
  Wenn vor dem Aktionsskript ein Bedingungsskript ausgeführt werden soll, muss der Name des Bedingungsskripts hier eingetragen werden. Das Ausführungsscript wird nur ausgeführt, wenn das Bedingungsskript „true“ zurückgegeben hat. Lassen Sie das Feld leer, wenn es keine Bedingung gibt.
- **Aktionsskript**
  Der Name des Aktionsskripts, das ausgeführt werden soll. Dies muss genau der Name sein, unter dem das Skript im Abschnitt *Skripte und Templates* des Admin Tools gespeichert ist.
- **Beschreibung**
  Geben Sie die Beschreibung ein, die im Web Client als Mouseover angezeigt werden soll (nur für manuelle Aktionen).

Speichern Sie die Aktion. Danach können Sie sie den Kundengruppen zuweisen. Siehe folgenden Schritt.

D.9.2.3 Schritt 3: Zuweisen von Kundenaktionen zu Kundengruppen

Damit die Kundenaktion wirksam wird, müssen Sie sie einer Kundengruppe zuweisen. Danach steht sie für alle Kunden dieser Kundengruppe zur Verfügung. Abhängig von der initialen Definition (Kontakt- oder Firmenaktion) wird die Aktion für Kontakte oder für Firmen in der Kundengruppe verfügbar sein. Um eine Kundenaktion einer Kundengruppe zuzuweisen, öffnen Sie im Admin Tool das Navigationselement *Kundengruppen* in der Navigationsgruppe *Kunden*. Wählen Sie die Kundengruppe aus, die Sie editieren möchten und klicken Sie auf den Button *Bearbeiten*, um das Pop-up-Fenster zu öffnen, in dem Sie die Kundenaktionen zuweisen können. Alle Kundenaktionen, die Sie unter *Aktionen* (siehe Schritt 2) gespeichert haben, werden hier für den jeweiligen Aktionstyp angeboten. Eine Aktion, die für den Typ *Aktualisieren* definiert wurde (siehe Schritt 2), wird z. B. nur als Aktualisierungsaktion angeboten.

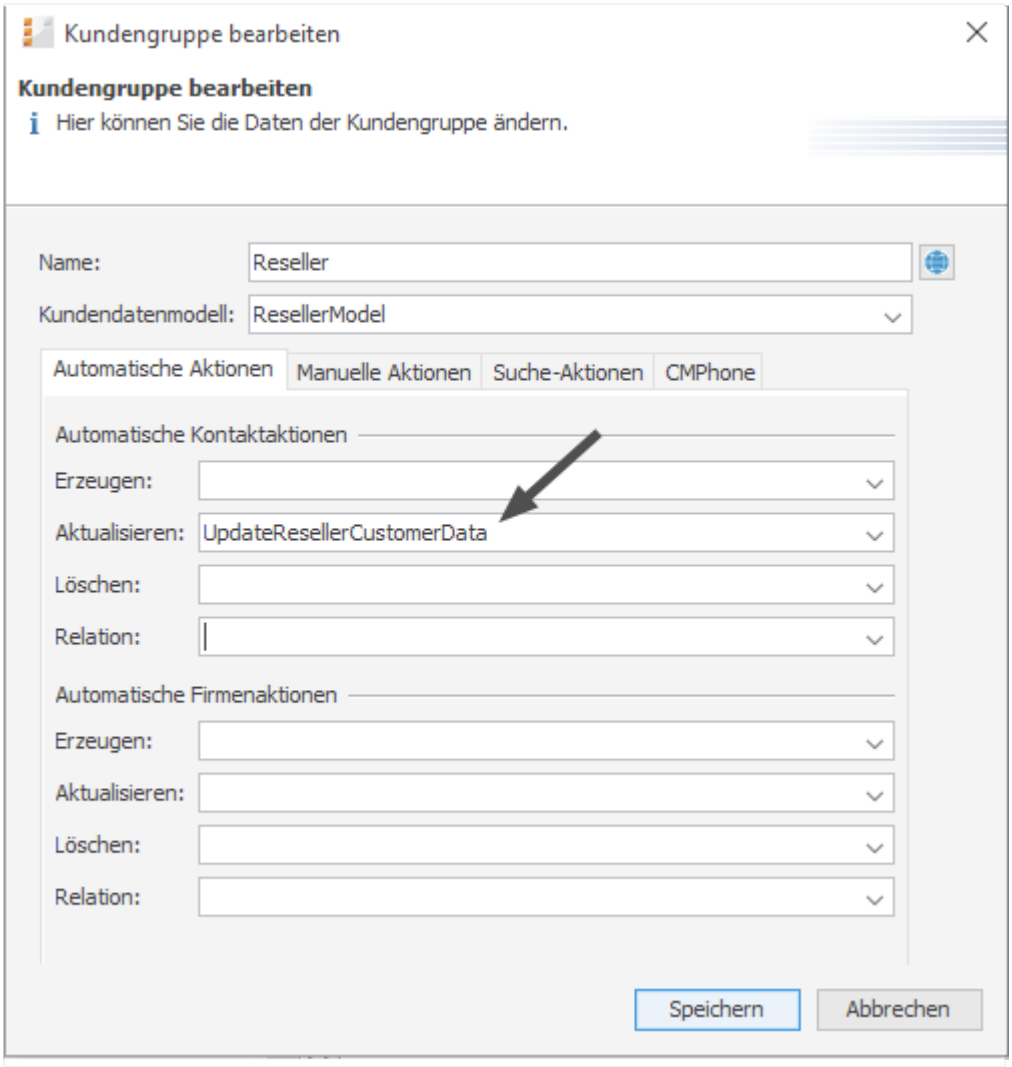


Abbildung 318: *ConSol CM Admin Tool - Kunden, Kundengruppen: Zuweisen von Kundenaktionen zu einer Kundengruppe*

Sie können die folgenden Aktionstypen einer Kundengruppe zuweisen:

- **Automatische Aktionen**
  - **Automatische Kontaktaktionen**
    Das Skript wird für den Kontakt ausgeführt.
  - **Automatische Firmenaktionen**
    Das Skript wird für die Firma ausgeführt.

  Für jeden Typ können Sie das Systemverhalten für die folgenden Operationen festlegen:

  - **Erzeugen**
    Das Skript wird automatisch ausgeführt, wenn der Kontakt bzw. die Firma erstellt wird.
  - **Aktualisieren**
    Das Skript wird automatisch ausgeführt, wenn der Kontakt bzw. die Firma aktualisiert wird, d. h. wenn die Daten (entweder manuell oder automatisch) verändert und gespeichert wurden. Ab CM-Version 6.10.5.4 können die Änderungen, die während der Aktion *Aktualisieren* durchgeführt wurden, mit dem Objekt `changes` überwacht werden. Dies ist detailliert im Abschnitt Arbeiten mit dem Changes-Objekt in Kundenaktionen des Typs Aktualisieren erklärt.
  - **Löschen**
    Das Skript wird automatisch ausgeführt, wenn der Kontakt bzw. die Firma gelöscht wird.
  - **Relation**
    Dieses Skript wird automatisch ausgeführt, wenn eine Relation von oder zu einem Kunden dieser Kundengruppe
    - erstellt
    - gelöscht

    wird. (Das Skript wird nicht ausgeführt, wenn der Kommentar einer Relation geändert wird.)
- **Manuelle Aktionen**
  Das Skript wird als manuelle Aktivität auf der Kontakt-/Firmenseite angeboten, vorausgesetzt der Bearbeiter hat die nötigen Berechtigungen (Berechtigung *Ausführen* für diese Kundengruppe).
- **Suche-Aktionen**
  Dies ist im Abschnitt Action Framework - Suche-Aktionen erklärt.

D.9.3 Verwendung von Kundenaktionen als Bearbeiter (Benutzer)

Als Bearbeiter (Benutzer) sind nur die Kundenaktionen des Typs *Manuell* für Sie relevant. Die Skripte *Erzeugen, Aktualisieren* und *Löschen* laufen im Hintergrund.

Manuelle Aktionen werden im Web Client angeboten, ähnlich wie Workflow-Aktivitäten für Tickets. Siehe *Beispiel 1* im nächsten Abschnitt.

D.9.4 Beispiele für Kundenaktionsskripte

D.9.4.1 Beispiel 1: Einfache manuelle Aktion

Eine manuelle Aktion wird als Admin-Tool-Skript programmiert und gespeichert. Dann wird eine Firmenaktion definiert, die dieses Skript verwendet, und die Aktion wird einer Kundengruppe zugewiesen.

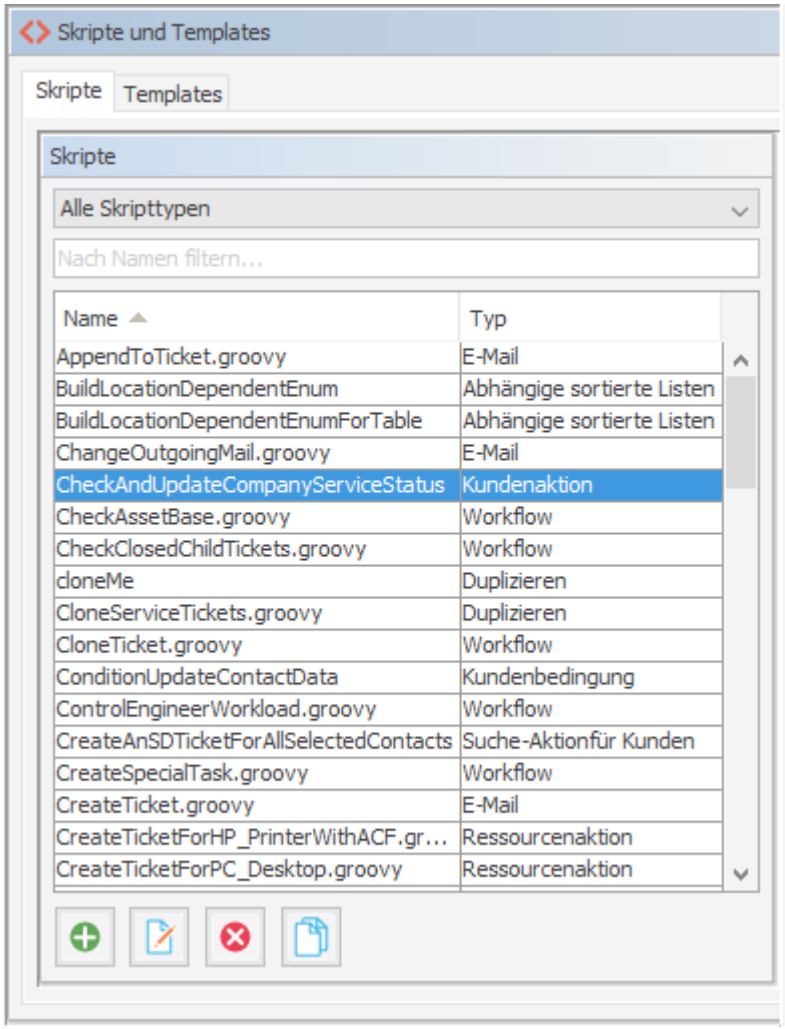


Abbildung 319: *ConSol CM Admin Tool - System, Skripte und Templates: Kundenaktionsskript für eine Firmenaktion*

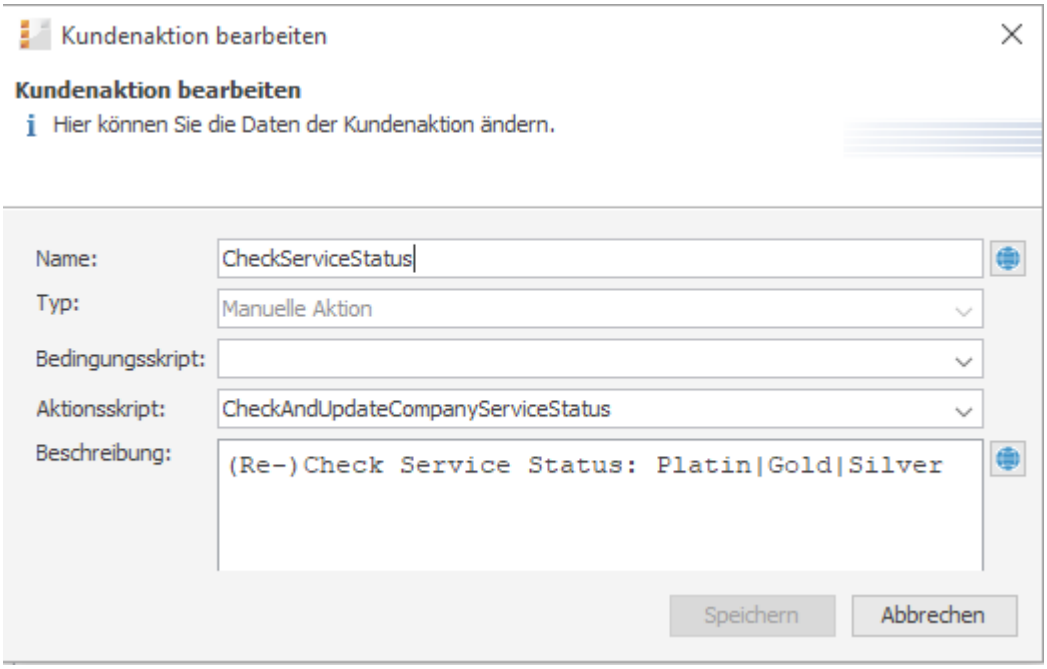


Abbildung 320: *ConSol CM Admin Tool - Kunden, Aktionen: Definieren der Firmenaktion*

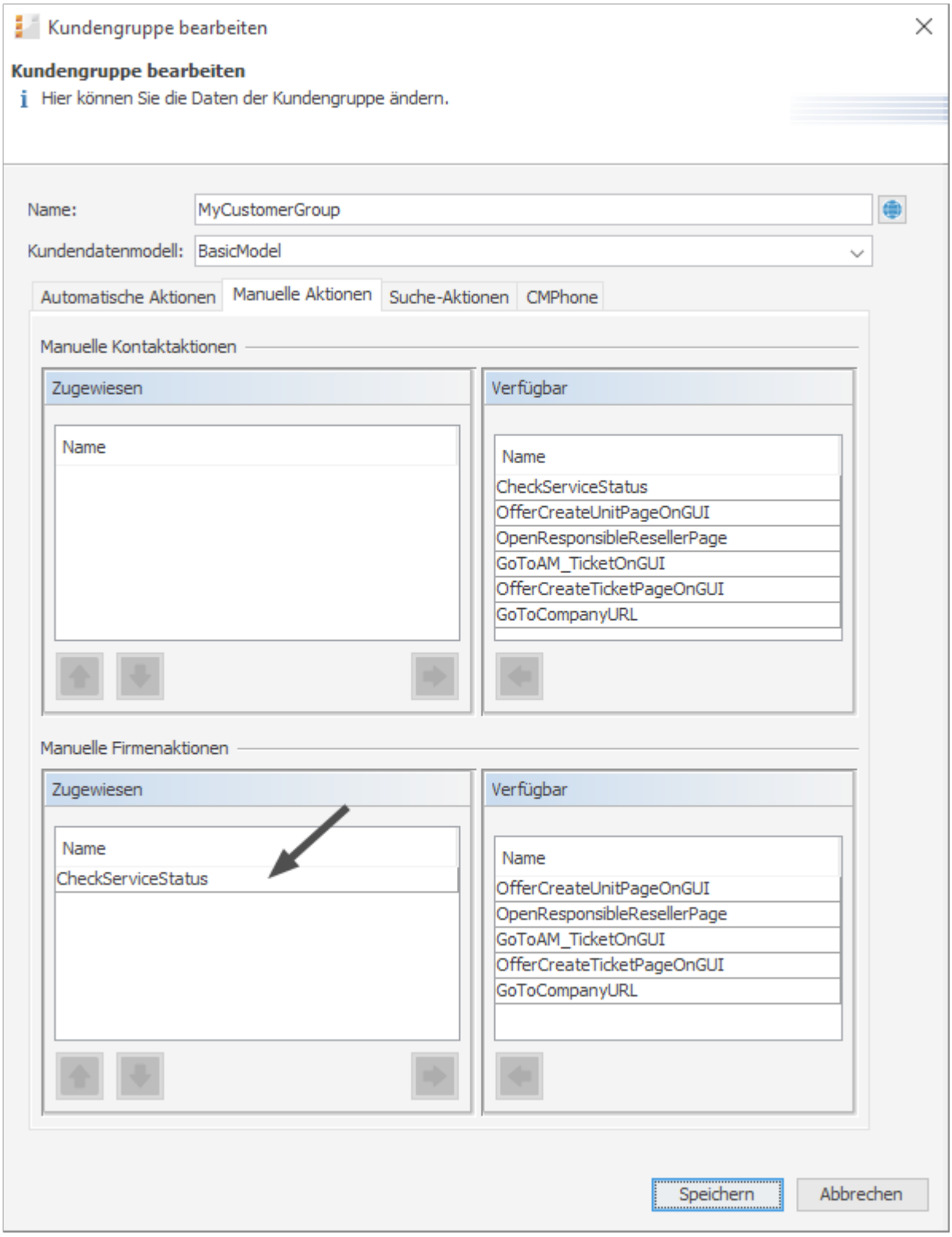


Abbildung 321: *ConSol CM Admin Tool - Kunden, Kundengruppen: Zuweisen einer Firmenaktion zu einer Kundengruppe*

Der Bearbeiter kann die Aktion manuell im Web Client ausführen.

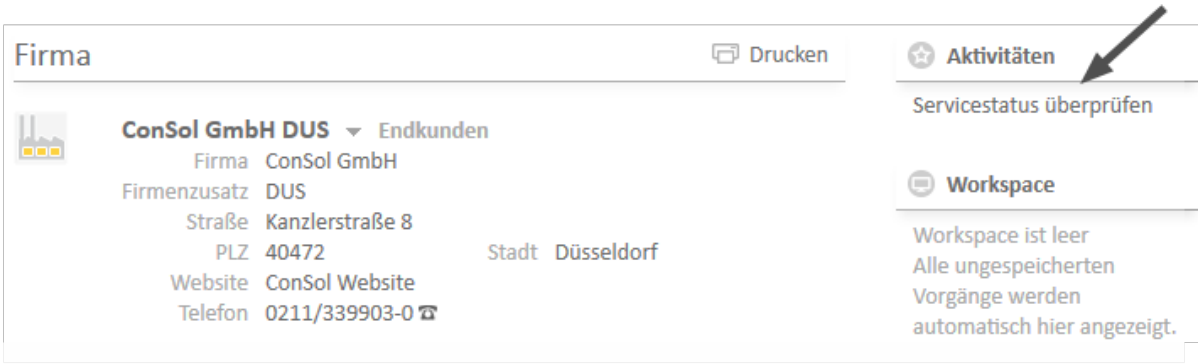


Abbildung 322: *ConSol CM Web Client - Verwenden einer manuellen Firmenaktion*

Das Skript könnte zum Beispiel folgendermaßen aussehen:

```
// check service status for a Reseller company and set new status
// this is an example for documentation purposes

import com.consol.cmas.common.model.customfield.enums.EnumValue
import com.consol.cmas.core.server.service.action.PostActionType

// ... do something, e.g., reference an external system to find the current service
 status of the company) ...
def ser_stat = enumService.getValueByName("service_status","ok")

// set the new service status for the company
unit.set("ResellerCompanyData:service_status","ok")
unitService.update(unit)

return actionScriptResultFactory.getPostAction(PostActionType.SUCCESS,
 "info.dataobject.action.success").withRefreshContent();
```

Code-Beispiel 40: *Kundenaktionsskript, CM-Version 6.10*

D.9.4.2 Beispiel 2: Neues Ticket für einen Kunden

Das folgende Skript öffnet die Seite *Neues Ticket* für den Kontakt oder die Firma, für den/die die Aktion ausgeführt wurde. Die Ziel-Queue ist *ServiceDesk*. Auf diese Weise kann man schnell ein neues Service-Desk-Ticket für den offenen Kontakt bzw. die offene Firma erstellen. Der folgende Abschnitt enthält eine Einführung in die Admin-Tool-Skripte für das Action Framework.

```
import com.consol.cmas.common.model.scripting.unit.PostActionParameterimport
import com.consol.cmas.core.server.service.UnitActionScriptResultFactoryimport
import com.consol.cmas.common.model.scripting.unit.PostActionType

def queueId = queueService.getByName("ServiceDesk").getId();
Map<String, Object> valuesMap = new HashMap<String, Object>()
valuesMap.put(PostActionParameter.UNIT_ID, unit.getId())
valuesMap.put(PostActionParameter.QUEUE_ID, queueId)
return unitActionScriptResultFactory.getPostAction("createTicket", valuesMap)
```

Code-Beispiel 41: *Kundenskript (CM-Version 6.9.4)*

```
import com.consol.cmas.core.server.service.action.PostActionType
import com.consol.cmas.common.model.ticket.Ticket
def newtic = new Ticket()
def qu = queueService.getByName("ServiceDesk")
newtic.setQueue(qu)
return actionScriptResultFactory.getPostAction(PostActionType.CREATE_TICKET,
 newtic, unit)
```

Code-Beispiel 42: *Kundenskript (CM-Version 6.10)*

Denken Sie daran, dass die Checkbox *Firma als Kunde* im Kundendatenmodell der jeweiligen Kundengruppe markiert sein muss, wenn es möglich sein soll, ein Ticket mit der Firma als Hauptkunde zu erstellen.

D.9.5 Skripte für das Action Framework: Programmierung von Kundenaktionen

Kundenaktionen werden in Admin-Tool-Skripten definiert, d. h. Groovy-Skripten, die im Bereich *Skripte und Templates* des Admin Tools gespeichert sind. Für diese Skripte ist das vordefinierte Objekt `unit` (d. h. ein Objekt der Klasse `Unit`) verfügbar. Objekte der Klasse `Unit` können je nach Kontext eine Firma oder einen Kontakt darstellen.

Es gibt zwei Arten von Skripten im Action Framework:

- Kundenaktionsskripte
- Kundenbedingungsskripte

D.9.5.1 Kundenaktionsskripte

Die Aktionen in diesem Skript werden entweder automatisch durch die Systemoperationen *Erzeugen*, *Aktualisieren* oder *Löschen* angestoßen oder durch eine manuelle Aktion des Bearbeiters (mit den *Aktivitäten* im Web Client) ausgelöst.

Skripte für automatische Kundenaktionen

```
unit.set("personalData.name", "Skywalker")
unitService.update(unit)
```

Code-Beispiel 43: *Setzen eines Wertes in den Kundendaten und Aktualisieren der Unit*

> ⚠ Wenn Sie `unitService.update(unit)` wie im obigen Beispiel verwenden, sollten Sie ein Kundenbedingungsskript einsetzen, um Endlosschleifen zu vermeiden. Siehe auch den Hinweis im Abschnitt Kundenbedingungsskripte.

Skripte für manuelle Kundenaktionen

In Skripten für manuelle Kundenaktionen können Sie einige spezielle Methoden und Objekte verwenden.

- **Methoden** (**Felder des Interfaces PostActionType**, com.consol.cmas.core.server.service.action.PostActionType)
  - **CREATE_RESOURCE**
    Ressourcenseite öffnen.
  - **CREATE_UNIT**
    Seite zum Erstellen einer Unit öffnen.
  - **CREATE_TICKET**
    Seite zum Erstellen eines Tickets öffnen.
  - **GOTO_RESOURCE**
    Detailseite der Ressource öffnen.

    - **GOTO_UNIT**
      Detailseite der Unit öffnen.
    - **GOTO_TICKET**
      Ticketseite öffnen.
    - **GOTO_PAGE**
      Webseite (URL) öffnen.
- **Objekte**
    - actionScriptResult

Eine detaillierte Erklärung der PostActionTypes finden Sie im Abschnitt Skripte für das Action Framework.

Erstellen einer Unit

(`PostActionType.CREATE_UNIT`) leitet den Benutzer auf die Seite zur Erstellung einer Unit weiter. Dabei wird der optionale Parameter `PostActionParameter.CUSTOMER_GROUP_ID` verwendet, um zu definieren, für welche Kundengruppe die neue Unit erstellt werden soll. Optional kann eine Map mit Kundenfelder übergeben werden (`PostActionParameter.FIELDS_MAP`), um die Kundenfelder der Unit mit übergebenen Werten zu füllen.

```
import com.consol.cmas.common.model.customfield.meta.FieldKey
import com.consol.cmas.common.model.customfield.AbstractField
import com.consol.cmas.common.model.customfield.StringField
import com.consol.cmas.common.model.scripting.unit.PostActionParameter
import com.consol.cmas.common.model.scripting.unit.PostActionType

Map<FieldKey, AbstractField<?>> fieldsMap = new HashMap<FieldKey, AbstractField<?>>
 ()
FieldKey firstName = new FieldKey("customer", "firstname")
FieldKey name = new FieldKey("customer", "name")
fieldsMap.put(firstName, new StringField(firstName, "Han"))
fieldsMap.put(name, new StringField(name, "Solo"))

Map<String, Object> valuesMap = new HashMap<String, Object>()
valuesMap.put(PostActionParameter.CUSTOMER_GROUP_ID, unit.getCustomerGroup().getId
 ())
valuesMap.put(PostActionParameter.FIELDS_MAP, fieldsMap)

return unitActionScriptResultFactory.getPostAction(PostActionType.CREATE_UNIT,
 valuesMap)
```

Code-Beispiel 44: *Firmenskript, mit dem einige Unit-Daten gefüllt werden, CM-Version 6.9.4*

```
// used for companies in MyCustomerGroup to create new contacts easily

import com.consol.cmas.common.model.customfield.meta.*
import com.consol.cmas.common.model.customfield.*
import com.consol.cmas.core.server.service.action.PostActionType

def myunit = new Unit()

def mycustomergroup = customerGroupService.getByName("MyCustomerGroup")
myunit.setCustomerGroup(mycustomergroup)

def mycustomerdefinition = unitDefinitionService.getByName("customer")
myunit.setDefinition(mycustomerdefinition)
myunit.set("company()", unit)

myunit.set("customer.firstname", "Han")
myunit.set("customer.name", "Solo")

return actionScriptResultFactory.getPostAction(PostActionType.CREATE_UNIT, myunit)
```

Code-Beispiel 45: *Firmenskript, mit dem einige Unit-Daten gefüllt werden, CM-Version 6.10*

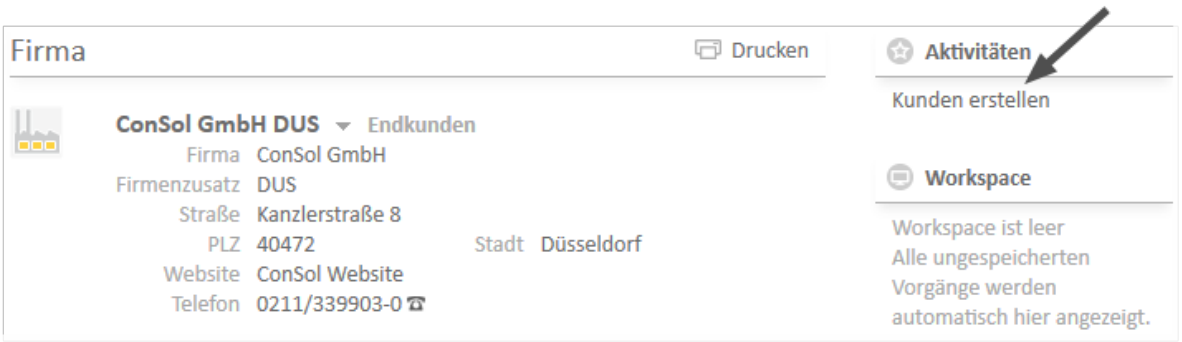


Abbildung 323: *ConSol CM Web Client - Manuelle Firmenaktion*

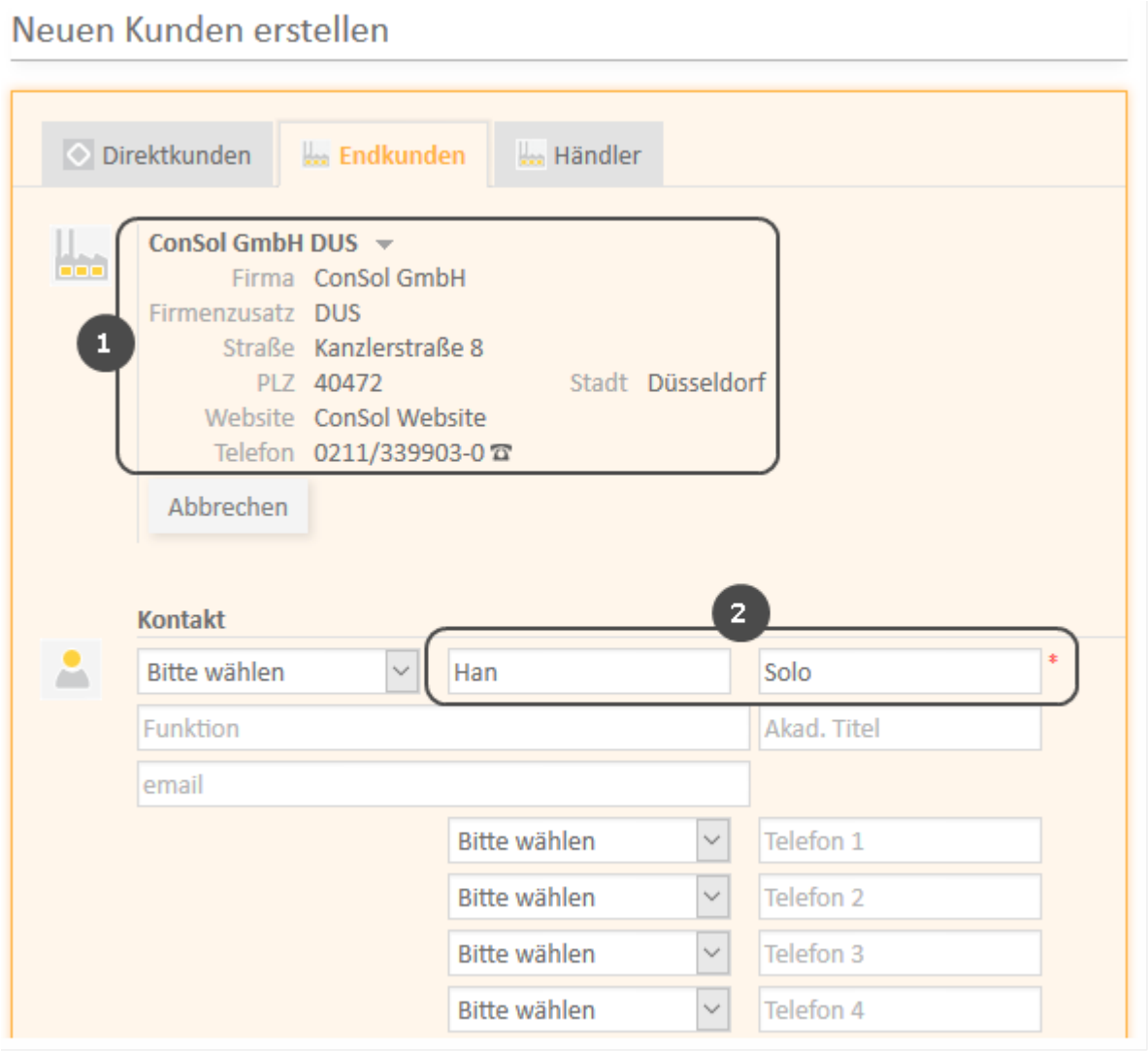


Abbildung 324: *ConSol CM Web Client - Seite zum Erstellen einer Unit geöffnet und durch Firmenaktion teilweise gefüllt*

- (1) Die Firma, für die die Aktion *Kunden erstellen* ausgeführt wird, ist schon ausgewählt.
- (2) Einige Datenfelder des Kontakts sind schon ausgefüllt.

Die Namen der im Skript verwendeten Kundenfelder müssen natürlich aus dem Kundendatenmodell stammen, das der Kundengruppe zugewiesen ist, für die ein neuer Kontakt erstellt werden soll.

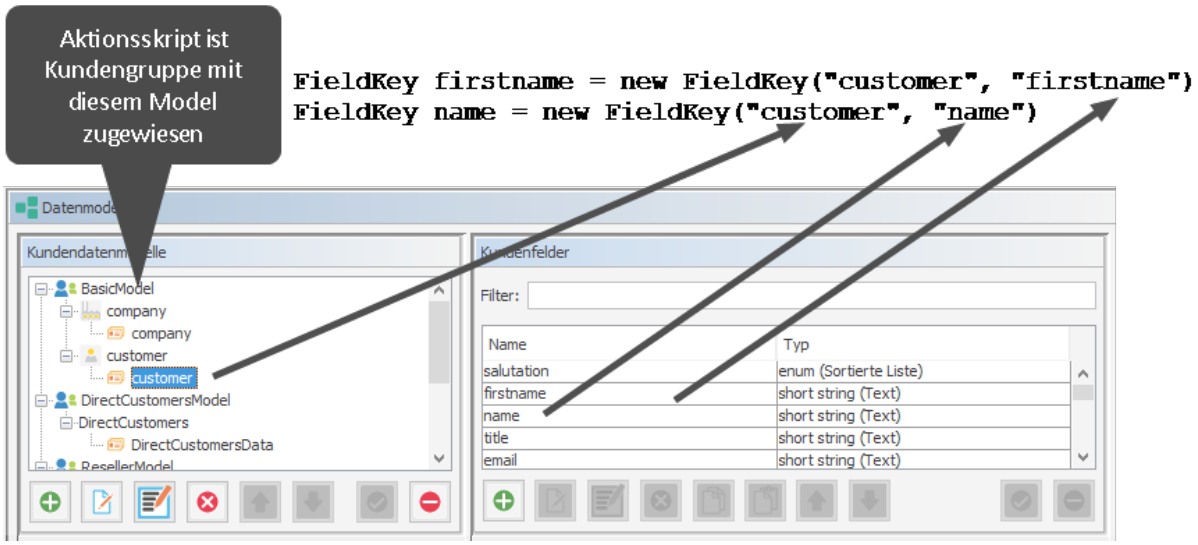


Abbildung 325: *ConSol CM Admin Tool - Kunden, Datenmodelle: Skript zum Erstellen einer Unit-Seite als Firmenaktion*

Erstellen eines Tickets

(`PostActionType.CREATE_TICKET`) leitet den Benutzer auf die Seite zum Erstellen eines Tickets weiter. Dabei werden die optionalen Parameter `PostActionParameter.UNIT_ID` mit der ID des Hauptkunden, `PostActionParameter.QUEUE_ID` mit der ID der Queue und die Map mit den Ticketfeldern `PostActionParameter.FIELDS_MAP` verwendet.

```
// offer Create Ticket page for a new HelpDesk 1st level ticket

import com.consol.cmas.common.model.scripting.unit.PostActionType
import com.consol.cmas.common.model.scripting.unit.PostActionParameter

def queueId = queueService.getByName("HelpDesk_1st_Level").getId()
Map<String, Object> valuesMap = new HashMap<String, Object>()
valuesMap.put(PostActionParameter.UNIT_ID, unit.getId())
valuesMap.put(PostActionParameter.QUEUE_ID, queueId)
return unitActionScriptResultFactory.getPostAction(PostActionType.CREATE_TICKET,
 valuesMap)
```

Code-Beispiel 46: *Mit dem Skript wird ein Aktionsergebnis erzeugt und zurückgegeben, das den Web Client anweist, ein neues Ticket mit der Unit als Hauptkontakt zu erstellen, CM-Version 6.9.4*

```
// offer Create Ticket page for a new Service Desk ticket

import com.consol.cmas.core.server.service.action.PostActionType
import com.consol.cmas.common.model.ticket.Ticket
def newtic = new Ticket()
def qu = queueService.getByName("ServiceDesk")
newtic.setQueue(qu)
return actionScriptResultFactory.getPostAction(PostActionType.CREATE_TICKET,
 newtic, unit)
```

Code-Beispiel 47: *Mit dem Skript wird ein Aktionsergebnis erzeugt und zurückgegeben, das den Web Client anweist, ein neues Ticket mit der Unit als Hauptkontakt zu erstellen, CM-Version 6.10*

Denken Sie daran, dass die Kundengruppe, auf die das Skript angewendet werden soll, der Queue zugewiesen sein muss, in der das Ticket erstellt werden soll (in diesem Beispiel *ServiceDesk*).

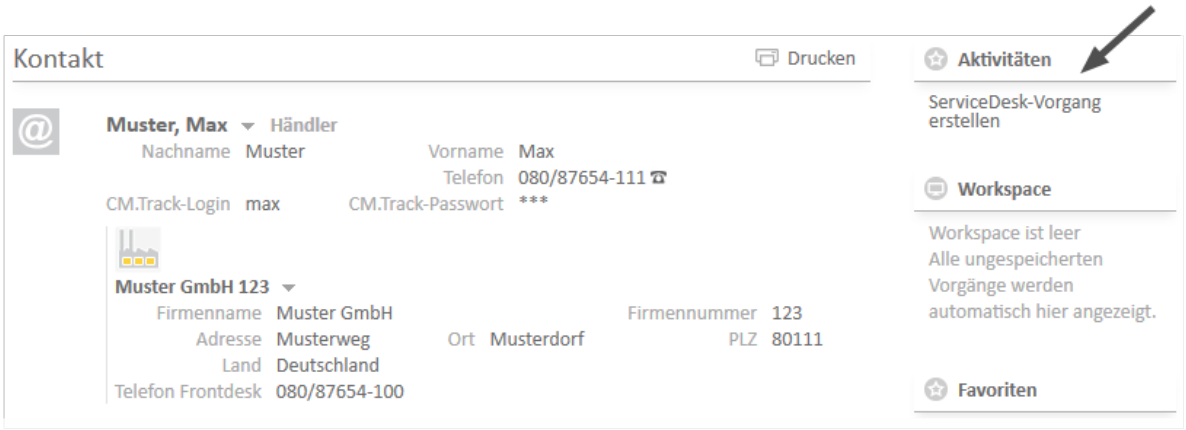


Abbildung 326: *ConSol CM Web Client - Manuelle Kontaktaktion*

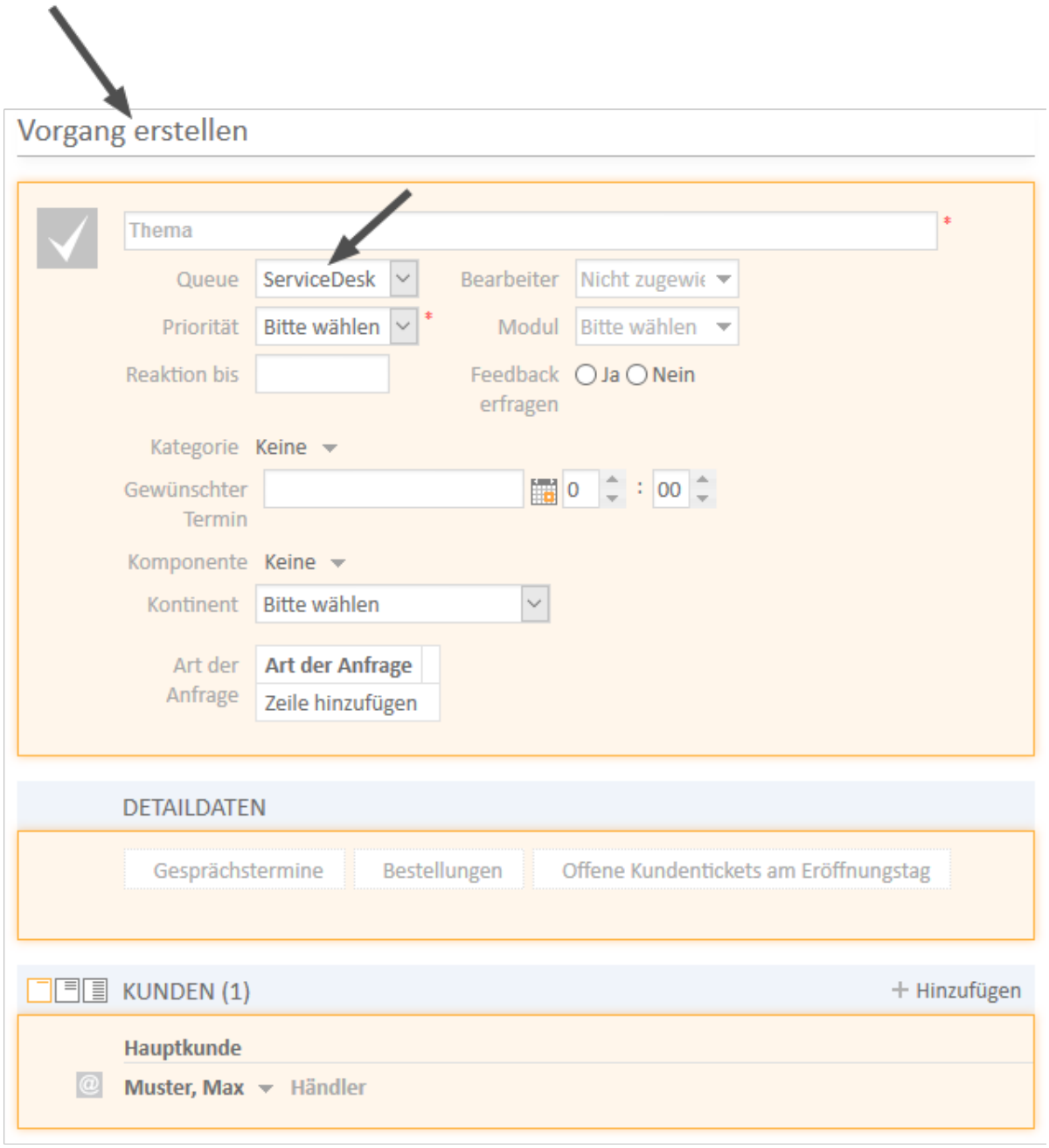


Abbildung 327: *ConSol CM Web Client - Seite zum Erstellen eines Tickets geöffnet und durch Kontaktaktion teilweise gefüllt*

Öffnen einer Unit-Seite

(`PostActionType.GOTO_UNIT`) leitet auf eine Unit-Seite weiter. Sie verwendet den obligatorischen Parameter `PostActionParameter.UNIT_ID` mit der ID der Unit.

```
import com.consol.cmas.common.model.scripting.unit.PostActionType
import com.consol.cmas.common.model.scripting.unit.PostActionParameter

Map<String, Object> valuesMap = new HashMap<String, Object>()
valuesMap.put(PostActionParameter.UNIT_ID, unit.get("company()").getId())
return unitActionScriptResultFactory.getPostAction(PostActionType.GOTO_UNIT,
 valuesMap)
```

Code-Beispiel 48: *Skript, das die Firmenseite öffnet, CM-Version 6.9.4*

Beispiel: Aufrufen der Kontaktdetailseite eines Endkunden und Öffnen der Firmendetailseite der Händlerfirma, die für diesen Endkunden zuständig ist. Eine Relation zwischen Firma und Kontakt muss zuvor hergestellt worden sein. Wenn mehr als eine Händlerrelation definiert wurde, wird die erste Relation in der Liste verwendet.

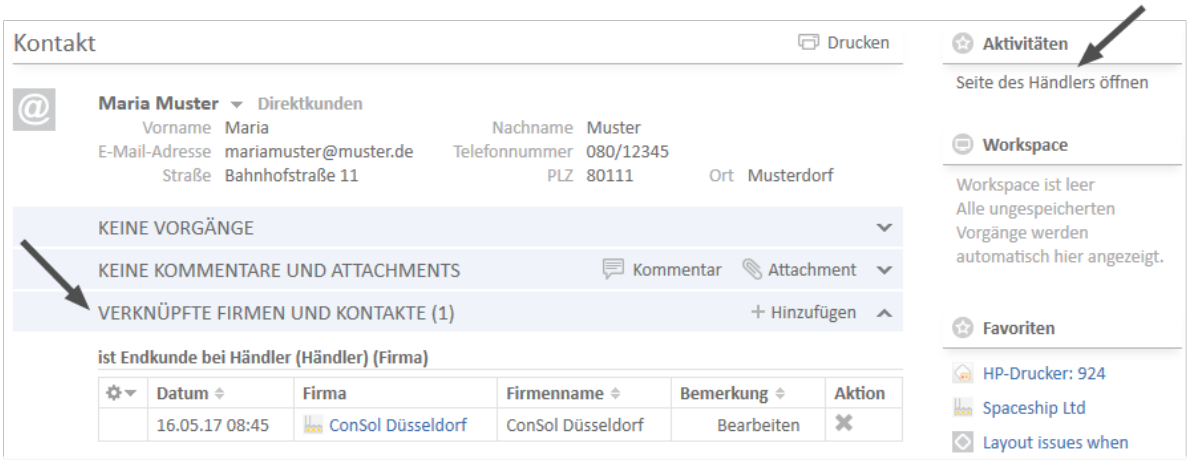


Abbildung 328: *ConSol CM Web Client - Kundenaktion auf Kontaktdetailseite (Aktivität öffnet die Firmendetailseite des zuständigen Händlers, CM-Version 6.10)*

```
// Open company detail page of responsible reseller company, uses first reseller in
 list
// should only be executed if this relation exists. A condition script is used to
 check

import com.consol.cmas.common.service.UnitRelationDefinitionService
import com.consol.cmas.core.server.service.action.PostActionType
import com.consol.cmas.common.model.customfield.UnitRelation

// find responsible reseller
def unit_rel_def = unitRelationDefinitionService.getByName
 ("ResellerDirectCustomersRelation")

Set<UnitRelation> res_relations = unitRelationService.getByDefinitionAndTarget
 (unit_rel_def, unit)

if (res_relations.size() > 0) {
   def source_unit = res_relations.toArray()[0].getSourceUnit()
   // log.info("SOURCE UNIT IS NOW " + source_unit.get
    ("ResellerCompanyData:company_name"))
   return actionScriptResultFactory.getPostAction(PostActionType.GOTO_UNIT, source_
    unit)

} else {
   log.info("ERROR -- no responsible reseller unit found")
   return actionScriptResultFactory.getPostAction(PostActionType.FAILURE,
    "action.result.failure" )
}
```

Code-Beispiel 49: *Skript, das die Firmenseite öffnet, CM-Version 6.10*

Falls die Aktion *Seite des Händlers öffnen (OpenResponsibleResellerPage)* nur im Web Client angeboten werden soll, wenn ein zuständiger Händler festgelegt wurde, können Sie mit einem Kundenbedingungsskript arbeiten. Dies muss als Admin-Tool-Skript des Typs *Kundenbedingung* erstellt werden und der Kundenaktion im Abschnitt *Aktionen* zugewiesen werden.

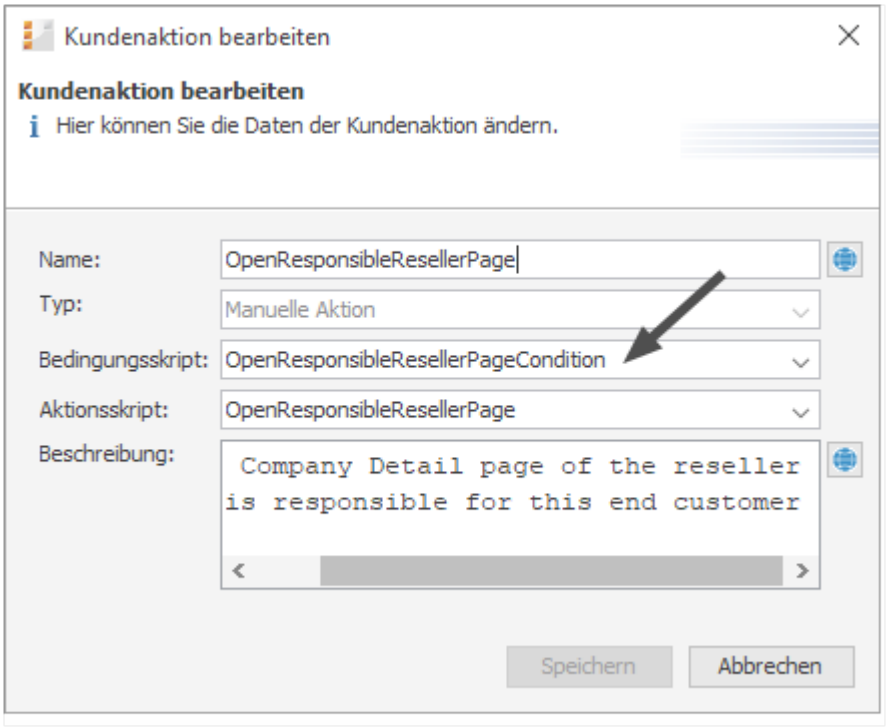


Abbildung 329: *ConSol CM Admin Tool - Kunden, Aktionen: Zuweisen eines Kundenbedingungsskripts*

```
// Checks if reseller relation is set

import com.consol.cmas.common.service.UnitRelationDefinitionService
import com.consol.cmas.core.server.service.action.PostActionType
import com.consol.cmas.common.model.customfield.UnitRelation

// find responsible reseller
def unit_rel_def = unitRelationDefinitionService.getByName
 ("ResellerDirectCustomersRelation")
Set<UnitRelation> res_relations = unitRelationService.getByDefinitionAndTarget
 (unit_rel_def, unit)
if (res_relations.size() > 0) {
   return true
} else {
   return false
}
```

Code-Beispiel 50: *Kundenbedingungsskript, das überprüft, ob es eine Reseller-Relation gibt*

Öffnen einer Ticketseite

(`PostActionType.GOTO_TICKET`) leitet auf eine Ticketseite weiter. Sie verwendet den obligatorischen Parameter `PostActionParameter.TICKET_ID` mit der ID des Tickets.

```
import com.consol.cmas.common.model.scripting.unit.PostActionType
import com.consol.cmas.common.model.scripting.unit.PostActionParameter
import com.consol.cmas.common.model.customfield.Unit
import com.consol.cmas.common.model.ticket.TicketCriteria
import com.consol.cmas.common.model.customfield.ListField
import com.consol.cmas.common.model.customfield.ContactReferenceField
import com.consol.cmas.common.model.customfield.UnitReferenceSearchField
import com.consol.cmas.common.model.customfield.ContactReferenceSearchField
import com.consol.cmas.common.model.customfield.meta.FieldKey
import com.consol.cmas.common.model.ticket.Ticket
import com.consol.cmas.common.model.ContactTicketRole
import com.consol.cmas.common.model.customfield.StringField
import com.consol.cmas.common.model.scripting.unit.UnitActionScriptResult


//get AM queue for search
def q_id = (workflowApi.getQueueByName("AccountManagement")).id
def q_ids = new HashSet()
q_ids.add(q_id)

//find AM ticket for the company
def crit = new TicketCriteria()
crit.setQueueIds(q_ids)

// create list field key
def contactSearchListFieldKey = new FieldKey("queue_fields","contacts")

// prepare list field
def contactsListField = new ListField(contactSearchListFieldKey )

// create member field key
def contactSearchFieldKey = new FieldKey("queue_fields","contacts_member")

// create unit member field with Unit and ticket main role

def contactsMember = new
ContactReferenceSearchField(contactSearchFieldKey, unit,
ContactTicketRole.MAIN_ROLE)

// put member field in Unit list field
contactsListField.addChild(contactsMember)

// put field(s) into the criteria
crit.setFields([contactsListField] as Set)

// seek and find
def foundTickets = ticketService.getByCriteria(crit)

if ( foundTickets ) {
   def AM_tic = foundTickets.first()
   def AM_tic_id = AM_tic.id

   // go to AM ticket
```

```
   Map<String, Object> valuesMap = new HashMap<String, Object>()
   valuesMap.put(PostActionParameter.TICKET_ID, AM_tic_id)
   return unitActionScriptResultFactory.getPostAction(PostActionType.GOTO_TICKET,
    valuesMap)
}

// Default: found nothing
return null
```

Code-Beispiel 51: *Öffnen einer Ticketseite im Ansichtsmodus, CM-Version 6.9*

```
import com.consol.cmas.common.model.customfield.Unit
import com.consol.cmas.common.model.ticket.TicketCriteria
import com.consol.cmas.common.model.customfield.ListField
import com.consol.cmas.common.model.customfield.ContactReferenceField
import com.consol.cmas.common.model.customfield.UnitReferenceSearchField
import com.consol.cmas.common.model.customfield.ContactReferenceSearchField
import com.consol.cmas.common.model.customfield.meta.FieldKey
import com.consol.cmas.common.model.ticket.Ticket
import com.consol.cmas.common.model.ContactTicketRole
import com.consol.cmas.common.model.customfield.StringField
import com.consol.cmas.core.server.service.action.PostActionType

//get AM queue for search
def q_id = (workflowApi.getQueueByName("AccountManagement")).id
def q_ids = new HashSet()
q_ids.add(q_id)
//find AM ticket for the company
def crit = new TicketCriteria()
crit.setQueueIds(q_ids)
// Listenfeld-Key erzeugen
def contactSearchListFieldKey = new FieldKey("queue_fields","contacts")
// Listenfeld vorbereiten
def contactsListField = new ListField(contactSearchListFieldKey )
// Memberfeld-Key erzeugen
def contactSearchFieldKey = new FieldKey("queue_fields","contacts_member")
// Unit-Memberfeld mit Unit und Ticket-Hauptrolle erzeugen
// COmpany is MAIN CONTACT at the AM ticket!
def contactsMember = new ContactReferenceSearchField(contactSearchFieldKey, unit,
 ContactTicketRole.MAIN_ROLE)
// Member-Feld in Unit-Listenfeld stopfen
contactsListField.addChild(contactsMember)
// Feld(er) in die Kriterien stopfen
crit.setFields([contactsListField] as Set)
// Suchen und finden
def foundTickets = ticketService.getByCriteria(crit)
println "Found tickets: ${foundTickets}"

if ( foundTickets ) {
   def AM_tic = foundTickets.first()
   return actionScriptResultFactory.getPostAction(PostActionType.GOTO_TICKET, AM_
    tic)
```

```
}
// Default: found nothing
return null
```

Code-Beispiel 52: *Öffnen einer Ticketseite im Ansichtsmodus, CM-Version 6.10*

> ⚠ Beachten Sie bei der Verwendung von einem der obigen Skripte, dass die Kundenfelder der Firma für die Ticketsuche indiziert sein müssen (Annotation `field-indexed` = „true“).

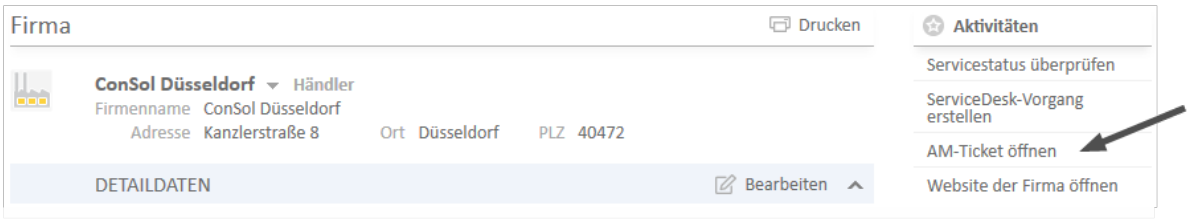


Abbildung 330: *ConSol CM Web Client - Firmenaktion auf der Firmenseite (1)*

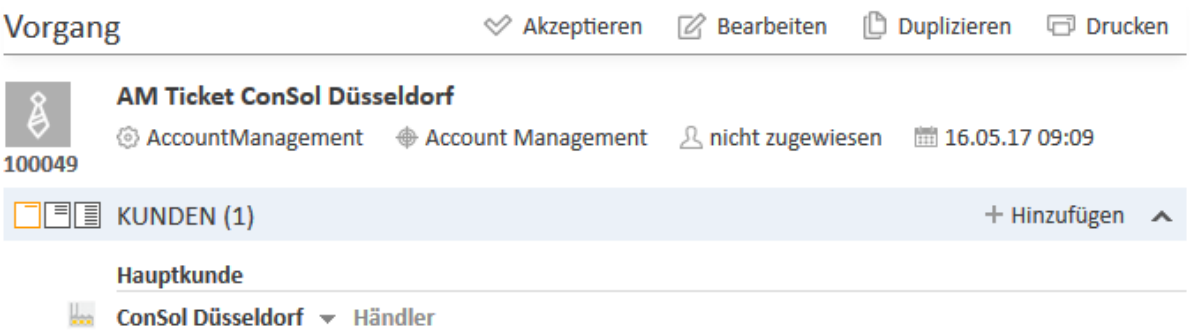


Abbildung 331: *ConSol CM Web Client - AM-Seite nach der Firmenaktion AM-Ticket öffnen*

Öffnen einer Webseite

(`PostActionType.GOTO_PAGE`) leitet an eine URL weiter. Sie verwendet den obligatorischen Parameter `PostActionParameter.URL` mit der URL.

Der folgende Code zeigt ein einfaches Beispiel für eine festgelegte URL für jede Firma.

```
import com.consol.cmas.common.model.scripting.unit.PostActionType
import com.consol.cmas.common.model.scripting.unit.PostActionParameter

Map<String, Object> valuesMap = new HashMap<String, Object>()
valuesMap.put(PostActionParameter.URL, unit.get("company:www"))
return unitActionScriptResultFactory.getPostAction(PostActionType.GOTO_PAGE,
 valuesMap)
```

Code-Beispiel 53: *Skript, das eine bestimmte Website (URL) öffnet, CM-Version 6.9.4*

```
// opens company's web site
import com.consol.cmas.core.server.service.action.PostActionType

def url = unit.get("url")

if (!url) {
   return actionScriptResultFactory.getPostAction(PostActionType.FAILURE,
    "error.script.no.url")
} else {
   return actionScriptResultFactory.getPostAction(PostActionType.GOTO_PAGE,url)
}
```

Code-Beispiel 54: *Skript, das eine bestimmte Website (URL) öffnet, CM-Version 6.10*

Der String `error.script.no.url` ist ein in der Navigationsgruppe *Bezeichnungen* definiertes Label, siehe Abschnitt Bezeichnungen.

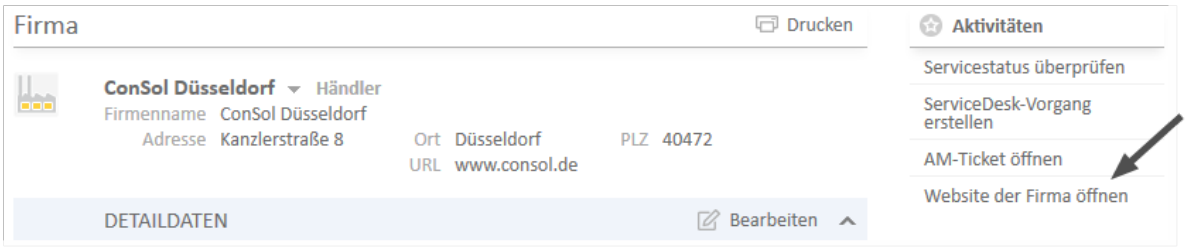


Abbildung 332: *ConSol CM Web Client - Firmenaktion auf der Firmenseite (2)*

⚠ Um eine festgelegte URL zu öffnen, können Sie ein Kundenfeld des Typs STRING mit der Annotation `text-type` = „url“ verwenden. Dies erzeugt automatisch einen Hyperlink. Deswegen ist die Verwendung des Parameters `GOTO_URL` in Kundenaktionsskripten nur empfehlenswert, wenn die URL mit dem Skript dynamisch gebildet wird.

D.9.5.2 Kundenbedingungsskripte

Ein Kundenbedingungsskript definiert, ob die Aktion im Web Client angezeigt werden soll oder nicht. Es wird vor dem Kundenaktionsskript ausgeführt. Wenn es „false“ zurückgibt, wird das Kundenaktionsskript nicht ausgeführt.

```
if(unit.getFieldValue("customer.personalData") == null) {
   return true
} else {
   return false
}
```

Code-Beispiel 55: *Kundenbedingungsskript*

Um ein Kundenbedingungsskript einsetzen zu können, müssen Sie folgende Schritte durchführen:

1. Schreiben Sie ein Admin-Tool-Skript des Typs *Kundenbedingung*. Das Skript muss die Bedingungen für die Rückgabe von „true“ und „false“ definieren. Siehe Beispiel oben. An dieser Stelle müssen Sie bereits wissen, für welche Kundengruppen und Datenmodelle das Skript verwendet werden soll, um sicherzustellen, dass das Skript die richtigen Datenfelder referenziert.
2. Weisen Sie das Kundenbedingungsskript einer oder mehreren Kundenaktionen zu. Dies wird im Abschnitt *Aktionen* der Navigationsgruppe *Kunden* gemacht. Auf diese Weise wird dieses Kundenbedingungsskript immer vor dem Kundenaktionsskript dieser Kundenaktion ausgeführt.

Das ist alles. Sie müssen das Bedingungsskript keiner Kundengruppe zuweisen. Dies geschieht implizit, wenn die entsprechende Kundenaktion den Kontakten oder Firmen bestimmter Kundengruppen zugewiesen wird.

D.9.5.3 Wichtige Groovy-Objekte

Objekt `UnitActionScriptResult` (CM-Version 6.9)

Das Objekt `UnitActionScriptResult` wird nur für manuelle Aktionen berücksichtigt. Es steht für Aktionen wie *Erzeugen*, *Aktualisieren* oder *Löschen* nicht zur Verfügung. Das Objekt `UnitActionScriptResult` wird von der Methode `unitActionScriptResultFactory.getPostAction(String, Map<String, Object>)` erzeugt. Mit dieser Klasse (bzw. diesem Objekt) werden Informationen gespeichert, die die Prozesse beeinflussen, die nach dem Ausführen der manuellen Aktion im Web Client stattfinden. Das Objekt `UnitActionScriptResult` enthält den manuellen Aktionstyp, die IDs des Tickets, der Unit, der Queue und der Kundengruppe. Nach dem Ausführen der manuellen Aktion kann der Benutzer auf eine andere Seite weitergeleitet werden.

Objekt `actionScriptResult` (CM-Version 6.10 und höher)

Ab CM-Version 6.10 bietet das Action Framework die Klassen `ActionScriptResult` und `ActionScriptResultFactory`. Ein Objekt (singleton) der Klasse `ActionScriptResultFactory` ist als `actionScriptResultFactory` in allen Aktionsskripten, unabhängig vom Typ des Aktionsskripts, verfügbar.

Details finden Sie im Abschnitt Skripte für das Action Framework.

> ⚠ Die Kundenaktionen *Erzeugen*, *Aktualisieren* und *Löschen* werden in den Basismethoden `create`, `update` und `delete` des Objekts `unitService` ausgeführt.
>
> Wenn also ein Aktionsskript das Typs *Aktualisieren* den Kunden mit der Methode `unitService.update(Unit)` aktualisiert, kann dies den Fehler `java.lang.StackOverflowError` verursachen, da die Aktualisierungsaktion unendlich oft ausgeführt wird. In diesem Fall sollten Sie ein Kundenbedingungsskript verwenden, um solche Endlosschleifen zu vermeiden.

D.9.5.4 Arbeiten mit dem Changes-Objekt in Kundenaktionen des Typs Aktualisieren

Ab CM-Version 6.10.5.4 ist es möglich, die Änderungen zu überwachen, die während einer Aktion zur Kundenaktualisierung durchgeführt wurden. (Das gleiche gilt für die Aktionen *Aktualisieren* von Ressourcen, siehe Abschnitt Arbeiten mit dem Changes-Objekt in Ressourcenaktionen des Typs Aktualisieren).

Mit dem Objekt der Klasse `UnitChanges` in Unit-Aktionen können Sie herausfinden, welche Änderungen erfolgt sind.

Denken Sie daran, dass das Skript *Aktualisieren* in folgenden Fällen ausgeführt wird:

- bei einer expliziten Aktion des Typs *Aktualisieren*
- wenn Kommentare und/oder Attachments hinzugefügt werden
- wenn Kommentare und/oder Attachments entfernt werden

Es gibt zwei Methoden des unitChanges-Objekts, die Informationen über die geänderten Daten bereitstellen:

- `getCustomFieldChangeInfo()`
  liefert Informationen über Änderungen an Unit-Daten (in Kundenfeldern)
- `getContentChangeInfo()`
  liefert Informationen über Änderungen im Unit-Protokoll (Kommentare, Attachments)

Da die Rückgabeparameter der Methode ziemlich komplexe Bestandteile haben, empfehlen wir, die API Doc der Klasse `UnitChanges` zu lesen. Das folgende Codebeispiel enthält ein Skript, in dem das Objekt `unitChanges` verwendet wird.

```
// Update Action Script UpdateContactData.groovy for contacts in Reseller group
import com.consol.cmas.common.model.content.unit.UnitCommentEntry
import com.consol.cmas.common.model.content.unit.UnitAttachmentEntry
log.info 'Contact data have been UPDATEd!'
// Are there any changes?
if (changes) {
   log.info 'Yes, changes have been made to unit'
   log.info 'Changes object is a ' + changes.class
}
// Have Custom Fields (Data Object Group Fields) been changed? If yes - which?
if (changes.customFieldChangeInfo) {
```

```
   log.info 'Yes, changes have been made to Custom Fields'
   log.info changes.customFieldChangeInfo
   log.info changes.customFieldChangeInfo.each { k, v ->
   log.info "Changed field: ${k.groupName}/ ${k.fieldName}"
   log.info "New value: ${v.value.value}"
   log.info "Old value: ${v.previousValue.value}"
}
} else {
   log.info 'No changes to Custom Fields'
}

// Have comments or attachmenst been changed? If yes - which?
log.info changes.contentChangeInfo
if (changes.contentChangeInfo){
   log.info 'Yes, changes have been made in detail section'
   if (changes.contentChangeInfo.value){
      log.info changes?.contentChangeInfo.each { ctEntry ->

         if (ctEntry?.value[0] instanceof UnitCommentEntry){
            log.info 'A comment has been added.'
            log.info 'Old value: ' + ctEntry?.previousValue
            log.info 'New value: ' + ctEntry.value[0]?.text
            log.info 'Made by the engineer ' + ctEntry.value[0]?.engineer?.name
            log.info 'Creation date of the comment: ' + ctEntry.value
             [0]?.creationDate
         } else if (ctEntry?.value[0] instanceof UnitAttachmentEntry){
            log.info 'An attachment has been added.'
            log.info 'Old value: ' + ctEntry?.previousValue
            log.info 'New value text: ' + ctEntry.value[0]?.text
            log.info 'New value file name: ' + ctEntry.value[0]?.filename
         }
      }
   } else {
      log.info 'Entry has been deleted.'
   }
}
```

Code-Beispiel 56: *Unit-Aktualisierungsskript, in dem Änderungen überwacht und in die Datei server.log geschrieben werden*

Wenn bei einem Kontakt der Kundengruppe *Resellers* Änderungen an den Kundenfeldern *phone* und *vip_person* gemacht wurden, wird folgender Text in der Datei `server.log` angezeigt.

```
database_UpdateContactData] [Susan-] Contact data have been UPDATEd!
database_UpdateContactData] [Susan-] Yes, changes have been made to unit
database_UpdateContactData] [Susan-] Changes object is a class
 com.consol.cmas.common.model.history.unit.UnitChanges
database_UpdateContactData] [Susan-] Yes, changes have been made to Custom Fields
database_UpdateContactData] [Susan-] {(phone,ResellerCustomerData)=Modification
 {value=AbstractField{key=(phone,ResellerCustomerData), value=0211/1231777},
 previousValue=AbstractField{key=(phone,ResellerCustomerData),
 value=0211/1231666}}, (vip_person,ResellerCustomerData)=Modification
 {value=AbstractField{key=(vip_person,ResellerCustomerData), value=false},
 previousValue=AbstractField{key=(vip_person,ResellerCustomerData), value=true}}}
database_UpdateContactData] [Susan-] Changed field: ResellerCustomerData/ phone
database_UpdateContactData] [Susan-] New value: 0211/1231777
database_UpdateContactData] [Susan-] Old value: 0211/1231666
database_UpdateContactData] [Susan-] Changed field: ResellerCustomerData/ vip_
 person
database_UpdateContactData] [Susan-] New value: false
database_UpdateContactData] [Susan-] Old value: true
```

Code-Beispiel 57: *Log-Ausgabe für das obige Skript*

D.10 Adress-Vervollständigung

In diesem Kapitel werden folgende Themen behandelt:

D.10.1 Einleitung 499

D.10.2 Einschalten der Adress-Vervollständigung mit dem Admin Tool 502

D.10.3 Importieren von PLZ-/Ort-/Adressdaten in die CM-Datenbank 504

D.10.4 Definieren der Adress-Vervollständigungskonfiguration mit dem Admin Tool 505

D.10.5 Editieren einer Adress-Vervollständigungskonfiguration 509

D.10.6 Löschen einer Adress-Vervollständigungskonfiguration oder von Adress-Vervollständigungsfeldern. 509

D.10.1 Einleitung

In einigen ConSol CM-Systemen müssen die Bearbeiter eine große Menge an Kundendaten manuell eingeben oder editieren. In solchen Fällen kann es hilfreich sein, dass das System folgende Funktionen unterstützt:

- Vorschläge für die Eingabe in einigen Datenfeldern (z. B. PLZ oder Adresse), um sicherzustellen, dass die eingegebene Adresse tatsächlich existiert
- Vermeidung von Duplikaten

Zur Unterstützung solcher Fälle bietet ConSol CM die Funktion *Adress-Vervollständigung*. Diese Funktion kann über eine System-Property ein- und ausgeschaltet werden. In den Standardinstallationen von ConSol CM ist sie ausgeschaltet. Wenn sie eingeschaltet ist, kann der Bearbeiter Vorschläge für die Eingabe in ein oder mehrere Kundenfelder erhalten:

- Ein oder mehrere Felder, die eine Postleitzahl enthalten
- Ein oder mehrere Felder, die einen Ort enthalten
- Ein oder mehrere Felder, die eine Adresse enthalten (Straße und Hausnummer)

Das gilt für folgende Vorgänge im CM Web Client:

- Erstellen eines Kunden auf der Seite *Neuer Kunde*
- Erstellen eines Kunden im Kundenbereich eines Tickets
- Editieren eines Kunden auf der Kundenseite
- Editieren eines Kunden im Kundenbereich eines Tickets
- Eingeben von Kundendaten auf der Seite *Detailsuche*
- Eingeben von Kundendaten in einem ACF (Aktivitätsformular)

Normalerweise dient ein öffentlich verfügbarer Datensatz, z. B. von einer CD ROM, als Quelle für den Import von Adressdaten. Auf diese Weise können die Bearbeiter implizit eine große Anzahl an Zuordnungen von PLZ/Ort/Adresse ausnutzen.

> ⚠ Beachten Sie, dass Ihr Unternehmen selbst eine Adressensammlung kaufen muss, um diese Funktion zu nutzen, d. h. die Funktion *Adress-Vervollständigung* basiert auf einem Import von externen Daten, die nicht Teil der ConSol CM-Lieferung sind!

Im folgenden Beispiel wurde eine deutsche Adressensammlung in ein ConSol CM-Demosystem importiert. In den drei folgenden Beispielen aus dem entsprechenden Web Client können Sie sehen, wie ConSol CM die Bearbeiter bei der Verbesserung der Qualität der eingegebenen Daten unterstützt.

Beispiel 1: Der Bearbeiter beginnt, in das Feld *PLZ* zu schreiben. Es werden nur existierende Postleitzahlen angeboten. Die Anzahl der angezeigten Vorschläge ist Teil der Adress-Vervollständigungskonfiguration, siehe Abschnitt Definieren der Adress-Vervollständigungskonfiguration mit dem Admin Tool.

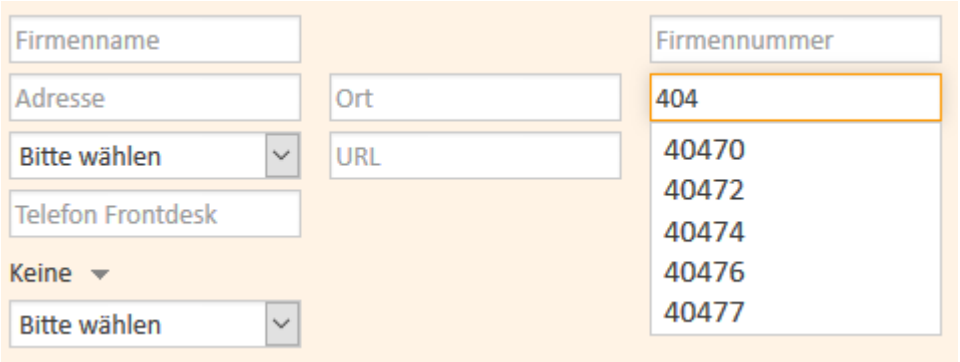


Abbildung 333: *ConSol CM Web Client - Vorschläge der Adress-Vervollständigungsfunktion, Beispiel 1*

Beispiel 2: Das Feld *PLZ* wurde bereits ausgefüllt. Für das Feld *Ort* werden nur mögliche korrekte Werte vorgeschlagen.

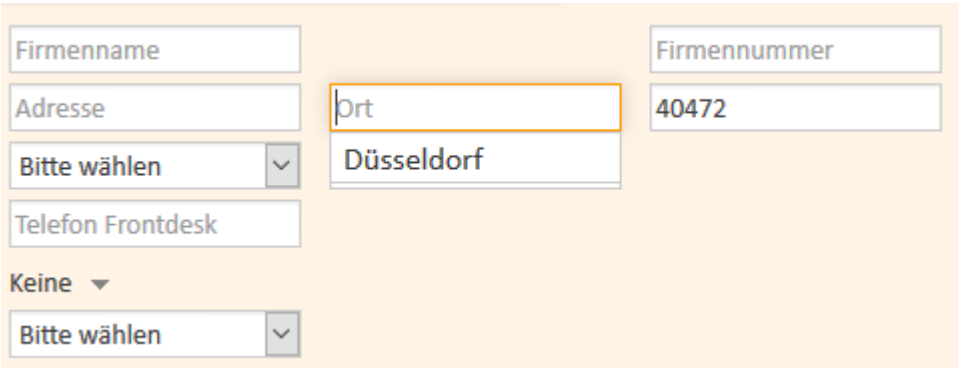


Abbildung 334: *ConSol CM Web Client - Vorschläge der Adress-Vervollständigungsfunktion, Beispiel 2*

Beispiel 3: Die Felder *PLZ* und *Ort* wurden bereits ausgefüllt. Für das Feld *Adresse* werden nur mögliche korrekte Werte vorgeschlagen.

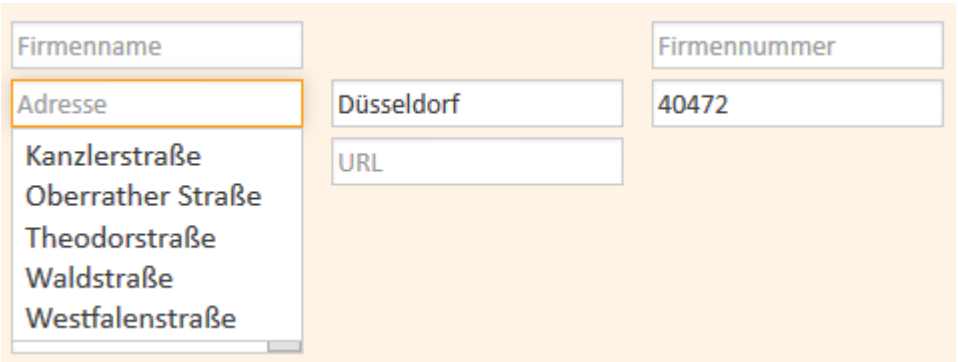


Abbildung 335: *ConSol CM Web Client - Vorschläge der Adress-Vervollständigungsfunktion, Beispiel 3*

Beachten Sie, dass es sich bei den angezeigten Werten lediglich um Vorschläge handelt. Der Bearbeiter kann die vorgeschlagenen Werte immer durch eine manuelle Eingabe in das Feld überschreiben.

Natürlich funktioniert parallel zu dieser Funktion auch die Standardfunktion von ConSol CM *Autocomplete-Suche*, die gleichzeitig Vorschläge für Kunden anzeigt, die schon in der CM-Datenbank gespeichert sind.



Abbildung 336: *ConSol CM Web Client - Vorschläge der Autocomplete-Suche in Kombination mit Adress-Vervollständigung*

Um das ConSol CM-System für diese Funktion zu konfigurieren, müssen Sie folgende Schritte durchführen:

1. Schalten Sie die Funktion *Adress-Vervollständigung* im Admin Tool ein, siehe Abschnitt Einschalten der Adress-Vervollständigung mit dem Admin Tool.
2. Importieren Sie die PLZ-/Ort-/Adressdaten in die CM-Datenbank. Siehe Abschnitt Importieren von PLZ-/Ort-/Adressdaten in die CM-Datenbank.
3. Definieren Sie die Adress-Vervollständigungsstrategie im ConSol CM Admin Tool, indem Sie eine oder mehrere Adress-Vervollständigungskonfigurationen anlegen, siehe Abschnitt Definieren der Adress-Vervollständigungskonfiguration mit dem Admin Tool.
4. Aktualisieren Sie den Index, siehe Abschnitt Aktualisieren des Index.

D.10.2 Einschalten der *Adress-Vervollständigung* mit dem Admin Tool

Geben Sie die System-Property cmas-app-admin-tool, autocomplete.enabled ein und setzen Sie ihren Wert auf „true“. Diese System-Property ist in der Standardinstallation von ConSol CM nicht vorhanden und muss manuell hinzugefügt werden.

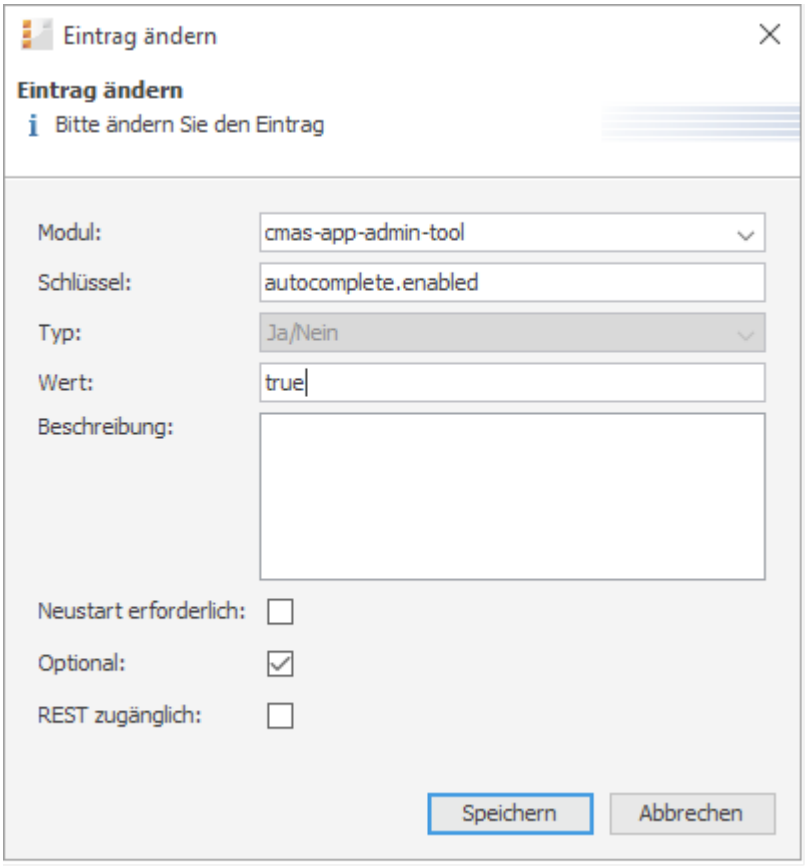


Abbildung 337: *ConSol CM Admin Tool - System, System-Properties: System-Property für die Funktion Adress-Vervollständigung*

Jetzt sehen Sie das neue Navigationselement *Adress-Vervollständigung* in der Navigationsgruppe *Kunden*. Im Abschnitt Definieren der Adress-Vervollständigungskonfiguration mit dem Admin Tool erfahren Sie, wie Sie die Adress-Vervollständigungsstrategie konfigurieren. Die Definition der Strategie ist aber erst nach dem Import der Quelldaten sinnvoll. Im nächsten Abschnitt erfahren Sie, wie dieser funktioniert.

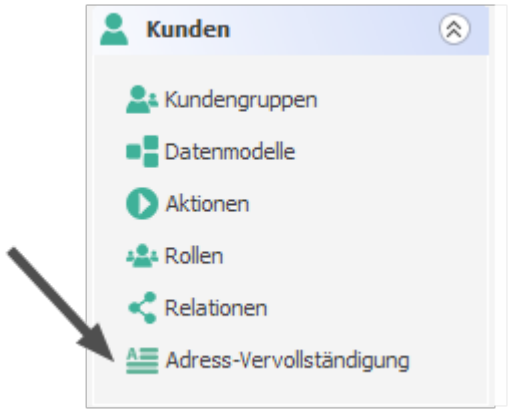


Abbildung 338: *ConSol CM Admin Tool - Navigationselement Adress-Vervollständigung in der Navigationsgruppe Kunden*

D.10.3 Importieren von PLZ-/Ort-/Adressdaten in die CM-Datenbank

Importieren Sie die Daten in die Tabelle `cmas_autocomplete_address` Ihrer ConSol CM-Datenbank.

Dieser Import muss von jemandem durchgeführt werden, der weiß, wie man Daten korrekt in eine relationale Datenbank einfügt! Weder der Import noch die Importskripte sind Teil der ConSol CM-Distribution oder Standardwartung. Sie können das Importskript mit einem Tool Ihrer Wahl implementieren. Falls Sie Unterstützung benötigen, wenden Sie sich bitte an Ihren ConSol CM-Consultant.

Der Import muss drei Felder für jeden Datensatz umfassen (siehe Abbildung unten)

- city
- street
- zip

```
mysql> show fields from cmas_autocomplete_address;
+--------+--------------+------+-----+---------+----------------+
| Field  | Type         | Null | Key | Default | Extra          |
+--------+--------------+------+-----+---------+----------------+
| id     | bigint(20)   | NO   | PRI | NULL    | auto_increment |
| city   | varchar(255) | YES  | MUL | NULL    |                |
| street | varchar(255) | YES  | MUL | NULL    |                |
| zip    | varchar(32)  | YES  | MUL | NULL    |                |
+--------+--------------+------+-----+---------+----------------+
```

Abbildung 339: *ConSol CM Datenbank - Tabelle cmas_autocomplete_address*

D.10.4 Definieren der Adress-Vervollständigungskonfiguration mit dem Admin Tool

Um die Adress-Vervollständigungsstrategie zu definieren, müssen Sie die folgenden Schritte ausführen:

1. Erstellen Sie eine oder mehrere Adress-Vervollständigungskonfigurationen.
2. Konfigurieren Sie das Verhalten für die einzelnen Adress-Vervollständigungskonfigurationen.

D.10.4.1 Erstellen von einer oder mehreren Adress-Vervollständigungskonfigurationen

Die Adress-Vervollständigungskonfiguration ist ein Mapping von einem Kundenfeld zu einem der Schlüsselfelder in der Tabelle `cmas_autocomplete_address`, z. B. können Sie folgende Definition umsetzen: "Wenn der Bearbeiter mit der Eingabe ins Feld *ResellerCompanyData*:*zip* beginnt, soll das System in *ZIP* suchen, und wenn der Bearbeiter mit der Eingabe im Kundenfeld *ResellerCompanyData*:*city* beginnt, soll das System im Feld city suchen".

Sie müssen die einzelnen Adress-Vervollständigungskonfigurationen im Admin Tool im Navigationselement *Adress-Vervollständigung* in der Navigationsgruppe *Kunden* definieren. Fügen Sie eine neue Adress-Vervollständigungskonfiguration ein, indem Sie auf den Button *Hinzufügen* klicken. Geben Sie den Namen der neuen Konfiguration ein und klicken Sie auf *Speichern*.

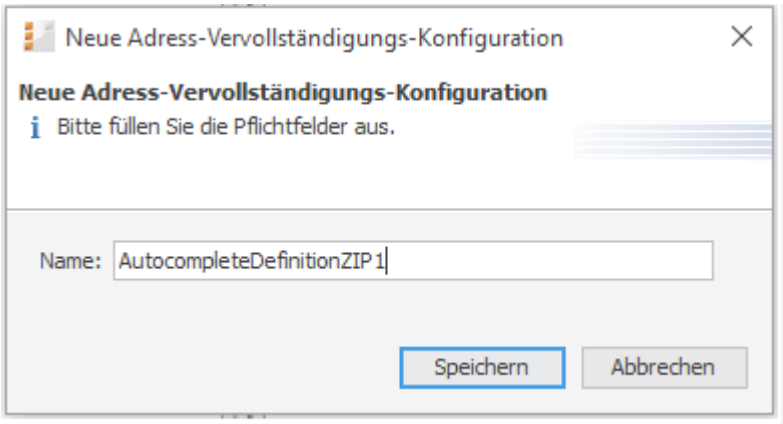


Abbildung 340: *ConSol CM Admin Tool - Kunden, Adress-Vervollständigung: Hinzufügen einer neuen Adress-Vervollständigungskonfiguration*

D.10.4.2 Konfigurieren des Verhaltens für die einzelnen Adress-Vervollständigungskonfigurationen

Um eine Adress-Vervollständigungskonfiguration zu konfigurieren, markieren Sie die Konfiguration in der Liste und fügen Sie ein oder mehrere Adress-Vervollständigungsfelder hinzu. Dies sind die Mapping-Regeln von den Kundenfeldern zu den Schlüsselfeldern in der Tabelle `cmas_autocomplete_address`.

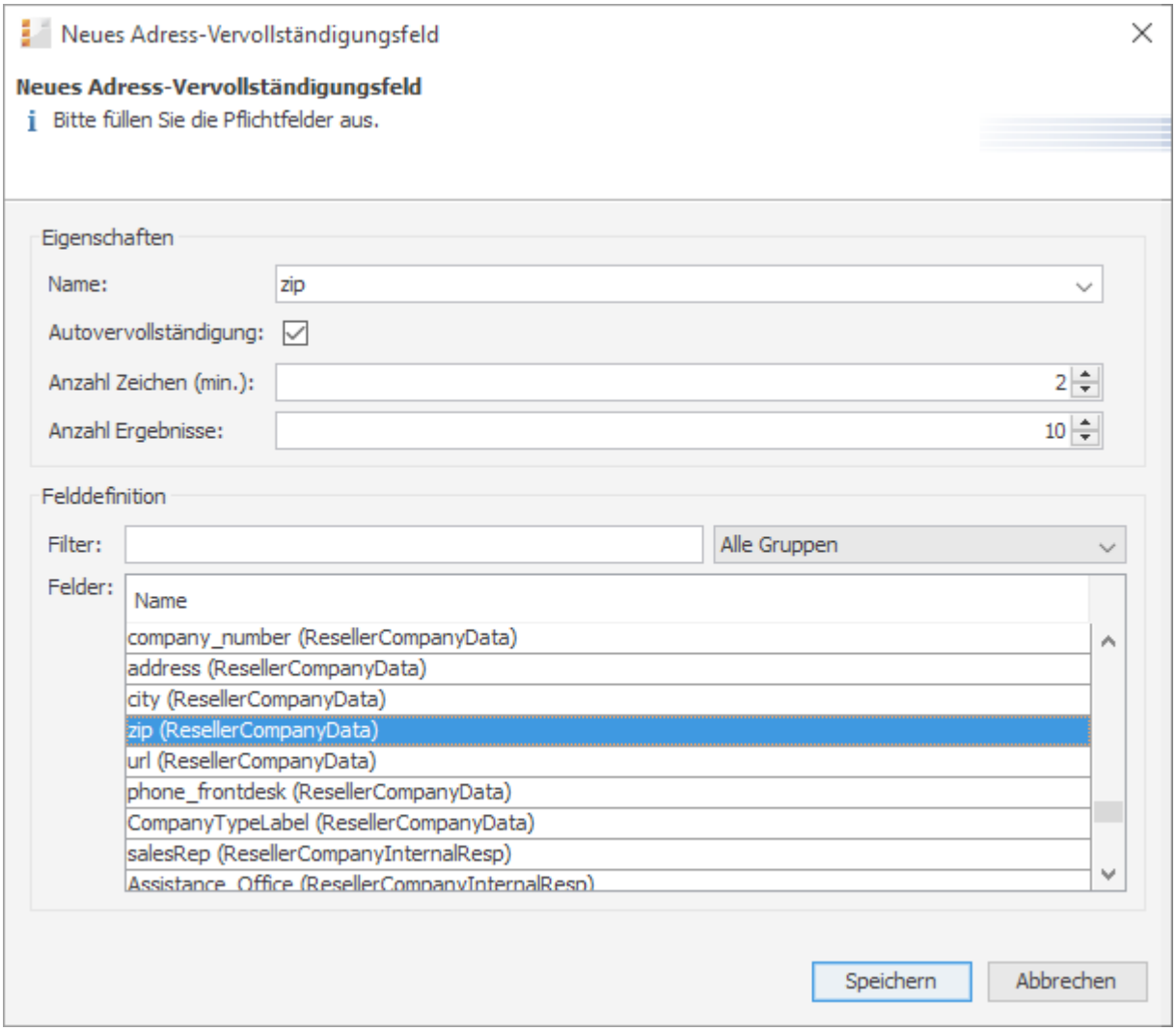


Abbildung 341: *ConSol CM Admin Tool - Kunden, Adress-Vervollständigung: Definition eines Adress-Vervollständigungsfeldes*

Für jedes Adress-Vervollständigungsfeld müssen folgende Felder definiert werden:

- **Name:**
  Wählen Sie *city*, *street* oder *zip*. Hier definieren Sie, welches Schlüsselfeld aus der Tabelle `cmas_autocomplete_address` verwendet wird.
- **Adress-Vervollständigung:**
  - **Wenn aktiviert:**
    Das Feld selber ist ein Autovervollständigen-Feld ohne Abhängigkeiten von anderen Feldern.
  - **Wenn deaktiviert:**
    Das Feld selber ist kein Autovervollständigen-Feld, wird aber durch Abhängigkeiten von

anderen Feldern gefüllt, z. B. ein Feld *city*, bei dem *Autocomplete* deaktiviert ist, wird automatisch gefüllt, wenn das Feld *zip* oder *address* gefüllt werden, reagiert selber aber nicht auf die Eingabe des Bearbeiters.

- **Anzahl Zeichen (min.):**
  Hier definieren Sie die Anzahl der Zeichen, die der Bearbeiter ins Kundenfeld eingeben muss, damit die Werterkennung und Autovervollständigung startet. Wenn das System die Liste anzeigen soll, sobald der Bearbeiter im Web Client den Cursor im Kundenfeld platziert, müssen Sie das Konfigurationsfeld an dieser Stelle leer lassen.
- **Anzahl Ergebnisse:**
  Hier definieren Sie die Anzahl der Vorschläge (in der Drop-down-Liste).
- **Felder:**
  Hier wählen Sie das Kundenfeld aus, das auf die Autovervollständigen-Eingabe reagieren soll. Es kann nur ein Feld ausgewählt werden.

Im folgenden Beispiel wurden zwei Adress-Vervollständigungsfelder für die Adress-Vervollständigungskonfiguration *AutocompleteDefinitionZIP1* definiert, siehe folgende Abbildung.

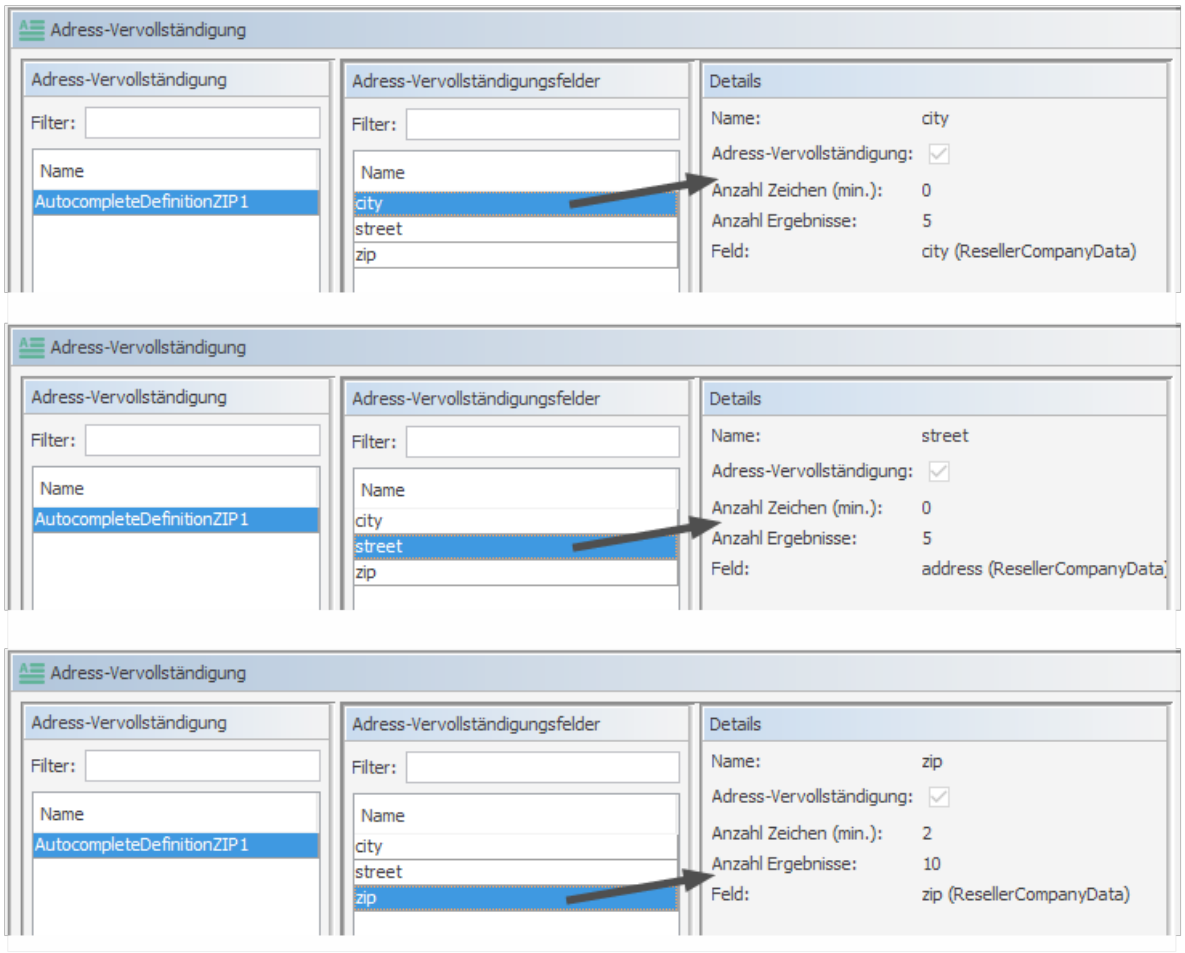


Abbildung 342: *ConSol CM Admin Tool - Kunden, Adress-Vervollständigung: Vollständige Adress-Vervollständigungskonfiguration*

D.10.4.3 Aktualisieren des Index

Nachdem Sie alle erforderlichen Adress-Vervollständigungskonfigurationen konfiguriert haben, müssen Sie den Index aktualisieren. Details über den Index finden Sie im Abschnitt Verwaltung von Indexer und Index mit dem Admin Tool.

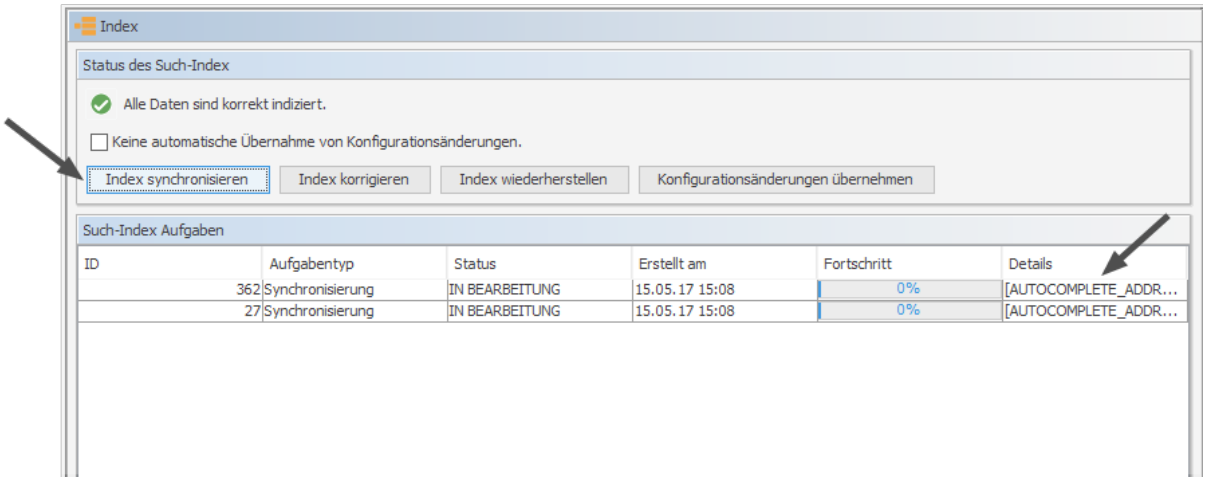


Abbildung 343: *ConSol CM Admin Tool - Dienste, Index: Indexaktualisierung*

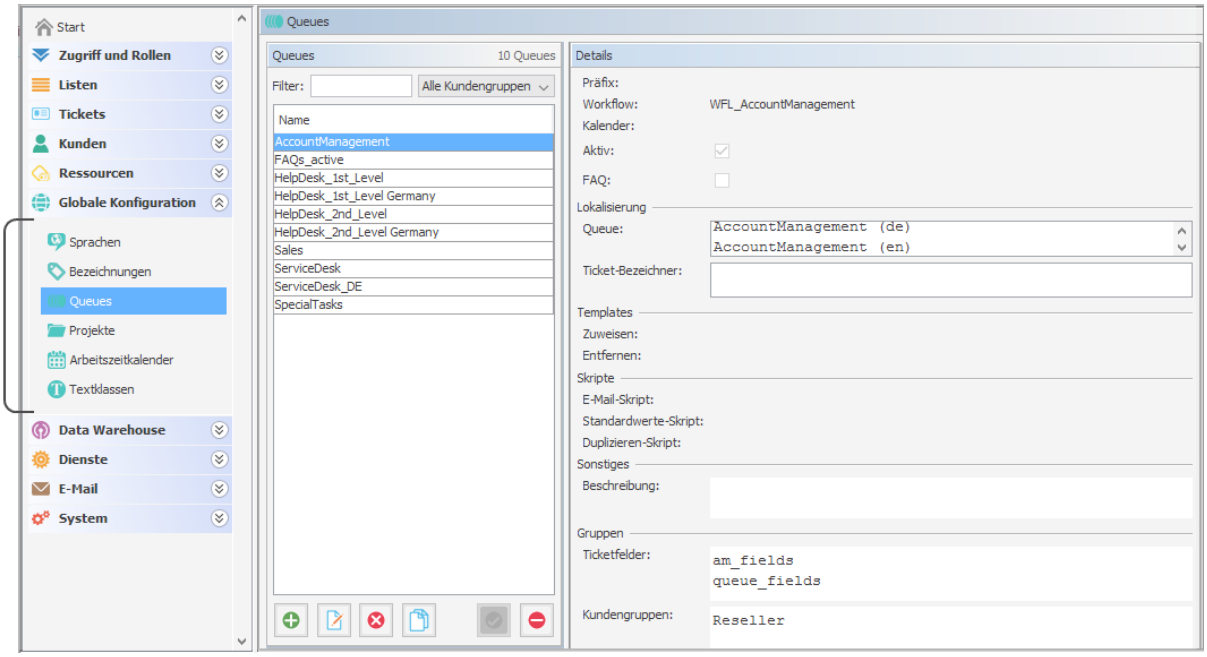
D.10.5 Editieren einer Adress-Vervollständigungskonfiguration

Um eine vorhandene Adress-Vervollständigungskonfiguration zu editieren, klicken Sie auf den Button *Bearbeiten*. Sie können die Adress-Vervollständigungskonfiguration und/oder die Adress-Vervollständigungsfelder editieren.

D.10.6 Löschen einer Adress-Vervollständigungskonfiguration oder von Adress-Vervollständigungsfeldern.

Um eine Adress-Vervollständigungskonfiguration oder Adress-Vervollständigungsfelder zu löschen, markieren Sie die Konfiguration oder das Feld in der entsprechenden Liste und klicken Sie auf den Button *Löschen*.

E - Abschnitt Globale Konfiguration



In diesem Abschnitt erfahren Sie, wie einige globale Einstellungen im ConSol CM-System konfiguriert werden.

- Sprachen
- Bezeichnungen
- Queue-Verwaltung
- Projekte
- Arbeiten mit Kalendern
- Arbeitszeitkalender
- Integration von Microsoft Exchange-Kalendern
- Textklassen

E.1 Sprachen

In diesem Kapitel werden folgende Themen behandelt:

E.1.1 Sprachen 511

E.1.2 Verwendung von Sprachumgebungen 513

E.1.1 Sprachen

ConSol CM kann so konfiguriert werden, dass die GUI des Web Clients in einer oder mehreren aus einer Vielzahl von Sprachen lokalisiert werden kann. Dies wird im Admin Tool über das Navigationselement *Sprachen* in der Navigationsgruppe *Globale Konfiguration* konfiguriert. Die dort vorgenommenen Einstellungen wirken sich auch auf die im Process Designer angebotenen Sprachen aus.

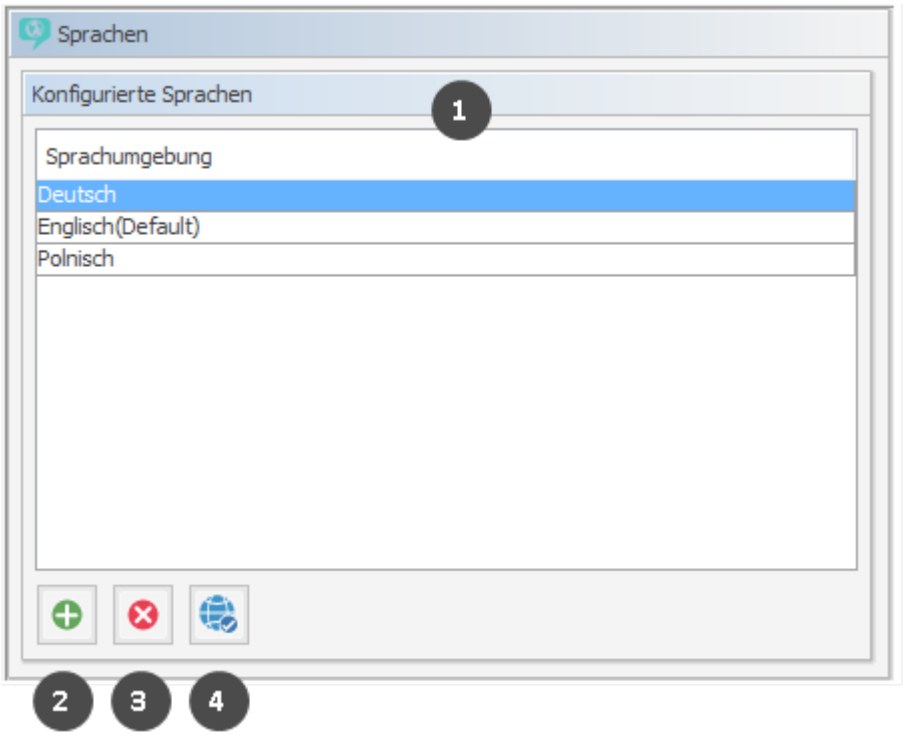


Abbildung 344: *ConSol CM Admin Tool - Globale Konfiguration, Sprachen*

Konfigurierte Sprachen:
In der Liste werden die Sprachumgebungen (1) konfiguriert, die im gesamten System zur Verfügung stehen. Dies betrifft die Liste der lokalisierten Werte im Process Designer (z. B. für Aktivitäten) und im Admin Tool (z. B. Ticketfelder). Die für diese Aktivitäten oder Felder im Web Client angezeigten Werte hängen dann von der Sprache ab, die der Benutzer in dem Browser, mit dem er den ConSol CM Web Client aufruft, eingestellt hat.

- Klicken Sie auf den Button *Hinzufügen*, um weitere Sprachumgebungen hinzuzufügen (2).
- Klicken Sie auf den Button *Löschen*, um die ausgewählte Sprachumgebung aus der Liste zu entfernen (3).
- Klicken Sie auf *Lokalisieren*, um die ausgewählte Sprachumgebung als Standardsprachumgebung festzulegen. Die Standardsprachumgebung wird verwendet, wenn die im Browser eingestellte Sprache in ConSol CM nicht verfügbar ist. Wenn der Bearbeiter zum Beispiel im Browser "FR" eingestellt hat und in der ConSol CM-Verwaltung nur Englisch (Standard), Deutsch und Polnisch verfügbar sind, werden als Standard die englischen Werte verwendet (4).

⚠ Stellen Sie sicher, dass die konfigurierten Sprachen auf allen Rechnern installiert sind, auf denen ConSol CM läuft oder verwendet wird. Dies wird nicht automatisch überprüft.

E.1.2 Verwendung von Sprachumgebungen

Der Bearbeiter, der mit dem ConSol CM Web Client arbeitet, sieht die Benutzeroberfläche in der Sprache, die er in seinem Browser eingestellt hat, sofern die entsprechende Sprache in ConSol CM konfiguriert ist. Wenn keine passende CM-Sprachumgebung gefunden wird, wird die im Admin Tool definierte Standardsprachumgebung verwendet.

Abhängig von der Stelle im Admin Tool ist der Mechanismus zum Eingeben von lokalisierten Begriffen leicht unterschiedlich. Eine Erklärung dazu finden Sie im Abschnitt Die unterschiedlichen Modi zum Lokalisieren von Begriffen und Bezeichnungen im Admin Tool.

Im Process Designer sind die Sprachumgebungen verfügbar, die im Admin Tool konfiguriert wurden. Sie können allerdings Sprachumgebungen im Process Designer löschen. Details finden Sie im *ConSol CM Process Designer Handbuch*.

E.2 Bezeichnungen

In diesem Kapitel werden folgende Themen behandelt:

E.2.1 Einleitung 514

E.2.2 Konfigurieren der Bezeichnungen mit dem Admin Tool 514

E.2.1 Einleitung

Es ist möglich, die auf der GUI, d. h. im Web Client angezeigten Bezeichnungen zu ändern. Auf diese Weise können Sie als Administrator die grafische Benutzeroberfläche an die Anforderungen Ihrer Firma hinsichtlich Terminologie und Corporate Design anpassen. Zum Beispiel können *Tickets* in Ihrer Firma *Fälle* genannt werden und die *Queue* kann *Prozess* heißen. Diese Änderungen können im Admin Tool über die Einstellungen der Bezeichnungen erfolgen.

E.2.2 Konfigurieren der Bezeichnungen mit dem Admin Tool

Bezeichnungen werden im Admin Tool über das Navigationselement *Bezeichnungen* in der Navigationsgruppe *Globale Konfiguration* konfiguriert.

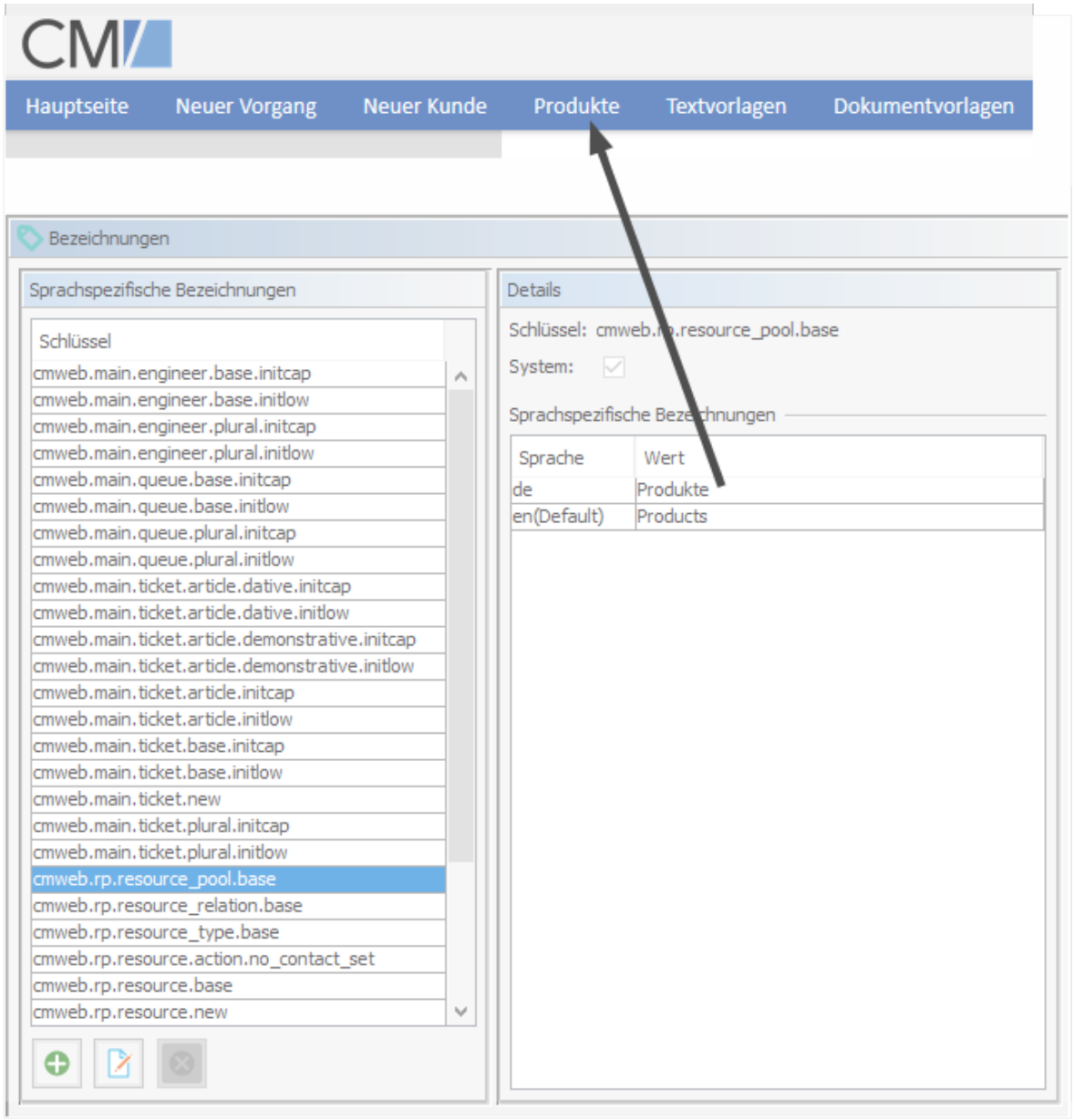


Abbildung 345: *ConSol CM Admin Tool - Globale Konfiguration, Bezeichnungen*

Aktuell sind die folgenden Bezeichnungen verfügbar:

Bezeichnung	Ort auf der GUI	Standardwert
cmweb.main.engineer.base.initcap	Begriff für einen Bearbeiter, Großbuchstabe am Wortanfang. Verwendet z. B. im Kopfbereich des Tickets.	Bearbeiter
cmweb.main.engineer.base.initlow	Begriff für einen Bearbeiter.	Bearbeiter
cmweb.main.engineer.plural.initcap	Begriff für mehr als einen Bearbeiter, Großbuchstabe am Wortanfang.	Bearbeiter
cmweb.main.engineer.plural.initlow	Begriff für mehr als einen Bearbeiter.	Bearbeiter
cmweb.main.queue.base.initcap	Begriff für eine Queue, Großbuchstabe am Wortanfang. Verwendet z. B. im Filter der Ticketliste.	Queue
cmweb.main.queue.base.initlow	Begriff für eine Queue.	Queue
cmweb.main.queue.plural.initcap	Begriff für mehr als eine Queue, Großbuchstabe am Wortanfang.	Queues
cmweb.main.queue.plural.initlow	Begriff für mehr als eine Queue.	Queues
cmweb.main.ticket.article.dative.initcap	Demonstrativpronomen im Dativ für ein Ticket, Großbuchstabe am Wortanfang	Diesem

Bezeichnung	Ort auf der GUI	Standardwert
cmweb.main.ticket.article.dative.initlow	Demonstrativpronomen im Dativ für ein Ticket	diesem
cmweb.main.ticket.article.demonstrative.initcap	Demonstrativpronomen im Nominativ für ein Ticket, Großbuchstabe am Wortanfang	Dieser
cmweb.main.ticket.article.demonstrative.initlow	Demonstrativpronomen im Nominativ für ein Ticket	dieser
cmweb.main.ticket.article.initcap	Artikel im Nominativ für ein Ticket, Großbuchstabe am Wortanfang	Der
cmweb.main.ticket.article.initlow	Artikel im Nominativ für ein Ticket	der
cmweb.main.ticket.base.initcap	Begriff für ein Ticket, Großbuchstabe am Wortanfang	Vorgang
cmweb.main.ticket.base.initlow	Begriff für ein Ticket	Vorgang
cmweb.main.ticket.new	Eintrag im Hauptmenü zum Erstellen eines neuen Tickets	Neuer Vorgang
cmweb.main.ticket.plural.initcap	Begriff für mehr als ein Ticket, Großbuchstabe am Wortanfang	Vorgänge
cmweb.main.ticket.plural.initlow	Begriff für mehr als ein Ticket	Vorgänge

Bezeichnung	Ort auf der GUI	Standardwert
cmweb.rp.resource.base	Titel der Ressourcenseite und der Ressourcenliste auf der Ressourcentypseite.	Ressource
cmweb.rp.resource.new	Eintrag zum Erstellen einer neuen Ressource	Neue Ressource
cmweb.rp.resource.plural	Ausdruck für mehr als eine Ressource	Ressourcen
cmweb.rp.resource.to	Eintrag im Kontextmenü zum Aufrufen einer Ressource	Zur Ressource
cmweb.rp.resource_pool.base	Eintrag im Hauptmenü	Ressourcenpool
cmweb.rp.resource_relation.base	Ausdruck für die Ressourcenrelation	Relation
cmweb.rp.resource_type.base	Ausdruck für den Ressourcentyp, z. B. auf der Seite der Detailsuche für Ressourcen	Ressourcentyp

E.2.2.1 Für CM-Programmierer: Bezeichnungen für Nachrichten

Sie können auch selbst Bezeichnungen definieren, die in bestimmten Situationen (Fehlermeldungen, Informationen) für den Benutzer im Web Client angezeigt werden sollen. So müssen Sie nicht mit den Standard CM-Properties arbeiten, sondern können sehr spezifische Bezeichnungen definieren. Die folgenden drei Abbildungen zeigen ein Beispiel.

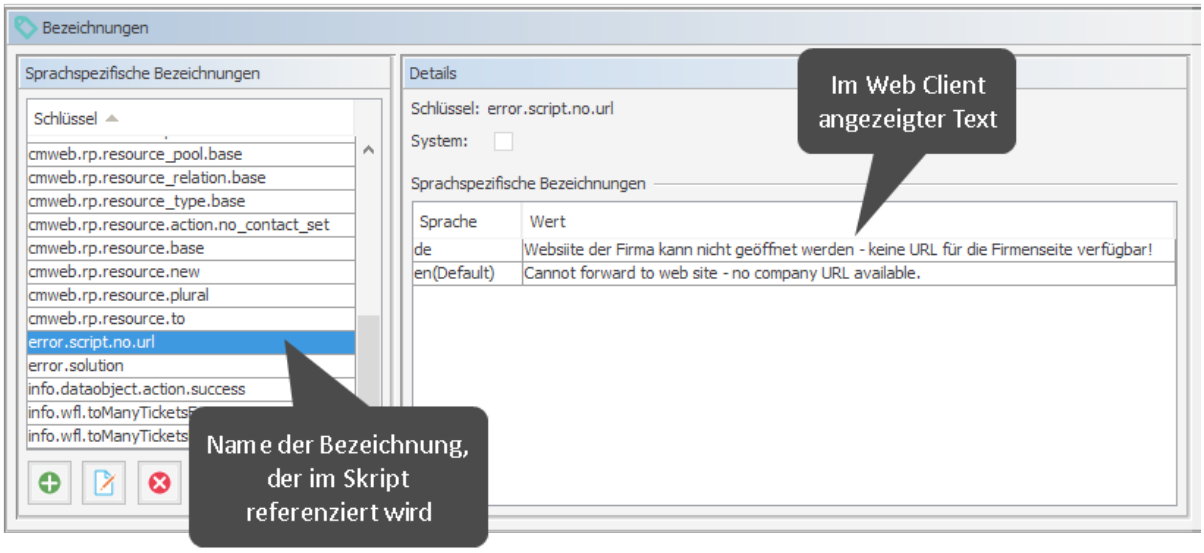


Abbildung 346: *ConSol CM Admin Tool - Globale Konfiguration, Bezeichnungen: Systemspezifische Bezeichnung für eine Fehlermeldung 1: Definition der Bezeichnung*

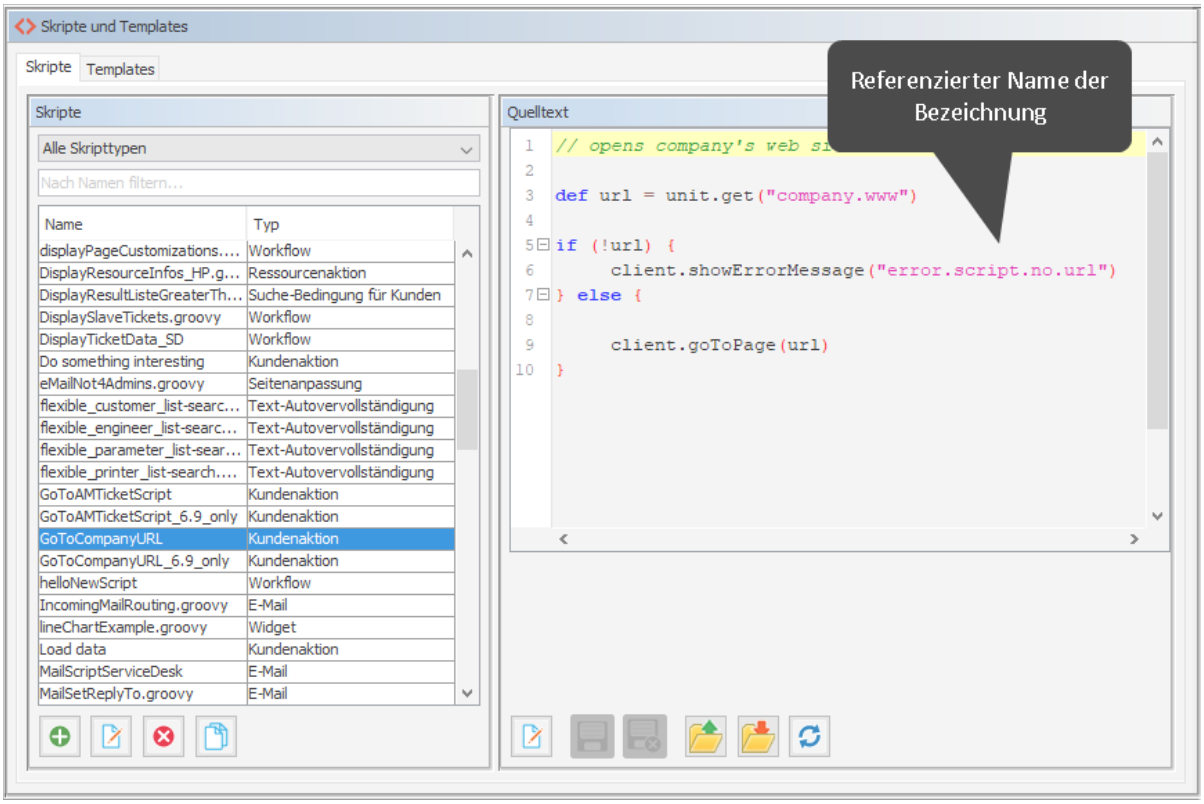


Abbildung 347: *ConSol CM Admin Tool - Skripte, Skripte und Templates: Systemspezifische Bezeichnungen, 2: Verwendung der Bezeichnung in einem Skript*



Abbildung 348: *ConSol CM Web Client - Skripte, Skripte und Templates: Systemspezifische Bezeichnungen, 3: Anzeige im Web Client*

Sie können auch mit `MessageProviderService` arbeiten. Diese Klasse bietet eine sehr generische Möglichkeit, Bezeichnungen zu verwenden, nicht nur für Meldungen, sondern auch für andere GUI-Elemente wie Namen in Charts oder anderen Objekten. Wie Sie es von anderen Services kennen, ist ein Singleton der Klasse `MessageproviderService` potenziell als Objekt `messageProviderService` in allen Skripten verfügbar.

Sie können die Methode `getMessage(String pKey)` verwenden. Diese benötigt als Parameter den Schlüssel einer Bezeichnung. Als Ergebnis wird der lokalisierte Wert der Bezeichnung in der Standardsprache des Systems angezeigt.

Beispiel aus einem Widget-Skript:

```
 //Some messages used by this widget
title = messageProviderService.getMessage
 ("web.chart.company.ticket.creators.title")
titleNoTicketsYet = messageProviderService.getMessage
 ("web.chart.company.ticket.creators.noResults")
seriesName = messageProviderService.getMessage
 ("web.chart.company.ticket.creators.series")
```

Wenn Sie die Meldung in einer bestimmten Sprache anzeigen möchten, können Sie mit der anderen Signatur der Methode `getMessage` arbeiten:

`getMessage(String pKey, Locale pLocale)`. Diese kann auch für andere Sprachen als die Standardsprache verwendet werden.

Beispiel aus einem Workflow-Skript, das die Auslastung des Bearbeiters überprüft. Hier wird die Standardsprache des angemeldeten Bearbeiters verwendet.

```
// Engineer can only accept ticket if he does not have too many tickets already
def curr_eng = workflowApi.currentEngineer
def max_tics = configurationService.getValue("custom-
 servicedesk","engineer.max.open.tickets")

// look for open tickets of current engineer
def engs = []
engs.add(curr_eng.id)

TicketCriteria tic_crit = new TicketCriteria()
tic_crit.engineerCriteria = TicketCriteria.EngineerCriteria.assigned(engs as Set)
tic_crit.status = TicketCriteria.Status.OPEN

List<Ticket> open_eng_tics = ticketService.getByCriteria(tic_crit)

def tic_number = open_eng_tics.size
def loc = engineerService.currentLocale

if (tic_number > max_tics) {
   log.info 'Too many tickets for engineer ' + engineerService.current + '. Current
    number is ' + tic_number
   // get text from labels defined in AT:

   def infoText1 = messageProviderService.getMessage
    ("info.wfl.toManyTicketsError1", loc)
   def infoText2 = messageProviderService.getMessage
    ("info.wfl.toManyTicketsError2", loc) + max_tics
   // alternative solution: workflowApi.addValidationError("INFO","You have too
    many tickets (" + tic_number + ") already, so you cannot accept another ticket.
    Maximum allowed number is " + max_tics)
   workflowApi.addValidationError("INFO",infoText1 + " " + infoText2)

} else {
   ticket.setEngineer(curr_eng)
}
workflowApi.reinitializeTrigger("defaultScope/Service_
 Desk/TimeTriggerDesiredDeadline")
```

Code-Beispiel 58: *Skript der Aktivität „Neues Ticket (Ticket annehmen)“, Auslastung des Bearbeiters wird geprüft*

In diesem Fall könnte eine Meldung wie die folgende angezeigt werden:

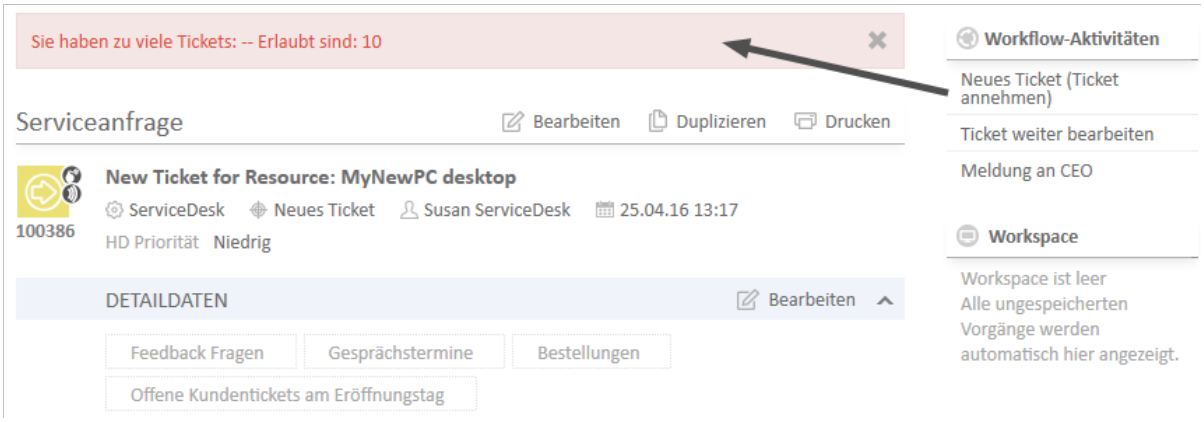


Abbildung 349: *ConSol CM Web Client - Meldung, wenn der Bearbeiter zu viele Tickets hat, um ein weiteres anzunehmen (Deutsch als Sprache im Browser eingestellt)*

E.3 Queue-Verwaltung

In diesem Kapitel werden folgende Themen behandelt:

E.3.1 Einleitung 523

E.3.2 Queue-Verwaltung mit dem Admin Tool 524

E.3.1 Einleitung

Queues sind zentrale Elemente von ConSol CM. Tickets werden in Queues gruppiert, z. B. für bestimmte Aufgaben oder Arbeitsgruppen. Jeder Queue wird ein einziger Workflow zugewiesen, mit dem die Bearbeitungsschritte aller Tickets in der Queue gesteuert werden. Zum Beispiel kann es eine Queue *ServiceDesk*, eine Queue *Marketing* und eine Queue *Vertrieb* geben.

Folgende Parameter und Objekte werden einer Queue zugewiesen (die Parameter und Objekte werden in anderen Tabs des Admin Tools **definiert** und hier nur einer neuen oder einer vorhandenen Queue **zugewiesen**):

- Der Workflow der Queue (obligatorisch), d. h. der Prozess, der für alle Tickets in der Queue eingesetzt werden soll (z. B. alle Tickets einer Abteilung). Eine Queue kann nur einen Workflow haben, aber ein Workflow kann in mehreren Queues verwendet werden.
- Das Template für E-Mails, die an die Bearbeiter gesendet werden, wenn ihnen ein Ticket zugewiesen oder entzogen wird (optional).
- Mehrere Skripte, die das Verhalten der Tickets in dieser Queue steuern (optional).
- Eine oder mehrere Kundengruppen, die mit der Queue verknüpft sind. Nur Tickets von Kunden dieser Kundengruppen können zur Queue hinzugefügt werden (eine Kundengruppe ist Pflicht, mehrere sind optional).
- Der Arbeitszeitkalender (d. h. die Arbeitszeiten), der für die Tickets in dieser Queue gelten soll (optional).
- Die Datenfelder (Ticketfelder), die für Tickets in dieser Queue verfügbar sein sollen. Diese werden definiert, indem der Queue Ticketfeldgruppen zugewiesen werden (einige sind obligatorisch, einige sind optional).
- Die Textklassen, die für Tickets in dieser Queue verfügbar sein sollen (optional).
- Die Projekte, die für die Zeitbuchung in Tickets dieser Queue verfügbar sein sollen (optional).



Als zentrales Element verwendet die Queue mehrere Objekte und Elemente, die an anderen Stellen definiert werden, d. h. auf einer anderen Seite des Admin Tools, weshalb man normalerweise zuerst die Elemente definiert, die für die Queue-Definition benötigt werden. Mit Ausnahme des Workflows können alle Parameter auch nach dem ersten Speichern der Queue-Definition noch geändert werden (d. h. nachdem die Queue erstellt wurde), sodass Sie die Queue bei Bedarf iterativ konfigurieren können.

Die Queue bildet zudem die Grundlage für die Zuweisung von Zugangsberechtigungen, siehe Abschnitt Rollenverwaltung.

E.3.2 Queue-Verwaltung mit dem Admin Tool

Im Admin Tool werden die Queues im Navigationselement *Queues* in der Navigationsgruppe *Globale Konfiguration* verwaltet.

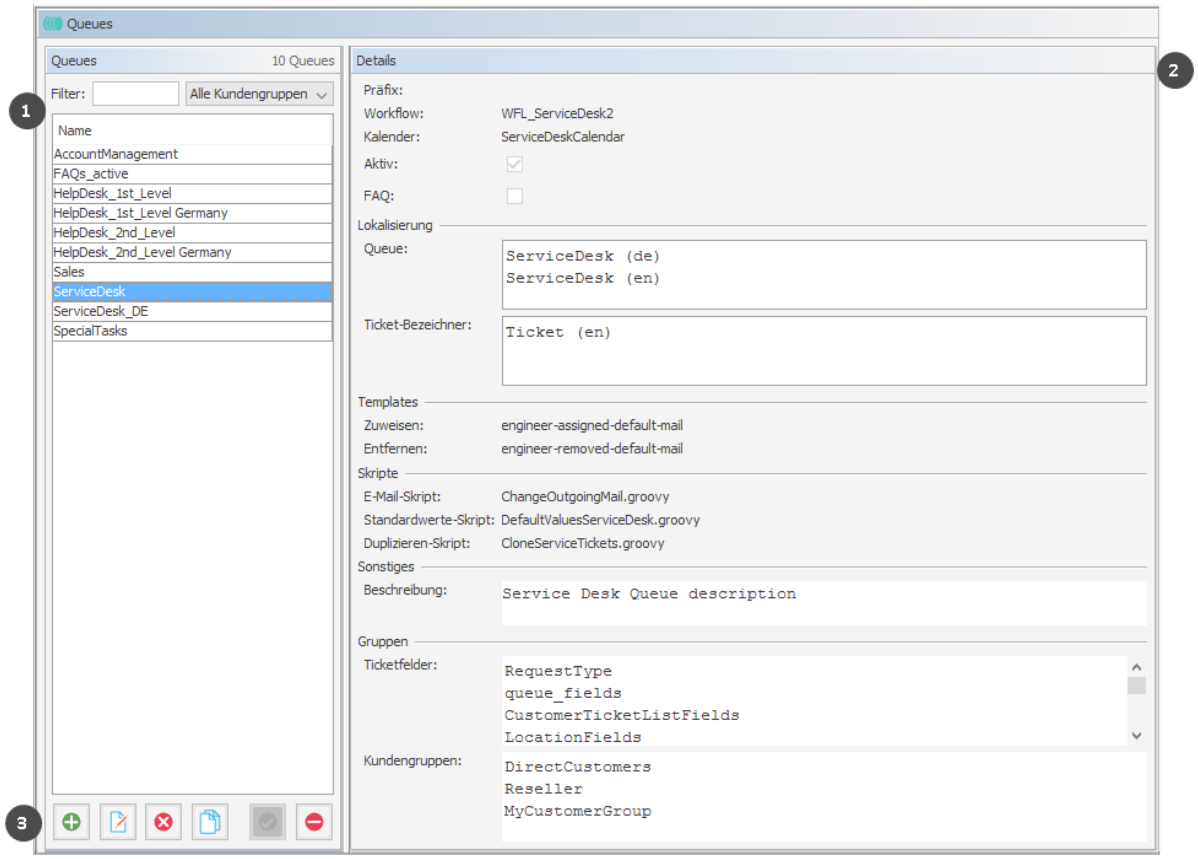


Abbildung 350: *ConSol CM Admin Tool - Globale Konfiguration, Queues: Queue-Verwaltung*

Das Navigationselement *Queues* enthält eine Liste aller Queues (1). Sie kann mit einem Textfilter und/oder einem Drop-down-Menü, in dem die Kundengruppe ausgewählt werden kann, gefiltert werden. Im Detailbereich (2) auf der linken Seite werden alle Queue-Details (schreibgeschützt) angezeigt.

Mit den entsprechenden Buttons in der Button-Leiste (3) unter der Queue-Liste können Sie eine Queue hinzufügen, bearbeiten, kopieren oder de(aktivieren).

E.3.2.1 Filtern der Queue-Liste

Sie finden die Queue, die Sie editieren oder kopieren möchten, am schnellsten, wenn Sie Filterinformationen in die Felder über der Queue-Liste eingeben.

Sie können nach Queues filtern, die

- eine bestimmte Zeichenfolge enthalten (Leerzeichen werden dabei berücksichtigt) und/oder
- bestimmten Kundengruppen zugewiesen sind.

E.3.2.2 Erstellen einer Queue

Sie können eine neue Queue erstellen, indem Sie unter der Queue-Liste auf den Button *Hinzufügen* klicken. Für das Erstellen und das Editieren einer Queue wird das gleiche Popup-Fenster geöffnet. In beiden Fällen müssen Sie die Details der Queue editieren.

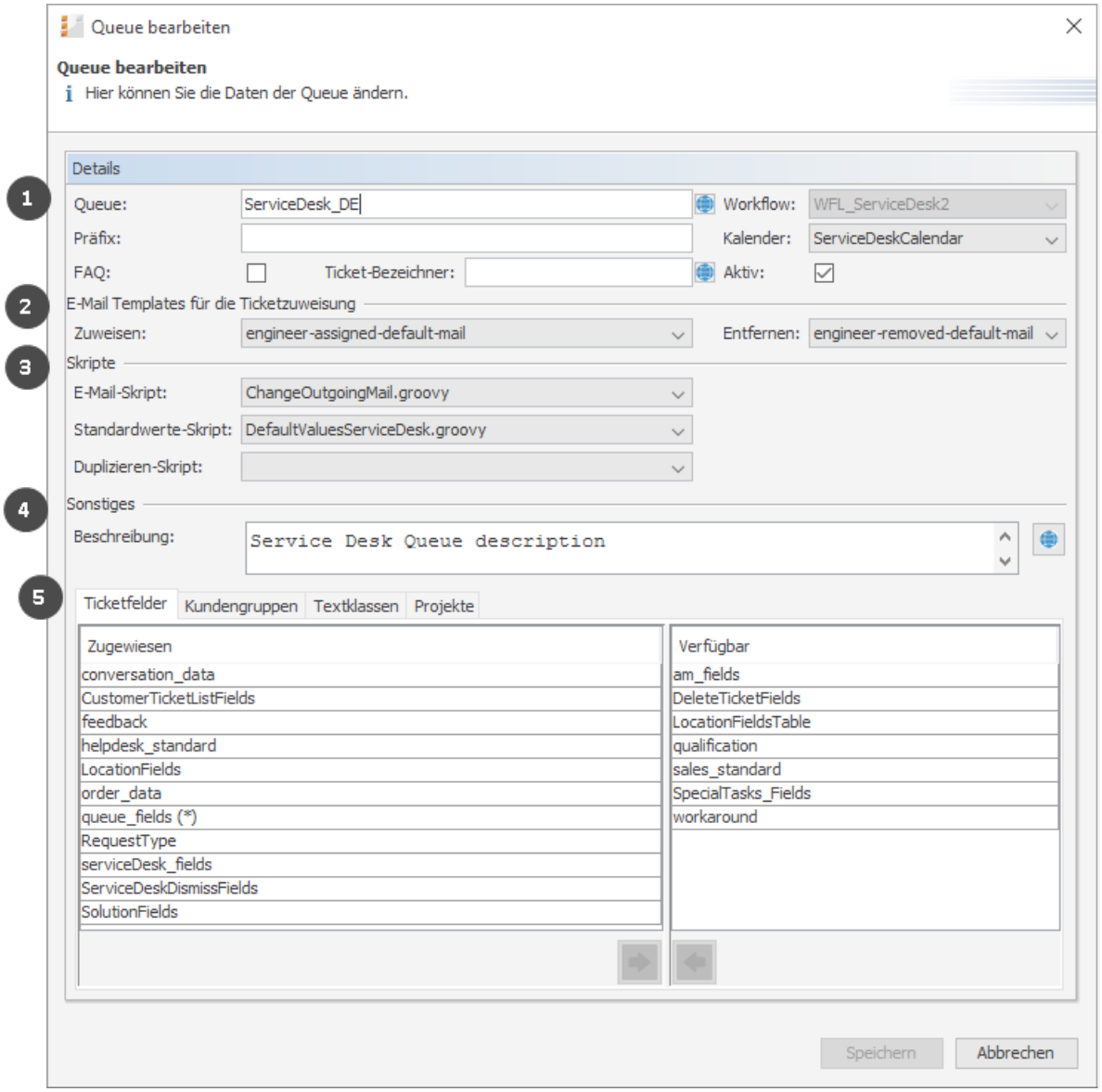


Abbildung 351: *ConSol CM Admin Tool - Globale Konfiguration, Queues: Erstellen einer Queue*

Es sind folgende Felder verfügbar:

- **Allgemeine Konfiguration (1):**
  - **Queue:**
    Geben Sie den technischen Namen der Queue in dieses Feld ein. Klicken Sie auf den Button *Lokalisieren*, um den lokalisierten Queue-Namen in allen im System verfügbaren

Sprachen einzugeben. Der lokalisierte Queue-Name wird (abhängig von der im Browser eingestellten Sprache) im Web Client im Kopfbereich der Tickets angezeigt. Wenn keine lokalisierten Werte angegeben sind, wird der technische Name angezeigt.

- **Workflow:**
  Wählen Sie den Workflow für die Queue aus dieser Liste.

  > Wenn Sie einen neuen Workflow entwickelt und installiert haben, ist dieser erst im Admin Tool verfügbar, nachdem Sie die Daten des Admin Tools aktualisiert haben.

  > Sobald Sie einen Workflow einer Queue zugewiesen haben, kann dieser nicht mehr geändert werden!

- **Präfix:**
  Sie können ein Präfix für die Ticket-IDs in der Queue eingeben, z. B. wenn die Ticket-ID zeigen soll, zu welcher Queue oder Organisationsstruktur das Ticket gehört.

  > Das Präfix bleibt mit dem Ticketnamen erhalten, wenn das Ticket in eine andere Queue verschoben wird.

- **Kalender:**
  Wählen Sie den Arbeitszeitkalender für die Queue aus der Liste. Arbeitszeitkalender in CM definieren die Arbeitszeiten, Feiertage und die gültige Zeitzone (siehe Abschnitt Arbeitszeitkalender). Sie werden für Zeit-Trigger im Workflow verwendet, und müssen für jeden Trigger explizit aktiviert werden, d. h. wenn Sie mit einer Zeitberechnung arbeiten möchten, die auf einem Arbeitszeitkalender basiert, müssen Sie dies an drei Stellen konfigurieren:
  - In der Navigationsgruppe *Globale Konfiguration*, Navigationselement Arbeitszeitkalender wird der Kalender erstellt, und die Arbeits- und Feiertagszeiten werden konfiguriert.
  - In der Queue-Verwaltung wird der Kalender der Queue zugewiesen.
  - Für jeden Zeit-Trigger im Workflow wird der queue-spezifische Kalender aktiviert oder nicht. Eine detaillierte Beschreibung der Arbeit mit Zeit-Triggern finden Sie im *ConSol CM Process Designer Handbuch*.

- **FAQ:**
  Wenn diese Checkbox markiert ist, dient die Queue als Wissensdatenbank für CM/Track-Benutzer. Diese können in dieser Queue über CM/Track, das ConSol CM-Webportal, nach Tickets suchen. Mehr zu diesem Thema erfahren Sie im Abschnitt CM/Track V1: FAQs in CM/Track bzw. CM/Track V2: FAQs in CM/Track.

- **Ticket-Bezeichner:**
  Optional. Der hier eingegebene String wird als Kopfzeile im Ticket angezeigt. Wenn ein lokalisierter Wert angegeben ist, wird dieser verwendet. Andernfalls wird der technische Name verwendet. Auf diese Weise können Sie Ihr CM-System an Anzeigenamen wie *Ticket*, *Anfrage* oder *Serviceanfrage* anpassen, und Sie sind nicht an den Begriff *Vorgang* gebunden. Wenn manuell kein anderer Begriff definiert wurde, wird im Web Client der Standardbegriff *Vorgang* verwendet. Beachten Sie, dass der Ticket-Bezeichner mit zwei Parametern manuell definiert werden kann:
  - Der Ticketbezeichner wird hier, in den Queue-Details, festgelegt. Diese Einstellung ist queue-spezifisch, d. h. es kann für jede Queue ein anderer Bezeichner definiert werden. Dies kann für die ServiceDesk-Queue zum Beispiel *Serviceanfrage* sein und in der Vertriebs-Queue *Opportunity*.
  - Der Wert wird mit der Bezeichnung `ticket.base.initcap` definiert. Diese wird nur verwendet, wenn kein queue-spezifischer Wert festgelegt ist. Details zu den Bezeichnungen finden Sie im Abschnitt über Bezeichnungen.

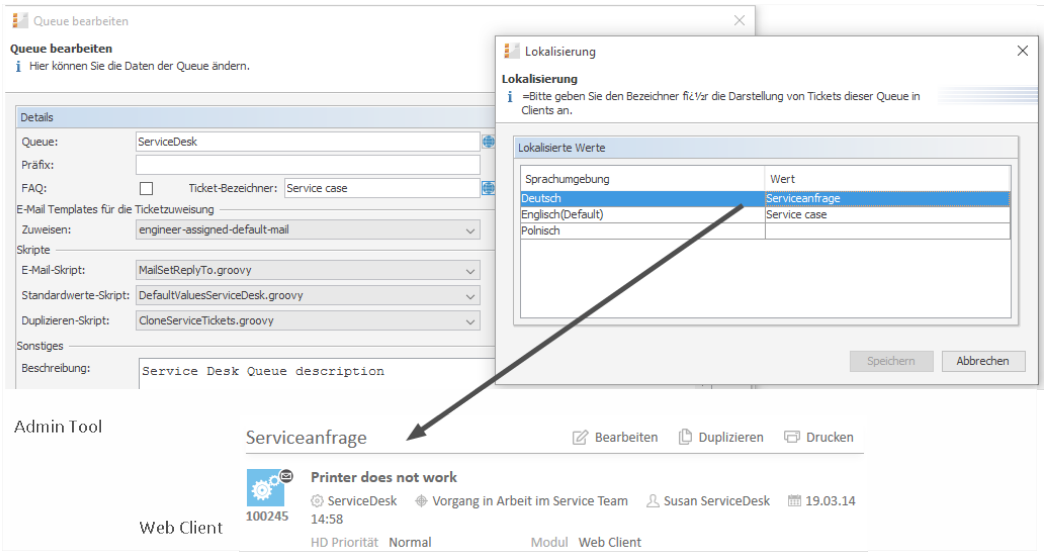


Abbildung 352: *ConSol CM Admin Tool und Web Client: Einstellen und Anzeigen einer queue-spezifischen Kopfzeile im Ticket*

- **Aktiv:**
  Wenn diese Checkbox markiert ist, ist die Queue sofort nach dem Speichern im System verfügbar. In aktivierten Queues können Sie Tickets erstellen. In deaktivierten Queues ist dies nicht möglich.

- **E-Mail-Templates für die Ticketzuweisung (2):**
  Hier können Sie E-Mail-Templates auswählen, die für automatische E-Mails verwendet werden, die an den betroffenen Bearbeiter gesendet werden, wenn ein Ticket einem (neuen) Bearbeiter zugewiesen wird (*Zuweisen*) oder wenn das Ticket einem Bearbeiter entzogen wird (*Entfernen*). In diesem Drop-down-Menü sind die Templates verfügbar, die Sie in der *Skript- und Template-Verwaltung* des Admin Tools (siehe Abschnitt Admin-Tool-Templates) definiert haben. Wenn

Sie nicht möchten, dass das ConSol CM-System beim Bearbeiterwechsel automatische E-Mails verschickt, lassen Sie dieses Feld leer. Denken Sie daran, dass die System-Properties cmas-core-server, mail.notification.engineerChange (= „true“) und cmas-core-server, mail.notification.sender gesetzt sein müssen (siehe Abschnitte System-Properties und Queue-spezifische E-Mails beim Zuweisen/Entfernen eines Bearbeiters).

- **Skripte (3):**
  Skripte werden zur Automatisierung von wiederkehrenden Aufgaben und Aktivitäten verwendet. Sie werden in der *Skript- und Template-Verwaltung* (siehe Abschnitt Admin-Tool-Skripte) verwaltet. Sie können folgende Skripte zuweisen:
  - **E-Mail-Skript**
    Wählen Sie ein Skript aus der Liste, wenn die ausgehenden E-Mails für diese Queue durch das Skript verändert werden sollen, z. B. damit sie für Felder wie Absender oder Adresse Standardwerte enthalten. Das hier angegebene E-Mail-Skript ist das letzte Skript, das eine ausgehende E-Mail verarbeitet, d. h. alle vorher vorgenommenen Einstellungen werden überschrieben, wenn eine Variable zuvor gesetzt wurde. Es sind alle im Skriptabschnitt gespeicherten Skripte des Typs *E-Mail* verfügbar, stellen Sie also sicher, dass Sie das richtige Skript auswählen

    ⚠ Beachten Sie das folgende technische Verhalten von ConSol CM bei der Konfiguration von Reply-To-Adressen und passen Sie Ihr System entsprechend an!

    Der technische Hintergrund:

    Es gibt vier mögliche Reply-To-Adressen, mit denen Sie zu tun haben:

    1. Die **Reply-To-Adresse, die in der System-Property `mail.reply.to` festgelegt ist.** Wenn sie gesetzt ist, wird sie im Ticket-E-Mail-Editor im Web Client angezeigt. Ob es tatsächlich die Reply-To-Adresse ist, die in einer E-Mail angewendet wird, hängt von der Konfiguration des queue-spezifischen Skripts für ausgehende E-Mails ab. Siehe nächster Punkt. Wenn das Attribut der Seitenanpassung *showReplyTo* für den Typ *mailTemplate* auf „false“ gesetzt ist, wird im Ticket-E-Mail-Editor keine Reply-To-Adresse angezeigt. Wenn die Property `mail.reply.to` gesetzt ist, wird diese Adresse trotzdem verwendet, außer es wird eine andere Adresse in einem Skript für ausgehende E-Mails gesetzt, siehe nächster Punkt.
    2. Die **Reply-To-Adresse, die in einem queue-spezifischen Skript für ausgehende E-Mails gesetzt wird**. Da das Skript für ausgehende E-Mails die letzte Instanz ist, die eine ausgehende E-Mail verarbeitet, ist die in diesem Skript gesetzte E-Mail-Adresse immer die angewendete Reply-To-Adresse. Falls die Property `mail.reply.to` gesetzt ist, wird diese mail-reply.to-address nicht verwendet (aber trotzdem im Ticket-E-Mail-Editor angezeigt, was verwirrend sein kann. Im nächsten Abschnitt ist erklärt, was dies für die Systemkonfiguration bedeutet).



3. Die **E-Mail-Adresse, die in der System-Property `mail.from` gesetzt ist.** Wenn diese gesetzt ist und weder `mail.reply.to` noch eine queue-spezifische Reply-To-Adresse gesetzt ist, setzen die meisten E-Mail-Clients die From-Adresse als Reply-To-Adresse.
4. Die **E-Mail-Adresse des aktuellen Bearbeiters** (der Bearbeiter, der im Web Client angemeldet ist). Diese persönliche E-Mail-Adresse wird als Reply-To-Adresse für E-Mails aus dem Web Client verwendet, wenn weder die Property `mail.reply.to` gesetzt ist, noch ein queue-spezifisches Skript für ausgehende E-Mails konfiguriert ist, noch die Property `mail.from` gesetzt ist.

Im Web Client wird im Ticketprotokoll immer die Reply-To-Adresse angezeigt, die für eine ausgehende E-Mail tatsächlich verwendet wurde. So wird immer die angewendete E-Mail-Adresse angezeigt, auch wenn die im Ticket-E-Mail-Editor angezeigte Adresse (die Property `mail.reply.to`) und die tatsächlich verwendete Reply-To-Adresse (die Reply-To-Adresse im queue-spezifischen Skript für ausgehende E-Mails) unterschiedlich sind. Die tatsächliche E-Mail-Adresse wäre in diesem Fall die Adresse aus dem Skript.

Unsere Empfehlung:

Es sollte immer eine Reply-To-Adresse für das System gesetzt werden! Sie können entscheiden, ob Sie

- mit der Reply-To-Adresse arbeiten, die in einem queue-spezifischen Skript für ausgehende E-Mails gesetzt wird

oder

- die System-Property `mail.reply.to` verwenden.

Da die E-Mail-Kommunikation immer über ConSol CM stattfinden soll und nicht über persönliche E-Mail-Adressen, sollte eine der beiden oben beschriebenen Systemeinstellungen verwendet werden, um zu verhindern, dass ConSol CM persönliche E-Mail-Adressen als Reply-To verwendet. Letzteres würde bedeuten, dass die E-Mails der Kunden an das persönliche E-Mail-Postfach des Bearbeiters gesendet werden statt an CM.

Auswirkungen auf die Systemkonfiguration:

1. Der einfachste Weg, um eine Reply-To-Adresse zu setzen, ist die Verwendung der System-Property `mail.reply.to`. Sie wird im Ticket-E-Mail-Editor angezeigt und ist die tatsächlich verwendete Reply-To-Adresse.
2. Wenn queue-spezifische Reply-To-Adressen erforderlich sind, empfehlen wir, ein Skript für ausgehende E-Mails zu schreiben, in dem die Queue-Namen den entsprechenden Reply-To-Adressen zugeordnet werden.



Dieses Skript kann dann für Bcc, Cc oder andere Adressen ergänzt werden.
Sie können die Property `mail.reply.to` und die queue-spezifischen Reply-To-Adressen auch kombinieren: Für alle Queues ohne ein spezielles Skript für ausgehende E-Mails wird die Adresse aus `mail.reply.to` verwendet; für alle Queues mit einem queue-spezifischen Skript für ausgehende E-Mails, das eine Reply-To-Adresse enthält, wird diese verwendet.

Auswirkungen auf Workflow-Skripte, die E-Mails versenden:

(Eine detaillierte Erklärung dazu finden Sie im *ConSol CM Process Designer-Handbuch*!)

- Verwenden Sie das Objekt und die Methode `configurationService.getValue("cmweb-server-adapter","mail.reply.to")`, um den Wert der System-Property abzurufen und in der ausgehenden E-Mail als Reply-To-Adresse zu setzen.
- Verwenden Sie das Objekt `Mail`, wenn das queue-spezifische Skript verwendet werden soll, z. B. `mail.useDefaultScript()` . Damit wird die Property `mail.reply.to` überschrieben!

Wenn weder die System-Property noch das queue-spezifische Skript für ausgehende E-Mails verwendet wird, d. h. wenn die Reply-To-Adresse nicht gesetzt ist, wird vom E-Mail-Client normalerweise die From-Adresse als Reply-To-Adresse verwendet.

- **Standardwerte-Skript**
  Hier können Sie ein Skript auswählen, mit dem Werte für Listenfelder vorausgefüllt werden, wenn im Web Client ein Ticket für diese Queue erstellt wird. Das Skript muss in der *Skript- und Template-Verwaltung* des Admin Tools gespeichert sein und den Typ *Standardwerte* haben. Details dazu finden Sie im Abschnitt Skripte des Typs Standardwerte.
- **Duplizieren-Skript**
  Hier können Sie ein Skript auswählen, das ausgeführt wird, wenn ein Ticket in dieser Queue (mit der Option *Duplizieren* im Ticketmenü) dupliziert wird. Das Skript muss in der *Skript- und Template-Verwaltung* des Admin Tools gespeichert sein und den Typ *Duplizieren* haben. Mit dem Duplizierungsskript werden die Standardwerte für ein Ticket gesetzt, das mit der Option *Duplizieren* erstellt wird. Details dazu finden Sie im Abschnitt Skripte des Typs Duplizieren.

- **Sonstiges (4)**
  - **Beschreibung:**
    In dieses Feld können Sie eine Freitextbeschreibung eingeben, z. B. um den Zweck der Queue zu dokumentieren. Diese Information wird nur im Admin Tool angezeigt.

- **Tabs für die Zuweisung von Feldern, Kundengruppen, Textklassen und Projekten (5):**
  - **Tab *Ticketfelder*:**
    Damit die Tickets der Queue Datenfelder (Ticketfelder) haben, müssen Sie die entsprechenden Ticketfeldgruppen hier zuweisen. Details über die Definition von Ticketfeldern finden Sie im Abschnitt Verwaltung von Ticketfeldern (Einrichten des Ticketdatenmodells).
  - **Tab *Kundengruppen*:**
    In dieser Queue können nur für Kunden der ausgewählten Kundengruppen Tickets erstellt werden. Stellen Sie sicher, dass die Bearbeiter, die mit den Tickets dieser Queue arbeiten sollen, auch die entsprechenden Zugangsberechtigungen für die Daten der Kundengruppe haben.
  - **Tab *Textklassen*:**
    Hier können Sie die Textklassen zuweisen, die in den Tickets dieser Queue verfügbar sein sollen. Eine Beschreibung der Definition von Textklassen finden Sie im Abschnitt Textklassen.
  - **Tab *Projekte*:**
    Hier können Sie der Queue Projekte zuweisen. Bearbeiter, die an einem Ticket in der Queue arbeiten, können Zeit auf die Projekte buchen, die der Queue zugewiesen wurden. Die Projekte werden auf der Seite Projekte definiert.

In jedem Tab können Sie einen ausgewählten Eintrag durch Klicken auf den Button *Zuweisen* zuweisen und durch Klicken auf den Button *Entfernen* entfernen.

Klicken Sie danach auf *Speichern*, um die Queue zu erstellen. Die Details der neuen Queue werden auf der rechten Seite angezeigt.

E.3.2.3 Editieren einer Queue

Wenn Sie eine Queue editieren möchten, wählen Sie sie in der Liste aus und klicken Sie auf den Button *Bearbeiten* oder doppelklicken Sie auf den Namen der Queue. Ändern Sie die Queue-Details und klicken Sie auf *Speichern*, um Ihre Änderungen zu speichern.

⚠ Sie können den Workflow einer Queue nicht mehr ändern, nachdem Sie die Queue zum ersten Mal gespeichert haben!

E.3.2.4 Löschen einer Queue

Wählen Sie die Queue, die Sie löschen möchten, in der Liste aus und klicken Sie auf den Button *Löschen*. Wenn Sie das daraufhin angezeigte Fenster mit *Ja* bestätigen, wird die Queue gelöscht und ist im System nicht mehr verfügbar.

⚠ Wenn es noch Tickets für die Queue gibt, kann diese nicht gelöscht werden. Sie müssen die Tickets in eine andere Queue verschieben, bevor Sie sie löschen können.

E.3.2.5 Kopieren einer Queue

Mit dem Button *Kopieren* können Sie beim Erstellen einer Queue Zeit sparen. Die ausgewählte Queue wird kopiert. Die neue Queue hat den gleichen Namen wie die kopierte Queue. Doppelklicken Sie auf den Namen oder klicken Sie auf den Button *Bearbeiten*, um das Editierfenster zu öffnen, in dem Sie den Namen und die Details der Queue ändern können. Klicken Sie auf *Speichern*, um Ihre Änderungen zu speichern.

⚠ Sie können den Workflow einer Queue nicht mehr ändern, nachdem Sie die Queue zum ersten Mal gespeichert haben!

E.3.2.6 Aktivieren oder Deaktivieren einer Queue

Sie können eine Queue deaktivieren, um zu verhindern, dass für diese Queue neue Tickets erstellt werden. Auf diese Weise können Sie die Queue später wieder aktivieren und müssen sie nicht löschen. Um eine Queue zu deaktivieren, wählen Sie sie in der Queue-Liste aus und klicken Sie auf den Button *Deaktivieren*. Der entsprechende Listeneintrag wird jetzt kursiv angezeigt. Klicken Sie unten auf der Seite einfach auf den Button *Aktivieren*, um die Queue wieder zu aktivieren.

Bearbeiter können im Web Client in einer deaktivierten Queue folgende Aktionen ausführen, sofern sie die erforderlichen Berechtigungen haben:

- Ausführen von Workflow-Aktivitäten, also Bewegen von Tickets durch den Geschäftsprozess
- Bearbeiten von Tickets
    - Editieren von Ticketfeldern
    - Hinzufügen von Kommentaren, E-Mails und Attachments

Bearbeiter können im Web Client folgende Aktionen in einer deaktivierten Queue NICHT ausführen:

- Öffnen/Erstellen von neuen Tickets
- Verschieben von Tickets aus einer anderen Queue in diese Queue mithilfe des Drop-down-Menüs des Tickets

E.4 Projekte

In diesem Kapitel werden folgende Themen behandelt:

E.4.1 Einleitung 533

E.4.2 Verwalten von Projekten mit dem Admin Tool 533

E.4.1 Einleitung

In ConSol CM können Sie Arbeitsstunden auf Projekte buchen. Eine detaillierte Erklärung dazu finden Sie in Zeitbuchung mit ConSol CM.

E.4.2 Verwalten von Projekten mit dem Admin Tool

Die Projekte werden im Navigationselement *Projekte* der Navigationsgruppe *Globale Konfiguration* verwaltet. Damit ein Projekt aktiv und im Web Client verfügbar ist, muss es einer Queue zugewiesen werden (siehe Abschnitt Queue-Verwaltung). Im Web Client können Sie dann Arbeitszeit auf Tickets buchen, die sich in einer der Queues befinden, denen das Projekt zugewiesen wurde. Bearbeiter können ihre Zeitbuchungen auf der Bearbeiterprofilseite sehen.

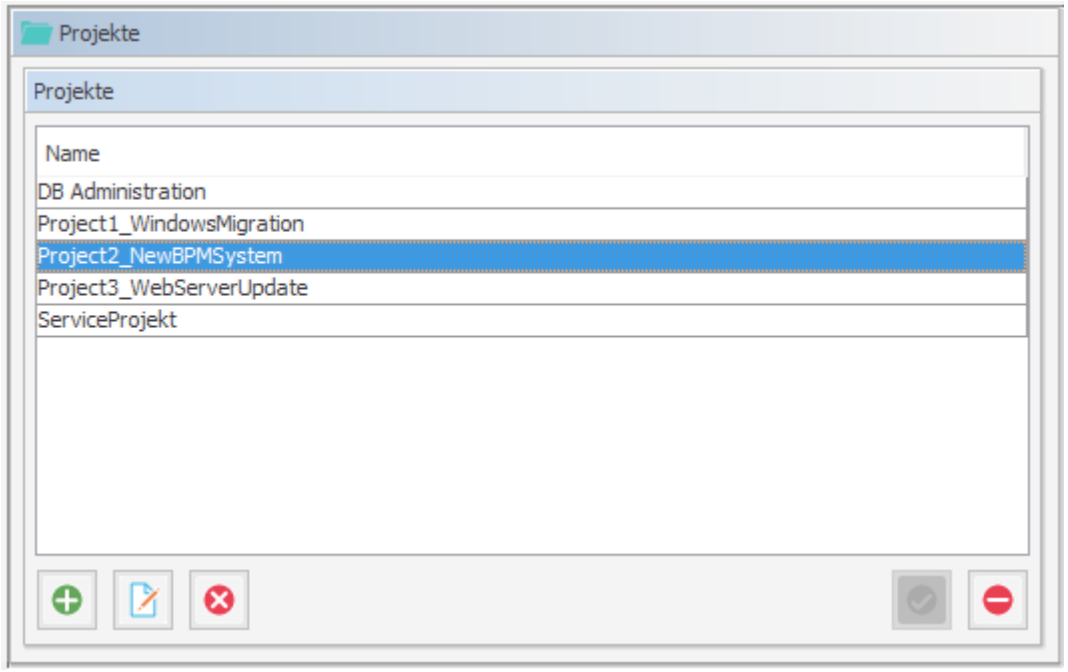


Abbildung 353: *ConSol CM Admin Tool - Globale Konfiguration, Projekte*

E.4.2.1 Erstellen oder Editieren eines Projekts

Ein Projekt wird über seinen Namen definiert. Wenn Sie auf *Hinzufügen* klicken, öffnet sich ein Pop-up-Fenster, in dem Sie den Namen eingeben können. Danach können Sie den Namen über den Button *Lokalisieren* neben dem Namensfeld lokalisieren (siehe Lokalisieren eines Projekts). Die Checkbox

Aktiviert ist bereits ausgewählt, damit das Projekt im System aktiv ist (siehe auch Deaktivieren oder Aktivieren eines Projekts). Wenn Sie zum Editieren eines Projekts auf den Button *Bearbeiten* klicken, wird das gleiche Fenster angezeigt.

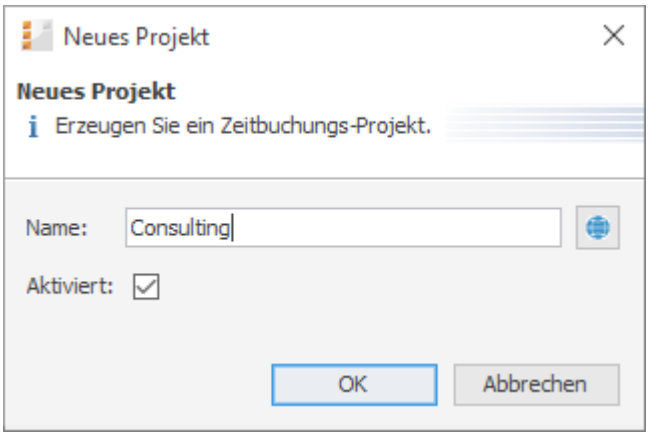


Abbildung 354: *ConSol CM Admin Tool - Globale Konfiguration, Projekte: Erstellen oder Editieren eines Projekts*

E.4.2.2 Löschen eines Projekts

Ein Projekt kann nur gelöscht werden, wenn es keiner Queue zugewiesen ist und nicht für Zeitbuchungen verwendet wurde. Andernfalls wird eine Warnung angezeigt, und Sie können dieses Projekt nur deaktivieren (siehe unten).

Um ein Projekt zu löschen, wählen Sie es in der Liste aus und klicken Sie auf den Button *Löschen*. Wenn Sie im daraufhin angezeigten Bestätigungsfenster *Ja* klicken, wird das Projekt aus der Liste und aus dem System gelöscht.

E.4.2.3 Deaktivieren oder Aktivieren eines Projekts

Wenn ein Projekt nicht mehr benötigt wird, aber noch einer Queue zugewiesen ist oder für Zeitbuchungen in einem Ticket verwendet wurde, können Sie es deaktivieren. Wählen Sie dazu das Projekt aus und klicken Sie auf den Button *Deaktivieren*. Der entsprechende Listeneintrag wird danach kursiv angezeigt. Das Projekt steht für Zeitbuchungen nicht mehr zur Verfügung. Klicken Sie unten auf der Seite einfach auf den Button *Aktivieren*, um das Projekt wieder zu aktivieren.

Sie können ein Projekt auch im Editierfenster aktivieren oder deaktivieren, indem Sie die Checkbox *Aktiviert* markieren oder leer lassen. Wenn Sie ein neues Projekt erstellen, ist diese Checkbox automatisch markiert.

E.4.2.4 Lokalisieren eines Projekts

Klicken Sie im Fenster zum Erstellen oder Editieren des Projekts auf den Button *Lokalisieren* und geben Sie den lokalisierten Namen des Projekts ein. Eine detaillierte Beschreibung des Lokalisierungsmechanismus finden Sie im Abschnitt Lokalisierung von Objekten allgemein, Typ 1.

E.5 Arbeiten mit Kalendern

In ConSol CM gibt es zwei Konfigurationsmöglichkeiten im Zusammenhang mit Kalendern:

1. Sie können Arbeitszeitkalender definieren, die zur Verwaltung von Geschäftszeiten für ein oder mehrere Teams, die mit ConSol CM arbeiten, verwendet werden. Dieses Thema wird im Abschnitt Arbeitszeitkalender behandelt.
2. Sie können ConSol CM so konfigurieren, dass Zugriff auf Microsoft Exchange Server-Kalender besteht. Dies ist im Abschnitt Integration von Microsoft Exchange-Kalendern erklärt.

E.5.1 Arbeitszeitkalender

E.5.1.1 Einleitung

Im Arbeitszeitkalender werden die Arbeitszeiten definiert. Er kann zum Beispiel zur Darstellung der Geschäftszeiten des Service-Desk-Teams verwendet werden, um zu vermeiden, dass Systemeskalationen während der freien Zeiten auftreten.

Sie können in ConSol CM so viele Arbeitszeitkalender definieren, wie für die Umgebung Ihres Unternehmens notwendig sind. Auf diese Weise können Sie für jedes Team bestimmte Arbeitszeiten konfigurieren.

Tickets, die mehr als eine Stunde nach ihrer Erstellung noch keinem Bearbeiter zugewiesen sind, werden automatisch in eine Eskalationsstufe verschoben. Wenn in einem Kalender Arbeitszeiten von 08:00 bis 17:00 festgelegt sind und ein Ticket um 16:45 eingeht, wird das Ticket nicht um 17:45 eskaliert, sondern erst um 08:45 des nächsten Arbeitstages. Die Zeit wird dabei folgendermaßen berechnet: 15 Minuten zwischen dem Eingang des Tickets und dem Ende der Geschäftszeiten plus 45 Minuten vom Beginn der Geschäftszeiten bis die im Eskalationslimit festgelegte volle Stunde erreicht ist.

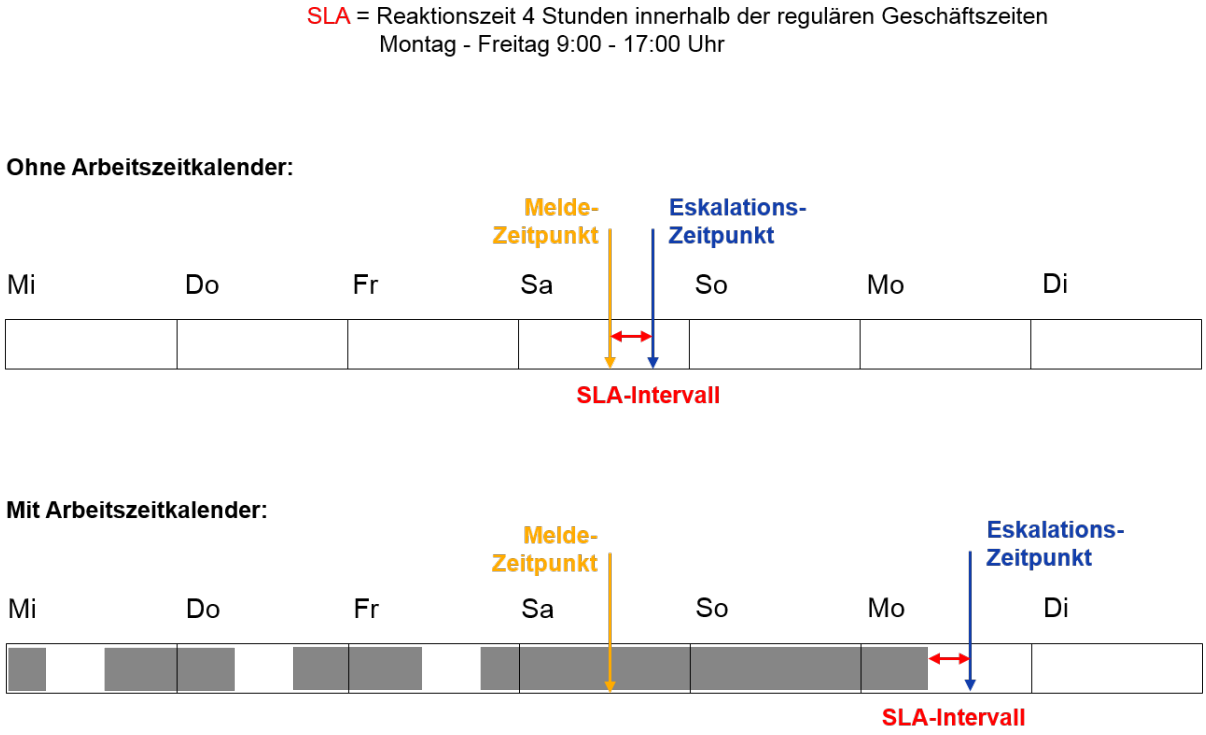


Abbildung 355: *ConSol CM-Prinzip - Arbeitszeitkalender*

Neben den Arbeitszeiten können Sie auch Feiertage definieren. An diesen Tagen pausiert die automatische Eskalation ganz. Die Feiertage müssen für jeden Kalender einzeln definiert werden. Es ist nicht möglich, einen Feiertag zu definieren, der für alle vorhandenen Kalender gleichzeitig gilt.

Wenn Sie mit in einem Arbeitszeitkalender definierten Zeiten arbeiten möchten (z. B. um die aktive Zeit im Workflow für einen Zeit-Trigger zu verwenden, der eine Eskalation auslöst), müssen Sie drei Schritte durchführen:

- Erstellen des Arbeitszeitkalenders mit den aktiven/inaktiven Zeiten (Admin Tool, siehe nachfolgende Beschreibung)
- Zuweisen des Arbeitszeitkalenders zu allen Queues, in denen er gelten soll, siehe Abschnitt Queue-Verwaltung (Admin Tool)
- Zuweisen der Verwendung eines Kalenders zu jedem einzelnen Workflow-Element, in dem der Kalender als Grundlage für Zeitberechnungen verwendet werden soll (Process Designer). Dies wird im *ConSol CM Process Designer Handbuch* beschrieben.

E.5.1.2 Konfiguration eines Arbeitszeitkalenders mit dem Admin Tool

Im Admin Tool werden Arbeitszeitkalender im Navigationselement *Arbeitszeitkalender* der Navigationsgruppe *Globale Konfiguration* definiert.

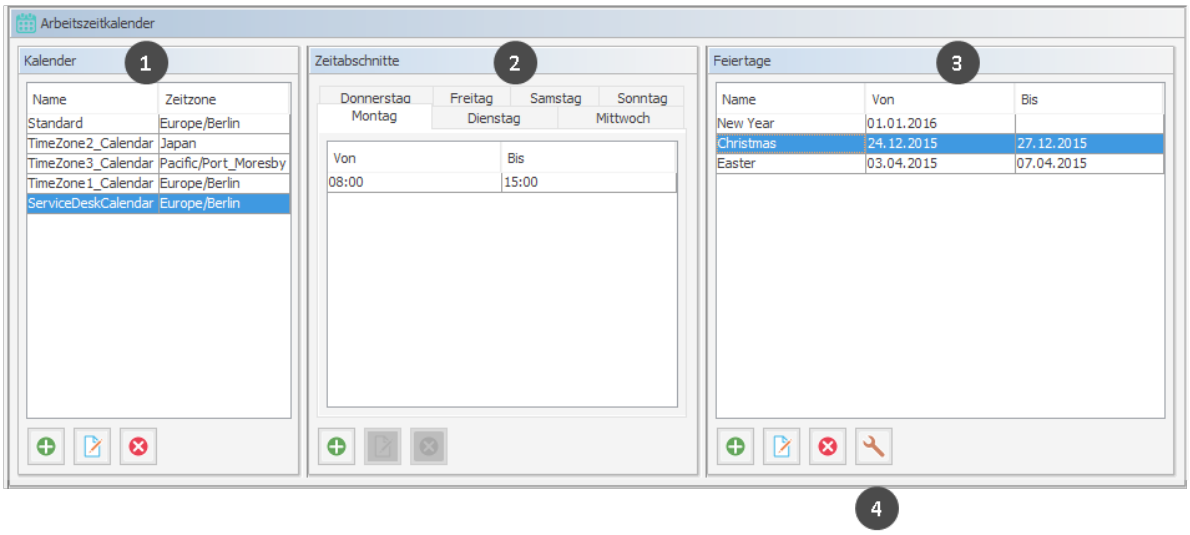


Abbildung 356: *ConSol CM Admin Tool - Globale Konfiguration, Arbeitszeitkalender*

- Liste der verfügbaren Arbeitszeitkalender (1)
- Konfigurierte Arbeitszeiten im ausgewählten Kalender (2)
- Feiertage im ausgewählten Kalender (3)
- Button zum Importieren von Feiertagen aus einer externen CSV-Datei (4)

Erstellen eines neuen Kalenders

Klicken Sie im linken Teil der Seite auf den Button *Hinzufügen*, um einen neuen Kalender zu erstellen. Es wird folgendes Fenster angezeigt:

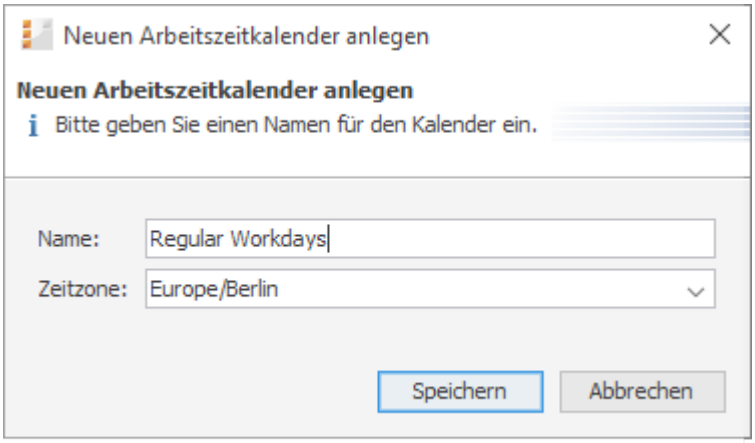


Abbildung 357: *ConSol CM Admin Tool - Globale Konfiguration, Arbeitszeitkalender: Neuer Kalender*

- **Name:**
  Geben Sie einen eindeutigen Namen für den Kalender ein.
- **Zeitzone:**
  Wählen Sie die Zeitzone aus, die für den Kalender verwendet werden soll.

> ⚠ In diesem Feld wird nur festgelegt, auf welche Zeitzone sich die definierten Zeiten beziehen. Der Kalender selbst gilt weltweit für den entsprechenden Workflow!
>
> **Beispiel:**
>
> Der ConSol CM-Server befindet sich in Detroit, MI, USA. Im Arbeitszeitkalender wird Europa/Berlin als Zeitzone festgelegt. Ein Zeit-Trigger, der diesen Business-Event-Trigger verwendet, wird entsprechend der Zeit in Berlin und nicht der Zeit in Detroit ausgelöst.

Klicken Sie danach auf *Speichern*, um den Kalender zu erstellen.

Sie können einen ausgewählten Kalender ändern, indem Sie auf den Button *Bearbeiten* klicken. Klicken Sie auf den Button *Löschen*, wenn Sie den ausgewählten Kalender löschen möchten.

Definieren der Arbeitszeiten für einen Kalender

Wählen Sie einen Kalender auf der linken Seite aus und klicken Sie in der Mitte der Seite auf den Button *Hinzufügen*, um die Tage und Stunden für diesen Kalender zu erstellen. Es wird folgendes Fenster angezeigt:

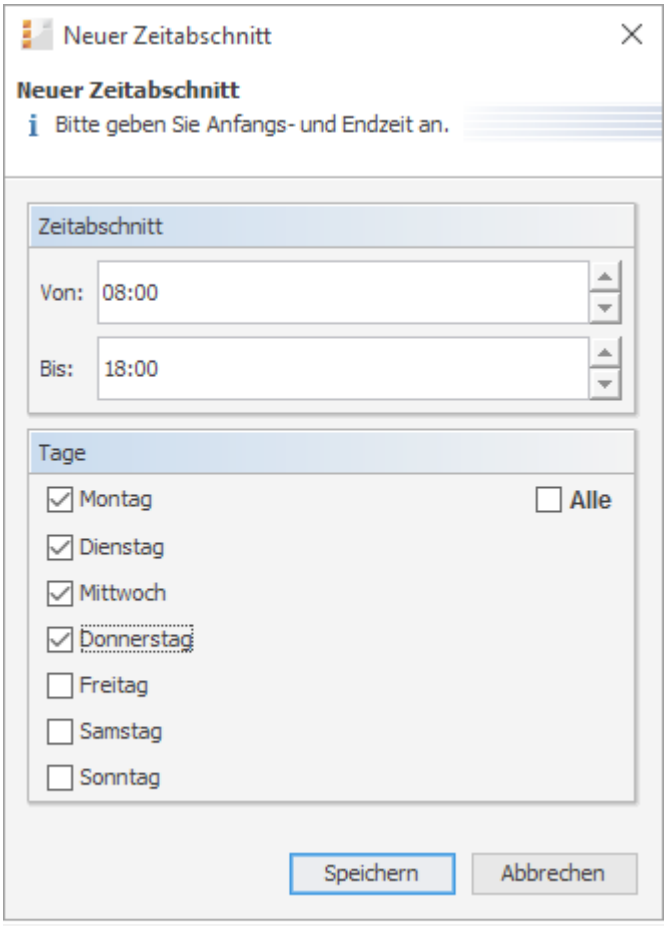


Abbildung 358: *ConSol CM Admin Tool - Globale Konfiguration, Arbeitszeitkalender: Geschäftszeiten eines Kalenders*

- **Zeitabschnitt**
  Geben Sie den Zeitabschnitt ein, für den automatische Workflow-Eskalationen aktiv sein sollen.
- **Tage**
  Markieren Sie die Checkboxen der Tage, für die der Zeitabschnitt gültig sein soll. Sie können einzelne Tage oder alle Tage auswählen (Checkbox *Alle*).

> Falls das System feststellt, dass die hier definierte Zeit im Widerspruch zu einer bereits vorhandenen Zeit steht, wird ein entsprechender Hinweis angezeigt.

Klicken Sie danach auf *Speichern*, um den Zeitabschnitt für die markierten Tage zu erstellen.

Wenn Sie den Zeitabschnitt zu einem späteren Zeitpunkt editieren möchten, müssen Sie jeden Tag einzeln ändern. Wählen Sie den entsprechenden Tag aus, klicken Sie auf den Button *Bearbeiten* und ändern Sie den Zeitabschnitt in dem daraufhin geöffneten Fenster. Alternativ können Sie auf *Löschen* klicken, wenn Sie den Zeitabschnitt für den ausgewählten Tag löschen möchten. Es ist nicht möglich, den Zeitabschnitt für mehrere Tage gleichzeitig zu editieren oder zu löschen.

Definieren der Feiertage für einen Kalender

Sie können die Daten und Zeiträume von Feiertagen auf zwei Wegen definieren:

- Manuelle Definition von Feiertagen.
- Importieren von Feiertagen aus einer Excel-Datei.

Manuelle Definition von Feiertagen für einen Kalender

Wählen Sie einen Kalender und klicken Sie auf der rechten Seite auf den Button *Hinzufügen*, um einen neuen Eintrag für einen Feiertag zu erstellen. Es wird folgendes Fenster angezeigt:

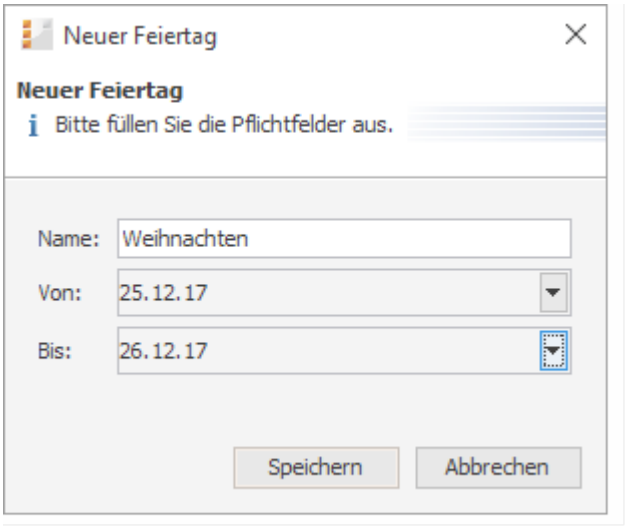


Abbildung 359: *ConSol CM Admin Tool - Globale Konfiguration, Arbeitszeitkalender: Feiertage eines Kalenders*

- **Name:**
  Geben Sie den Namen des Feiertags hier ein.
- **Von:**
  Geben Sie das Datum des Feiertags hier ein.
- **Bis:**
  Wenn dieser Feiertag mehrere Tage dauert, z. B. Weihnachten, können Sie das letzte Datum des Feiertags hier eingeben.

> Es ist nicht möglich, Feiertage zu definieren, die nur einen halben Tag dauern.

Klicken Sie danach auf *Speichern*, um den Feiertag zu erstellen.

Wenn Sie einen ausgewählten Feiertag editieren möchten, klicken Sie auf den Button *Bearbeiten*. Durch Klicken auf den Button *Löschen* werden ein oder mehrere ausgewählte Einträge gelöscht.

Importieren von Feiertagen für einen Kalender aus einer .csv-Datei

Feiertagsdaten können aus einer .csv-Datei importiert werden, die folgendem Format entspricht:

- Erste Spalte: Überschrift/Name des Feiertags
- Zweite Spalte: Startdatum
- Dritte Spalte: Enddatum (Verwenden Sie das gleiche Datum wie das Startdatum, wenn es sich um einen eintägigen Feiertag handelt.)
- Trennzeichen: Komma (kein Komma am Ende der Zeile)
- Schrägstriche innerhalb des Datums

```
Christmas,24/12/2015,27/12/2015
New Year,01/01/2016,01/01/2016
Easter,03/04/2015,07/04/2015
```

Bitte beachten Sie, dass alle Datumsangaben für den Feiertagsimport im folgenden Format geschrieben sein müssen: DD/MM/YYYY.

Wählen Sie im Admin Tool in der Navigationsgruppe *Globale Konfiguration*, Navigationselement *Arbeitszeitkalender* einen Kalender aus, klicken Sie auf *Feiertage importieren* und geben Sie den Pfad zur .csv-Importdatei ein.

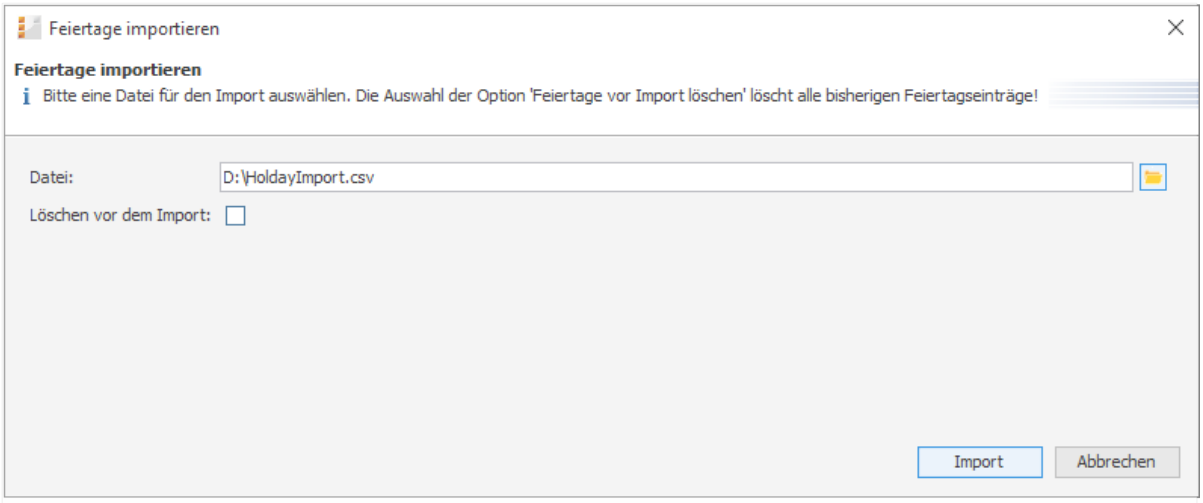


Abbildung 360: *ConSol CM Admin Tool - Globale Konfiguration, Arbeitszeitkalender: Importieren von Feiertagen*

Die neuen Feiertage werden in die Liste der Feiertage des ausgewählten Arbeitszeitkalenders importiert.

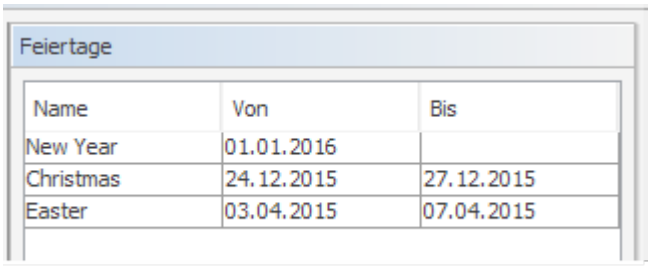


Abbildung 361: *ConSol CM Admin Tool - Globale Konfiguration, Arbeitszeitkalender: Neu importierte Feiertage*

E.5.2 Integration von Microsoft Exchange-Kalendern

E.5.2.1 Einleitung

Ab CM-Version 6.10 ist es möglich, eine Microsoft Exchange-Kalenderansicht in den Web Client zu integrieren.

Eine Liste der unterstützten Versionen von Exchange Server finden Sie in den ConSol CM Systemanforderungen.

Die Kalenderansicht kann an folgenden Stellen angeboten werden ...

- auf der Ticketseite
- auf der Kundenseite, d. h.
    - auf der Kontaktseite
    - auf der Firmenseite

Der Kalender wird in einem eigenen Bereich der Ticket- bzw. Kundenseite angezeigt.

Ein Bearbeiter, der mit der Kalenderansicht arbeitet, kann ...

- zwischen der Monats- und der Wochenansicht wechseln
- vorhandene Termine per Drag-and-Drop verschieben
- neue Termine erstellen (sofern ein Vollzugriff konfiguriert wurde)

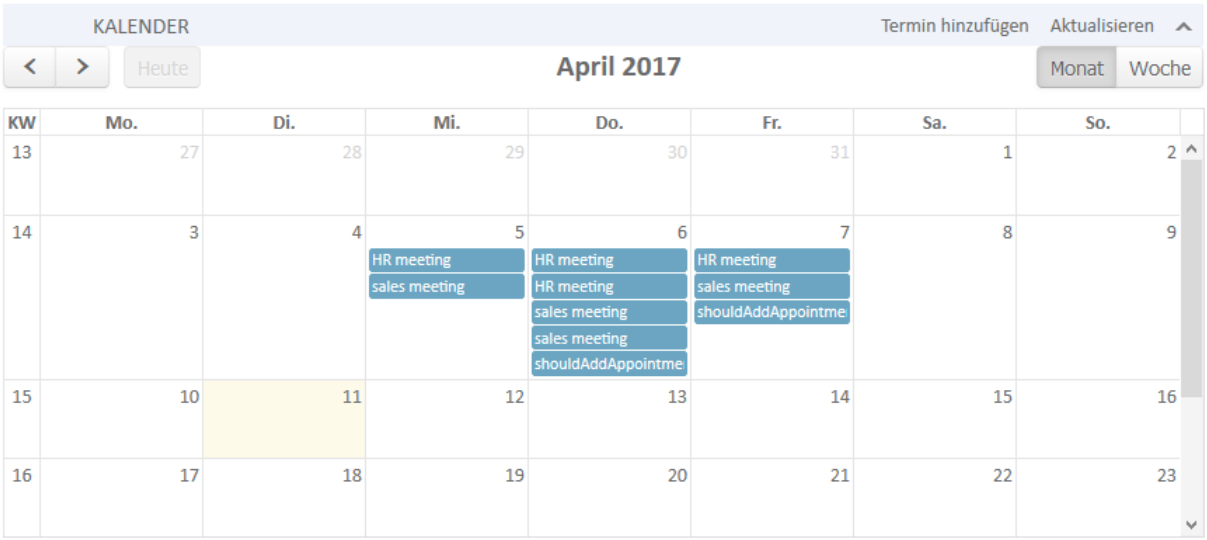


Abbildung 362: *ConSol CM Web Client - Ticket mit Kalenderbereich (Monatsansicht)*

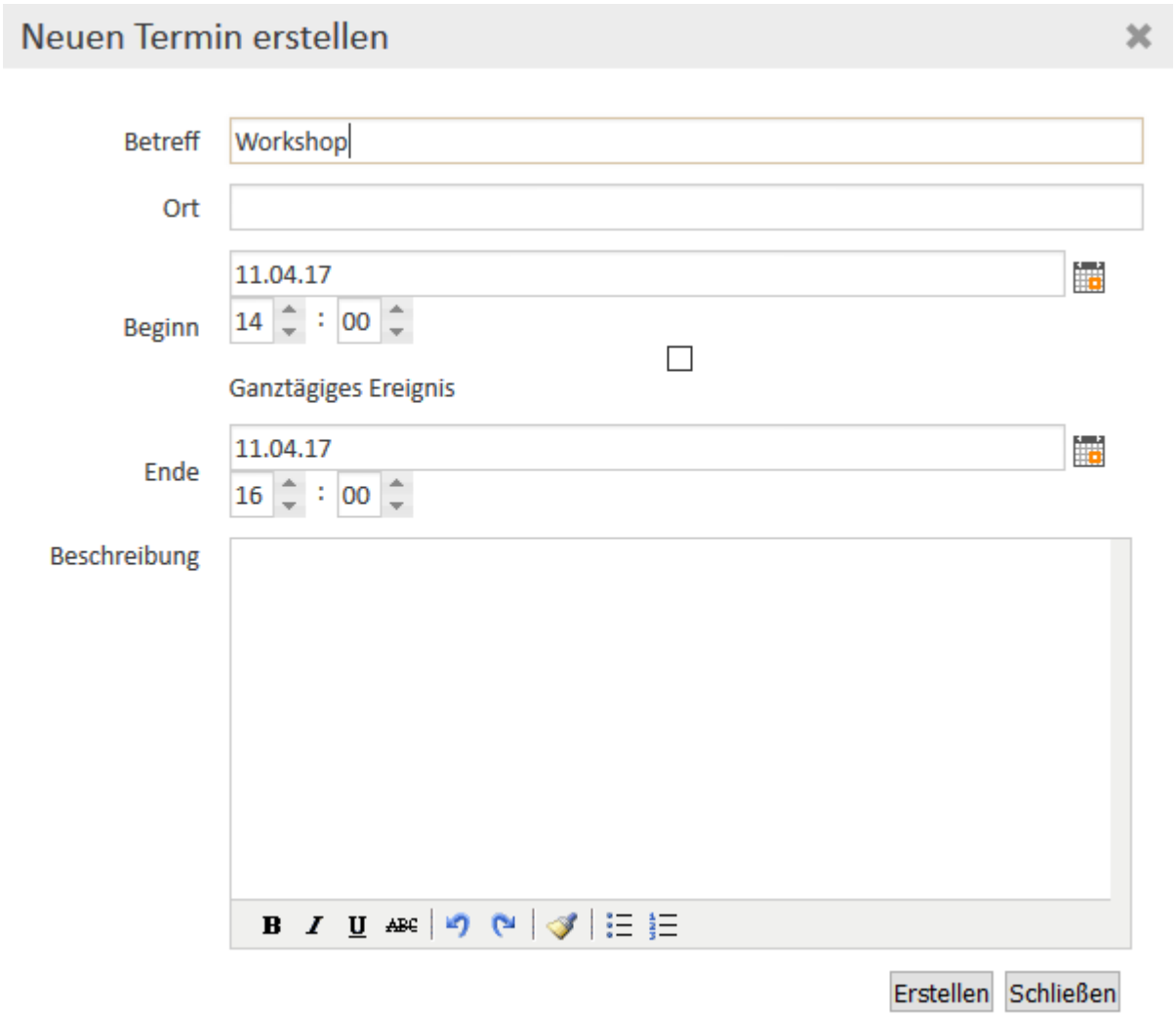


Abbildung 363: *ConSol CM Web Client - Hinzufügen eines Termins im Kalender*

E.5.2.2 Konfigurieren der Integration eines Microsoft Exchange-Kalenders

Grundlegende Konfiguration

Die Integration eines Microsoft Exchange-Servers, der die Kalenderdaten bereitstellen soll, basiert auf der *Seitenanpassung*. Eine detaillierte Einführung in dieses Thema finden Sie im Abschnitt Seitenanpassung. In diesem Kapitel wird nur die Konfiguration erklärt, die den Kalender betrifft.

Führen Sie folgende Schritte durch:

1. **Blenden Sie den Kalenderbereich ein (Beispiel für die Ticketseite):**

   Melden Sie sich als Administrator an und öffnen Sie ein Ticket. Wählen Sie im Hauptmenü *Seitenanpassung aktivieren*. Da der Kalenderbereich noch nicht angezeigt wird, können Sie das Element, das Sie konfigurieren möchten, nicht markieren. Stattdessen können Sie es in der

Baumstruktur der Seitenanpassung auswählen. Wählen Sie *calendar/ticketEditPage/calendarSection* und setzen Sie das Attribut *state* von „hidden“ auf „expanded“. Alternativ können Sie „collapsed“ verwenden. Damit ist der Kalenderbereich am Anfang eingeklappt und der Bearbeiter kann ihn manuell ausklappen. In beiden Fällen ist der Kalenderbereich im Ticket sichtbar. In der Kopfzeile wird *Kein Kalender* angezeigt. Die Konfiguration des Kalenders folgt in Schritt 2.

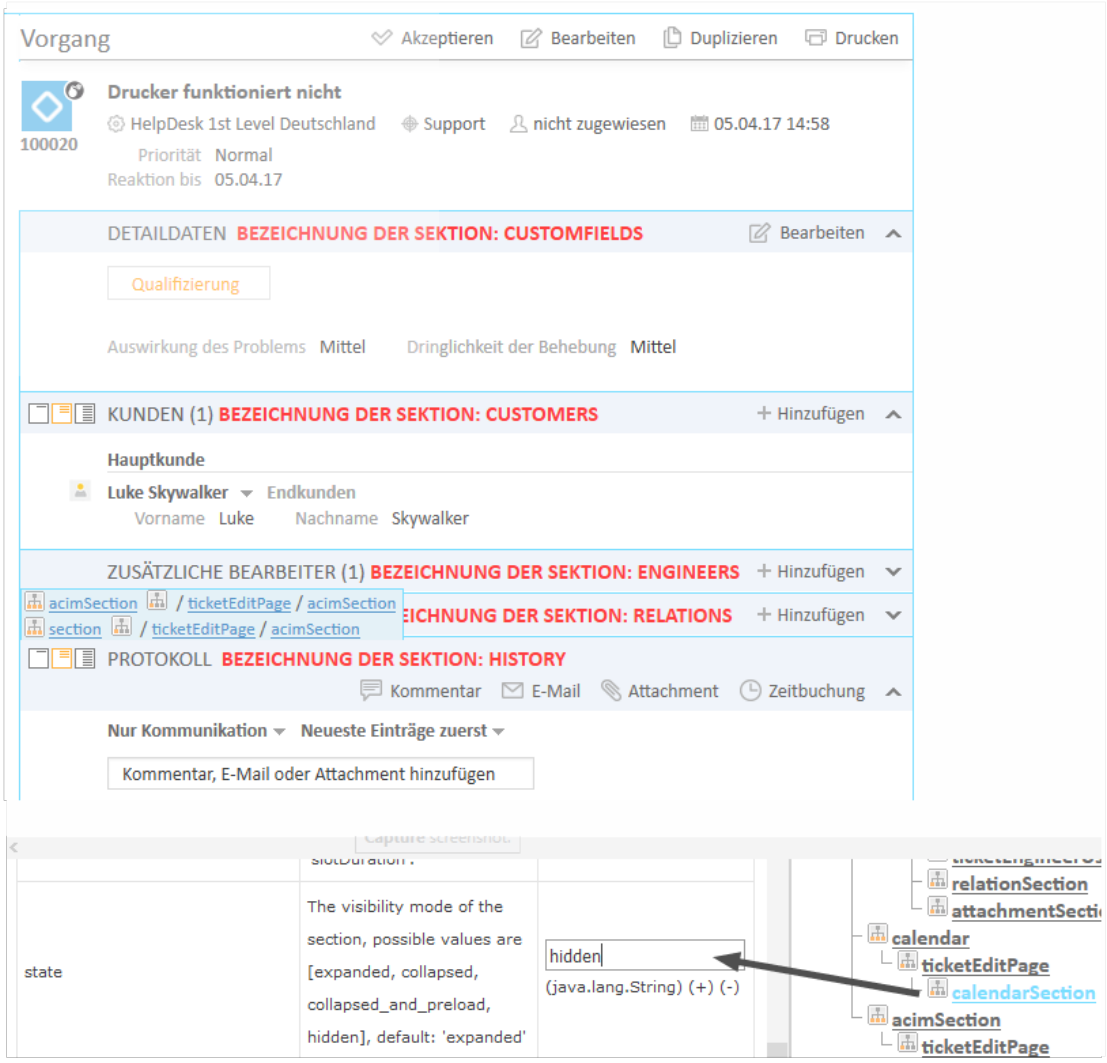


Abbildung 364: *ConSol CM Web Client - Einblenden des Kalenderbereichs im Ticket mit der Seitenanpassung*

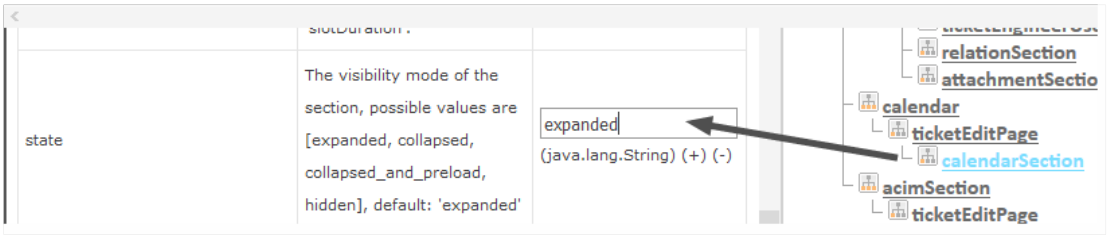


Abbildung 365: *ConSol CM Web Client - Definitionsbereich der Seitenanpassung für den Kalenderbereich des Tickets*

2. **Konfigurieren Sie das Kalenderskript:**

   Der Verbindung zum Kalender auf dem Microsoft Exchange-Server wird mithilfe eines Admin-Tool-Skripts implementiert. Das Skript muss im Skriptbereich des Admin Tools gespeichert werden, d.h. in der Navigationsgruppe *System*, Navigationselement *Skripte und Templates*, und muss vom Skripttyp *Kalenderintegration* sein. Der Name des Skripts muss in der Seitenanpassung als Wert für das Attribut *calendarInitializationScript* eingegeben werden (siehe Schritt 1). Das Admin-Tool-Skript wird zum Initialisieren der Kalenderquelle verwendet und muss eine Map zurückgeben, die alle Parameter enthält, mit denen die Quelle beschrieben wird. Wenn die zurückgegebene Map leer oder null ist, wird kein Kalender angezeigt. (Der Bereich ist leer und hat die Beschreibung *Kein Kalender*.)

   Das Skript muss wie im folgenden Beispiel gezeigt codiert werden:

```
return [
   name: 'Exchange Source',
   'access.type': 'EXCHANGE',
   'access.url': 'https://exchange1.server.net/EWS/Exchange.asmx',
   'access.username': 'exchange-user',
   'access.password':'exchange-password',
   'access.domain': 'SSO',
   'access.impersonation':'somebody@sso.server.net',
   'access.version': '2013'
]
```

Code-Beispiel 59: *Beispiel für Kalenderintegrationsskript*

Im Skript können folgende Parameter verwendet werden:

1. **name**
   String. Der Name der Quelle. Noch hat diese Angabe keine Funktion. In zukünftigen CM-Versionen, wenn mehrere Quellen im Kalenderbereich verwendet werden können, dient sie der Identifizierung der Quelle auf der Benutzeroberfläche.
2. **color**
   String. Hintergrundfarbe für Termineinträge. Format: HTML-Farbformat (z. B. red oder #D80000)

3. **editable**
   Boolean. Gibt an, ob das Erstellen/Bearbeiten/Entfernen von Terminen aktiviert sein soll. Wenn dieser Wert nicht explizit gesetzt ist, wird der Wert aus der Seitenanpassung des Bereichs verwendet. Dieser ist standardmäßig auf „false“ gesetzt. Hinweis: Termine können nur bearbeitet werden, wenn der aktuelle Benutzer Schreibrechte auf das Ticket hat, in dem der Kalenderbereich angezeigt wird.
4. **access.***
   Properties für den Zugriff auf den Kalender: Anmeldeinformationen, Verbindungsdetails, URLs ... usw.
   a. **access.type**
      String-Parameter. Mögliche Werte EXCHANGE (Microsoft Exchange Server), RANDOM (zu Testzwecken zufällig generierte Termine).

      Für access.type EXCHANGE:

      i. **access.url**
         URL. Die URL der .asmx-Datei (Active Server Method File) auf dem Exchange-Server. Diese Datei befindet sich normalerweise im Verzeichnis EWS (Exchange Web Services) und stellt den Exchange Access Web Service zur Verfügung.

      ii. **access.username**
         String. Der Name des technischen Benutzers, mit dem der Zugriff / die Anmeldung beim Exchange-Server durchgeführt wird.

      iii. **access.password**
         String. Das Passwort des technischen Benutzers, mit dem der Zugriff / die Anmeldung beim Exchange-Server durchgeführt wird.

      iv. **access.domain**
         String. Die Windows-Domäne des Exchange-Servers. Der technische Benutzer (für access.username und access.password, siehe oben) muss ein Mitglied dieser Domäne sein.

      v. **access.impersonation**
         String. E-Mail-Konto eines Exchange-Kalenderbenutzers. Der Kalender auf der Ticketseite oder der Kundenseite basiert auf der Ansicht, die dieser Benutzer auf den Exchange-Kalender hat. So könnte *somebody* zum Beispiel das Exchange-Login des momentan angemeldeten ConSol CM-Bearbeiters sein. Wenn der Name des Bearbeiters in Exchange und ConSol CM identisch ist, können Sie über die Methode `workflowApi.getCurrentEngineer().getName()` oder `engineerService.current.name` den Namen des aktuellen Bearbeiters an Exchange weitergeben.

      Für access.type RANDOM:

i. **access.calendar**
Name der Kalenderdatei. Beim zufälligen Kalender werden die erzeugten Termine auf der Festplatte gespeichert. (Die Dateien werden automatisch entfernt, wenn der CM-Server angehalten wird.) Durch die Verwendung des gleichen Namens in der Konfiguration wird sichergestellt, dass der Benutzer den gleichen Satz an Terminen hat.)

Zusätzliche Variablen im Skript:

- Ticketkontext: **ticket**
- Kontakt-/Kundenkontext: `unit`

3. **Aktivieren Sie den Editiermodus für den Kalender:**
Um den Bearbeitern vollen Zugriff zu ermöglichen (d. h. Erstellen und Editieren von Terminen), setzen Sie das Attribut *editable* auf „true“.

Grundprinzip des Zugriffs auf Kalender auf Exchange-Servern

Der konfigurierte Exchange-Server, genauer gesagt der in der URL angegebene Web Service, wird kontaktiert. Die Anmeldung auf dem Server erfolgt mit dem technischen Benutzer (access.username, access.password). Danach wird der Benutzer in die Person / den Benutzer geändert, der unter `access.impersonation` angegeben ist. Letzteres wird mithilfe der Funktion *Impersonation* (Identitätswechsel) des Exchange-Servers durchgeführt. Der Kalender des Benutzers, dessen Identität der technische Benutzer angenommen hat, wird im Ticket oder auf der Kundenseite angezeigt.

> Die Funktion zum Wechseln der Identität kann nur von einem Konto verwendet werden, dem der Exchange-Administrator die Rolle *ApplicationImpersonation* erteilt hat. Stellen Sie beim Einrichten dieser Rolle sicher, dass alle Sicherheitsaspekte berücksichtigt werden!

Erweiterte Konfiguration: Seitenanpassungsparameter für die Integration von Microsoft Exchange-Kalendern

Kurze Hintergrundinformationen über die Integration von Microsoft Exchange-Kalendern

Die Integration (oder genauer gesagt, die *Anzeige*) von Microsoft Exchange-Kalendern in ConSol CM basiert auf dem jQuery-Plugin ***fullcalendar***. Vollständige Details über diese API finden Sie auf der Blendarkatur-Website.

Seiten für die Konfiguration von calendarSection

Der Kalenderbereich kann auf drei Seiten konfiguriert werden:

- **Ticket**
Verwenden Sie *calendar/ticketEditPage/calendarSection*.
- **Kundenseiten**
  - **Kontaktseite**
  Verwenden Sie *calendar type/contactEditPage/calendarSection*.
  - **Firmenseite**
  Verwenden Sie *calendar/companyEditPage/calendarSection*.

calendarSection

Mit den folgenden Attributen können die Darstellung und das Verhalten des integrierten Exchange-Kalenders konfiguriert werden.

Attribute:

- **allDaySlot**
  Boolean. Legt fest, ob das Zeitfenster "Ganztägig" oben im Kalender angezeigt werden soll. Der Standardwert ist „true“.
- **appointmentBackgroundColor**
  java.lang.String. Legt die Hintergrundfarbe für alle Termine im Kalender fest. Sie können ein beliebiges CSS-Farbformat verwenden, z. B. „#f00“, „#ff0000“, „rgb(255,0,0)“ oder „red“.
- **appointmentBorderColor**
  Legt die Rahmenfarbe für alle Termine im Kalender fest. Sie können ein beliebiges CSS-Farbformat verwenden, z. B. „#f00“, „#ff0000“, „rgb(255,0,0)“ oder „red“. (java.lang.String)
- **appointmentColor**
  java.lang.String. Legt die Hintergrund- und Rahmenfarbe für alle Termine im Kalender fest. Sie können ein beliebiges CSS-Farbformat verwenden, z. B. „#f00“, „#ff0000“, „rgb(255,0,0)“ oder „red“.
- **appointmentConstraint**
  java.lang.String. Beschränkt das Verschieben, Verlängern und Verkürzen von Terminen auf ein bestimmtes Zeitfenster.

  Mögliche Werte:

  - **<Termin-ID>**
    Termine, die verschoben, verlängert oder verkürzt werden, müssen komplett innerhalb von mindestens einem der über die Termin-ID angegebenen Terminen liegen.
  - **businessHours**
    Termine, die verschoben, verlängert oder verkürzt werden, müssen komplett innerhalb der Geschäftszeiten liegen (standardmäßig Montag-Freitag 09:00-17:00), siehe Attribut *businessHours*.
  - **<Startzeit>-<Endzeit>;<Wochentage>**
    Ein benutzerdefiniertes Zeitfenster im gleichen Format wie das Attribut *businessHours*. Die Wochentage sind optional.
    Beispiele: *10:00-18:00; 1,2,3,4* oder *10:00-18:00*
- **appointmentDurationEditable**
  Boolean. Ermöglicht das Editieren der Termindauer über die Änderung der Größe. Der Standardwert ist „true“.
- **appointmentOverlap**
  Boolean. Legt fest, ob sich die Termine im Kalender nach dem Verschieben oder Verlängern überlappen dürfen. Der Standardwert ist „true“.
- **appointmentStartEditable**
  Boolean. Ermöglicht es, die Startzeit eines Termins durch Verschieben zu verändern. Der Standardwert ist „true“.

- **appointmentTextColor**
  java.lang.String. Legt die Textfarbe für alle Termine im Kalender fest. Sie können ein beliebiges CSS-Farbformat verwenden, z. B. „#f00“, „#ff0000“, „rgb(255,0,0)“ oder „red“.
- **aspectRatio**
  java.lang.String. Legt das Verhältnis von Breite zu Höhe des Kalenders fest. Der Standardwert ist „2.8“ Wenn die Kalenderkomponente leer ist, wird der interne Standardwert 1.35 verwendet.
- **businessHours**
  java.lang.String. Hebt bestimmte Zeiträume im Kalender hervor.
  Format: <Startzeit>-<Endzeit>;<Wochentage>
  Beispiel: 10:00-18:00; 1,2,3,4 (von 10:00 bis 18:00, Montag bis Donnerstag)
- **calendarEventHandlerScript**
  java.lang.String. Enthält den Namen des Skripts, das die Kalendererereignisse steuert. Neben den Standardvariablen des Kontextes, wie **ticket**, gibt es folgende zusätzliche Variablen:
  - **eventType**
    enum (Werte: CREATE, UPDATE, DELETE)
  - **appointment**
    mit Termindaten (uid, subject, location usw.).

  Details finden Sie in der Dokumentation. (java.lang.String)
- **calendarInitializationScript**
  java.lang.String. Enthält den Namen des Skripts, das die Kalenderkonfiguration erzeugt. Wenn der Wert leer ist oder das Skript „null“ zurückgibt, wird der Kalender nicht angezeigt.
- **contentHeight**
  java.lang.String. Gibt die Höhe des Inhaltsbereichs des Kalenders in Pixel an. Standardmäßig ist die Option nicht gesetzt und die Höhe des Kalenders wird anhand des Attributs *aspectRatio* berechnet.
- **defaultAllDayAppointmentDuration**
  java.lang.String. Die Standarddauer für ganztägige Termine, für die keine *Endzeit* angegeben ist. Der Standardwert ist „1“ (ein Tag).
- **defaultDate**
  java.lang.String. Die initialen Daten, die beim ersten Laden des Kalenders angezeigt werden. Möglich ist ein ISO8601-Datumsstring wie „2014-02-01“.
- **defaultTimedAppointmentDuration**
  java.lang.String. Die Standarddauer für nicht ganztägige Termine, für die keine *Endzeit* angegeben ist. Wenn nicht gesetzt, wird der Standardwert „02:00:00“ (2 Stunden) verwendet. Dieses Attribut wirkt sich auch auf die Standarddauer von Terminen bei der Terminerstellung aus.
- **DefaultView**
  java.lang.String. Standardansicht des Kalenders.
  Mögliche Werte: month, basicWeek, basicDay, agendaWeek, agendaDay. Der Standardwert ist „agendaWeek“.
  Beispielansichten unter Verfügbare Ansichten.

- **editable**
  Boolean. Ob Termine erstellt, verschoben, verlängert und verkürzt werden können. Dieser Wert überschreibt die Quellkonfiguration. Der Standardwert ist „false“.
- **firstDay**
  java.lang.String. Der Tag, an dem die Woche beginnt. (Sonntag=0, Montag=1, Dienstag=2 usw.). Wenn leer, wird der Wert aus der Sprachumgebung des Browsers verwendet.
- **forceAppointmentDuration**
  Boolean. Ein Flag, mit dem die Berechnung eines Terminendes erzwungen werden kann, wenn kein Ende angegeben wird. Der Standardwert ist „false“.
- **handleWindowResize**
  Boolean. Ob die Größe des Kalenders automatisch angepasst werden soll, wenn der Benutzer die Größe des Browser-Fensters verändert. Der Standardwert ist „true“.
- **headerCenter**
  java.lang.String. Definiert die Buttons und die Überschrift in der Mitte über dem Kalender. Details dazu finden Sie in der Beschreibung von *headerLeft*. Der Standardwert ist „title“.
- **headerLeft**
  java.lang.String. Definiert die Buttons und die Überschrift links über dem Kalender. Liste mit durch Kommas oder Leerzeichen getrennten Werten (durch Kommas getrennte Werte werden nebeneinander angezeigt). Der Standardwert ist „prev,next today“.

  Mögliche Werte:

  - **title**
    Text, der den aktuellen Monat/Woche/Tag enthält.
  - **prev**
    Button, mit dem der vorherige Monat/Woche/Tag im Kalender angezeigt wird.
  - **next**
    Button, mit dem der nächste Monat/Woche/Tag im Kalender angezeigt wird.
  - **prevYear**
    Button, mit dem das letzte Jahr im Kalender angezeigt wird.
  - **nextYear**
    Button, mit dem das nächste Jahr im Kalender angezeigt wird.
  - **today**
    Button, mit dem der aktuelle Monat/Woche/Tag im Kalender angezeigt wird.
  - **<Name der Ansicht>**
    Button, mit dem eine der verfügbaren Ansichten des Kalenders geladen wird (eine Beschreibung der verfügbaren Ansichten finden Sie unter *defaultView*).

  Die Kopfzeile wird nicht angezeigt, wenn alle drei header-Attribute (Center, Left, Right) leer sind.
- **headerRight**
  java.lang.String. Definiert die Buttons und die Überschrift rechts über dem Kalender. Details dazu finden Sie in der Beschreibung des Attributs *headerLeft*. Der Standardwert ist „month,agendaWeek“.

- **height**
  java.lang.String. Gibt die Höhe des gesamten Kalenders (einschließlich Kopfzeile) in Pixel an. Standardmäßig ist die Option nicht gesetzt und die Höhe des Kalenders wird anhand des Attributs *aspectRatio* berechnet.
- **hiddenDays**
  java.lang.String. Schließt bestimmte Wochentage aus der Anzeige aus. Durch Kommas getrennte Liste der Wochentagsindizes (Beispiel: '1,3,5'). Jeder Index beginnt mit der Null (Sonntag=0) und umfasst die Werte 0 bis 6.
- **lazyFetching**
  Boolean. Legt fest, wann die Termine abgerufen werden sollen. Details finden Sie in der Dokumentation. Es ist sinnvoll, dieses Attribut auf „false“ zu setzen, wenn am Kalender des Benutzers viele externe Änderungen vorgenommen werden. Der Standardwert ist „true“.
- **maxTime**
  java.lang.String. Legt die Endzeit fest, die (ausschließlich) angezeigt wird, auch wenn der Benutzer ganz nach unten gescrollt hat. Der Standardwert ist „24:00:00“.
- **minTime**
  java.lang.String. Legt die Startzeit fest, die angezeigt wird, auch wenn der Benutzer ganz nach oben gescrollt hat. Der Standardwert ist „00:00:00“.
- **nextDayThreshold**
  java.lang.String. Wenn die Endzeit eines Termins an einem anderen Tag liegt, muss der Termin mindestens bis zu dieser Zeit dauern, damit er an diesem Tag angezeigt wird. Der Standardwert ist „09:00:00“ (09:00). Dies betrifft nur Termine, die nicht als ganztägige Termine konfiguriert sind, aber trotzdem den ganzen Tag ausfüllen. Ganztägige Felder kommen in den Kalenderansichten *month*, *basicDay* und *basicWeek* und in *ganztägigen* Zeitfenstern vor.
- **rightToLeftMode**
  Boolean. Zeigt den Kalender von rechts nach links an. Der Standardwert ist „false“.
- **scrollTime**
  java.lang.String. Legt fest, wie weit der scrollbare Bereich am Anfang heruntergescrollt wird. Der Standardwert ist „06:00:00“ (6:00).
- **slotAppointmentOverlap**
  Boolean. Legt fest, ob Termine sich in der Kalenderansicht überlappen sollen. Der Standardwert ist „true“.
- **slotDuration**
  java.lang.String. Das Intervall für die Anzeige von Zeitfenstern. Der Standardwert ist „00:30:00“ (30 Minuten).
- **snapDuration**
  java.lang.String. Das Zeitintervall, bei dem sich ein Termin beim Verschieben an das Zeitraster der Kalenderansicht anheftet. Diese Einstellung wirkt sich auch auf die Genauigkeit der Zeitauswahl aus. Der Standardwert ist „slotDuration“.
- **state**
  java.lang.String. Der Sichtbarkeitsmodus dieses Bereichs, mögliche Werte sind [*expanded, collapsed, collapsed_and_preload, hidden*]. Der Standardwert ist „expanded“.

- **weekNumbers**
  Boolean. Legt fest, ob die Nummer der Woche im Kalender angezeigt werden soll. Der Standardwert ist „true“.
- **weekends**
  Boolean. Legt fest, ob die Spalten für den Samstag und Sonntag in der Kalenderansicht enthalten sein sollen. Der Standardwert ist „true“.

Definieren von Event-Handlern für Terminereignisse

Mithilfe des Seitenanpassungsattributs *calendarEventHandlerScript* können Sie Aktionen definieren, die angestoßen werden sollen, wenn ein bestimmtes Ereignis eingetreten ist. Die Ereignisse, die verwendet werden können, sind:

- ein neuer Termin wurde erstellt (Ereignistyp CREATE)
- ein Termin wurde bearbeitet (Ereignistyp UPDATE)
- ein Termin wurde entfernt (Ereignistyp DELETE)

Zum Beispiel können Sie definieren, dass automatisch eine E-Mail an alle Kontakte des Tickets gesendet wird, wenn ein neuer Termin gemacht wird.

In einem calendarEventHandlerScript sind folgende Variablen verfügbar:

- **ticket** (nur im Ticketkontext)
- **unit** (nur im Kontakt-/Firmenkontext)
- **eventType** - Ereignistyp, mögliche Werte sind: CREATE, UPDATE oder DELETE. Dies ist ein Enum des Typs com.consol.cmweb.server.common.model.calendar.AppointmentEventType.
- **appointment** - Terminobjekt (Klasse com.-consol.cmweb.server.common.model.calendar.AppointmentVo).
  Properties (Einige Properties sind möglicherweise nicht verfügbar. Dies hängt vom Typ des Kalender-Servers und der entsprechenden Version ab):

  Grundeinstellungen:

  - **subject**
    String. Betreff/Titel des Termins.
  - **startDate**
    Datum. Startdatum/-zeit des Termins.
  - **endDate**
    Datum. Enddatum/-zeit des Termins.
  - **allDayEvent**
    Boolean. Definiert, ob ein Termin ein ganztägiges Ereignis ist, d. h. den ganzen Tag (oder mehrere Tage) lang dauert.
  - **location**
    String. Ort des Termins.

- **meeting**
  Boolean. Ist „true“, wenn der Termin eine Besprechung ist. (Dies ist eine spezielle Property von Microsoft Exchange Server; wenn Teilnehmer eingeladen wurden, wird aus dem Termin eine Besprechung)
- **cancelled**
  Boolean. Gibt an, ob der Termin als *abgesagt* markiert ist.
- **recurring**
  Boolean. Gibt an, ob der Termin Teil einer Terminserie ist.
- **busyStatus**
  AppointmentVo.BusyStatus. Mögliche Werte: FREE, TENTATIVE, BUSY, OUT_OF_OFFICE, WORKING_ELSEWHERE oder NONE.
- **body**
  String. Text, Inhalt des Termins. Dies kann Text oder HTML sein (abhängig von der Property *bodyType*)
- **bodyType**
  AppointmentVo.BodyType. Mögliche Werte TEXT, HTML oder NONE

Erweiterte Einstellungen:

- **uid**
  String. Eindeutige Identifikation eines Termin auf dem Kalender-Server.
- **start**
  org.joda.time.DateTime. Startdatum/-zeit (joda)
- **end**
  org.joda.time.DateTime. Enddatum/-zeit (joda)
- **timeZone**
  org.joda.time.DateTimeZone. Zeitzone des Termins (wird verwendet, um ganztägige Termine, die in jeder Zeitzone zu einer anderen Uhrzeit beginnen, richtig anzuzeigen.

Das folgende Skript zeigt ein Beispiel für ein `calendarEventHandlerScript`.

```
import static
 com.consol.cmweb.server.common.model.calendar.AppointmentEventType.*;
import com.consol.cmas.common.model.customfield.meta.UnitDefinitionType;
// Check if you are on ticket page or on customer page:
def inTicket = false
def inUnit = false
if ( binding.variables.containsKey("ticket") ) {
   log.info "Context: Ticket" inTicket = true
}
if ( binding.variables.containsKey("unit") ) {
   log.info "Context: Unit" inUnit = true
}
def recip def mailField
// if you are on ticket page: write e-mail to engineer
if (inTicket){
```

```
   recip = ticket.engineer?.email
// if you are in unit context, CONTACT, write to contact
} else if (inUnit) {
def unitDefName = unit.definitionName
def unitDefType = unit.definition.type

log.info ' Definition is now ' + unitDefName
if (unitDefType == UnitDefinitionType.CONTACT){
   switch (unitDefName) {
      case 'DirCustCustomer': mailField = "dir_cust_email";
         break
      case 'customer': mailField= "email"
         break
      case 'PartnersContact': mailField = "email"
         break
      case 'ResellerCustomer': mailField = "email"
         break
   }
} else if (unitDeftype == COMPANY) {
   log.info "No email for Company!"
}


recip = unit.get(mailField) }
log.info 'mailField is now ' + mailField
log.info 'recip is now ' + recip


// EXAMPLE! log.info only :
log.info "Appointment '${appointment.subject}' has been "
if (eventType == CREATE) {
   log.info "created"
} else if (eventType == UPDATE) {
   log.info "modified"
} else if (eventType == DELETE) {
   log.info "removed"
}


// send mail here ...
```

Code-Beispiel 60: *Admin-Tool-Skript, Beispiel für ein calendarEventHandlerScript*

E.6 Textklassen

In diesem Kapitel werden folgende Themen behandelt:

E.6.1 Einleitung 556

E.6.2 Verwaltung von Textklassen mit dem Admin Tool 558

E.6.1 Einleitung

Eine **Textklasse** ist eine Klassifizierung, die Sie einem Ticketeintrag zuweisen. Dies kann sein:

- ein Kommentar
- eine E-Mail, die aus dem Ticket gesendet wurde
- eine E-Mail, die vom Ticket erhalten wurde
- ein Attachment

Die Zuweisung von Textklassen kann mehreren Zwecken dienen:

- Hervorheben des Texts im Ticket in einer bestimmten Farbe, damit er leichter wiedergefunden wird (z. B. ein wichtiger Hinweis wie in der folgenden Abbildung gezeigt). Zusätzlich kann für jede Textklasse ein Icon verwendet werden.
- Kennzeichnen eines Ticketeintrags, damit er in CM/Track sichtbar ist, d. h. damit er für die Kunden verfügbar ist, die sich im ConSol CM-Kundenportal anmelden.
- Kennzeichnen eines Eintrags zur Steuerung des Prozessablaufs, z. B. kann ein Ticket nur abgeschlossen werden, wenn mindestens ein Eintrag als *Lösung* markiert ist.
- Kennzeichnen eines Eintrags für die Übergabe an einen anderen Prozess, z. B. werden die Einträge, die als *Frage* und *Antwort* gekennzeichnet sind, automatisch für ein FAQ-Ticket verwendet.

Auf diese Weise können Sie mit Textklassen die Informationen innerhalb des Tickets organisieren und den Prozessablauf sowie die Verfügbarkeit von Informationen steuern.

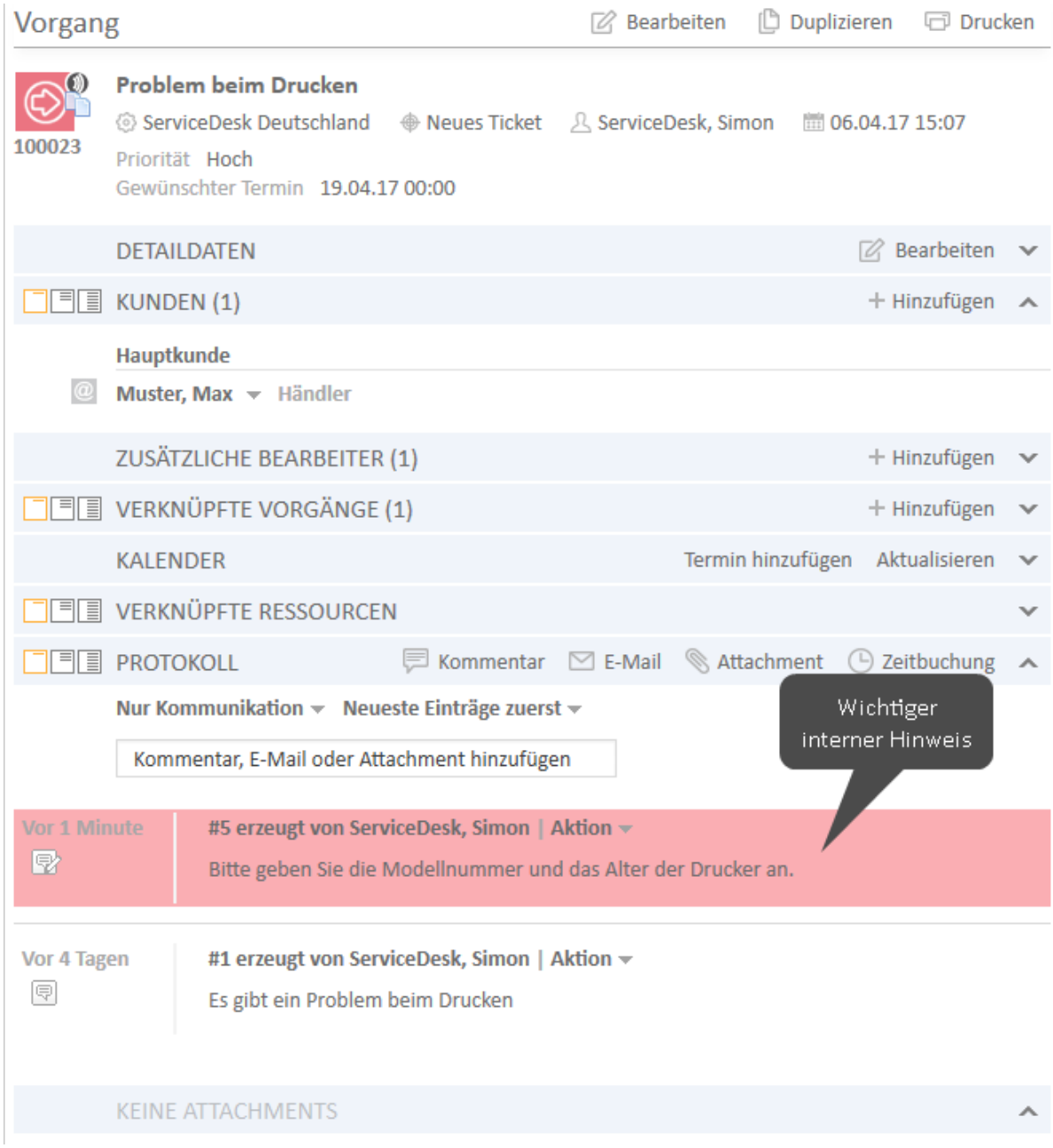


Abbildung 366: *ConSol CM Web Client - Verwenden einer Textklasse für einen wichtigen internen Hinweis*

E.6.2 Verwaltung von Textklassen mit dem Admin Tool

E.6.2.1 Hinzufügen einer neuen Textklasse

Zum Hinzufügen einer neuen Textklasse für Tickets in einer bestimmten Queue sind zwei Schritte erforderlich:

1. Definieren der Textklasse im Navigationselement *Textklassen*.
2. Zuweisen der Textklasse zu der Queue, in der sie für Tickets zur Verfügung stehen soll (siehe Abschnitt Queue-Verwaltung).

Definieren einer Textklasse

Textklassen werden im entsprechenden Navigationselement der Navigationsgruppe *Globale Konfiguration* im Admin Tool konfiguriert (siehe folgende Abbildung).

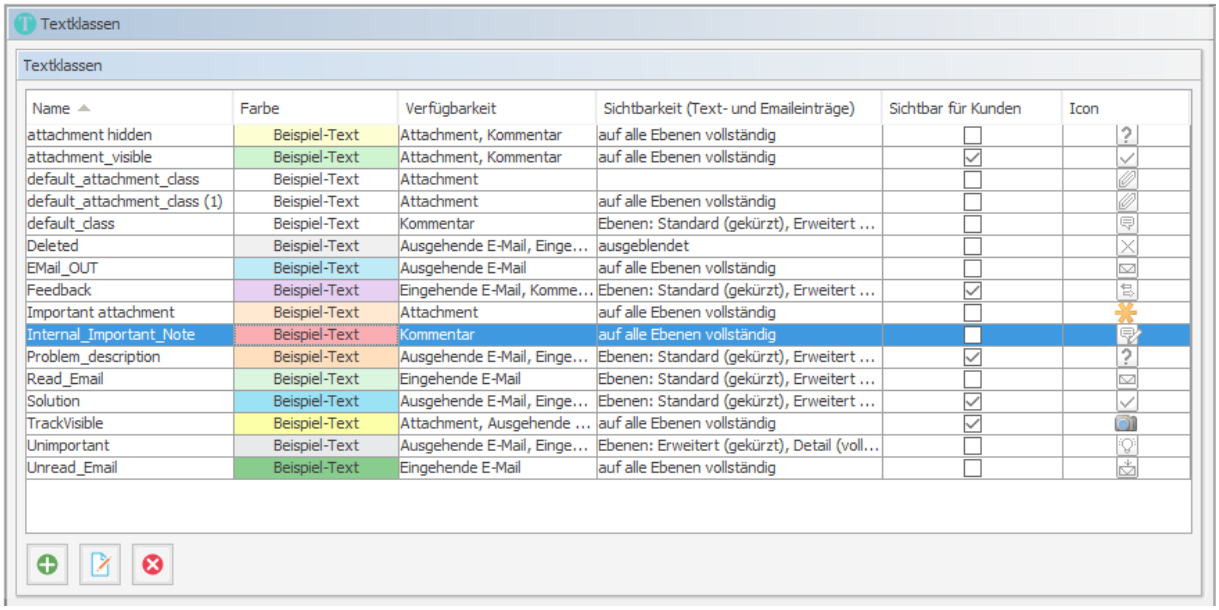


Abbildung 367: *ConSol CM Admin Tool - Globale Konfiguration, Textklassen*

Sie können eine neue Textklasse definieren, indem Sie unter der Liste auf den Button *Hinzufügen* klicken. Es wird folgendes Pop-up-Fenster geöffnet:

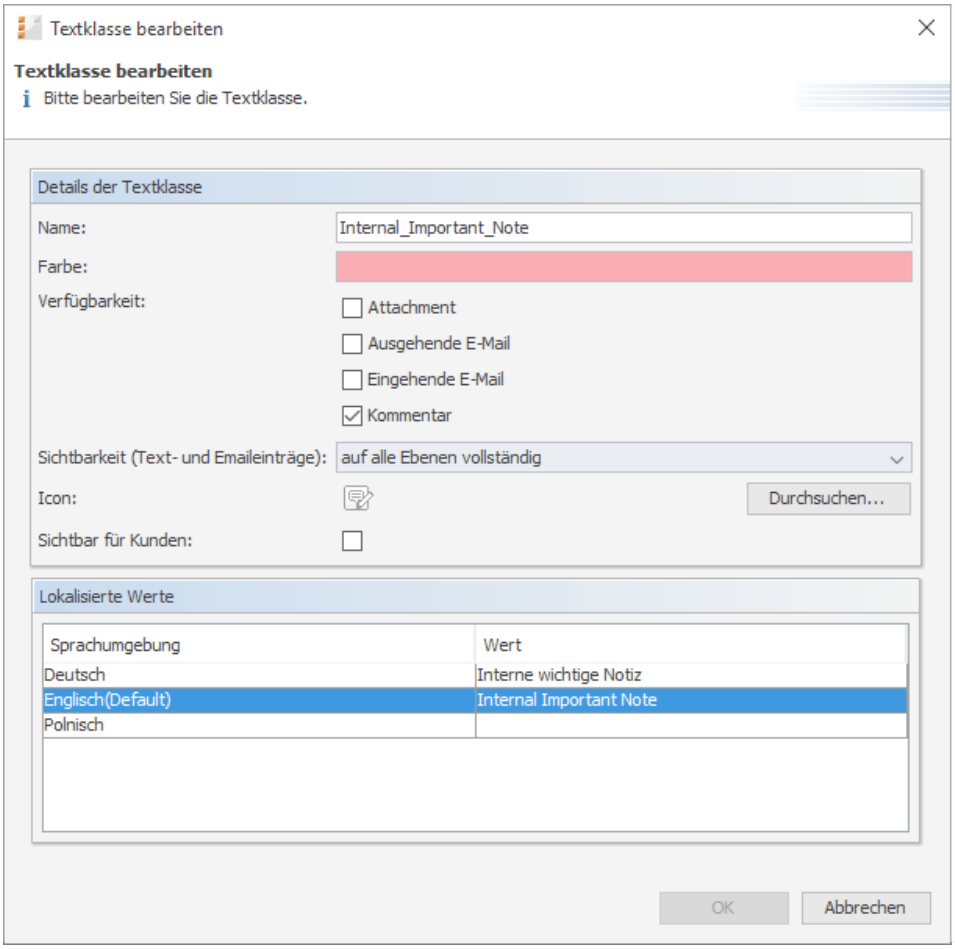


Abbildung 368: *ConSol CM Admin Tool - Globale Konfiguration, Textklassen: Neue Textklasse*

Hier müssen Sie die Details der Klasse definieren:

- **Name**
  Geben Sie einen Namen für die neue Textklasse ein. Der Name muss eindeutig sein.
- **Farbe**
  Wenn Sie in das Feld *Farbe* klicken, wird ein Pop-up-Fenster geöffnet. Es enthält eine Reihe an Farben, aus denen Sie die gewünschte Farbe für die Textklasse auswählen können. Sie können die ausgewählte Farbe im *Vorschau*-Bereich prüfen. Klicken Sie auf *OK*, um Ihre Auswahl zu speichern. Klicken Sie auf *Zurücksetzen*, um die zuletzt gespeicherte Farbe wiederherzustellen.



Abbildung 369: *ConSol CM Admin Tool - Globale Konfiguration, Textklassen: Auswählen einer Farbe*

- **Verfügbarkeit**
  Hier können Sie auswählen, für welche Ticket-Informationen die Textklasse verfügbar sein soll. Markieren Sie eine oder mehrere der folgenden Optionen:
  - Attachment
  - Kommentar
  - Eingehende E-Mail
  - Ausgehende E-Mail
- **Sichtbarkeit**
  Setzt das Sichtbarkeitslevel für Kommentare und E-Mails, wird nicht auf Attachments angewendet. Siehe auch Allgemeine Informationen über die Sichtbarkeit von Ticketprotokolleinträgen im Web Client.

  Es gibt im Web Client drei Levels für das Ticketprotokoll:

  - Standard (1. Level)
  - Erweitert (2. Level)
  - Detail (3. Level)

  Die Konfiguration einer Textklasse für ein oder mehrere dieser Levels bestimmt, ob ein mit dieser Textklasse gekennzeichneter Texteintrag auf dem entsprechenden Level angezeigt wird.

  Die Zusätze *vollständig* und *gekürzt* geben an, ob die Einträge *innerhalb* des ausgewählten Sichtbarkeitslevels in ganzer Länge (*vollständig*) angezeigt werden, oder ob sie nach einer bestimmten Anzahl an Zeichen abgeschnitten werden (*gekürzt*):

- **gekürzt** - abgeschnitten
- **vollständig** - volle Länge

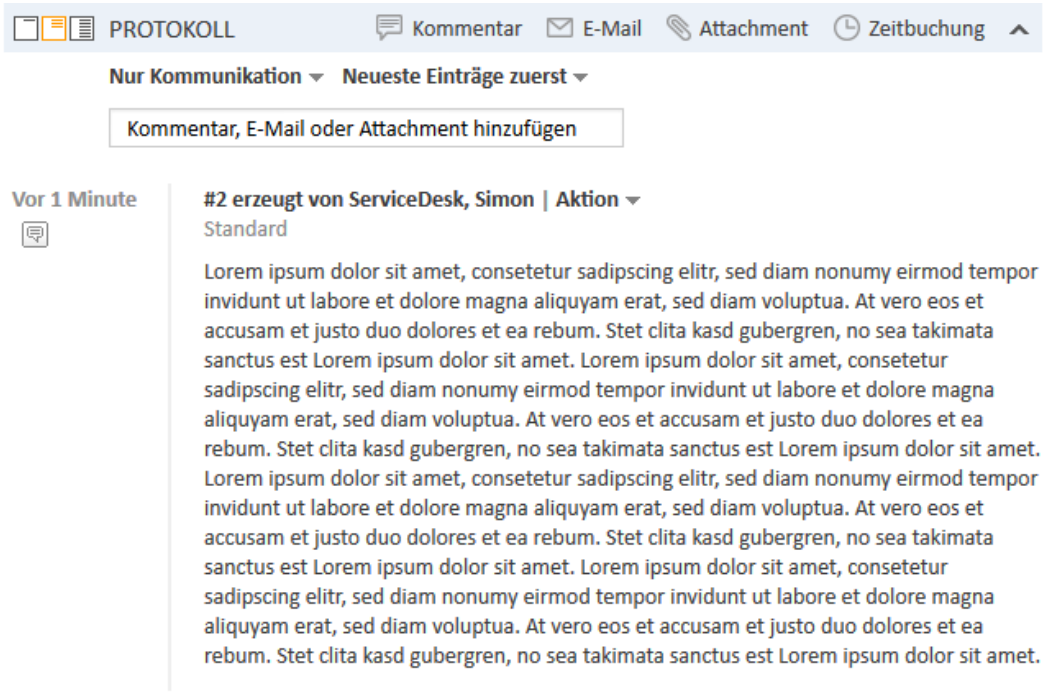


Abbildung 370: *ConSol CM Web Client - Beispiel: Sichtbarkeitslevel "Erweitert" (2. Level) zeigt Einträge in voller Länge ("vollständig")*

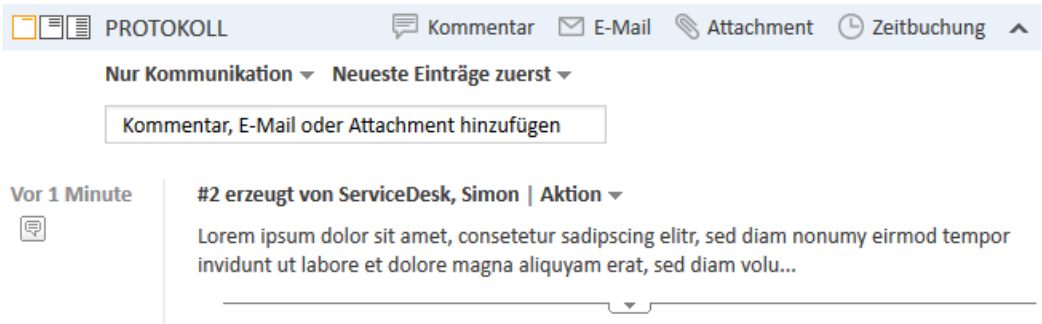


Abbildung 371: *ConSol CM Web Client - Beispiel: Sichtbarkeitslevel "Standard" (1. Level) zeigt Einträge gekürzt ("gekürzt")*

Wählen Sie im Drop-down-Menü auf welchen Levels die markierten Ticket-Informationen sichtbar sein sollen (siehe Abbildung unten).

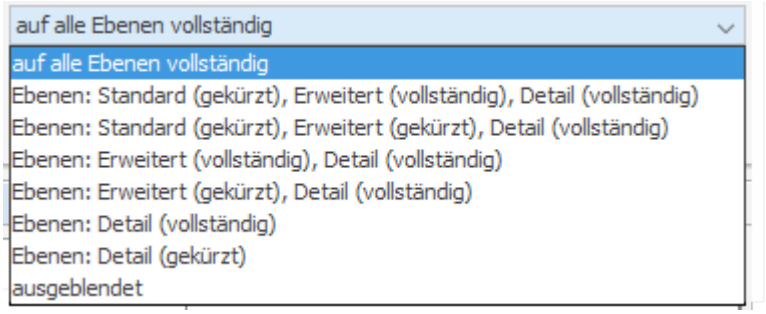


Abbildung 372: *ConSol CM Admin Tool - Globale Konfiguration, Textklassen: Auswählen eines Sichtbarkeitslevels*

Wenn Sie *hidden* auswählen, ist der mit dieser Textklasse gekennzeichnete Protokolleintrag im Ticketprotokoll nicht sichtbar. Eine detaillierte Beschreibung der Sichtbarkeitslevels und des Anzeigemodus für Einträge im Ticketprotokoll finden Sie im Abschnitt Allgemeine Informationen über die Sichtbarkeit von Ticketprotokolleinträgen im Web Client.

- **Icon**
  Wenn Sie in das Feld neben *Icon* klicken, wird eine Auswahl der Standardicons vom ConSol CM angezeigt. Wählen Sie eines dieser Icons für die neue Textklasse oder laden Sie ein eigenes Icon hoch, indem Sie auf den Button *Durchsuchen...* klicken. Ein Icon für eine Textklasse muss die Größe 16 (Höhe) * 24 (Breite) px haben, damit es im Web Client im entsprechenden Drop-down-Menü richtig platziert und ausgerichtet ist.



Abbildung 373: *ConSol CM Admin Tool - Globale Konfiguration, Textklassen: Auswählen eines Icons*

- **Sichtbar für Kunden**
  Markieren Sie diese Checkbox, wenn die mit dieser Textklasse gekennzeichneten Ticket-Informationen für Kunden in CM/Track, dem ConSol CM-Kundenportal, sichtbar sein sollen.
- **Lokalisierte Werte**
  Sie können den Namen der Textklasse lokalisieren. Geben Sie im Feld *Wert* in jeder zusätzlichen Sprache den Namen der Textklasse ein. Eine detaillierte Beschreibung des Lokalisierungsmechanismus finden Sie im Abschnitt Lokalisierung von Objekten allgemein, Typ 2.

Klicken Sie auf *OK*, um die Details der neuen Textklasse zu speichern und das Fenster zu schließen.

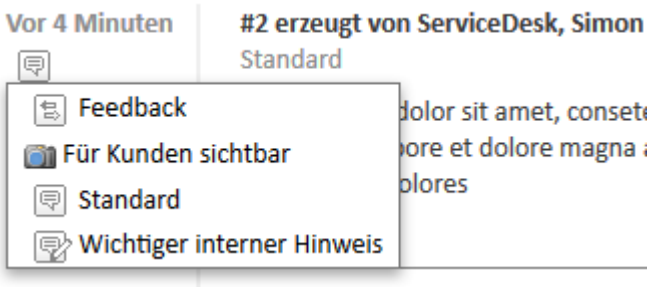


Abbildung 374: *ConSol CM Web Client - Kontextmenü für Textklassen im Ticketprotokollbereich*

Zuweisen von Textklassen zu einer Queue

Die Textklasse ist im Web Client für Tickets einer Queue erst verfügbar, wenn sie der Queue in der Queue-Verwaltung zugewiesen wurde.

E.6.2.2 Editieren einer Textklasse

Wenn Sie eine Textklasse editieren möchten, wählen Sie sie in der Liste aus und klicken Sie auf den Button *Bearbeiten*. Es wird dasselbe Fenster geöffnet wie für die Erstellung einer Textklasse. Sie können alle Details ändern und Ihre Änderungen durch Klicken auf *OK* speichern.

E.6.2.3 Löschen einer Textklasse

Sie können eine Textklasse nur löschen, wenn sie in keinem Ticket verwendet wird und keiner Queue zugewiesen ist. Um eine Textklasse zu löschen, wählen Sie sie in der Liste aus und klicken Sie auf den Button *Löschen*. Wenn Sie das daraufhin angezeigte Fenster mit *Ja* bestätigen, wird die Textklasse aus der Liste und aus dem System gelöscht.

E.6.2.4 Festlegen der Standardtextklasse

Verwenden Sie zum Definieren der Standardtextklasse die System-Property cmweb-server-adapter, defaultContentEntryClassName (siehe System-Properties). Die Standardtextklasse wird auf alle Ticketeinträge angewendet, die nicht explizit mit einer anderen Textklasse gekennzeichnet wurden.

Die Anzeige von normalen Kommentaren im Ticket (d. h. den Kommentaren mit der Textklasse *default_class*) hängt von der Sichtbarkeit dieser Textklasse ab, siehe Abschnitt Sichtbarkeit.

E.6.2.5 Arbeiten mit Textklassen in Skripten

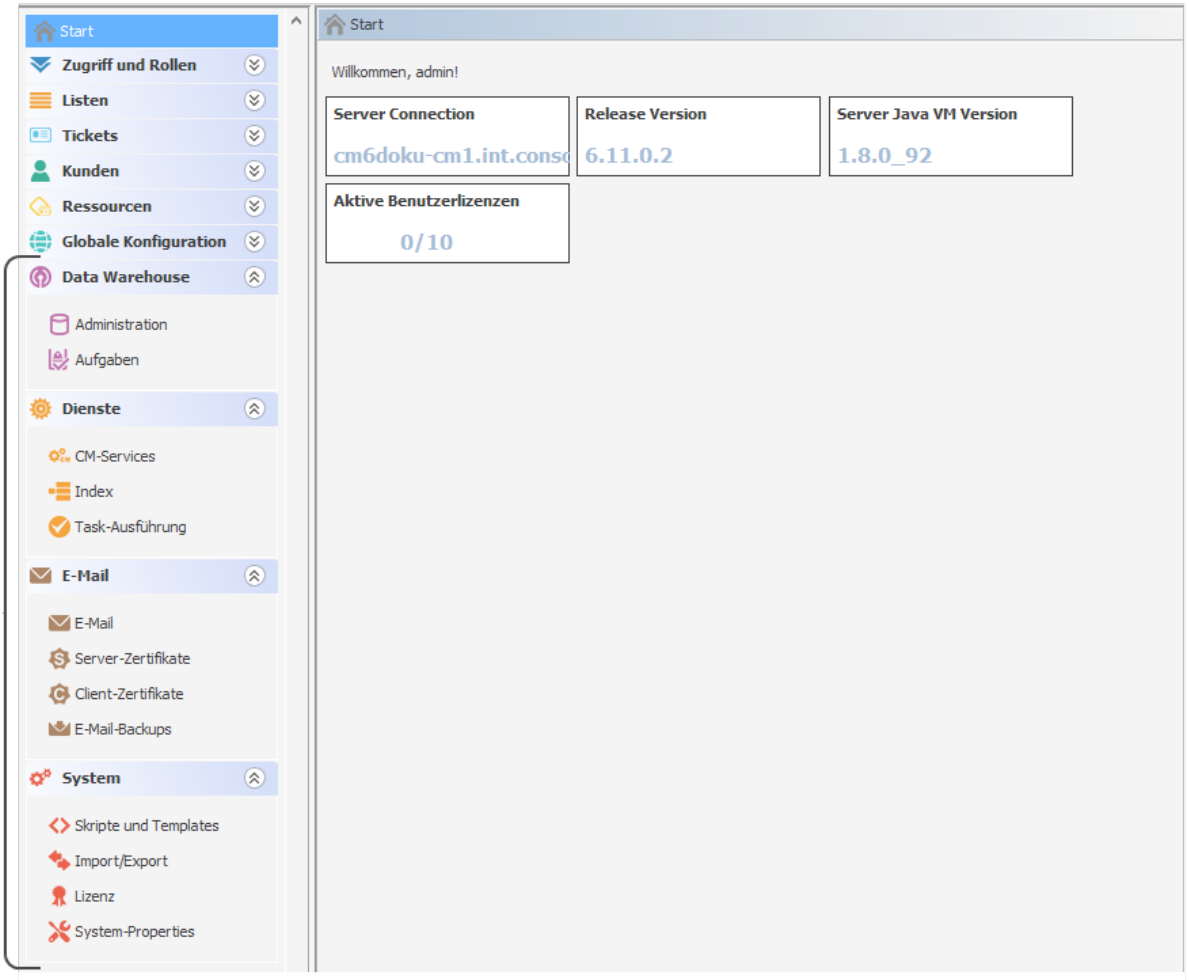
Sie können in Skripten, die in Workflow-Aktivitäten verwendet werden oder *Skriptbereich* des Admin Tools gespeichert sind, mit Textklassen arbeiten. Details zur Programmierung finden Sie im *ConSol CM Process Designer Handbuch*.

In Skripten können Sie einem Texteintrag eine Textklasse zuweisen, auch wenn diese der Queue nicht zugewiesen ist! Dies kann bei der Automatisierung von bestimmten Prozessen helfen.

E.6.2.6 Übertragen von Textklassen an das DWH

Alle Textklassen (Standardklassen und spezielle Klassen), die in Protokolleinträgen (Kommentaren, E-Mails) verwendet werden, werden von ConSol CM automatisch in allen Übertragungsmodi (LIVE und ADMIN) an das Data Warehouse (DWH) übertragen. Für diese Funktion ist keine Konfiguration erforderlich, die Installation wird automatisch entsprechend aktualisiert. Es sind allerdings eine neue Initialisierung und eine neue initiale Übertragung erforderlich.

F - Abschnitt Experten



In diesem Abschnitt erfahren Sie, wie einige erweiterte Einstellungen in ConSol CM konfiguriert werden.

Es werden folgende Themen behandelt:

- Ticketverwaltung: Hier erfahren Sie, wie man Tickets löscht und wieder öffnet.
- Verwaltung des Data Warehouse (DWH): Hier erfahren Sie, wie man ConSol CM für die Initialisierung und Arbeit mit dem Data Warehouse, der Grundlage für das ConSol CM-Reporting, konfiguriert.
- CM-Services: Hier erfahren Sie, welche Dienste innerhalb der ConSol CM-Applikation ausgeführt werden.
- Suche in ConSol CM: Hier erfahren Sie alles über die Suche in ConSol CM, z. B. wie man den Indexer verwaltet oder wie man Suche-Aktionen für das Action Framework implementiert.

- Das Task Execution Framework (TEF): Hier erfahren Sie, was das TEF ist und wie man es im ConSol CM-System konfiguriert.
- E-Mail-Konfiguration: Hier erfahren Sie alles über die E-Mail-Funktion in ConSol CM und wie man die entsprechenden Module konfiguriert.
- Verwaltung von Admin-Tool-Skripten und -Templates: Hier erfahren Sie, welche Arten von Skripten und Templates im Admin Tool verwaltet werden, und wie Sie diese selber schreiben können.
- Deployment (Import/Export): Hier erfahren Sie alles über die ConSol CM-Szenarien, die für den Export und Import des ConSol CM-Systems verwendet werden.
- Lizenzverwaltung: Hier erfahren Sie, wie man eine neue Lizenz im ConSol CM-System installiert.
- System-Properties: Hier erfahren Sie, was System-Properties sind und wie man sie zur Konfiguration von grundlegenden Einstellungen von ConSol CM verwendet.
- Arbeiten mit Textvorlagen: Hier erfahren Sie alles über Text- und Dokumentvorlagen, die als Grundbestandteil des ConSol CM-Systems genutzt werden können.
- Zeitbuchung mit ConSol CM: Hier erfahren Sie, wie man das ConSol CM-System für die Nutzung der Zeitbuchungsfunktion vorbereitet.
- Authentifizierungsmethoden in ConSol CM: Hier lernen Sie die unterschiedlichen Konfigurationsmöglichkeiten für die Authentifizierung der Bearbeiter im Web Client kennen.
- Systemarchitektur: Hier erhalten Sie einen Überblick über die Systemarchitektur von ConSol CM. Weitere Informationen über dieses Thema finden Sie im *ConSol CM Set-Up Manual* und *ConSol CM Operations Manual*.

F.1 Ticketverwaltung

In diesem Kapitel werden folgende Themen behandelt:

F.1.1 Einführung in die Ticketverwaltung 567

F.1.2 Ticketverwaltung mit dem Admin Tool 567

F.1.1 Einführung in die Ticketverwaltung

In der Ticketverwaltung können Sie:

- **Tickets löschen**
  z. B. wenn ein Ticket versehentlich erstellt wurde.
- **Tickets wieder öffnen**
  z. B. wenn ein Ticket zu schnell geschlossen wurde.

⚠ Bedenken Sie dabei, dass ein Ticket, das wieder geöffnet wird, den Prozess am Startknoten des entsprechenden Workflows beginnt. Wenn ein Ticket Knoten durchläuft, die Events auslösen, die nur einmal durchgeführt werden sollen (z. B. das Ticket wird an einen Genehmiger übergeben), ist es möglicherweise besser ein neues Ticket zu erstellen. Als Alternative können Sie den Workflow so konfigurieren, dass er eine "Abkürzung" enthält, mit der bestimmte Schritte umgangen werden, damit sie für das Ticket nicht mehrmals ausgeführt werden.

F.1.2 Ticketverwaltung mit dem Admin Tool

Im Admin Tool können Sie Tickets im Navigationselement *Verwaltung* der Navigationsgruppe *Tickets* verwalten.

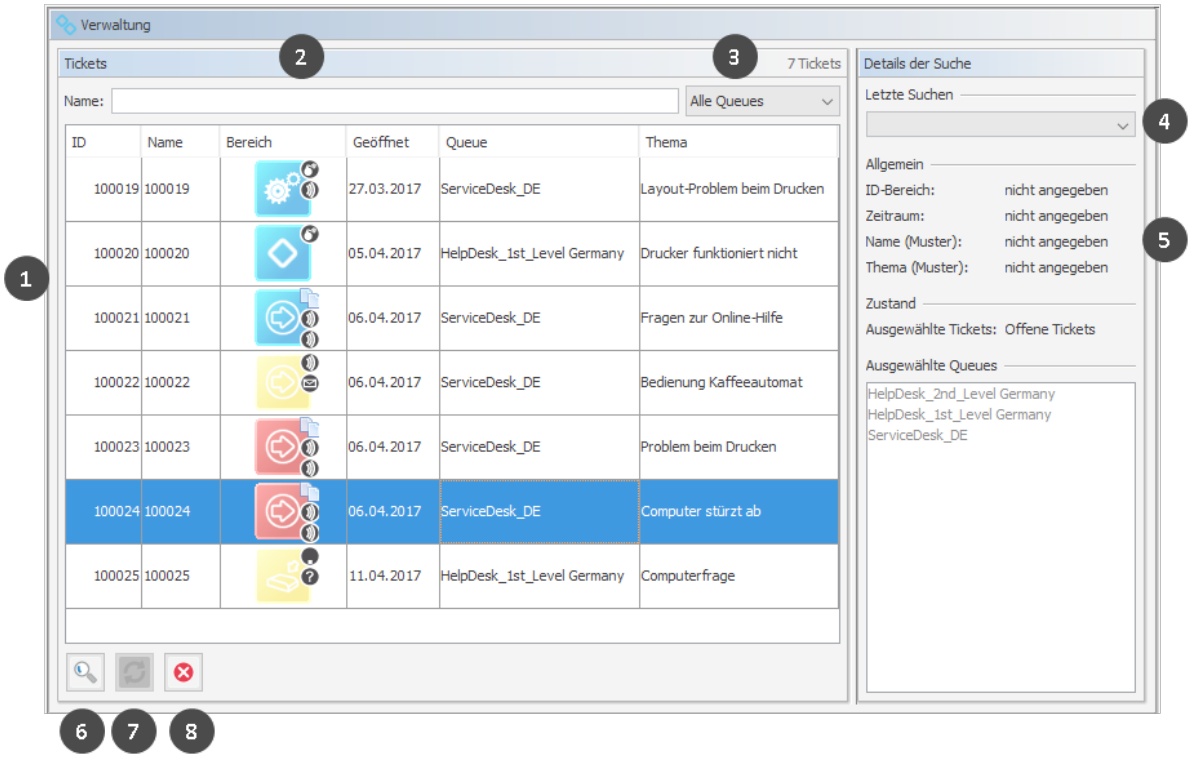


Abbildung 375: *ConSol CM Admin Tool - Tickets, Verwaltung: Liste der Tickets nach der Suche*

Die Seite der Ticketverwaltung umfasst folgende Elemente:

- Liste aller Tickets, die den Suchkriterien entsprechen (1)
- Tickets nach Namen filtern (2)
- Tickets nach Queue filtern (3)
- Liste aller zuletzt ausgeführten Suchen (4)
- Details der aktuellen Suche (5)
- Button zum Definieren der Suchkriterien zum Starten einer neuen Suche (6)
- Button zum Wiedereröffnen der ausgewählten Tickets (7)
- Button zum Löschen der ausgewählten Tickets (8) - nur verfügbar, wenn im CM-System aktiviert

F.1.2.1 Löschen oder Wiedereröffnen von Tickets

Für diese Vorgänge können Sie entweder die Buttons unter der Liste oder das Kontextmenü verwenden.

1. **Buttons:**
   Wählen Sie die gewünschten Tickets in der Liste aus und klicken Sie auf den Button *Löschen*, um Tickets zu löschen, bzw. auf den Button *Wieder öffnen*, um Tickets wieder zu öffnen. Wenn

Sie das daraufhin angezeigte Fenster mit *Ja* bestätigen, wird die entsprechende Aktion ausgeführt.

2. **Kontextmenü:**
   Wählen Sie die gewünschten Tickets in der Liste und öffnen Sie das Kontextmenü durch Klicken der rechten Maustaste. Wählen Sie den gewünschten Vorgang.

F.1.2.2 Ausschalten der Löschfunktion mit einer System-Property

Die Löschfunktion kann mit der System-Property cmas-app-admin-tool, delete.ticket.enabled deaktiviert werden. Diese Property ist ein Boolean-Wert. Wenn sie auf „false“ gesetzt ist, wird der Button *Löschen* nicht mehr angezeigt und die Löschfunktion ist im Kontextmenü nicht mehr verfügbar.

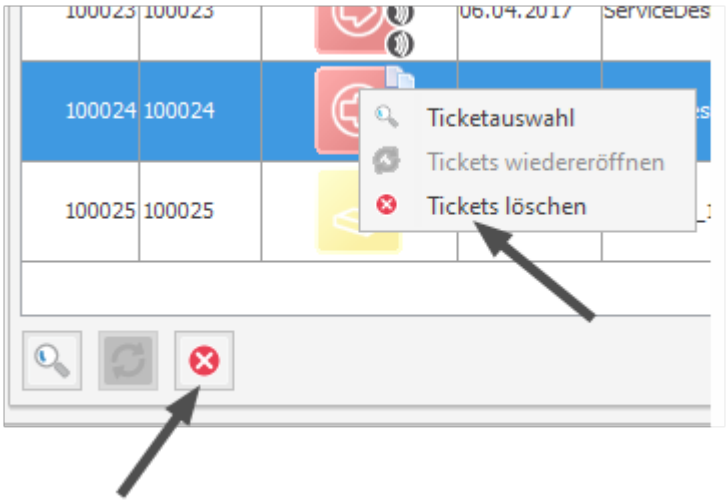


Abbildung 376: *ConSol CM Admin Tool - Tickets, Verwaltung: cmas-app-admin-tool, delete.ticket.enabled = true*

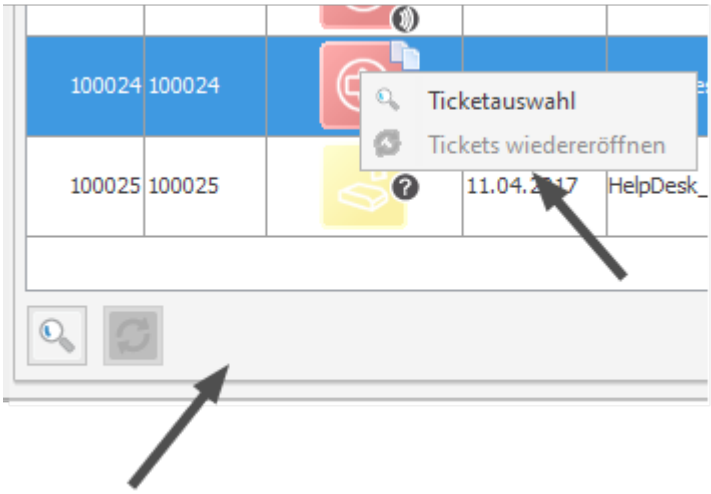


Abbildung 377: *ConSol CM Admin Tool - Tickets, Verwaltung: cmas-app-admin-tool, delete.ticket.enabled = false*

F.1.2.3 Suchen von Tickets

Zum Suchen der Tickets, die gelöscht oder wieder geöffnet werden sollen, klicken Sie in der linken unteren Ecke der Seite auf den Button *Suchen* oder verwenden Sie das Kontextmenü. Es wird ein Pop-up-Fenster angezeigt, in dem Sie die Suchkriterien eingeben können.

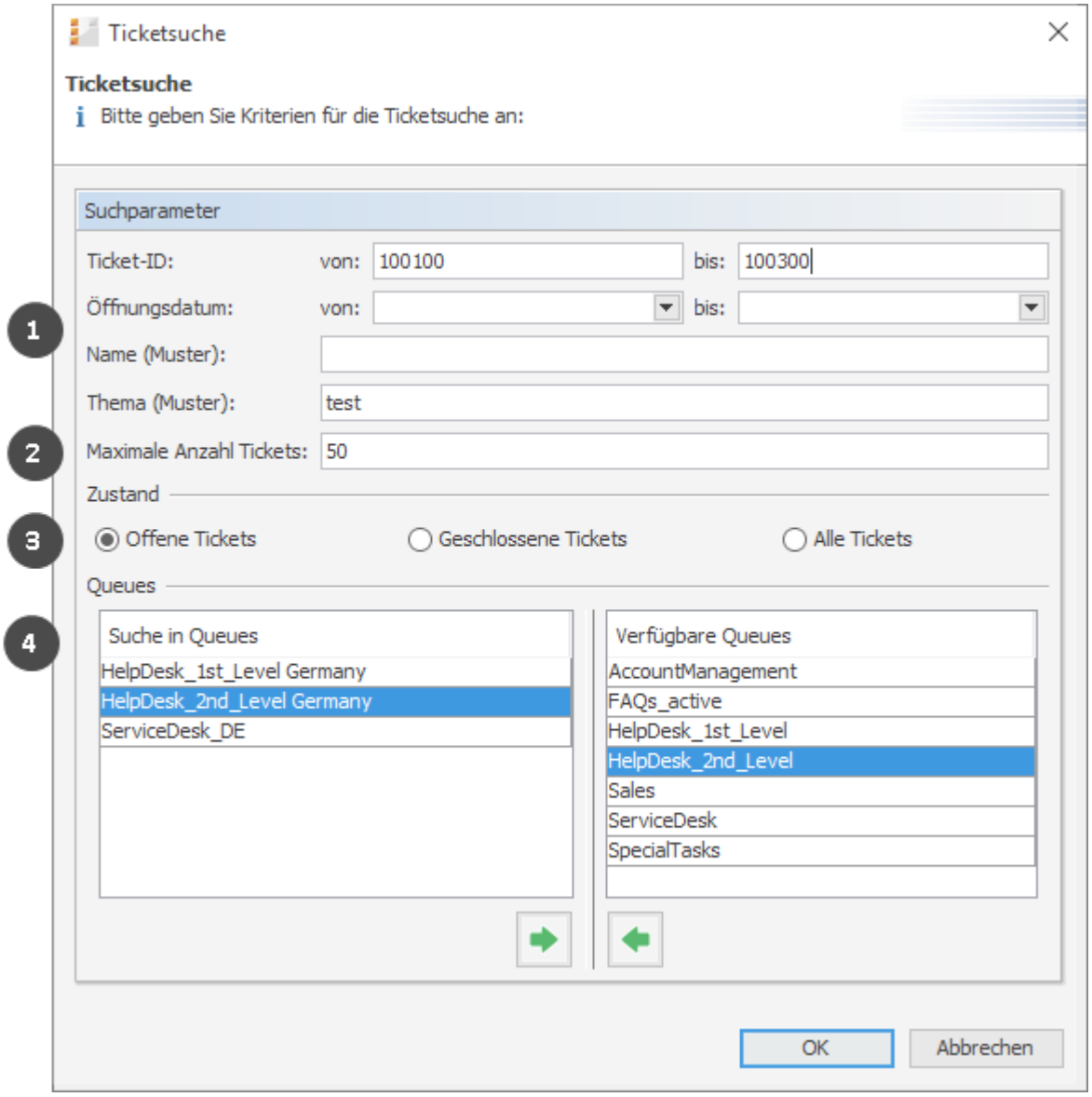


Abbildung 378: *ConSol CM Admin Tool - Tickets, Verwaltung: Ticketsuche*

Für die Suche können folgende Parameter verwendet werden:

- **Suchkriterien (1)**
  - **Ticket-ID**
    Hier können Sie einen Wertebereich für die Ticket-ID eingeben.
  - **Öffnungsdatum**
    Über den Kalender können Sie die Suche auf Tickets beschränken, die innerhalb eines bestimmten Zeitraums geöffnet wurden.
  - **Name (Muster)**

Hier können Sie Schlüsselwörter oder ein Suchmuster für den Ticketnamen eingeben.

- **Thema (Muster)**
  Hier können Sie Schlüsselwörter oder ein Suchmuster für das Ticketthema eingeben.

- **Maximale Anzahl Tickets (2)**
  Hier können Sie die maximale Anzahl der Tickets eingeben, die in der Liste angezeigt werden sollen.
- **Zustand (3)**
  Mithilfe der Radio-Buttons können Sie festlegen, ob Sie nach *offenen*, *geschlossenen* oder *allen* Tickets suchen möchten.
- **Queues (4)**
  Die Liste auf der rechten Seite zeigt die verfügbaren Queues. Wählen Sie die Queues aus, die durchsucht werden sollen, und klicken Sie auf den Button *Zuweisen*, um sie in die Suchliste auf der linken Seite zu verschieben. Wenn Sie keine Queues auswählen, werden alle verfügbaren Queues durchsucht.



Bitte beachten Sie den Unterschied zwischen der Ticket-ID und dem Ticketnamen. Die Ticket-ID ist eine interne ID in der CM-Datenbank, die im Normalfall nicht in der Benutzeroberfläche angezeigt wird. Der Ticketname wird unter dem Ticket-Icon angezeigt und wird - im alltäglichen Arbeitsleben - oft als Ticket-ID bezeichnet, obwohl dies technisch gesehen nicht korrekt ist. Der Ticketname kann ein Präfix (z. B. SUP) enthalten, je nach queue-spezifischer Konfiguration. Die Ticket-ID enthält nie ein Präfix!

Die obigen Abbildungen zeigen ein Beispiel:

Die Ticket-ID ist *100134*, der Ticketname ist *SUP-61*. Um eine Suche nach diesem Ticket zu starten, können Sie beispielsweise entweder als Suchkriterium verwenden: *Ticket-ID* ist zwischen *10.000* und *200.000*, oder Sie suchen nach dem Namen (Muster) mit dem Kriterium *Name (Muster)* ist *SUP-*1**.

Klicken Sie auf *OK*, um die Suche zu starten. Das Ergebnis wird auf der Seite *Ticketverwaltung* angezeigt. Wenn die Liste zu lang ist, können Sie die Anzeige mit dem Namen- und Queue-Filter über der Liste beschränken.

Rechts neben der Ticketliste finden Sie eine Übersicht der angewendeten Suchkriterien. Das Listenfeld *Letzte Suchen* über diesem Bereich enthält Ihre letzten Suchen. Wenn Sie auf einen Listeneintrag klicken, wird ein Pop-up-Fenster mit den Kriterien der ausgewählten Suche geöffnet. Sie können diese Suche dort ändern oder einfach neu starten.

F.2 Verwaltung des Data Warehouse (DWH)

In diesem Kapitel werden folgende Themen behandelt:

F.2.1 Einleitung 574
F.2.2 DWH-Verwaltung mit dem Admin Tool 576
F.2.3 System-Properties für das DWH 590
F.2.4 Übertragungsmodus 590
F.2.5 DWH-Informationen für Experten 592

⚠ Zum Einrichten eines DWH wird eine laufende Instanz von ConSol CM Reporting Framework (CMRF) benötigt. Wenn Ihr System noch kein CMRF beinhaltet, wenden Sie sich bitte an Ihren ConSol CM-Consultant oder an ConSol Software.

F.2.1 Einleitung

F.2.1.1 Data Warehouse

Ein Data Warehouse (auch ***DWH*** genannt) ist eine Sammlung an Daten auf einem oder mehreren Systemen und/oder Datenbanken, auf deren Grundlage das Reporting und die Datenanalyse stattfinden. Die importierten Daten müssen dazu häufig kombiniert oder neu angeordnet (integriert) werden, damit eine optimale Nutzung für das Reporting und die Datenanalyse möglich ist. Eine detailliertere Einführung in das Funktionsprinzip des Data Warehouse und das ConSol CM Data Warehouse finden Sie im *ConSol CM DWH-Handbuch*.

F.2.1.2 ConSol CM Data Warehouse und ConSol CM Reporting Framework

Eine Standardinstallation von ConSol CM enthält alle Module, die für den Aufbau eines CM Data Warehouse benötigt werden. Eine der beiden Kernkomponenten ist das ***ConSol CM Reporting Framework (CMRF)***.

Dabei handelt es sich um eine Java EE-Applikation, die die Daten zwischen der ConSol CM-Datenbank und der DWH-Datenbank synchronisiert. Die folgende Abbildung zeigt eine Übersicht der Systemarchitektur einer typischen DWH- und CMRF-Installation. Wir empfehlen Ihnen die Verwendung von zwei Servern für die ConSol CM- und CMRF-Instanzen. Informationen über die unterstützten Application Server und RDBMS finden Sie in den aktuellen *Systemanforderungen*.

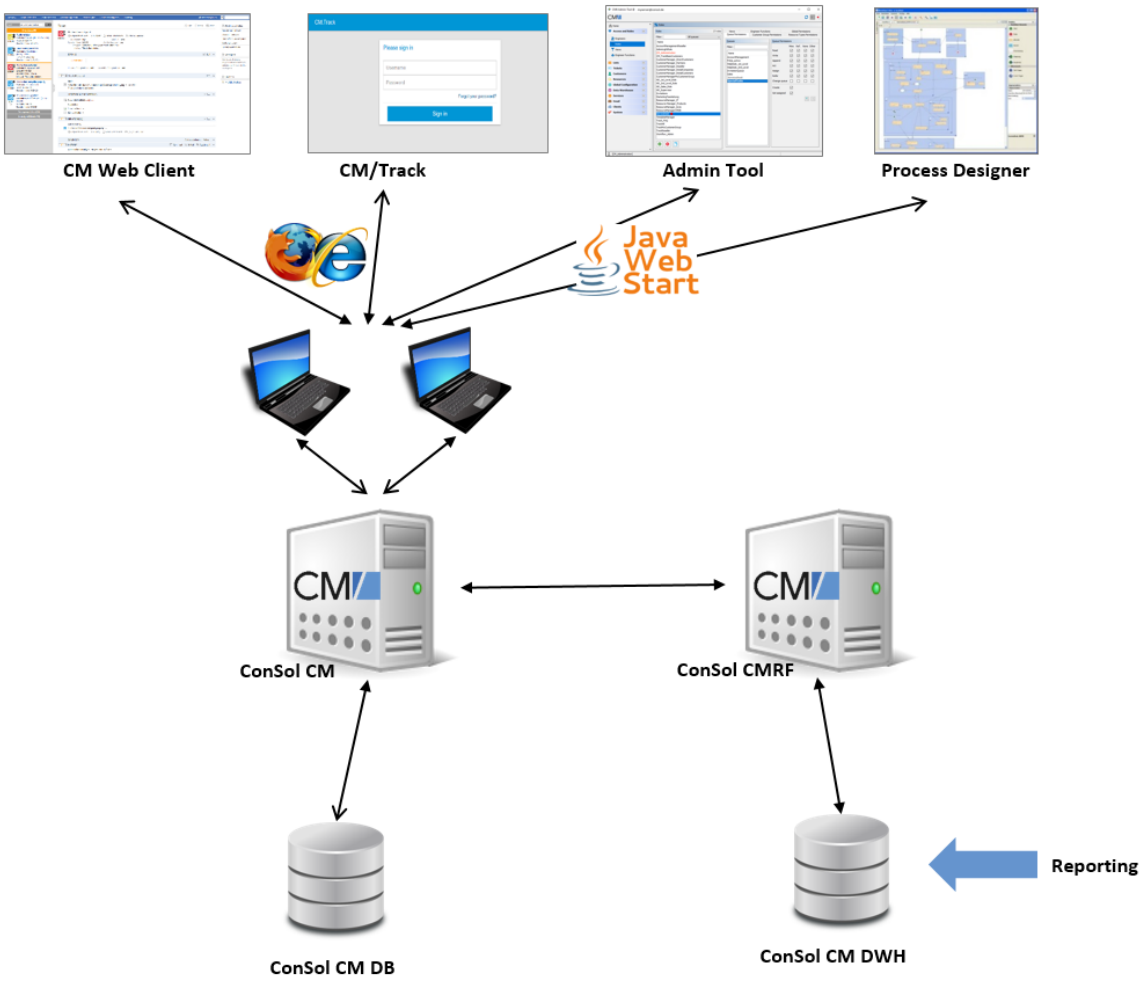


Abbildung 379: *ConSol CM - Systemarchitektur mit DWH und CMRF (2 Server)*

Es gibt zwei verschiedene Synchronisierungsmodi für die Übertragung von Daten von ConSol CM an die DWH-Datenbank:

- **LIVE-Modus**
  In diesem Modus wird jede Änderung an der ConSol CM-Datenbank sofort mit dem DWH synchronisiert.
- **ADMIN-Modus**
  In diesem Modus muss der Administrator die Synchronisierung manuell anstoßen.

⚠ Es werden nur die Daten aus Ticketfeldern, Kundenfeldern und Ressourcenfeldern, die die Annotation `reportable` = „true“ haben, mit dem DWH synchronisiert. Zusätzlich werden alle Daten übertragen, die standardmäßig an das DWH übertragen werden. Eine detaillierte Erklärung dazu finden Sie im *ConSol CM DWH-Handbuch*.

F.2.2 DWH-Verwaltung mit dem Admin Tool

Die DWH-Verwaltung findet in den Navigationselementen der Navigationsgruppe *Data Warehouse* statt:

- Administration
- Aufgaben

F.2.2.1 Administration

Öffnen Sie zur Administration aller DWH-Operationen die Navigationsgruppe *Data Warehouse*, Navigationselement *Administration*.

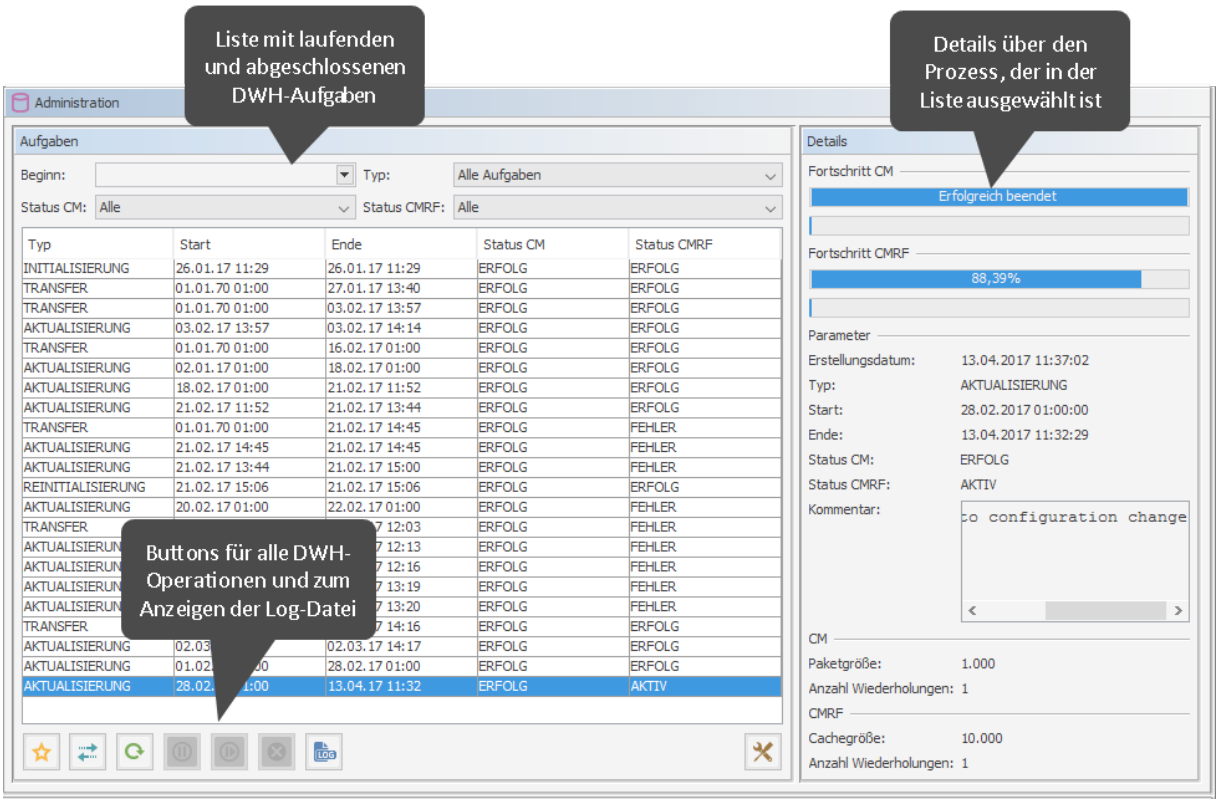


Abbildung 380: *ConSol CM Admin Tool - DWH, Aufgaben: Konfiguration und Logs*

In der Tabelle *Aufgaben* auf der linken Seite wird eine Liste aller DWH-Operationen angezeigt, die ausgeführt wurden oder noch ausgeführt werden.

Sie können die Liste mithilfe von einem der Filter über der Liste filtern. Die Filterkriterien werden mittels UND kombiniert.

- **Beginn**
  Öffnet ein Feld zur Datumsauswahl. Es werden nur die DWH-Aufgaben angezeigt, die nach dem ausgewählten Datum gestartet wurden.

- **Typ**
  Es werden nur die DWH-Aufgaben des ausgewählten Typs angezeigt.
  - **Alle Aufgaben**
    Alle verfügbaren Aufgaben
  - **Abgeschlossene**
    Abgeschlossene Aufgaben. Der CM- und CMRF-Status ist ERFOLG oder FEHLER.
  - **Aktuell ausgeführte**
    Aufgaben, die gerade ausgeführt werden. Der CM- bzw. CMRF-Status ist AKTIV.
  - **Anstehende**
    Aufgaben, die konfiguriert, gespeichert und in die Liste aufgenommen wurden, aber noch nicht ausgeführt wurden. Der CM- bzw. CMRF-Status ist NEU.
- **Status CM**
  Es werden nur die DWH-Aufgaben angezeigt, die den ausgewählten CM-Status haben.
  - Alle
  - NEU - erster Status nach der Erstellung
  - AKTIV - Daten werden ans CMRF gesendet
  - PAUSE - aktiver Status wird gestoppt, bis der Button *Fortsetzen* geklickt wird
  - ERFOLG - Senden von Daten an CMRF erfolgreich abgeschlossen
  - FEHLER - Senden von Daten an CMRF nicht erfolgreich abgeschlossen
- **Status CMRF**
  Es werden nur die DWH-Aufgaben angezeigt, die den ausgewählten CMRF-Status haben.
  - Alle
  - NEU - erster Status nach der Erstellung
  - AKTIV - Daten werden im CMRF verarbeitet
  - PAUSE - aktiver Status wird gestoppt, bis der Button *Fortsetzen* geklickt wird
  - ERFOLG - Verarbeiten von Daten erfolgreich abgeschlossen
  - FEHLER - Verarbeiten von Daten nicht erfolgreich abgeschlossen



Das *ConSol CM Operations Manual* enthält im Abschnitt *CMDB / CMRF / Data Warehouse Synchronization Process* eine sehr ausführliche Beschreibung des ConSol CM-DWH-Übertragungsprozesses. Der folgende Abschnitt des vorliegenden Handbuchs enthält nur eine Kurzfassung.

Die Liste enthält Spalten und Werte, die aus der CM-Datenbanktabelle `cmas_dwh_synchronization` abgerufen werden.

- **Typ**
  - **AKTUALISIERUNG**
    Es werden neue Daten in ein bereits vorhandenes DWH übertragen.
  - **INITIALISIERUNG**
    Das DWH ist neu/leer. Beim Initialisierungsschritt wird die Datenbankstruktur gebaut.
  - **REINITIALISIERUNG**
    Eine neue Initialisierung eines bereits vorhandenen DWHs.
  - **TRANSFER**
    Initiale Datenübertragung nach der Einrichtung
- **Start**
  Das Startdatum der DWH-Aufgabe
- **Ende**
  Das Enddatum der DWH-Aufgabe
- **Status CM**
  Während der DWH-Aufgabe werden Daten von der CM-Datenbank ans DWH übertragen. Diese Spalte zeigt den Status des CM-Teils der Aufgabe an, d. h. den Status der Übertragung aus dem CM an die DWH-Steuerungstabellen. Eine Erklärung des Status finden Sie in der Liste oben.
- **Status CMRF**
  Während der DWH-Aufgabe werden Daten von der CM-Datenbank ans DWH übertragen. Diese Spalte zeigt den Status des CMRF-Teils der Aufgabe an. Eine Erklärung des Status finden Sie in der Liste oben.

Unter der Liste gibt es Buttons für alle möglichen DWH-Aufgaben:

- **Initialisieren**
  - Fall a) Erstellen der Tabellen bei der DWH-Einrichtung (siehe Initialisierung des DWH (Initialisieren))
  - Fall b) Wenn das DWH bereits vorhanden ist, können Sie hier die **Neuinitialisierung** starten. Aktivieren Sie dazu die Option *Vorhandene Daten löschen*. Die Datenbank wird komplett neu aufgebaut.
- **Übertragen**
  Startet die initiale Datenübertragung, siehe Erste DWH-Synchronisierung (Übertragen)
- **Aktualisieren**
  Überträgt neue/zusätzliche Daten ans DWH, siehe DWH-Synchronisierung während des Systembetriebs (Aktualisieren)
- **Anhalten**
  Hält eine gerade ausgeführte DWH-Aufgabe an
- **Fortsetzen**
  Setzt eine angehaltene DWH-Aufgabe fort

- **Löschen**
  Löscht eine noch nicht abgeschlossene Data Warehouse-Operation aus der Liste. Funktioniert für geplante noch nicht gestartete Aufgaben und für zuvor angehaltene Aufgaben. Der häufigste Anwendungsfall ist, wenn eine Initialisierungs-, eine Übertragungs- und eine Aktualisierungsaufgabe hintereinander ohne Wartezeit angestoßen werden.
- **Logs**
  Öffnet ein Popup, das die Datei `cmrf.log` anzeigt. Diese enthält die Einträge der Log-Datei, unabhängig von dem in der Liste ausgewählten Eintrag.
- **Konfiguration**
  Öffnet das Popup zur DWH-Konfiguration, siehe Grundlegende DWH-Konfiguration

> ⚠ **Wichtige Hintergrundinformationen über DWH-Operationen**
>
> Berücksichtigen Sie folgendes Verhalten von ConSol CM/CMRF bezüglich DWH-Operationen!
>
> Wenn Sie auf *Initialisieren*, *Übertragen* oder *Aktualisieren* klicken, wird eine Operation des jeweiligen Typs als Eintrag in der ConSol CM-Datenbanktabelle `cmas_dwh_synchronization` erzeugt. Bei jedem Klick auf einen Button wird eine neue Operation des entsprechenden Typs erzeugt. Die Operationen werden dann nacheinander ausgeführt. Dabei gilt das FIFO-Prinzip (first in, first out) nach dem Erzeugungsdatum! Sie können alle diese Aufgaben im Bereich *Administration* überwachen.

Rechts neben dem Bereich *Administration* befindet sich der Bereich *Details*, in dem alle Details der in der Liste ausgewählten Aufgabe aufgeführt sind.

- **Fortschritt CM**

  Fortschrittsbalken, die den Fortschritt der Übertragung der CM-Daten in die DWH-Steuerungstabellen anzeigen:

  - oberer Balken: der Gesamtfortschritt der Operation auf der Senderseite
  - unterer Balken: der Fortschritt der aktuellen Suboperation
- **Fortschritt CMRF**

  Fortschrittsbalken, die den Fortschritt der Aktualisierung der DWH-Datenbank durch das CMRF anzeigen:

  - oberer Balken: der Gesamtfortschritt der Operation auf der Empfängerseite
  - unterer Balken: der Fortschritt der aktuellen Suboperation
- **Parameter**
  Alle Parameter der ausgewählten DWH-Aufgabe
  - **Erstellungsdatum**
    Automatisch gesetzt, Zeitstempel der Erstellung der DWH-Aufgabe
  - **Typ**
    DWH-Aufgabentyp (INITIALISIERUNG | AKTUALISIERUNG usw.), siehe Liste oben

- **Start**
  Das Startdatum der DWH-Übertragungsaufgabe, kann vom Erstellungsdatum abweichen, wenn eine Aufgabe für einen zukünftigen Zeitpunkt geplant wurde.
- **Ende**
  Das Enddatum der DWH-Aufgabe. Nur für abgeschlossene Aufgaben.
- **Status CM**
  Siehe oben
- **Status CMRF**
  Siehe oben
- **Kommentar**
  Der Kommentar, der beim Definieren der DWH-Aufgabe gesetzt wurde, kann auch leer sein.

- **CM**
  - **Paketgröße**
    Anzahl der pro Transaktion vom CM an die DWH-Steuerungstabellen übertragenen Objekte
  - **Anzahl Wiederholungen**
    Anzahl der Wiederholungen, die beim Auftreten von Fehlern ausgeführt werden. Nachdem die Anzahl an Wiederholungen erreicht wurde, wird die Operation mit dem CM-Status FEHLER abgeschlossen.
- **CMRF**
  - **Cachegröße**
    Anzahl der pro Transaktion von den DWH-Steuerungstabellen an die DWH-Tabellen übertragenen Objekte
  - **Anzahl Wiederholungen**
    Anzahl der Wiederholungen, die beim Auftreten von Fehlern ausgeführt werden. Nachdem die Anzahl an Wiederholungen erreicht wurde, wird die Operation mit dem CMRF-Status FEHLER abgeschlossen.

Grundlegende DWH-Konfiguration

Bevor Sie ein ConSol CM-DWH einrichten können, müssen Sie die Datenbank (oder das Datenbankschema), das die DWH-Daten enthalten soll, vorbereiten. Der entsprechende Datenbankserver muss für den CMRF-Server verfügbar sein. Eine detaillierte Beschreibung dazu finden Sie im *ConSol CM Set-Up Manual*. Sobald die für das DWH bestimmte Datenbank (bzw. das Datenbankschema) vorliegt und das CMRF in Betrieb ist, können Sie mit den folgenden Schritten fortfahren.

Öffnen Sie zur Konfiguration des DWH-Synchronisierungsmodus und der Benachrichtigungen im Zusammenhang mit dem DWH die Navigationsgruppe *Data Warehouse* und das Navigationselement *Administration*. Klicken Sie auf den Button *Konfiguration* und geben Sie alle erforderlichen Werte ein.

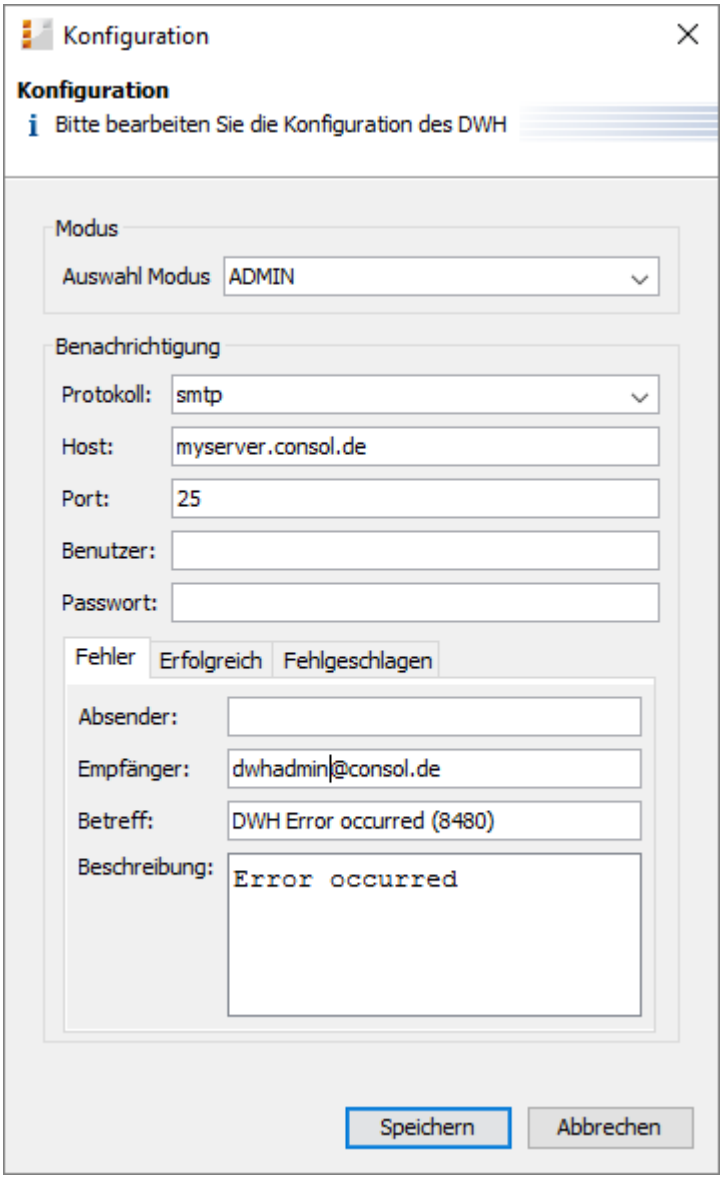


Abbildung 381: *ConSol CM Admin Tool - Data Warehouse, Administration: DWH Konfiguration*

- **Auswahl Modus**
  - **LIVE**
    In diesem Modus wird jede Änderung an der ConSol CM-Datenbank sofort mit dem DWH synchronisiert.
  - **ADMIN**
    In diesem Modus muss der Administrator die Synchronisierung manuell anstoßen.
  - **OFF**
    Es findet keine Datenübertragung an das DWH statt.

Sie können den aktuellen DWH-Modus auch der DWH-System-Property cmas-dwh-server, dwh.mode entnehmen (siehe System-Properties).

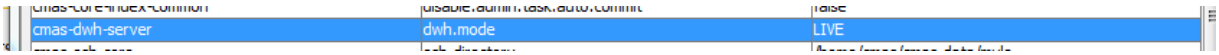


Abbildung 382: *ConSol CM Admin Tool - System-Property für DWH-Modus*

- **Benachrichtigung**
  Hier werden die Parameter für die E-Mails konfiguriert, die aufgrund von DWH-Ereignissen gesendet werden:
  - **Protokoll**
    Pflichtfeld - Das Protokoll, das beim Senden der Nachricht verwendet wird. Dies ist normalerweise SMTP.
  - **Host**
    Pflichtfeld - Der E-Mail-Server. Sie können einen (über DNS auflösbaren) Namen oder eine IP-Adresse eingeben.
  - **Port**
    Pflichtfeld - Der Port des E-Mail-Servers, den der Mail Daemon abhört.
  - **Benutzer**
    Optional - Benutzername, falls der E-Mail-Server eine Benutzerauthentifizierung erfordert.
  - **Passwort**
    Optional - Passwort des E-Mail-Benutzers, falls der E-Mail-Server eine Benutzerauthentifizierung erfordert.
  - Tabs **Fehler/Erfolgreich/Fehlgeschlagen**
    Hier können Sie die Parameter für die E-Mails konfigurieren, die das System im Zusammenhang mit dem DWH versendet. Es gibt drei Arten von Nachrichten: Nachrichten bei Fehlern, Nachrichten bei erfolgreichen Operationen und Nachrichten bei fehlgeschlagenen Operationen.
    - **Absender**
      Die E-Mail-Adresse des Absenders der Nachrichten (sie kann von der E-Mail-Adresse abweichen, die für E-Mails an Kunden oder Bearbeiter verwendet wird).
    - **Empfänger**
      Die E-Mail-Adresse des Empfängers der DWH-Nachrichten. Dies ist anfänglich die E-Mail-Adresse des ConSol CM-Administrators (CM-System-Property cmas-core-security, admin.email).
    - **Betreff**
      Der Betreff der E-Mail für die einzelnen Nachrichtentypen (Fehler/Erfolgreich/Fehlgeschlagen).
    - **Beschreibung**
      Der Text der Nachricht.

Initialisierung des DWH (Initialisieren)

Nachdem die grundlegende Konfiguration vorgenommen wurde, kann mit der DWH-Initialisierung begonnen werden. Klicken Sie auf *Initialisieren*. Es wird eine neue DWH-Aufgabe des Typs INITIALISIERUNG zur Liste hinzugefügt.

In diesem Schritt wird die Datenbankstruktur des DWH mit allen Tabellen und Relationen erstellt. Es werden noch keine Daten übertragen.

Wenn das DWH schon einmal in Betrieb war und neu eingerichtet wird, muss eine Reinitialisierung durchgeführt werden. Markieren Sie dazu die Option *Vorhandene Daten löschen*, um die alte Datenbankstruktur zu löschen und eine neue Datenbankstruktur anzulegen.

Erste DWH-Synchronisierung (Übertragen)

Um das Data Warehouse zum ersten Mal mit ConSol CM-Daten zu füllen, klicken Sie auf *Übertragen*. Damit wird die initiale Übertragung gestartet. Je nach Anzahl der Tabellen kann dies eine Weile (bis zu mehreren Stunden) dauern. Sie können die Log-Einträge nachverfolgen, indem Sie mit dem Button *Logs* die Datei `cmrf.log` öffnen.

Für jede Übertragungsoperation können mehrere Parameter gesetzt werden, siehe Abschnitt Parameter für Übertragungs- und Aktualisierungsoperationen.

> ⚠ Die Aktion *Übertragen* sollte nur direkt nach der Initialisierung verwendet werden, da sie alle vorhandenen Daten aus den Standardtabellen löscht und alle benutzerdefinierten Tabellen löscht. Bei großen Datenbanken kann dies (die Übertragung) zu Problemen bezüglich des auf dem Datenbank-Server verfügbaren Speicherplatzes führen, da das Transaktions-Log sehr groß ist.

DWH-Synchronisierung während des Systembetriebs (Aktualisieren)

Für jede Aktualisierungsoperation können mehrere Parameter gesetzt werden, siehe Abschnitt Parameter für Übertragungs- und Aktualisierungsoperationen.

System läuft im ADMIN-Modus

Wenn das DWH im ADMIN-Modus läuft, muss der DWH-Administrator die Aktualisierung manuell starten, indem er auf *Aktualisieren* klickt. Beim Klicken auf *Aktualisieren* wird eine neue Aufgabe des Typs *Aktualisieren* für die Operation erstellt. Diese Aufgabe wird in der Aufgabenliste im Navigationselement *Aufgaben* angezeigt. Dann werden alle Daten, die übertragen werden sollen, d. h. alle Daten aus Feldern mit der Annotation `reportable` = „true“, die seit der letzten Übertragung oder Aktualisierung hinzugefügt oder geändert wurden, übertragen.

Wenn ein Ticketfeld, Kundenfeld oder Ressourcenfeld bei der letzten Übertragung die Annotation `reportable` noch nicht hatte und sie jetzt hat, wird der Inhalt des Feldes von allen Tickets, Kunden und Ressourcen übertragen.

System läuft im LIVE-Modus

Wenn das System im DWH-LIVE-Modus läuft, wird wie beim ADMIN-Modus eine Aufgabe des Typs *Aktualisieren* erstellt und verwaltet. Zusätzlich wird der LIVE-Modus für die Dauer der Aktualisierung angehalten und automatisch reaktiviert, wenn die Aktualisierung abgeschlossen ist.

Hinweise für ADMIN- und LIVE-Modus



Entfernen Sie die Annotation `reportable` = „true“ nicht aus einem Feld, wenn Sie nicht absolut sicher sind, dass die Daten nicht mehr in Reports verwendet werden! Wenn Sie ein Feld entfernen, das in Reports und/oder Cubes verwendet wird, treten zur Laufzeit Fehler im Reporting auf!



Die DWH-Aktualisierung kann auch über die Kommandozeile oder über ein Skript gestartet werden. In diesem Fall müssen Sie ein Tool verwenden, das über die Kommandozeile auf die ConSol CM-MBeans zugreifen kann, z. B. Twiddle für JBoss. Die zu verwendende MBean ist `consol.cmas.global.dwh.synchronizationService`. Der Befehl (die Methode) ist `update`.

Die folgende Kommandozeile zeigt einen Beispielbefehl mit Twiddle. Twiddle als Standalone-Installation.

```
$TWIDDLE_STANDALONE_HOME/bin/twiddle.sh -s service:jmx:remoting-
 jmx://127.0.0.1:9999 invoke
 consol.cmas:type=admin,topic=global,name=dwh.synchronizationService update
```

Parameter für Übertragungs- und Aktualisierungsoperationen

Für jede Operation des Typs *Übertragen* oder *Aktualisieren* können im Pop-up-Menü, das geöffnet wird, wenn der Button *Übertragen* oder *Aktualisieren* geklickt wird, mehrere Parameter eingestellt werden.

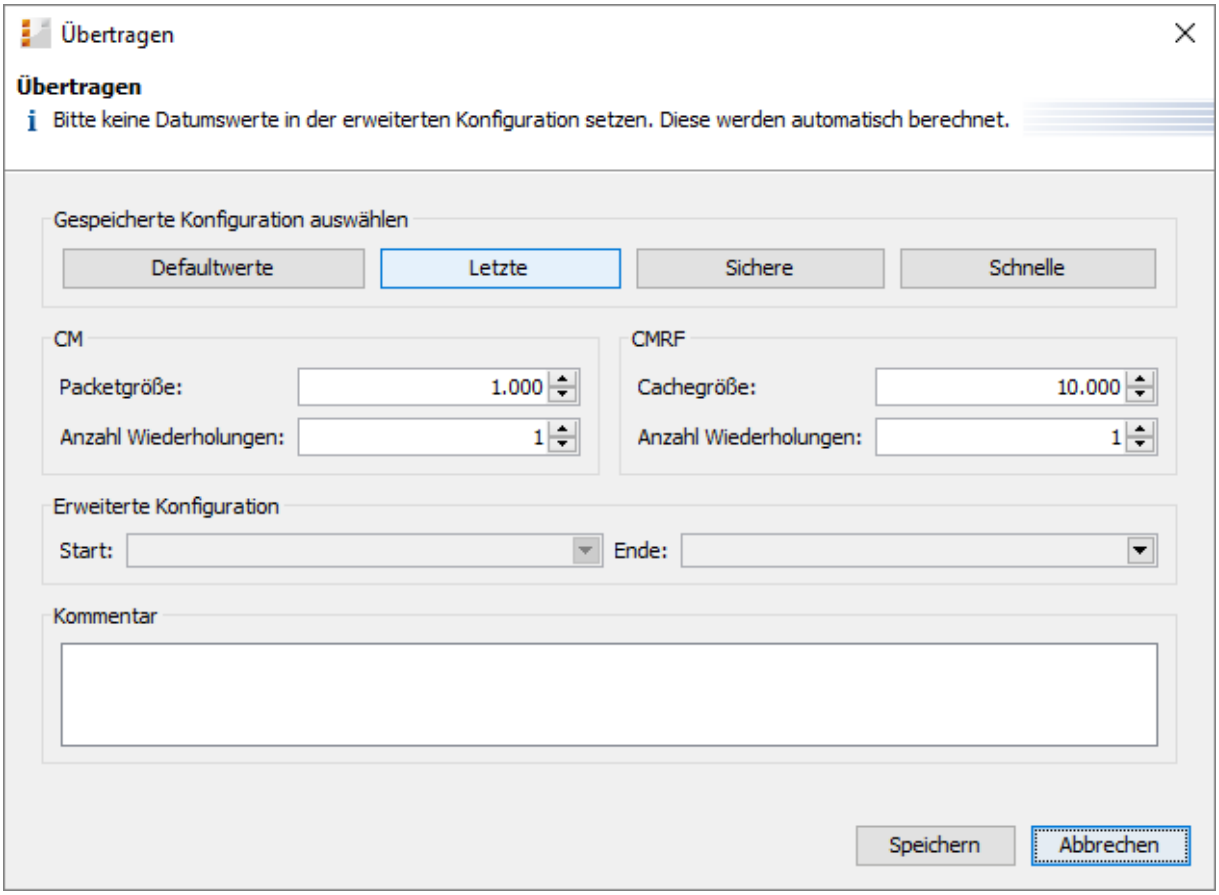


Abbildung 383: *ConSol CM Admin Tool - Data Warehouse, Administration: Konfigurieren der Übertragungsoperation*

Gespeicherte Konfiguration auswählen für die einzelne zu startenden Aufgabe:

- **Defaultwerte**
  Die Aufgabe mit den Standardeinstellungen ausführen, so wie wenn in diesem Dialog keine Änderungen vorgenommen werden.
- **Letzte**
  Die Aufgabe mit der gleichen Konfiguration ausführen wie bei der vorherigen Ausführung.
- **Sichere**
  Die Aufgabe mit vorsichtigen Einstellungen ausführen, die dazu führen können, dass die Ausführung länger dauert, aber dafür sorgen, dass keine OutOfMemory-Ausnahmen auftreten.
- **Schnelle**
  Die Aufgabe mit für den schnellen Abschluss optimierten Einstellungen ausführen. Es können allerdings OutOfMemory-Ausnahmen auftreten, wenn die Java Virtual Machine nicht richtig eingestellt ist.

Wenn Sie auf einen der Buttons klicken, werden die Parameter für die anderen Felder (z. B. *Paketgröße*) aus der entsprechenden Konfiguration in die Felder geladen. Die Operation wird noch nicht gestartet. Sie können weitere Änderungen an den Werten für die geplante Operation vornehmen.

Es sind folgende Felder verfügbar:

- **CM** (Senderseite)
  - **Paketgröße**
    Anzahl der in einer DWH-Nachricht gesendeten Objekte. Höhere Werte bedeuten eine bessere Performance und mehr Speichernutzung. Wenn das System genug RAM hat, kann der Wert 1000 oder mehr betragen. (In CM-Versionen vor 6.11 wurde diese Einstellung in der CM-System-Property cmas-dwh-server, batch-commit-interval vorgenommen, aber dies war ein systemweiter Parameter. Ab CM-Version 6.11 ist es möglich, ihn für jede DWH-Operation einzeln einzustellen.)
  - **Anzahl Wiederholungen**
    Anzahl an Wiederholungen, bis ein FEHLER geworfen wird.
- **CMRF** (Empfängerseite)
  - Cachegröße
  - **Anzahl Wiederholungen**
    Anzahl an Wiederholungen, bis ein FEHLER geworfen wird.
- **Erweiterte Konfiguration**
  - Start
  - Ende

> ⚠ **Verwenden Sie die Optionen der erweiterten Konfiguration nur, wenn Sie explizit von einem ConSol-Vertreter dazu aufgefordert werden!**
>
> Die im Abschnitt *Erweiterte Konfiguration* angebotenen Parameter zum Einstellen eines Datumsintervalls, das für die Aufgabe berücksichtigt werden soll, sollte nur verwendet werden, wenn Sie sich über die Folgen im Klaren sind. Sie können zu einem inkonsistenten Data Warehouse führen und sollten nur verwendet werden, wenn Sie mit detaillierten Anweisungen explizit vom ConSol-Support oder ConSol CM-Consulting dazu aufgefordert werden.

- **Kommentar**
  Optional. Ein Kommentar, der die DWH-Aufgabe beschreibt. Er wird in die Datenbank geschrieben und im Abschnitt *Details* angezeigt.

F.2.2.2 DWH-Aufgaben

Wenn es aktive DWH-Aufgaben gibt, wird dies im Admin Tool angezeigt, siehe folgende Abbildung. Die Navigationsgruppe *Data Warehouse* hat ein Ausrufezeichen, sodass Sie auch dann, wenn die Navigationsgruppe geschlossen ist, sehen, dass es aktive Aufgaben gibt. Die Anzahl der aktiven Aufgaben ist im Navigationselement *Aufgaben* angegeben.



Abbildung 384: *Anzeige aktiver DWH-Aufgaben im Admin Tool*

Diese Liste enthält Einträge (einen Eintrag für jede Aufgabe), wenn ...

- das DWH im ADMIN-Modus ausgeführt wird und der Administrator eine Aktualisierung gestartet hat: Es werden alle auszuführenden Aufgaben aufgeführt.
- das DWH im LIVE-Modus ausgeführt wird und die Checkbox *Automatische Übernahme von Konfigurationsänderungen* nicht markiert ist.
- Annotationen von Ticketfeldern, Kundenfeldern oder Ressourcenfeldern auf `reportable` = „true“ gesetzt wurden und die Checkbox *Automatische Übernahme von Konfigurationsänderungen* nicht markiert ist.
- Annotationen von Ticketfeldern, Kundenfeldern oder Ressourcenfeldern hinzugefügt und auf `reportable` = „true“ gesetzt wurden und die Checkbox *Automatische Übernahme von Konfigurationsänderungen* nicht markiert ist.
- Annotationen von Ticketfeldern, Kundenfeldern oder Ressourcenfeldern von `reportable` = „true“ auf „false“ gesetzt wurden und die Checkbox *Automatische Übernahme von Konfigurationsänderungen* nicht markiert ist.

Wenn die Checkbox *Automatische Übernahme von Konfigurationsänderungen* nicht markiert ist, verbleiben die Aufgaben im Status "zur Ausführung bereit".

Sie können eine oder mehrere Aufgaben in der Liste markieren und manuell ausführen (Button *Aufgaben ausführen*).

Wenn die Checkbox *Automatische Übernahme von Konfigurationsänderungen* markiert ist, werden die Aufgaben automatisch vom System ausgeführt. Die Ausführung kann mehrere Minuten dauern.

Folgende Sonderfälle sind zu beachten:

- Beim Entfernen der Annotation `reportable` = „true“ oder `reportable group` = „true“ wird keine DWH-Aufgabe erstellt. Diese Spalte und die entsprechenden DWH-Tabellen werden erst bei der nächsten DWH-Aktualisierung entfernt.
- Im ADMIN-Modus wird eine Aufgabe zum Ändern von `reportable` in „false“ ignoriert. Diese Aufgabe wird erstellt und sofort gelöscht. Die DWH-Tabelle wird nicht geändert. Diese Änderung wird bei der nächsten DWH-Aktualisierung durchgeführt.
- Im LIVE-Modus wird die Aufgabe sofort ausgeführt, auch wenn die Checkbox *Automatische Übernahme von Konfigurationsänderungen* nicht markiert ist.
- Bei einer DWH-Aktualisierung wird die Tabellenstruktur durch ein neu annotiertes Feld aus den offenen Aufgaben erweitert, sie wird erst bei der Ausführung der entsprechenden DWH-Aufgabe initial mit Daten befüllt.

F.2.2.3 DWH-Monitor

Sie können den DWH-Monitor öffnen, indem Sie in der Navigationsgruppe *Data Warehouse* auf das Navigationselement *Monitor* klicken.

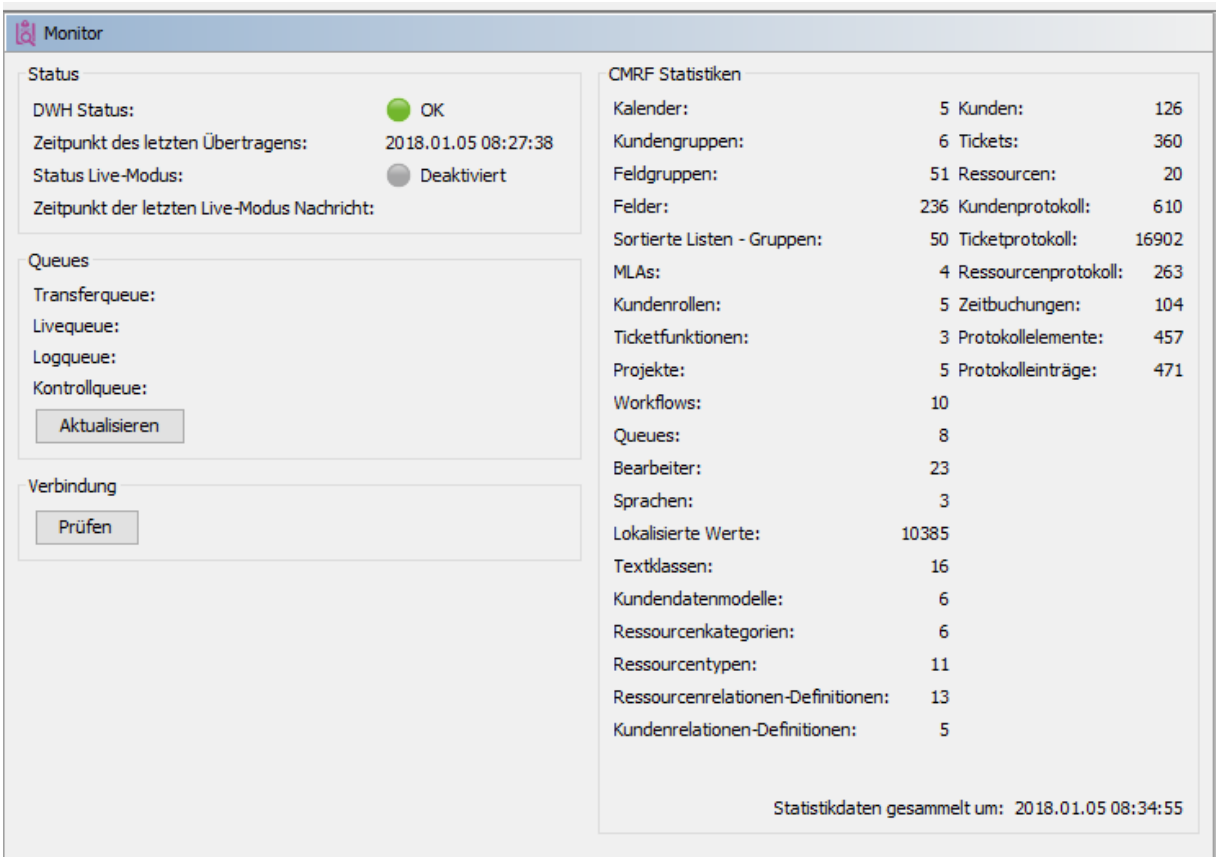


Abbildung 385: *ConSol CM Admin Tool - Data Warehouse, Monitor: DWH Monitor*

Der DWH-Monitor bietet eine schnelle Übersicht über alle DWH-Parameter. Das Datum des aktuellen Snapshots steht in der unteren rechten Ecke des Bildschirms.

Es werden folgende Parameter angezeigt:

- **Status**
  Eine schnelle Übersicht über die wichtigsten Parameter.

- **DWH Status**

Farbe des Statussymbols	DWH-Statusmeldung
Grün	OK
Gelb	Nicht initialisiert
	Aktion wird ausgeführt
	Aktion angehalten
	Aktion geplant
Rot	Fehler
Grau	Deaktiviert

- **Zeitpunkt des letzten Übertragens**
  Datum der letzten Übertragungs- oder Aktualisierungsoperation
- **Status Live-Modus**

Farbe des Statussymbols	DWH-Live-Statusmeldung
Grün	OK
Gelb	Nicht initialisiert
	Aktion wird ausgeführt
	Aktion angehalten
	Aktion geplant
	Gestartet (Aktualisierung erforderlich)
Rot	Fehler
	Fehler (Aktualisierung erforderlich)
Grau	Deaktiviert

- **Queues**
  Informiert über die Anzahl der Meldungseinträge in den internen Meldungs-Queue-Tabellen des Data Warehouse. Mit dem Button *Aktualisieren* können Sie die Werte aktualisieren.

  - Transferqueue: Anzahl der Meldungen in der Übertragungstabelle des Data Warehouse (INT_TRANSFER_QUEUE)
  - Livequeue: Anzahl der Meldungen in der LIVE-Meldungstabelle (INT_LIVE_QUEUE)
  - Logqueue: Anzahl der Meldungen in der Log-Meldungstabelle (INT_LOG_QUEUE)
  - Kontrollqueue: Anzahl der Meldungen in der Kontrollmeldungstabelle (INT_CONTROL_QUEUE)
- **Verbindung**
  Testet die Verbindung zur DWH-Datenbank mit dem Button *Prüfen*.
- **CMRF Statistiken**
  Liefert eine Liste mit der Anzahl der übertragenen Objekte.

F.2.2.4 DWH-Fehlerbehebung und -Reparatur

Wenn bei der Initialisierung, Übertragung oder Aktualisierung Fehler aufgetreten sind, werden die Log-Einträge im entsprechenden Log-Bereich angezeigt.

Die Originaldateien finden Sie unter folgendem Pfad:

- **JBoss 7 (ein Server):**

```
<JBOSS7_HOME>/standalone/log/cmrf.log
```

- **Weblogic:**

```
<ORACLE_HOME>\Middleware\user_projects\domains\consolcm6_domain\cmrf-
 logs\cmrf.log
```

Beachten Sie, dass dies die Standardpfade sind. Die Verwendung von anderen Pfaden kann in der Datei `cm6-cmrf.xml` konfiguriert sein. Eine ausführliche Beschreibung des Logging und der entsprechenden Dateien finden Sie im *ConSol CM Setup Manual* und *ConSol CM Operations Manual*.

Normalerweise geben die Log-Datei und/oder die Einträge im Log-Bereich gute Hinweise über den Grund des Übertragungsfehlers. Wenn ein Problem auftritt, das Sie nicht beheben können und Sie einen Wartungsvertrag mit ConSol haben, wenden Sie sich an unser Supportteam.

F.2.3 System-Properties für das DWH

Eine Liste aller System-Properties, die für eine bestimmte DWH-Konfiguration relevant sind, ist im Abschnitt CMRF- und DWH-Konfiguration enthalten.

F.2.4 Übertragungsmodus

Alle Daten mit der Annotation `reportable` = „true“ müssen an das Data Warehouse übertragen werden. Dies geschieht durch ConSol CM und das CMRF.

In ConSol CM-Version 6.11 und höher ist ein Übertragungsmodus verfügbar:

- DIRECT-Modus

Der Modus wird mit der System-Property cmas-dwh-server, communication.channel gesetzt. Mögliche Werte sind:

- „DIRECT“: Datenbank-Kommunikationskanal, einziger verfügbarer Wert seit Version 6.11.

Im DIRECT-Modus sendet der CM-Server die Nachrichten über direkten Zugriff auf Datenbanktabellen in der DWH-Datenbank:

- INT_CONTROL_QUEUE
- INT_LIVE_QUEUE
- INT_TRANSFER_QUEUE

Dies ist möglich, weil ConSol CM im DIRECT-Modus Zugriff auf die CMRF-Datenquelle erhält, d. h.

- Im Overlay-Modus:
  ConSol CM und CMRF verwenden sowieso den gleichen Application Server
- Im Standalone-Modus:
  Die Konfigurationsdatei der CMRF-Datenbank ist auch auf dem ConSol CM-Application-Server gespeichert (sie wird dahin kopiert)

Der CMRF-Server liest die Einträge aus den Tabellen und schreibt die Daten ins DWH.

F.2.5 DWH-Informationen für Experten

F.2.5.1 Lokalisierung von statischen DWH-Tabellen

Ab CM-Version 6.10.1 gibt es in statischen DWH-Tabellen Datenbankfelder für die lokalisierten Werte. Dies bedeutet, dass die lokalisierten Beschreibungen der Parameter ebenfalls an das DWH übertragen werden. Das folgende Beispiel zeigt die Felder im Admin Tool und im DWH für die Projekte in ConSol CM.

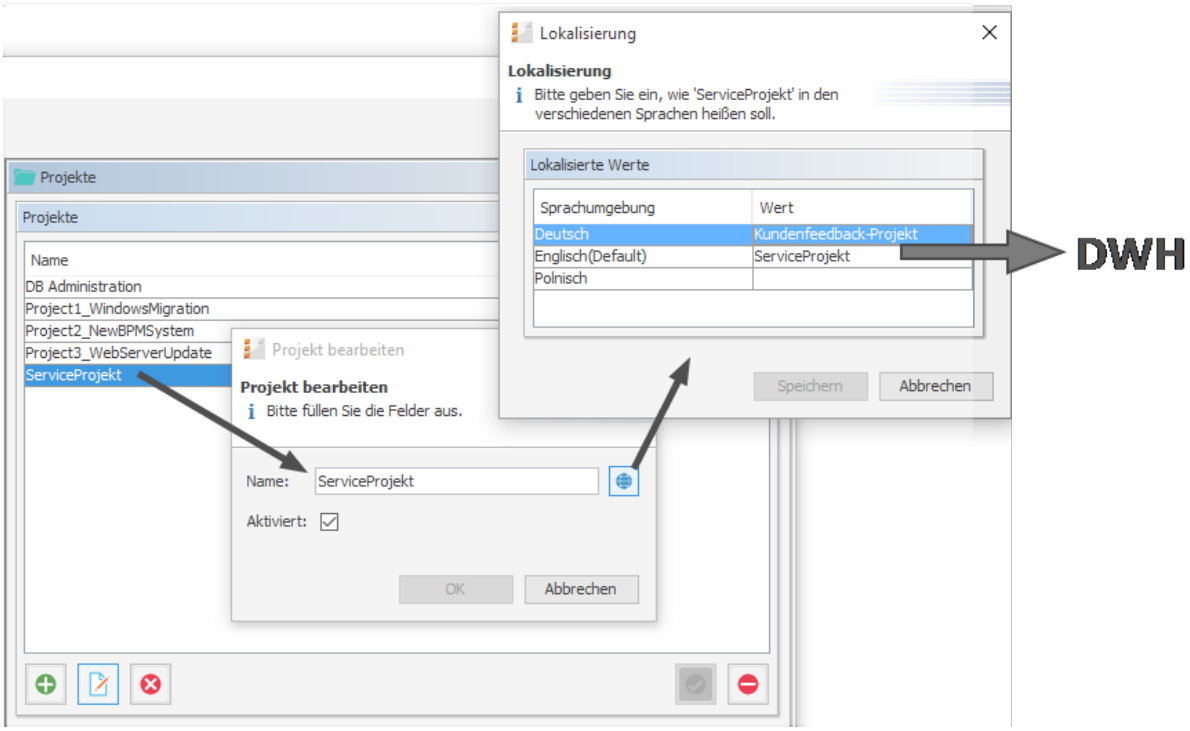


Abbildung 386: *ConSol CM Admin Tool - Globale Konfiguration, Projekte: Lokalisierte Werte von Projekten*

table *dim_project*

project_id	project_uid	name	name_en	name_de
46	2fc3d1c9-2df4-11e4-b9c4-ad888261acc9	ServiceProjekt	ServiceProjekt	Kundenfeedback-Projekt
47	4bb09eda-9e5f-4142-b33e-41e9b03d1e8f	Project3_WebServerUpdate	Project 3	Projekt 3
48	6db541f8-8a05-42e9-8eff-64620369ee9c	Project1_WindowsMigration	Project 1 Windows Migration	Projekt 1 Windows-Migration
49	aa570a20-3322-4696-9ffc-fd0750aebe25	Project2_NewBPMSystem	Project 2 New ERP System	Projekt 2 Neues ERP-System
50	5fce3e53-c8d1-11e5-998a-67528c2f9cca	DB Administration	DB Administration	NULL
NULL	NULL	NULL	NULL	NULL

Abbildung 387: *DWH-Tabelle mit lokalisierten Werten*

Die Verwendung dieser Felder kann die Dauer der DWH-Aktualisierungen erhöhen. Um zu verhindern, dass eine Aktualisierung zu lange läuft, können Sie CM mit der Java-System-Property `cmrf.localization.enabled` starten. Diese Property kann verwendet werden, um die Übertragung der lokalisierten Werte an das DWH ein- oder auszuschalten.

Beispiel für den Startbefehl:

```
nohup $JBOSS_HOME/bin/standalone.sh --server-config=cm6-cmrf.xml -b=0.0.0.0 -
 Dcmrf.localization.enabled=false
```

Eine detaillierte Erklärung dazu finden Sie im *ConSol CM DWH Manual*.

F.2.5.2 MBeans für die DWH-Verwaltung

Ab Version 6.10.5.4 ermöglicht die MBean `dwh.admin.service` (in `consol.cmas.admin.global`) Zugang zum DWH-Modus (OFF |ADMIN | LIVE). Es sind die Methoden `getMode()` und `setMode()` verfügbar.

Der Zugriff auf die MBean erfolgt über grafische Tools, z. B. JConsole für JBoss, oder über REST API. Dies ist in der *ConSol CM REST API Dokumentation* beschrieben.

F.3 CM-Services

Die CM-Services werden im Navigationselement *CM-Services* der Navigationsgruppe *Dienste* verwaltet. Dieses Navigationselement enthält die Liste (1) aller verfügbaren CM-Services. Angehaltene oder deaktivierte Dienste sind kursiv dargestellt. In diesem Navigationselement können Sie einzelne Dienste des ConSol CM-Systems, wie z. B. die Datenindizierung oder die DWH-Übertragung starten (2) oder stoppen (3).

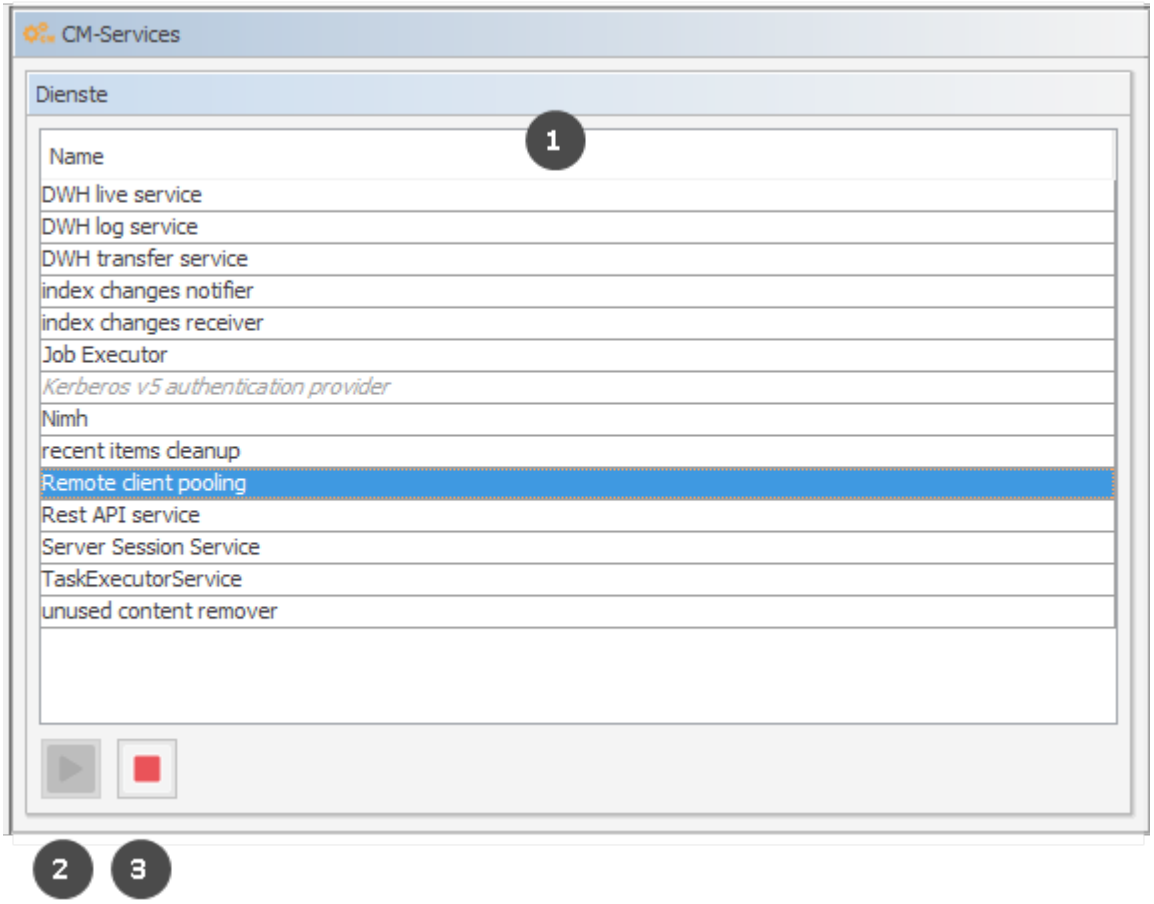


Abbildung 388: *ConSol CM Admin Tool - Dienste, CM-Services*

> ⚠ Der Status eines Dienstes sollte nur von einem erfahrenen ConSol CM-Consultant oder einem Mitglied des ConSol CM-Supportteams geändert werden! Wenn ein Dienst nicht läuft, funktionieren einige Schlüsselfunktionen von ConSol CM möglicherweise nicht.

Liste der Dienste:

- **DWH live service**
  Steuert die Just-in-Time-DWH-Aktualisierung im LIVE-Modus.

- **DWH log service**
  Liest und verarbeitet CMRF/DWH-Log-Meldungen für das Admin Tool und speichert sie in der ConSol CM-Datenbank. Die Einträge werden im Log-Protokoll des Admin Tools verwendet. Siehe Abschnitt Verwaltung des Data Warehouse (DWH).
- **DWH transfer service**
  Steuert die DWH-Übertragungen.
- **Job Executor**
  Steuert die Eskalationen für Prozesse bzw. Workflows.
- **Kerberos v5 authentication provider**
  Erforderlich, wenn die Kerberos-Authentifizierung aktiviert ist.
- **NIMH**
  Ruft eingehende E-Mails ab, wenn der Server im NIMH-Modus läuft (einziger Modus in 6.11.1).
- **Remote client pooling**
  Sorgt dafür, dass die Web Clients die Änderungen aus dem Admin Tool erhalten.
- **Rest API service**
  Aktiviert oder deaktiviert das REST Interface (*Representation State Transfer-Interface*).
- **Server Session Service**
  Prüft die Sessions und stoppt sie, wenn eine Client- oder Admin-Tool-Session abgelaufen ist. Siehe beispielsweise die System-Properties `admin.tool.session.check.interval` und `server.session.timeout`.
- **TaskExecutorService**
  Die Engine für die Ausführung der Aufgaben im Task Execution Framework. Sie umfasst einen Hauptverarbeitungs-Thread (mit Watchdog), der die Datenbank nach Aufgaben im Status *NEW* durchsucht, und eine zweite Komponente, die einen dedizierten Thread-Pool steuert, der für die Ausführung der Aufgaben genutzt wird.
- **Index changes notifier**
  Erstellt JMS-Meldungen (*Java Message Service*) mit Benachrichtigungen, wenn Änderungen vorliegen, die den Index betreffen.
- **Index changes receiver**
  Liest die JMS-Queue aus und startet die Aktualisierung im Indexer.
- **Unused content remover**
  Entfernt Attachments und Kommentare, die im Web Client (im Protokollbereich eines Tickets) als *gelöscht* markiert sind.

F.4 Suche in ConSol CM

ConSol CM bietet eine leistungsstarke Suchfunktion.

Im Abschnitt Konfiguration der Suche erfahren Sie, wie Sie das ConSol CM-System an Ihre Suchanforderungen anpassen. Der Abschnitt ConSol CM-Indexer enthält Informationen über den Indexer, der für die Suche benötigt wird.

Ab ConSol CM-Version 6.10.1 ist es zusätzlich möglich, Aktionen für Suchergebnisse auszuführen. Diese Funktion ist Teil des ConSol CM Action Framework und wird im Abschnitt Action Framework - Suche-Aktionen beschrieben.

Die Ergebnisse können als CSV-Liste aus dem Web Client exportiert werden. Diese Funktion wurde mit ConSol CM-Version 6.10.1 eingeführt. Sie ist im Abschnitt CSV-Export von Suchergebnissen beschrieben.

F.4.1 Konfiguration der Suche

ConSol CM bietet eine leistungsstarke Suche nach allen an den Geschäftsprozessen beteiligten Objekten (Kunden, Tickets und Ressourcen). Technisch gesehen basiert die Suche auf dem ***Indexer***, einem Modul von ConSol CM.

In den folgenden Abschnitten wird das Thema Suche in ConSol CM umfassend aus der Perspektive eines Administrators beschrieben. Eine detaillierte Beschreibung über die Verwendung der Suche als Bearbeiter finden Sie im *ConSol CM Benutzerhandbuch*.

F.4.1.1 Suchmodi

Den Bearbeitern stehen in ConSol CM mehrere Arten von Suchen zur Verfügung:

Schnellsuche

Die Schnellsuche wird über das Suchfeld in der oberen rechten Ecke des Web Clients durchgeführt. Die Anzeige der Ergebnisse (d. h. die Felder und die Reihenfolge der Felder in der Ergebnisliste) kann mithilfe von Templates formatiert werden. Details dazu finden Sie in den Abschnitten Templates für Kundendaten und CM/Resource Pool - Templates für Ressourcendaten. Denken Sie daran, dass Sie die Anzahl der Suchergebnisse mit der System-Property cmweb-server-adapter, globalSearchResultSizeLimit begrenzen können. Details dazu finden Sie im Abschnitt System-Properties. Der Ergebnissatz wird auch durch das Attribut der Seitenanpassung *appendWildcardAutomatically* beeinflusst, siehe Abschnitt Seitenanpassung, appendWildcardAutomatically.

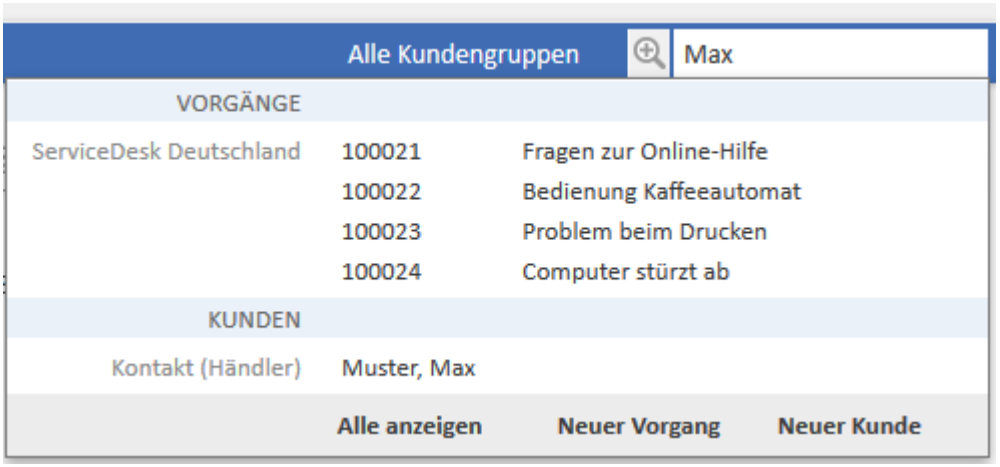


Abbildung 389: *ConSol CM Web Client - Schnellsuche*

Phonetische Suche

Für die Schnellsuche kann eine phonetische Suche für String-Felder konfiguriert werden. Eine Erklärung der Konfiguration Ihres ConSol CM-Systems für diese Funktion finden Sie in Konfigurieren der phonetischen Suche.

Mit der phonetischen Suche findet der Bearbeiter nicht nur genaue Treffer, sondern auch Ergebnisse, die ähnlich klingen aber nicht genau so geschrieben werden, wie der eingegebene Suchbegriff. Die Implementierung basiert auf den Bibliotheken von ***Apache Commons Codec***.

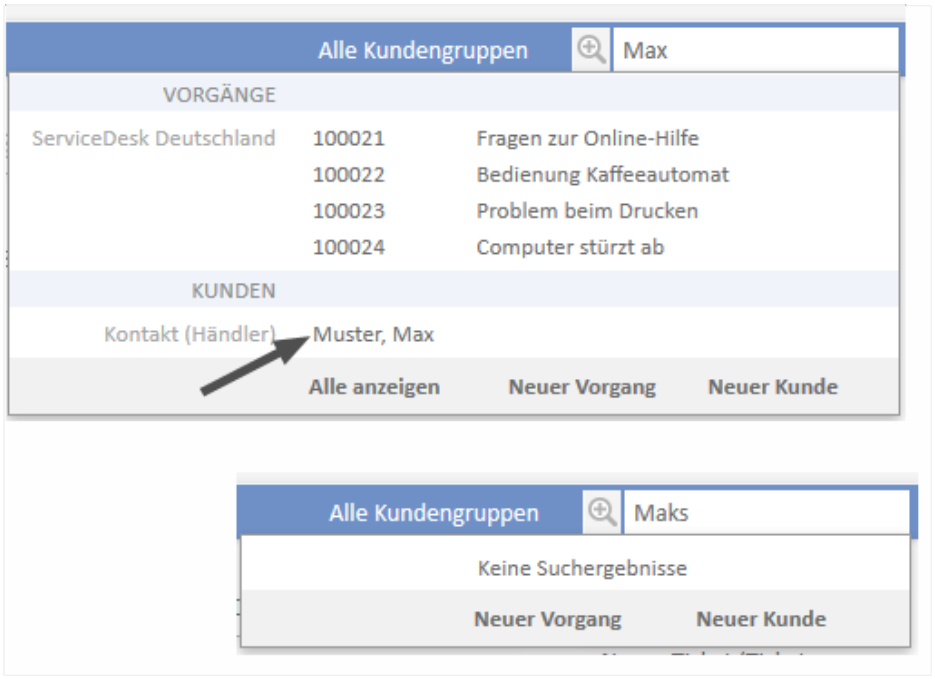


Abbildung 390: *ConSol CM Web Client - Suchergebnisse ohne phonetische Suche*

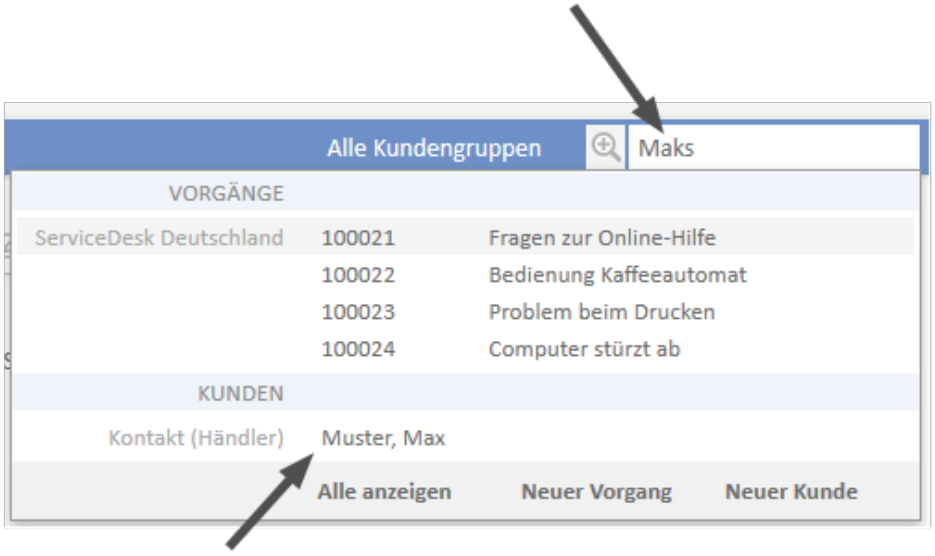


Abbildung 391: *ConSol CM Web Client - Suchergebnisse mit phonetischer Suche*

Detailsuche

Die Detailsuche wird auf der Seite *Detailsuche* durchgeführt. Um diese Seite zu öffnen, klicken Sie neben dem Eingabefeld der Schnellsuche auf die Lupe.

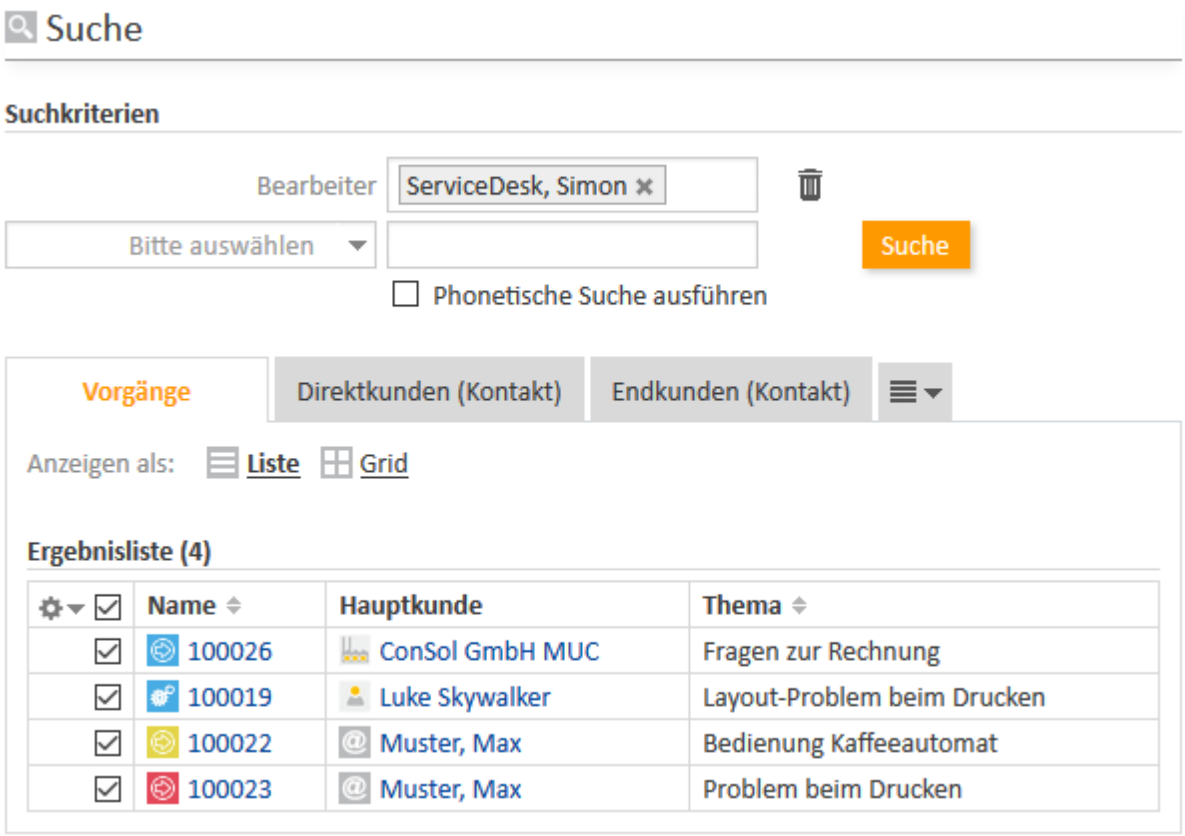


Abbildung 392: *ConSol CM Web Client - Detailsuche*

Denken Sie daran, dass Sie die Anzahl der Suchergebnisse pro Seite (Standardwert und Optionen) mit den System-Properties cmweb-server-adapter, searchPageSize und cmweb-server-adapter, searchPageSizeOptions konfigurieren können. Eine Erklärung dazu finden Sie in System-Properties.

Für die Bearbeitersuche können Sie als Administrator Bearbeiternamen festlegen, die nicht im Dropdown-Menü in den Suchkriterien angezeigt werden sollen. Diese Konfiguration erfolgt in der Seitenanpassung. Details dazu finden Sie im Abschnitt detailSearch (Typ).

Informationen über die Verwendung der Suchfunktion finden Sie im *ConSol CM Benutzerhandbuch*, Abschnitt *Suchen nach Tickets, Kunden und Ressourcen*.

Wichtiger Hinweis über Felder mit Lists of Structs

Beachten Sie, dass in den ConSol CM-Versionen 6.10.6.x und älter zwei Felder, die zur gleichen List of Structs (d. h. Tabelle) gehören, in der Detailsuche mit ODER kombiniert werden. In CM-Versionen 6.10.7.0 und höher werden die beiden Felder mit UND kombiniert. Dies führt dazu, dass die Suchergebnisse in unterschiedlichen CM-Versionen verschieden sind!

Ein detailliertes Beispiel finden Sie im *ConSol CM Benutzerhandbuch*.

Konfiguration der Liste mit den Suchergebnissen

Als Administrator können Sie das Layout der Liste mit den Suchergebnissen mithilfe der Annotation `order-in-result` konfigurieren. Diese Annotation wirkt sich auf die Spalten der objektspezifischen Suchergebnisse aus. Die Kundenfeldgruppe *ResellerCompanyData*, die zum Kundendatenmodell *ResellerModel* gehört, enthält beispielsweise folgende Kundenfelder:

- *company_name*: order-in-result = 1
- *company_number*: order-in-result = 2

Die Liste mit den Suchergebnissen, z. B. aus einer Detailsuche, die einen Satz Firmen aus einer Kundengruppe mit dem ResellerModel zeigt, enthält also die beiden Spalten *Firmenname (company-name)* und *Firmennummer (company_number)*.

Setzen Sie den Wert der Annotation `order-in-result` auf „0“ (Null), wenn ein Feld (d. h. eine Spalte in der Ergebnistabelle) nicht im Web Client angezeigt werden soll (d. h. das Feld soll nicht in der Feldauswahl für die Ergebnisliste und als Tabellenspalte verfügbar sein).

Die folgende Abbildung zeit die Annotation `order-in-result` eines Kundenfeldes. Sie gelangen auf diese Seite, indem Sie das Navigationselement *Datenmodelle* in der Navigationsgruppe *Kunden* öffnen.

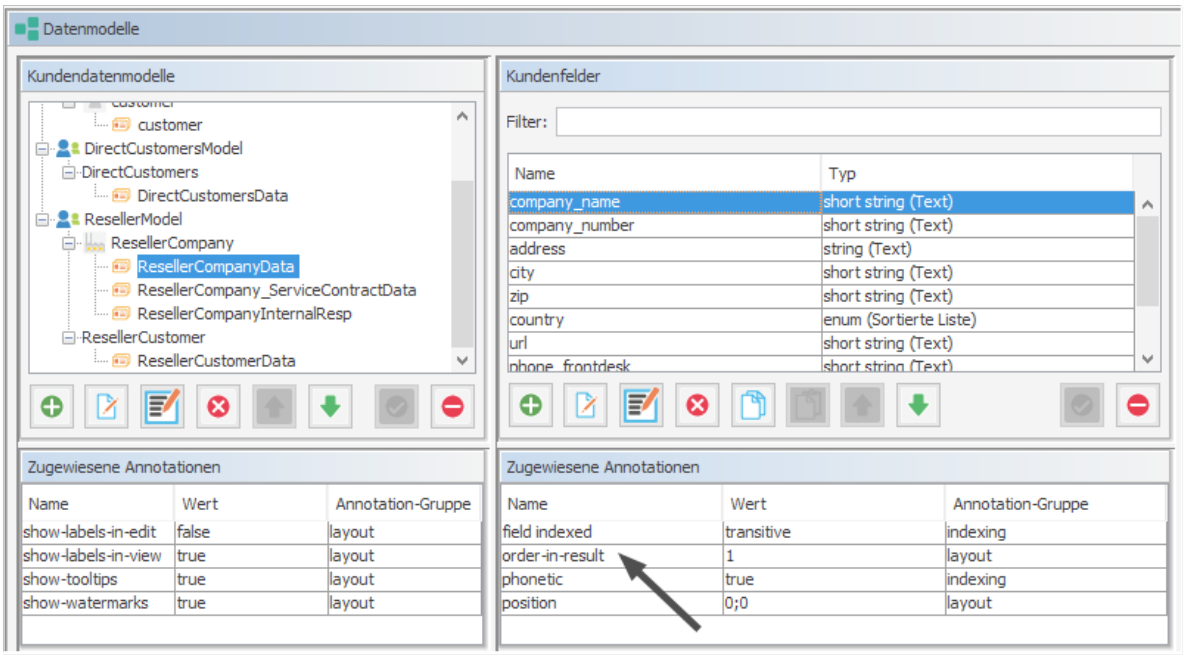


Abbildung 393: *ConSol CM Admin Tool - Kunden, Datenmodelle: Setzen der Annotation order-in-result für ein Kundenfeld*

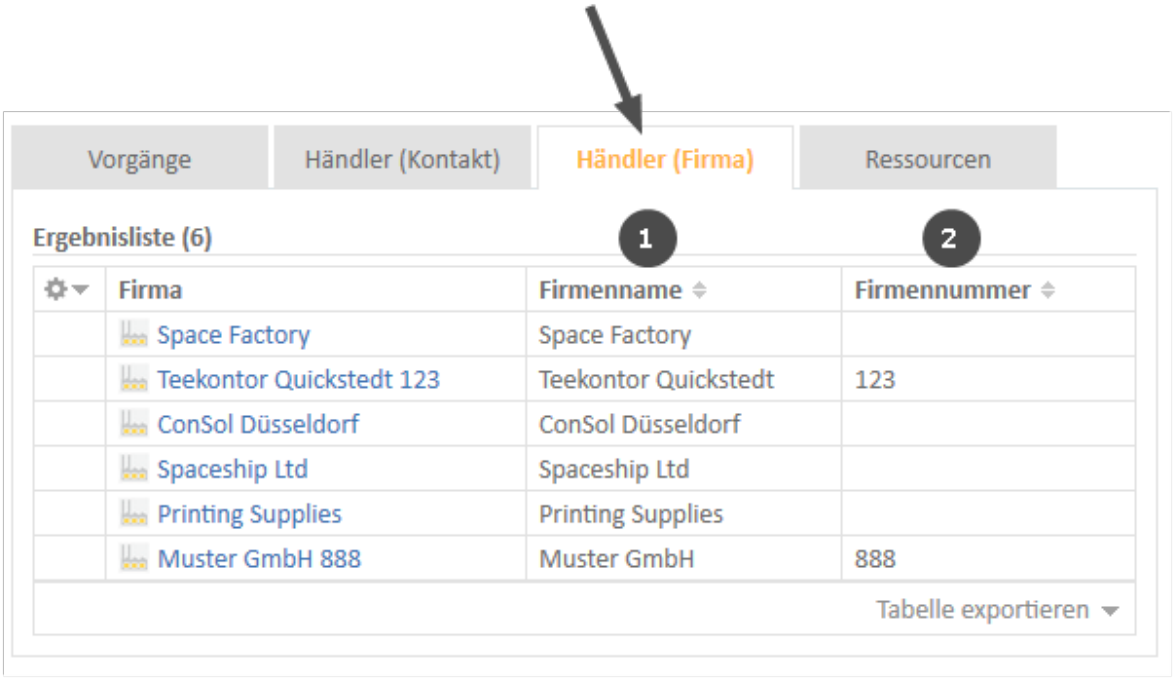


Abbildung 394: *ConSol CM Web Client - Suchergebnisse mit zwei annotierten Spalten (order-in-result)*

Hinweise zur Reihenfolge der Spalten mit den Suchergebnissen

Da die Annotation `order-in-result` nicht der einzige Parameter ist, der die Reihenfolge von Spalten in den Suchergebnissen (Tabellen) beeinflusst, finden Sie hier eine detaillierte Erklärung.

Im folgenden Abschnitt wird der Ausdruck *Benutzereinstellungen* für das vom Benutzer manuell konfigurierte Layout der Tabelle mit den Suchergebnissen verwendet, d. h. die anzuzeigenden Spalten, die Reihenfolge der Spalten und die Sortierung.

Regeln:

1. Die Spalten einer Tabelle mit Suchergebnissen können für jeden der folgenden CM-Objekttypen unterschiedliche Felder enthalten:
   a. für Tickets: Ticketfelder sowie interne Felder wie Queue, Erstellungsdatum, Bearbeiter
   b. für Kunden (Kontakte oder Firmen): Kundenfelder sowie interne Felder wie der Objektname eines Kunden
   c. für Ressourcen: Ressourcenfelder sowie interne Felder wie der Objektname einer Ressource
2. Mit der Annotation `order-in-result` wird nur die Reihenfolge der Ticketfelder, Kundenfelder und Ressourcenfelder bestimmt, nicht aber die der internen Felder.

3. Die internen Feldtypen haben standardmäßig folgende Sichtbarkeit und Reihenfolge:
    a. in Ticketlisten:
        i. Bearbeiter
        ii. Hauptkunde
        iii. Ticketname
        iv. Ticketthema
    b. in Kundenlisten (Kontakt/Firma):
        i. Name des Objekts (d. h. Kontakt oder Firma)
    c. in Ressourcenlisten:
        i. Name des Objekts (=Ressource)
4. Die Datenfelder (Ticketfelder, Kundenfelder und Ressourcenfelder) sind primär nach den Benutzereinstellungen geordnet. Diese überschreiben jegliche andere Konfiguration.
   Wenn zum Beispiel ein Ticketfeld die Annotation `order-in-result` = „5“ hat und sichtbar ist, wird es (Fall a) nicht angezeigt, wenn der Bearbeiter die Spalte nicht ausgewählt hat, und (Fall b) als Spalte Nr. 2 angezeigt, wenn der Bearbeiter es an diese Stelle gezogen hat.
5. Wenn durch die Benutzereinstellungen keine Reihenfolge definiert ist, ist die Reihenfolge der Datenfelder die aufsteigende Reihenfolge entsprechend der als Wert für die Annotation `order-in-result` angegebenen Zahlen.. Diese Zahlen werden global über alle Datenfelder in allen Ticketfeldgruppen, Kundenfeldgruppen und Ressourcenfeldgruppen ausgewertet. Es ist also theoretisch möglich, dass unterschiedliche Datenfelder identische Werte für die Annotation `order-in-result` haben.
   Um die Reihenfolge exakt zu definieren, setzen Sie für die gesamte CM-Installation eine Konvention ein und verwenden Sie eindeutige Zahlen, z. B. hohe vierstellige Werte.
6. Wenn es immer noch zwei Felder mit identischen Werten für `order-in-result` gibt, die beide angezeigt werden sollen, sind die Datenfelder alphabetisch nach ihren lokalisierten Namen geordnet. Dabei wird die Sprache verwendet, die der Bearbeiter im Browser eingestellt hat.

Konfigurieren des Verhaltens der Detailsuche für Werte aus Sortierten Listen und MLAs

Die Werte in einer Sortierten Liste (Enum) oder einem MLA können sich ändern. Zum Beispiel wird eine neue Software-Version zur Liste hinzugefügt und eine alte Version ist nicht mehr verfügbar. In diesem Fall wird der alte (deprecated) Wert im aktuellen System deaktiviert. Es ist natürlich wichtig, dass die Bearbeiter noch nach Tickets suchen können, die diesen Wert enthalten. Deshalb sind deaktivierte Listen- und MLA-Werte in der Detailsuche verfügbar, wie in der folgenden Abbildung für den deaktivierten Listenwert *Resource Pool* gezeigt.

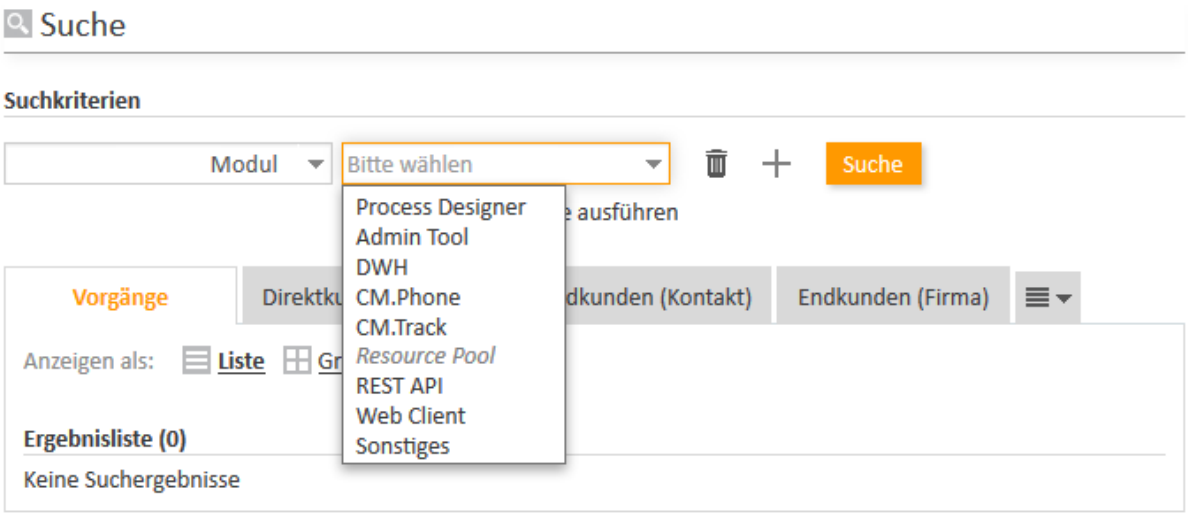


Abbildung 395: *ConSol CM Web Client - Detailsuche nach Modulen, ein Listenwert deaktiviert*

Das Verhalten von ConSol CM bezüglich der Detailsuche nach deaktivierten Listen- und MLA-Werten kann mit folgendem Attribut der Seitenanpassung konfiguriert werden: detailSearch (Typ).

Autocomplete-Suche (Suche in intelligenten Feldern)

Die Autocomplete-Suche wird implizit durchgeführt, wenn Sie mit der Eingabe in ein Autocomplete-Feld beginnen, z. B. bei der Ticketerstellung die Firmen- oder Kundendaten eingeben (siehe folgende Abbildung).

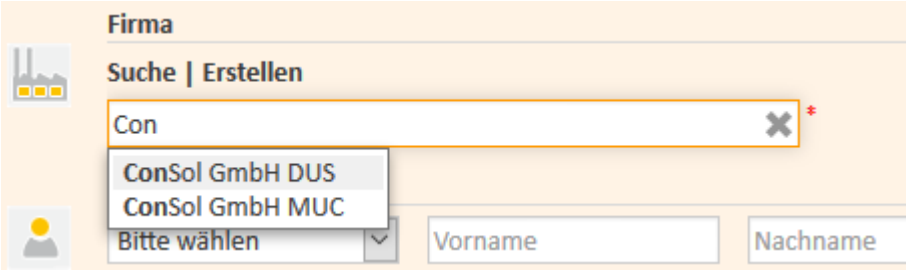


Abbildung 396: *ConSol CM Web Client - Autocomplete-Suche*

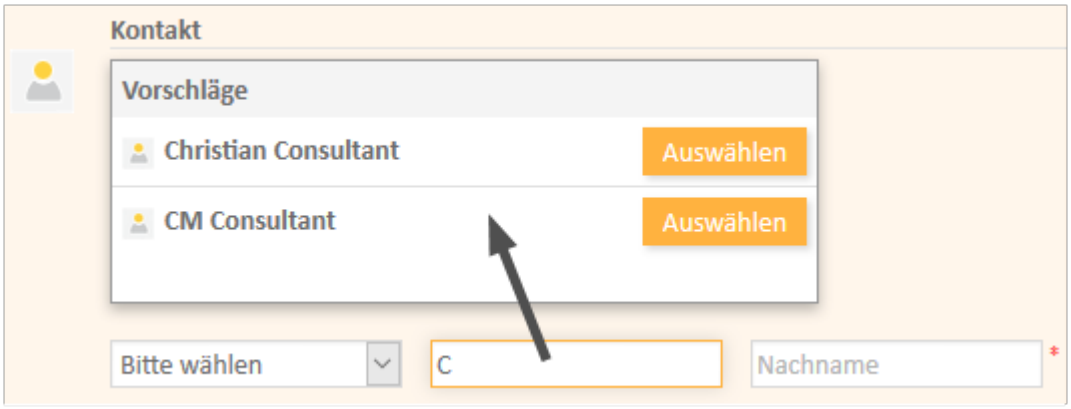


Abbildung 397: *ConSol CM Web Client - Vorschläge einer Autocomplete-Suche*

Ab ConSol CM-Version 6.10.2 wird auch im Ticket-E-Mail-Editor eine Suche in allen Kontakten der ConSol CM-Datenbank durchgeführt, wenn der Bearbeiter mit der Eingabe in das Feld An, Cc oder Bcc beginnt:

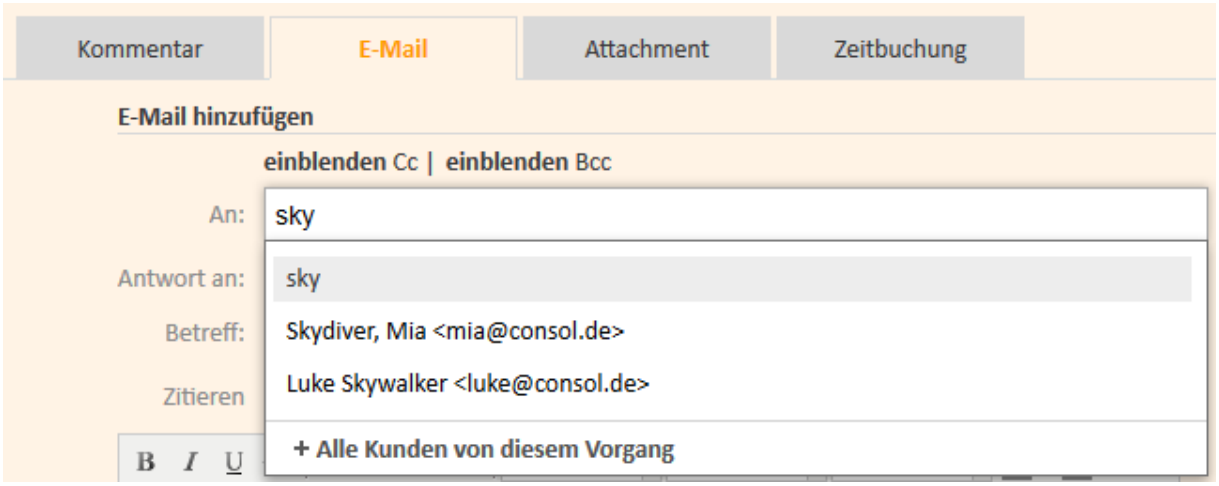


Abbildung 398: *ConSol CM Web Client - Ticket-E-Mail-Editor: Automatische Suche nach Kontakten als E-Mail-Empfänger*

Die Anzahl der Zeichen, die eingegeben werden müssen, damit die automatische Suche beginnt, kann in der Seitenanpassung konfiguriert werden. Verwenden Sie dazu das Seitenanpassungsattribut *minMailInputLength* (Integer, Standardwert: 1) im Bereich *mailTemplate*.

F.4.1.2 Durchsuchbare Felder

Für Ticketfelder, Kundenfelder und Ressourcenfelder nach denen gesucht werden soll, muss die Annotation `field indexed` gesetzt werden. Siehe Abschnitt Konfiguration der Suche. Dann ist das Feld in der Schnellsuche und in der Detailsuche verfügbar.

Die Suchfunktion verarbeitet zwei Arten von Feldern:

1. Datenfelder und Inhalt, die standardmäßig indiziert werden
2. Datenfelder, die mit der Annotation `field indexed` annotiert sind

Datenfelder und Inhalt, die standardmäßig indiziert werden

- Bearbeiterdaten
    - E-Mail
    - Vorname
    - Nachname
    - ID
- Ticketdaten
    - Attachment-Inhalt (Dies wir durch die CM-System-Property cmas-core-index-common, index.attachment gesteuert, die standardmäßig auf „true“ steht.)
    - Erstellungsdatum
    - Bearbeiter
    - Protokoll
    - ID (technisch gesehen der Ticketname)
    - Queue
    - Zusätzliche Bearbeiter
    - Thema
    - Ansicht

Datenfelder, die mit der Annotation `field indexed` annotiert sind

Für alle Datenfelder, die von ConSol CM-Administratoren hinzugefügt und konfiguriert wurden, muss die Annotation `field indexed` gesetzt werden, damit die Felder in der Suche verfügbar sind. Ein indiziertes Feld ist verfügbar in:

- der Schnellsuche
- der Detailsuche
- allen Suchen in Workflow- und Admin-Tool-Skripten, die beispielsweise Klassen/Objekte mit Kriterien verwenden (z. B. `TicketCriteria`, `UnitCriteria`, `ResourceCriteria`). Dies betrifft:
    - Skripte, die manuell durch den Administrator hinzugefügt werden
    - Skripte, die standardmäßig vorhanden sind (z. B. das Skript `createTicket.groovy`. In diesem Skript kann der korrekte Kontakt für ein neues Ticket nur gesetzt werden, wenn das Kundenfeld, mit dem der Kontakt oder die Firma abgerufen wird, indiziert ist.)

Wenn Werte einer Tabelle, d. h. einer List of Structs, in der Suche verfügbar sein sollen, müssen alle Elemente der Datenstruktur mit `field-indexed` = „true“ annotiert sein. Das bedeutet, dass die Liste, der Struct und jedes einzelne Datenfeld im Struct so annotiert sein muss!

Als Administrator können Sie das Systemverhalten bezüglich der Suchergebnisse definieren. Abhängig von den Werten der Annotation `field indexed` enthalten die Ergebnisse der Suche nach einem Kontakt auch alle Tickets dieses Kontakts. Die folgende Tabelle zeigt die Auswirkungen aller möglichen Werte der Annotation `field indexed` für die unterschiedlichen Objekttypen. Die Annotation `field indexed` muss für jedes einzelne Datenfeld gesetzt werden, z. B. *name* und *e-mail* für einen Kontakt, *zip* und *address* für eine Firma, *priority* und *software module* für Tickets oder *name* und *model* für einen Ressourcentyp.

Beachten Sie, dass Sie es hier mit zwei Perspektiven zu tun haben:

1. Das **Suchkriterium**, d. h. das Feld, nach dem Sie suchen, z. B. alle Firmen, deren Name mit ConS* beginnt. Ein Feld steht in der Suche zur Verfügung, wenn die Annotation `field-indexed` gesetzt ist.
2. Der **Ergebnissatz**, d. h. die Objekte, die in der Ergebnisliste angezeigt werden. Die Objekte in der Ergebnisliste werden durch den Wert der Annotation `field-indexed` definiert. Abhängig von diesem Wert sehen Sie nur die Objekte, nach denen Sie gesucht haben (z. B. nur Kontakte) oder auch Objekte, die mit diesen Objekten verknüpft sind (z. B. Firmen der Kontakte). Eine detaillierte Beschreibung finden Sie in der folgenden Tabelle.

Objekttyp / Wert der Annotation field indexed	transitive	unit	local	not indexed
TICKET	• kein Unterschied zwischen den drei Werten • die Ticketdaten sind in der Suche verfügbar • es werden keine anderen mit dem Ticket verknüpften Objekte abgerufen • wir empfehlen die Verwendung des Standardwerts „transitive“			• die Ticketdaten sind NICHT in der Suche verfügbar

Objekttyp / Wert der Annotation field indexed	transitive	unit	local	not indexed
KONTAKT	• die Kontaktdaten sind in der Suche verfügbar • die Tickets des Kontakts werden bei der Suche nach dem Kontakt ebenfalls gefunden • die Suche nach der Firma über ihr Kontaktfeld ist NICHT möglich	• die Kontaktdaten sind in der Suche verfügbar • die Suche nach der Firma über ihr Kontaktfeld ist NICHT möglich	• die Kontaktdaten sind in der Suche verfügbar • es werden keine anderen mit dem Kontakt verknüpften Objekte abgerufen	• die Kontaktdaten sind NICHT in der Suche verfügbar
FIRMA	• die Firmendaten sind in der Suche verfügbar • die Tickets der Firma werden bei der Suche nach der Firma ebenfalls gefunden • die Kontakte der Firma werden bei der Suche nach der Firma ebenfalls gefunden	• die Firmendaten sind in der Suche verfügbar • die Kontakte der Firma werden bei der Suche nach der Firma ebenfalls gefunden	• die Firmendaten sind in der Suche verfügbar • es werden keine anderen mit der Firma verknüpften Objekte abgerufen	• die Firmendaten sind NICHT in der Suche verfügbar
RESSOURCE	• kein Unterschied zwischen den drei Werten • die Ressourcendaten sind in der Suche verfügbar • es werden keine anderen mit der Ressource verknüpften Objekte abgerufen • wir empfehlen die Verwendung des Standardwerts *transitive*			• die Ressourcendaten sind NICHT in der Suche verfügbar



Bitte beachten Sie Folgendes: Wenn es möglich sein soll, Ergebnistabellen im Web Client (durch Klicken auf die Kopfzeile der Spalte) nach einer Spalte zu sortieren, muss das entsprechende Feld indiziert sein!

Konfigurieren der phonetischen Suche

Um die phonetische Suche für ein Datenfeld zu aktivieren, setzen Sie die Annotation `phonetic` für dieses Feld auf „true“. Die Annotation kann nur für String-Felder gesetzt werden.

F.4.2 ConSol CM-Indexer

F.4.2.1 Einführung

ConSol CM speichert die meisten Daten in einer relationalen Datenbank. Zur Verbesserung der Geschwindigkeit von Suchvorgängen verwendet ***ConSol CM Apache Lucene***-Indizes. Für jedes Datenfeld, das als Suchkriterium dienen soll, wird ein Index erzeugt. Das ConSol CM-Modul, das die Indizes verwaltet, heißt ***CM-Indexer***. Die Indizes werden im Dateisystem in einem Unterverzeichnis des bei der Systemeinrichtung angegebenen Datenverzeichnisses gespeichert. Der Pfad des Datenverzeichnisses wird als System-Property gespeichert: cmas-core-shared, data.directory.

F.4.2.2 Indexer-Datenverzeichnis

Die folgende Abbildung zeigt ein Beispiel für Indexdateien einer ConSol CM-Installation. `demo_Datadir` ist das Datenverzeichnis, das bei der Einrichtung angegeben wurde. Alle Unterverzeichnisse werden automatisch durch ConSol CM erstellt.

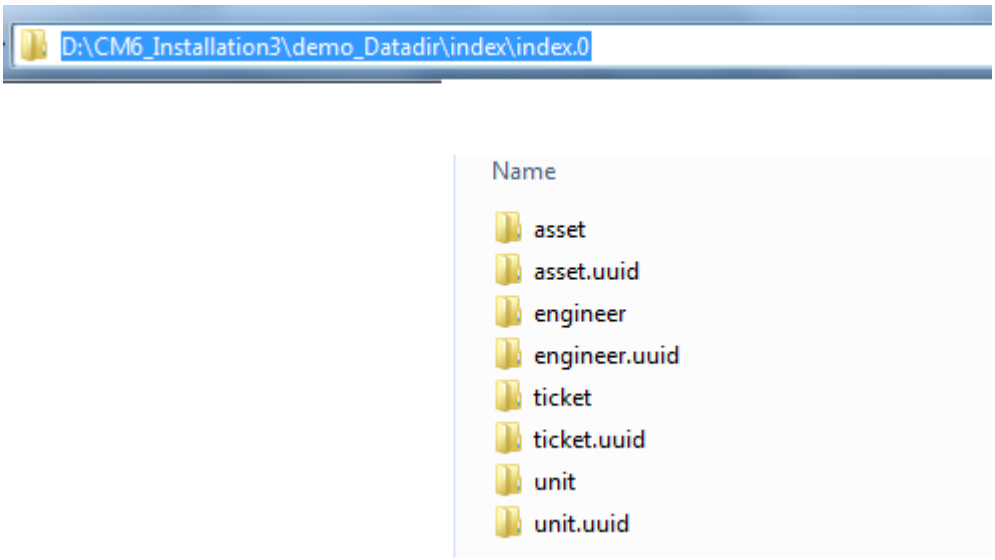


Abbildung 399: *ConSol CM Indexer - Verzeichnis demo_Datadir*

Stellen Sie sicher, dass ...

- das Datenverzeichnis dem ConSol CM-Server immer zur Verfügung steht, wenn es auf einem anderen Server erstellt wurde und mit dem Application Server verknüpft oder darauf gemountet wurde.
- im Datenverzeichnis ausreichend Speicherplatz für die Indizes vorhanden ist.
- das ConSol CM-Datenverzeichnis Teil der täglichen Datensicherung ist (und bei Bedarf wiederhergestellt werden kann).

Wenn das Indexverzeichnis beschädigt oder nicht verfügbar ist, kann der Index neu gebaut oder repariert werden. Details dazu finden Sie in den folgenden Abschnitten.

F.4.2.3 Indexer-Dienste

Für die Indizierung sind zwei ConSol CM-Dienste wichtig:

- **Index changes notifier**
  Dieser Dienst erstellt Nachrichten im persistenten Speicher mit Benachrichtigungen, wenn für den Index relevante Änderungen erfolgen.
  Das Stoppen von *index changes notifier* ist **nicht** ungefährlich. Wenn das Indexer-Modul feststellt, dass *index changes notifier* gestoppt wurde, und es eine Nachricht gibt, die an den persistenten Speicher gesendet werden muss, setzt der Indexer die Property des Indexstatus (cmas-core-index-common, index.status) auf „RED“, d. h. der Index muss komplett synchronisiert werden.
- **Index changes receiver**
  Das Verhalten dieses Dienstes hängt von dem Typ des Nodes, auf dem er läuft, ab.
  - Auf einem Master-Node lädt der *index changes receiver* die Tickets und erstellt die entsprechenden Lucene-Dokumente.
  - Auf einem Slave-Node (nur im Cluster möglich) fragt er den Master-Indizierungsserver über HTTP ab, um die Lucene-Dateien mit den indizierten Daten herunterzuladen.
  - In beiden Fällen liest dieser Dienst aus dem persistenten Speicher und aktualisiert den Indexer.
    Das Stoppen von *index changes receiver* ist ungefährlich. Nach einem Neustart werden alle fehlenden Änderungen aus dem persistenten Speicher übernommen.

Siehe auch Abschnitt CM-Services.

F.4.2.4 Aufgaben des Administrators bezüglich des Indexers

Für den Administrator ist es wichtig, zu wissen, wie man ConSol CM so konfiguriert, dass ...

- alle erforderlichen Felder durchsuchbar sind
- keine Overhead entsteht (d. h. nicht zu viele Felder für die Suche konfiguriert sind).
- die Suchergebnisse wie gewünscht dargestellt werden:
  - in der Ergebnistabelle der Detailsuche
  - als Vorschläge in der Autocomplete-Suche

F.4.2.5 System-Properties für den Indexer

Es gibt mehrere System-Propertiesfür die Konfiguration des Indexers. Eine Liste der wichtigsten Properties finden Sie in Liste der System-Properties nach Bereich.

F.4.2.6 Verwaltung von Indexer und Index mit dem Admin Tool

Die Annotation `field indexed`

Standardmäßig werden der gesamte Tickettext und alle Anhänge indiziert. Alle Ticketfelder, Kundenfelder und Ressourcenfelder, die indiziert werden sollen, müssen die Annotation `field indexed` haben. Details über die Annotationen für Ticketfelder (Ticketdaten), Kundenfelder (Kundendaten) und Ressourcenfelder (Ressourcendaten) finden Sie in den Abschnitten Verwaltung von

Ticketfeldern (Einrichten des Ticketdatenmodells), Verwaltung von Kundenfeldern und GUI-Design für Kundendaten und CM/Resource Pool - Einrichten des Ressourcenmodells. Diese Annotation kann vier mögliche Werte haben:

- **local**
- **unit**
- **transitive**
- **not indexed**

Eine detaillierte Beschreibung des durch diese Werte bestimmten Systemverhaltens finden Sie in der obigen Tabelle.

Verschachtelte Felder müssen den gleichen Indextyp haben, andernfalls können sie nicht durchsucht werden. Wenn Sie zum Beispiel mit einer List of Structs arbeiten, müssen die Liste, das Struct und alle Datenfelder im Struct, die durchsucht werden sollen, die Annotation `field indexed` mit dem Wert `transitive` haben.

Indexer-Verwaltung: Navigationselement *Index*

Normalerweise ist keine manuelle Pflege des Index erforderlich. ConSol CM führt alle Indizierungsvorgänge automatisch aus. Es gibt nur zwei Fälle, in denen Sie Administrationsvorgänge manuell ausführen müssen:

- Sie möchten die Konfiguration ändern, z. B. durch Ändern von `field-indexed` Annotationen.
- Beim Indizierungsvorgang sind Fehler aufgetreten.

Öffnen Sie im Admin Tool die Navigationsgruppe *Dienste*, Navigationselement *Index*, um den Indexer zu konfigurieren und zu verwalten. Wenn gerade Indexer-Aufgaben ausgeführt werden, wird dies durch ein Ausrufezeichen neben dem Namen der Navigationsgruppe (*Dienste*) angezeigt. Neben dem Namen des Navigationselements (*Index*) steht die Anzahl der offenen Aufgaben. Auf diese Weise sehen Sie als Administrator, dass es in der Gruppe *Dienste* offene Aufgaben gibt, auch wenn die Navigationsgruppe nicht aufgeklappt ist. Zudem können Sie die Anzahl der offenen Aufgaben schnell herausfinden.

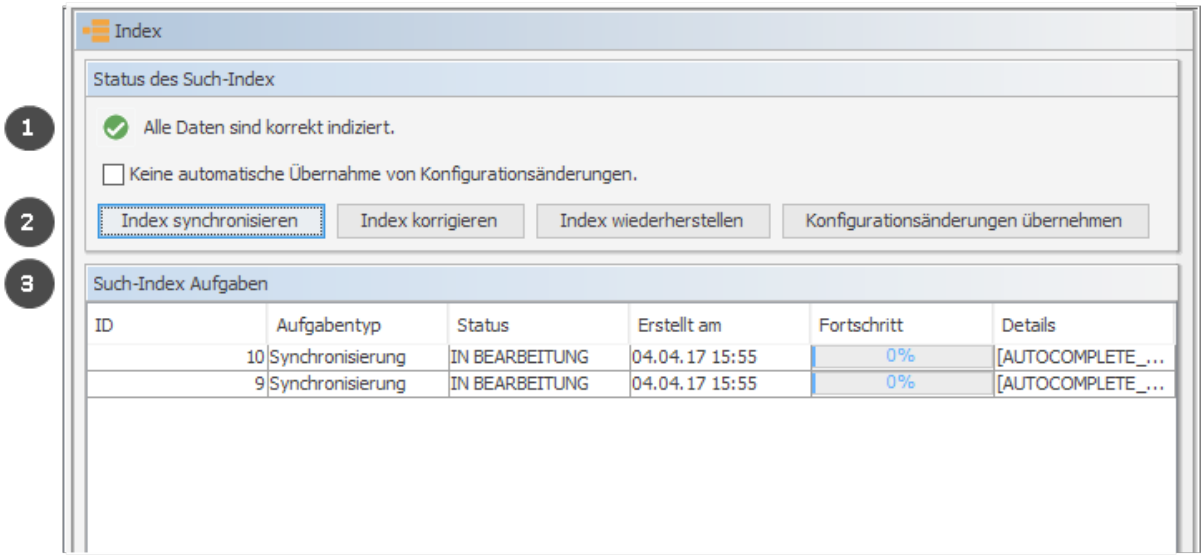


Abbildung 400: *ConSol CM Admin Tool - Dienste, Index*

Die Seite *Index* enthält drei Elemente:

- Status des Indexers (1)
- Buttons zum Verwalten des Indexers (2)
- Liste der offenen/aktiven Indexer-Aufgaben (3)

Die Anzahl der offenen Indexer-Aufgaben steht auch neben dem Navigationselement *Index*.

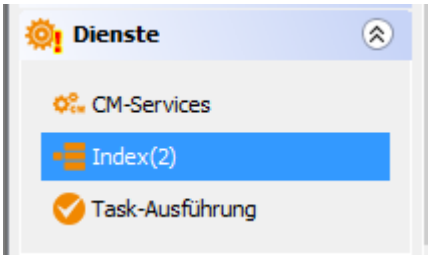


Abbildung 401: *ConSol CM Admin Tool - Services: Angabe offener Indexer-Aufgaben*

In der ersten Zeile wird der aktuelle Status des Indexers angezeigt (das ist der Wert der System-Property cmas-core-index-common, index.status):

- **GREEN**
  Alle Indexer-Aufgaben wurden korrekt ausgeführt, es ist keine Aktion erforderlich. Bei Beginn des Synchronisierungsvorgangs wird der Indexstatus auf Grün gesetzt. Wenn er erfolgreich abgeschlossen wurde, bleibt er grün. Wenn ein Problem aufgetreten ist, wird er auf Gelb oder Rot gesetzt.

- **YELLOW**
  Es wurden behebbare Probleme erkannt, gesammelt und registriert. Dieser Status wird gesetzt, wenn eine Administrationsaufgabe (bei **ausgeschalteter** automatischer Übernahme) oder eine Wiederherstellungsaufgabe erstellt wird.
- **RED**
  Es sind Fehler aufgetreten. Bitte überprüfen Sie diese. Der Index muss komplett synchronisiert werden.

Es können folgende Vorgänge durchgeführt werden:

- **Index synchronisieren**
  Der Index wird komplett (von Null) neu erstellt. Alle offenen Indexer-Aufgaben werden verworfen. Wählen Sie einen Wert für jede der folgenden Optionen, um die Reihenfolge der Indizierung zu definieren. Dies wirkt sich auf die benötigte Zeit, bis spezielle Suchergebnisse verfügbar sind, aus.
  - **Tickets**
    Indiziere offene Tickets aller Queues, dann geschlossene Tickets aller Queues - Indiziert zuerst alle offenen Tickets und dann alle geschlossenen Tickets.
    Indiziere offene und geschlossene Tickets jeder Queue - Indiziert eine Queue nach der anderen.
  - **Attachments**
    Indiziere Tickets zusammen mit deren Attachments - Komplette Indizierung, dauert länger.
    Indiziere Tickets ohne Attachments, dann erneut mit Attachments - Schnelle Indizierung aller Tickets, damit die Ticketsuche schnell auf dem aktuellen Stand ist. Danach werden auch die Attachments indiziert, was länger dauern kann.
- **Index korrigieren**
  Indexer-Aufgaben, die nicht erfolgreich ausgeführt wurden, werden neu gestartet. Die Aufgaben können in der Aufgabenliste des Indexers ausgewählt werden.
- **Index wiederherstellen**
  Es kann ein Zeitabschnitt ausgewählt werden. Alle Änderungen, die in ConSol CM während dieses Zeitraums vorgenommen wurden, werden (neu) indiziert.
- **Konfigurationsänderungen übernehmen**
  Klicken Sie auf diesen Button, um die Änderungen zu übernehmen, wenn Sie ein zuvor nicht indiziertes Ticketfeld, Kundenfeld oder Ressourcenfeld auf `field-indexed` gesetzt haben. Dies ist erforderlich, wenn die Checkbox *Keine automatische Übernahme* von Konfigurationsänderungen ausgewählt ist. Wenn die Checkbox inaktiv ist, werden die Änderungen automatisch übernommen, wenn Sie eine neue Annotation gesetzt haben.

  Beachten Sie, dass die automatische Übernahme von Konfigurationsänderungen die Systemleistung beeinträchtigen kann.

Nur für erfahrene Administratoren:

Die Operation *Konfigurationsänderungen übernehmen* kann auch über die ConSol CM MBean `consol.cmas.admin.global.core.indexManagement`, Methode `commitAdministrativeChanges()` ausgeführt werden. Sie können dazu grafische Tools wie JConsole oder einen Befehl, z. B. mit REST und Jolokia verwenden. Wenn Sie zu diesem Thema Hilfe benötigen, wenden Sie sich bitte an Ihren ConSol CM-Consultant.

Wenn es offene Aufgaben in der Liste der Indexer-Aufgaben gibt, werden folgende Daten für die einzelnen Aufgaben angezeigt:

- **ID**
  ID der Aufgabe
- **Aufgabentyp**
  Es sind drei Typen verfügbar:
  - **Synchronisierung**
    - Erstellt den gesamten Index neu.
    - Wird manuell im Admin Tool über den Befehl *Index synchronisieren* gestartet.
    - Vor dem Start werden alle anderen Indexer-Aufgaben entfernt.
  - **Konfigurationsänderungen**
    - Wird automatisch erstellt, wenn eines der folgenden Elemente aktualisiert wurde: Bereich, Queue, Wert in Sortierter Liste, Ticketfunktion, Ticketbearbeiter, unterstützte Sprachumgebung, Rolle.
    - Wird automatisch verarbeitet, wenn die Option *Keine automatische Übernahme von Konfigurationsänderungen* nicht markiert ist.
    - Mit dem Befehl *Konfigurationsänderungen übernehmen* werden alle die Konfigurationsänderungen betreffenden Aufgaben gestartet.
  - **Wiederherstellung**
    - Wird automatisch erstellt, wenn während der Indexaktualisierung ein Fehler aufgetreten ist.
    - Enthält Informationen über die Objekte, die die Probleme hervorgerufen haben.
    - Alle Wiederherstellungsaufgaben werden mit dem Button *Index korrigieren* gestartet.
- **Status**
  z. B. RUNNING
- **Erstellt am**
  Zeitstempel, wann die Aufgabe erstellt wurde.
- **Fortschritt**
  Ein Fortschrittsbalken, der den Fortschritt der Aufgabe in Prozent angezeigt.
- **Details**
  Eine Liste der Objekte, die in dieser Aufgabe (neu) indiziert wurden.

Beachten Sie, dass die Daten im Index immer mit der ConSol CM-Datenbank synchronisiert werden, d. h. während der Indexaktualisierung werden keine Daten aus den Indexdateien gelöscht/entfernt. Der Index ist während des Synchronisierungsvorgangs voll funktionsfähig, d. h. die Suchvorgänge (Detail- und Schnellsuche) können in ihrem vollen Funktionsumfang und mit den vollständigen Datensätzen genutzt werden. Änderungen, die vorgenommen wurden, nachdem die Synchronisierung gestartet wurde, spiegeln sich sofort im Index wider, da diese Daten eine höhere Priorität haben, als die Daten, die aufgrund einer manuell mit dem Admin Tool ausgelösten Indexaktualisierung synchronisiert werden.

Wenn ein Administrator die Indexaktualisierung manuell gestartet hat (*Index synchronisieren*), werden alle anderen Index-Aufgaben entfernt, bevor diese Aktualisierung gestartet wird.

Für Indexer und Index relevante System-Properties

Die folgenden System-Properties sind für den Indexer relevant (siehe Abbildung). Eine detaillierte Beschreibung der System-Properties für den Indexer finden Sie in System-Properties und im Abschnitt Konfiguration von Indexer und Suche der Liste der System-Properties nach Bereich.

Die folgende Abbildung zeigt die Properties des Moduls `cmas-core-index-common`. Sie gelangen auf diese Seite, indem Sie das Navigationselement *System-Properties* in der Navigationsgruppe *System* öffnen und das Modul auswählen.

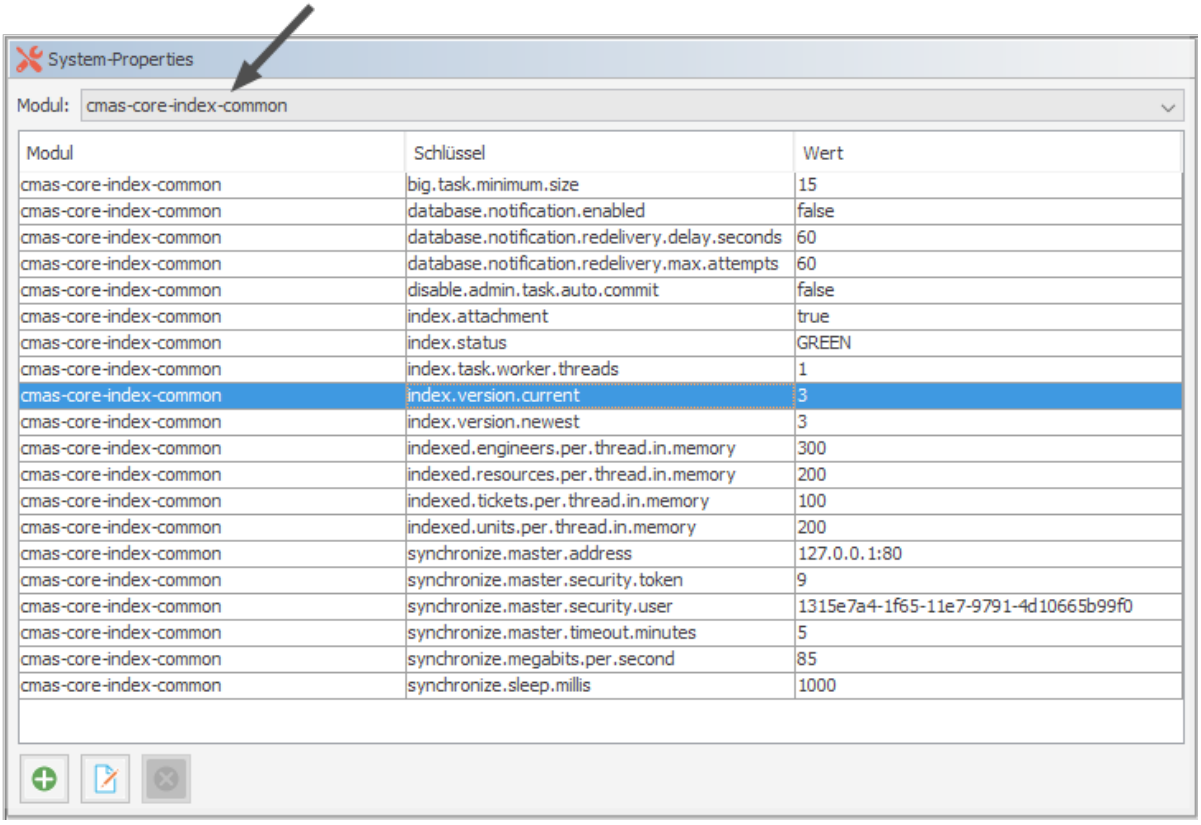


Abbildung 402: *ConSol CM Admin Tool - System, System-Properties: Indexer-Properties*

Indexer-Dienste

Für die Indizierung sind zwei ConSol CM-Dienste wichtig:

- **Index changes notifier**
  Dieser Dienst erstellt Nachrichten im persistenten Speicher mit Benachrichtigungen, wenn für den Index relevante Änderungen erfolgen.
  Das Stoppen von *index changes notifier* ist **nicht** ungefährlich. Wenn das Indexer-Modul feststellt, dass *index changes notifier* gestoppt wurde, und es eine Nachricht gibt, die an den persistenten Speicher gesendet werden muss, setzt der Indexer die Property des Indexstatus (cmas-core-index-common, index.status) auf „RED“, d. h. der Index muss komplett synchronisiert werden.
- **Index changes receiver**
  Das Verhalten dieses Dienstes hängt von dem Typ des Nodes, auf dem er läuft, ab.
  - Auf einem Master-Node lädt der *index changes receiver* die Tickets und erstellt die entsprechenden Lucene-Dokumente.
  - Auf einem Slave-Node (nur im Cluster möglich) fragt er den Master-Indizierungsserver über HTTP ab, um die Lucene-Dateien mit den indizierten Daten herunterzuladen.
  - In beiden Fällen liest dieser Dienst aus dem persistenten Speicher und aktualisiert den Indexer.
    Das Stoppen von *index changes receiver* ist ungefährlich. Nach einem Neustart werden alle fehlenden Änderungen aus dem persistenten Speicher übernommen.

Siehe auch Abschnitt CM-Services.

F.4.2.7 Indexer-Aktualisierung

Dieser Abschnitt beschreibt die E-Mail-Funktionen in einer Umgebung mit einem Server. Informationen über Umgebungen mit mehreren Servern finden Sie im *ConSol CM Cluster Manual*.

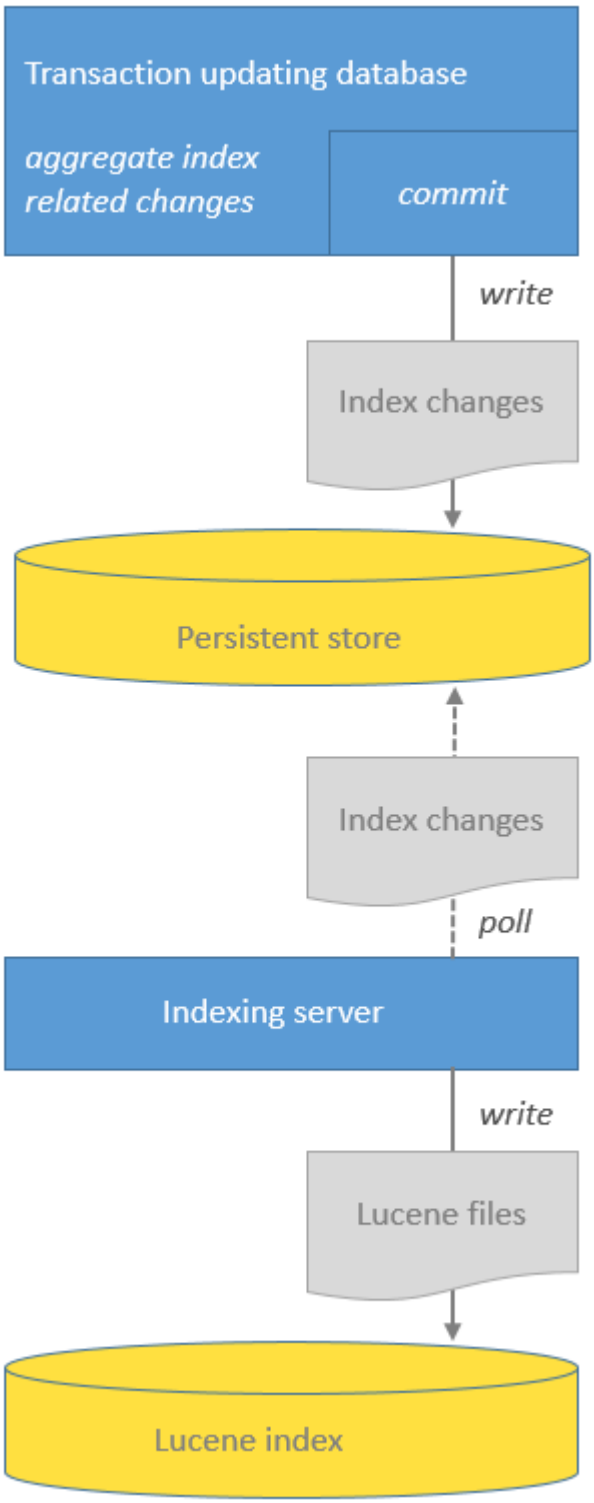


Abbildung 403: *Indexer-Architektur in einer Umgebung mit einem Server*

Änderungen, die Index-Aktualisierungen benötigen

Für den Indexer sind zwei Arten von Änderungen relevant:

- **Operative Änderungen**

  Dies sind Änderungen an folgendem Inhalt:

  - **Werte von Datenfeldern:** Werte von Feldern (Ticket-, Kunden- und Ressourcenfelder), die mit der Annotation `field-indexed` für die Indizierung konfiguriert sind
  - **Bearbeiterdaten:** E-Mail, Vorname, Nachname
  - **Ticketdaten:** Attachments (sofern nicht anderweitig konfiguriert), Erstellungsdatum, Bearbeiter, Protokoll, Name, Queue, zusätzliche Bearbeiter, Thema, Sicht

  Der Index wird automatisch aktualisiert (angestoßen vom Service `Index changes notifier`), wenn diese Änderungen erfolgen.

- **Administrative Änderungen**

  Dies sind Änderungen von bestimmten Konfigurationsparametern:

  - Bereiche
  - Queues
  - Werte von Sortierten Listen
  - Ticketfunktionen
  - Ticketbearbeiter
  - Unterstützte Sprachen
  - Rollen
  - Annotation `field-indexed` von Ticket-, Kunden oder Ressourcenfeldern

  Die Verarbeitung von administrativen Änderungen hängt von der Option *Keine automatische Übernahme von Konfigurationsänderungen* ab:

  - Wenn sie nicht markiert ist, werden die Änderungen automatisch verarbeitet.
  - Wenn sie markiert ist, werden die Änderungen für eine nachgelagerte Verarbeitung in der Tabelle `cmas_index_administrative_task` gespeichert. Die Verarbeitung kann durch Klicken auf den Button *Konfigurationsänderungen übernehmen* im Admin Tool gestartet werden.



Nur für erfahrene Administratoren:

Die Operation *Konfigurationsänderungen übernehmen* kann auch über die ConSol CM MBean `consol.cmas.admin.global.core.indexManagement`, Methode `commitAdministrativeChanges()` ausgeführt werden. Sie können dazu grafische Tools wie JConsole oder einen Befehl, z. B. mit REST und Jolokia verwenden. Wenn Sie zu diesem Thema Hilfe benötigen, wenden Sie sich bitte an Ihren ConSol CM-Consultant.

Modi der Indexer-Aktualisierung

Der Indexer speichert die Entitäten, die eine Index-Aktualisierung benötigen, in einem persistenten Speicher. Dieser persistente Speicher kann entweder eine JMS-Queue (`queue/cm6-index`) oder eine Datenbanktabelle (`cmas_index_update_serialized`) sein.

Welche von beiden verwendet wird, ist in der System-Property cmas-core-index-common, database.notification.enabled definiert.

- **false**
  JMS-Modus, die JMS-Queue `queue/cm6-index` wird verwendet.
- **true**
  Datenbank-Modus, die Datenbanktabelle `cmas_index_update_serialized` wird (als persistenter Speicher) für die Indexer-Transaktionen verwendet. Der Indizierungsserver fragt diese Datenbank ab. (Sollte nicht in CM-Versionen vor 6.9.4.1 verwendet werden.)

Prinzip der Indexer-Aktualisierung (Umgebungen mit einem Server)

Der **Indizierungsserver** fragt den persistenten Speicher, ob neue Einträge vorhanden sind. Wenn es neue Einträge gibt, werden Indexer-Tasks in den Tabellen `cmas_index_update_task` und `cmas_index_update_part` erstellt. Der Indizierungsserver führt dann die Tasks aus `cmas_index_update_task` und `cmas_index_batch_update_task` aus. Auf diese Weise wird der Index aktualisiert.

Auf dem Indizierungsserver werden die administrativen Tasks in der Datenbanktabelle `cmas_index_administrative_task` gespeichert.



Beachten Sie, dass die Daten im Index immer mit der ConSol CM-Datenbank synchronisiert werden, d. h. während der Indexaktualisierung werden keine Daten aus den Indexdateien gelöscht/entfernt. Der Index ist während des Synchronisierungsvorgangs voll funktionsfähig, d. h. die Suchvorgänge (Detail- und Schnellsuche) können in ihrem vollen Funktionsumfang und mit den vollständigen Datensätzen genutzt werden. Änderungen, die vorgenommen wurden, nachdem die Synchronisierung gestartet wurde, spiegeln sich sofort im Index wider, da diese Daten eine höhere Priorität haben, als die Daten, die aufgrund einer manuell mit dem Admin Tool ausgelösten Indexaktualisierung synchronisiert werden.

Wenn ein Administrator die Indexaktualisierung manuell gestartet hat (*Index synchronisieren*), werden alle anderen Index-Aufgaben entfernt, bevor diese Aktualisierung gestartet wird.

Fehlschlagen einer Aktualisierungs-Task

Wenn die Ausführung einer `cmas_index_update_task` fehlschlägt, wird eine neue Task `cmas_index_update_task` des Typs `REPAIR` erzeugt. Diese Task wartet, bis der Administrator über das Admin Tool, Navigationsgruppe *Dienste*, Navigationselement *Index*, die Aktion *Index reparieren* ausführt. Das Vorhandensein einer Reparatur-Task setzt die System-Property des Indexstatus auf „YELLOW“.

F.4.3 Action Framework - Suche-Aktionen

F.4.3.1 Einführung in Suche-Aktionen

Es ist möglich, Aktionen für Suchergebnisse zu definieren. Diese Aktionen werden so ähnlich dargestellt wie Workflow-Aktivitäten. Sie beziehen sich auf die Ergebnislisten mit:

- Tickets
- Kunden (= Units)
- Ressourcen

Suche-Aktionen werden für den gesamten Satz an Suchergebnissen ausgeführt. Sie können Suche-Aktionen konfigurieren, um den ConSol CM-Bearbeitern verschiedene Massenaktionen anzubieten, z. B. Zuweisen aller Tickets in der Ergebnisliste zum aktuellen Bearbeiter (siehe folgende Abbildung), Senden einer bestimmten E-Mail an alle Kunden in der Ergebnisliste, oder Erstellen eines Wartungstickets für alle PCs (Ressourcen) in der Ergebnisliste.

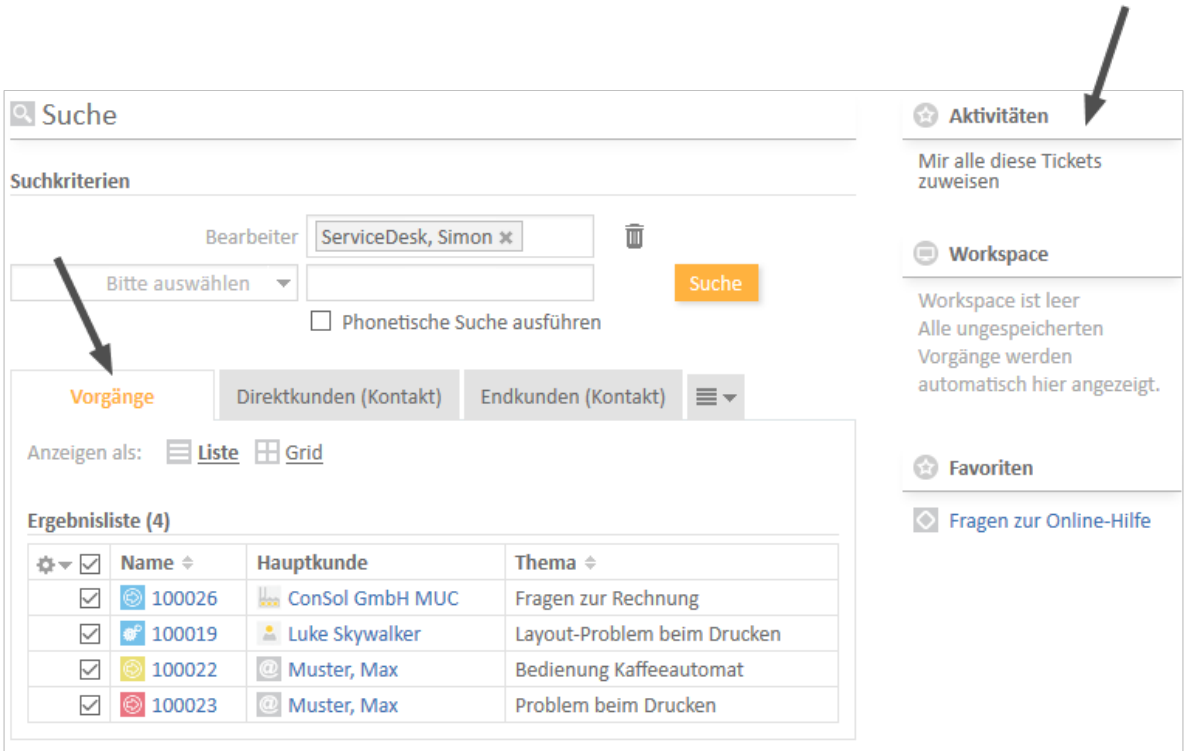


Abbildung 404: *ConSol CM Web Client - Suche-Aktionen für Tickets*

Eine Suche-Aktion (Suchergebnis-Aktion) wird im Web Client nur angezeigt, wenn

- der Bearbeiter die Berechtigung *Ausführen* für alle Elemente der Ergebnisliste hat.
- das Bedingungsskript „true“ zurückgegeben hat (sofern es ein Bedingungsskript gibt).

Jede Suche-Aktion wird auf Grundlage von Admin-Tool-Skripten der folgenden Skripttypen implementiert.

- **Verwendung für Tickets:**
  - Suche-Aktion für Tickets
  - Suche-Bedingung für Tickets
- **Verwendung für Ressourcen:**
  - Suche-Aktion für Ressourcen
  - Suche-Bedingung für Ressourcen
- **Verwendung für Kunden:**
  - Suche-Aktion für Kunden
  - Suche-Bedingung für Kunden

Beachten Sie, dass Suche-Aktionen für folgende Mengen durchgeführt werden können:

a) nur für den momentan angezeigten Teil der Ergebnisliste, z. B. für 20 von 100 Ergebnissen

ODER

b) für die gesamte Ergebnisliste, z. B. alle 100 Ergebnisse

Das Verhalten der Suche-Aktion hängt von Folgendem ab:

- Die Implementierung des Admin-Tool-Skripts. Eine detaillierte Erklärung dazu finden Sie im Abschnitt Der Ergebnissatz in Admin-Tool-Skripten für Suche-Aktionen.
- Die Auswahl der Zeilen, siehe Seitenanpassung, enableRowSelection.

F.4.3.2 Konfigurieren von Suche-Aktionen mit dem Admin Tool

Suche-Aktionen werden im Admin Tool definiert. Je nach Typ der Suche-Aktion müssen Sie zur Implementierung zwei oder drei Schritte ausführen.

Für die Implementierung einer Suche-Aktion für Tickets durchzuführende Schritte

1. Definieren und schreiben Sie das Admin-Tool-Skript (Navigationsgruppe *System*, Navigationselement *Skripte und Templates*)
   - Obligatorisch: ein Skript des Typs *Suche-Aktion für Tickets* (siehe folgende Abbildung)
   - Optional: ein Skript des Typs *Suche-Bedingung für Tickets*

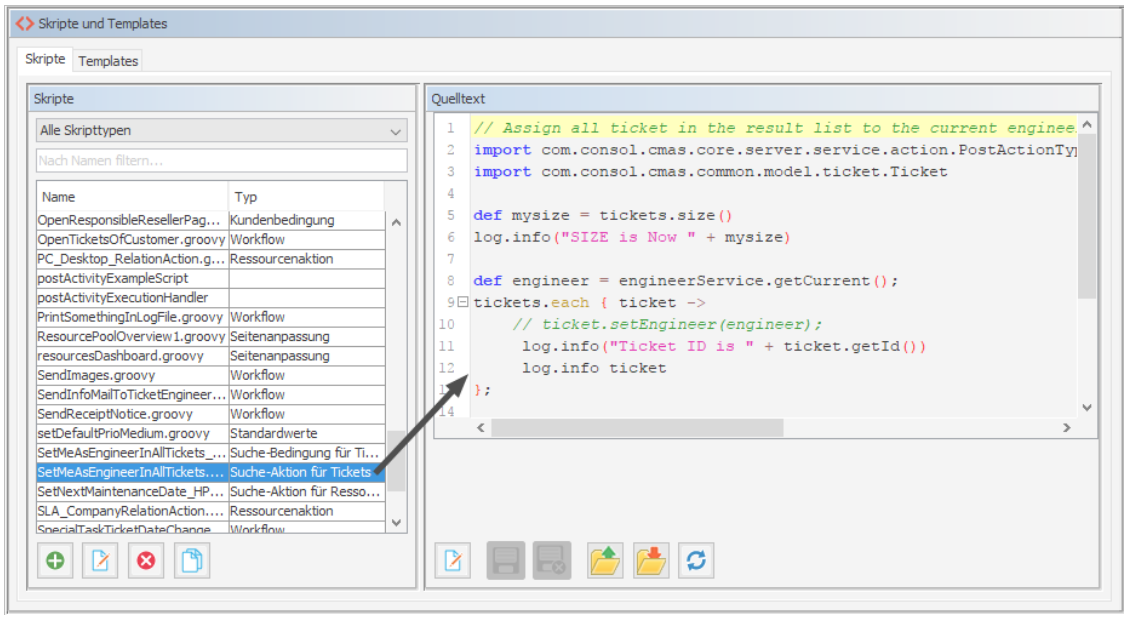


Abbildung 405: *ConSol CM Admin Tool - System, Skripte und Templates: Skript für Suche-Aktion für Tickets*

Beispielskript

```
// Assign all tickets in the result list to the current engineer
import com.consol.cmas.core.server.service.action.PostActionType
import com.consol.cmas.common.model.ticket.Ticket

def engineer = engineerService.getCurrent();
tickets.each { ticket ->
   ticket.setEngineer(engineer);
};
return actionScriptResultFactory.getPostAction("success",
 "cmweb.search.assigned").withRefreshContent();
```

Code-Beispiel 61: *Suche-Aktionsskript für Tickets*

> Eine detaillierte Erklärung über das Schreiben von Skripten für das ConSol CM Action Framework finden Sie in Skripte für das Action Framework.

2. Definieren Sie die Suche-Aktion im Navigationselement *Suche-Aktionen* der Navigationsgruppe *Tickets*. Dabei wird das Aktionsskript für die Suche-Aktion zugewiesen. Optional wird auch ein Bedingungsskript zugewiesen. Sie können den lokalisierten Begriff für die Ticketaktion über den Button *Lokalisieren* eingeben. Eine detaillierte Beschreibung des Lokalisierungsmechanismus finden Sie im Abschnitt Lokalisierung von Objekten allgemein, Typ 1.

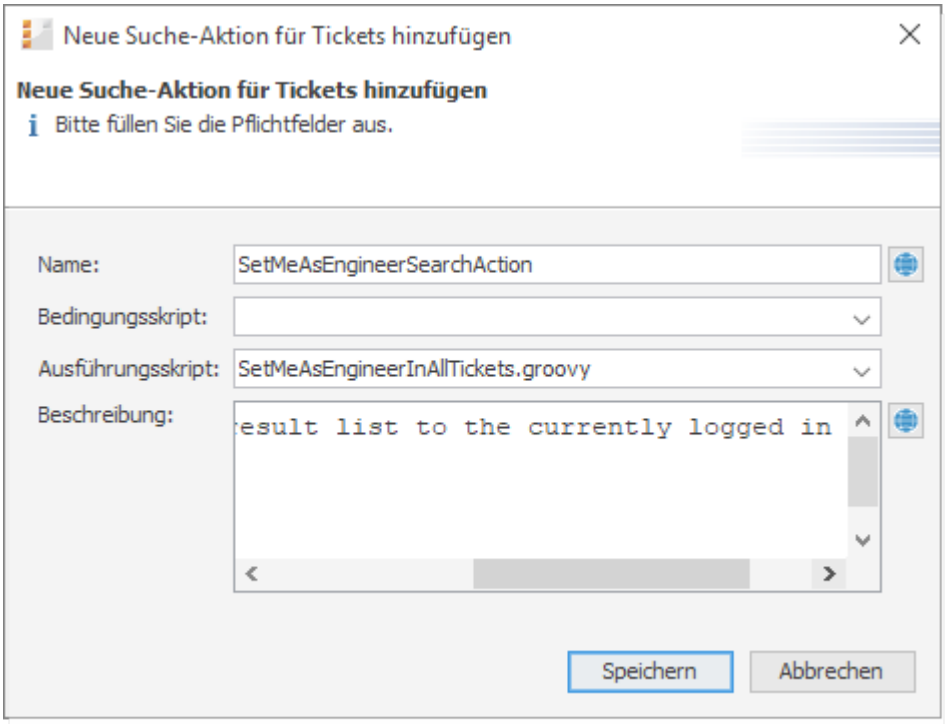


Abbildung 406: *ConSol CM Admin Tool - Tickets, Suche-Aktionen: Definieren einer neuen Suche-Aktion*

Für die Implementierung einer Suche-Aktion für Ressourcen durchzuführende Schritte

1. Definieren und schreiben Sie das Admin-Tool-Skript
   - Obligatorisch: ein Skript des Typs *Suche-Aktion für Ressourcen*
   - Optional: ein Skript des Typs *Suche-Bedingung für Ressourcen*

   Beispielskript, mit dem das nächste Wartungsdatum aller gefundenen HP-Drucker auf ein Datum in zwei Wochen gesetzt wird:

```
// Schedule maintenance date for all selected HP printers to a date in two
 weeks from today

import groovy.time.TimeCategory
import com.consol.cmas.core.server.service.action.PostActionType

println 'RUNNING Search action resources script!'

def now = new Date()

use(TimeCategory) {
   nextMaintenanceDate = now + 2.weeks
}

resources.each { res ->
   res.setFieldValue("HP_Printer_Fields_basic", "NextMaintenanceDate",
    nextMaintenanceDate)
}

return actionScriptResultFactory.getPostAction(PostActionType.SUCCESS,
 "cmweb.search.assigned").withRefreshContent();
```

Code-Beispiel 62: *Suche-Aktionsskript für Ressourcen*



Eine detaillierte Erklärung über das Schreiben von Skripten für das ConSol CM Action Framework finden Sie in Skripte für das Action Framework.

2. Definieren Sie die Suche-Aktion im Navigationselement *Aktionen* der Navigationsgruppe *Ressourcen*. Dabei wird das Aktionsskript für die Suche-Aktion zugewiesen. Optional wird auch ein Bedingungsskript zugewiesen. Sie können den lokalisierten Begriff für die Ressourcenaktion über den Button *Lokalisieren* eingeben. Eine detaillierte Beschreibung des Lokalisierungsmechanismus finden Sie im Abschnitt Lokalisierung von Objekten allgemein, Typ 1.

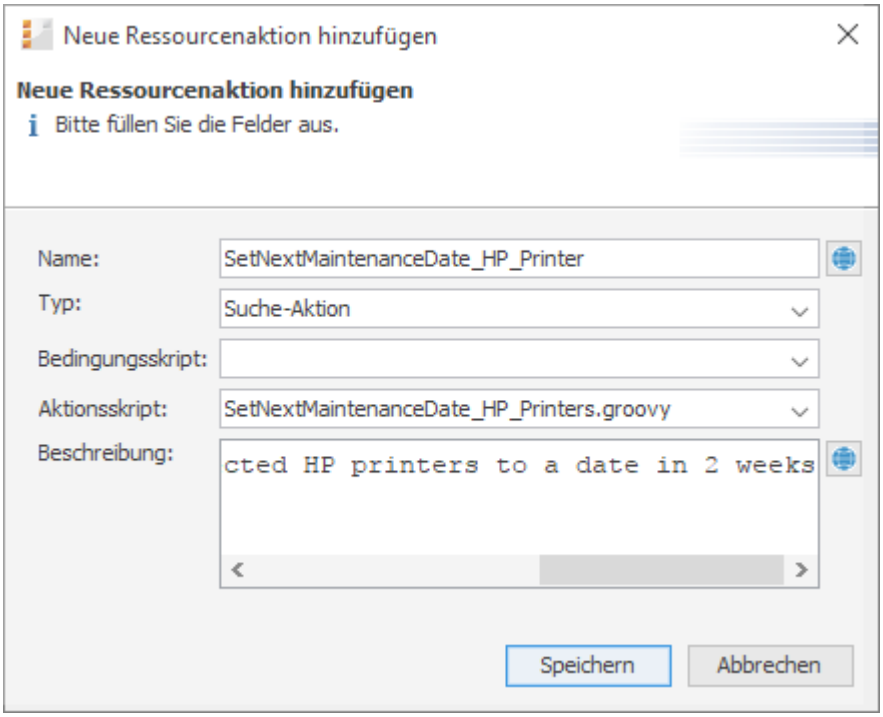


Abbildung 407: *ConSol CM Admin Tool - Ressourcen, Aktionen: Definieren einer Suche-Aktion für Ressourcen*

3. Weisen Sie die Aktion einem oder mehreren Ressourcentypen zu (Navigationsgruppe *Ressourcen*, Navigationselement *Datenmodelle*, Tab *Suche-Aktionen* jedes einzelnen Ressourcentyps).

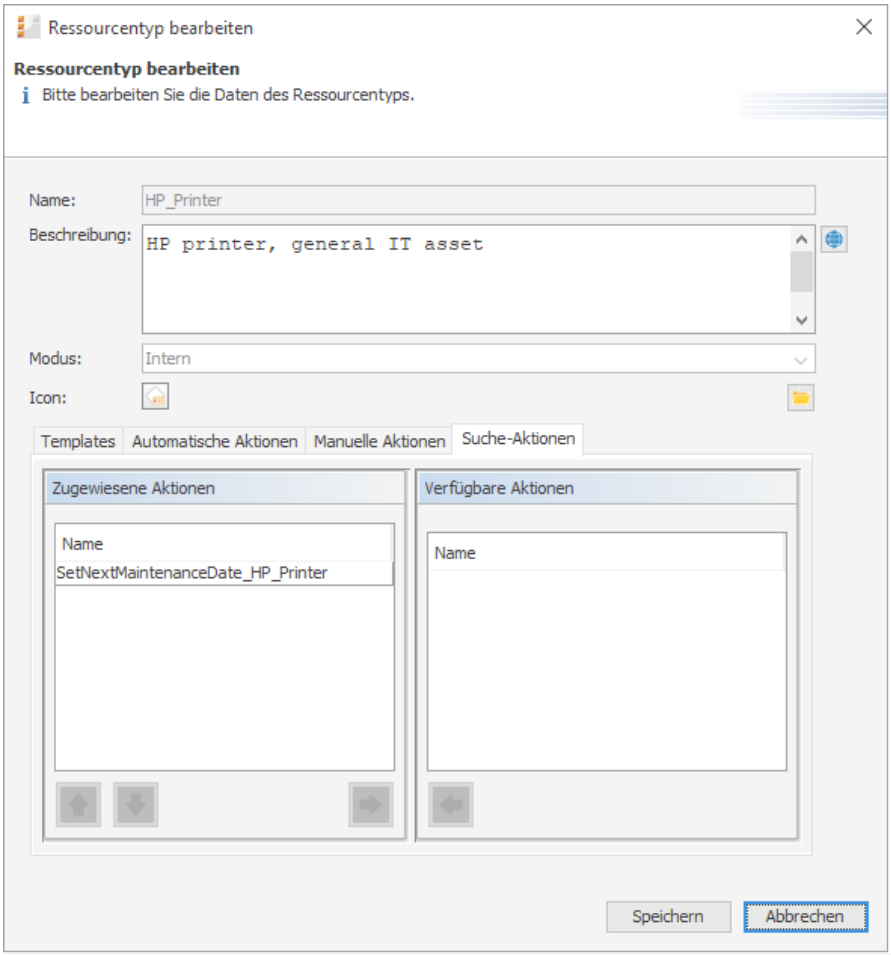


Abbildung 408: *ConSol CM Admin Tool - Ressourcen, Aktionen: Zuweisen einer Suche-Aktion zu einem Ressourcentyp*

4. Überprüfen Sie im Web Client, dass die Aktion funktioniert.

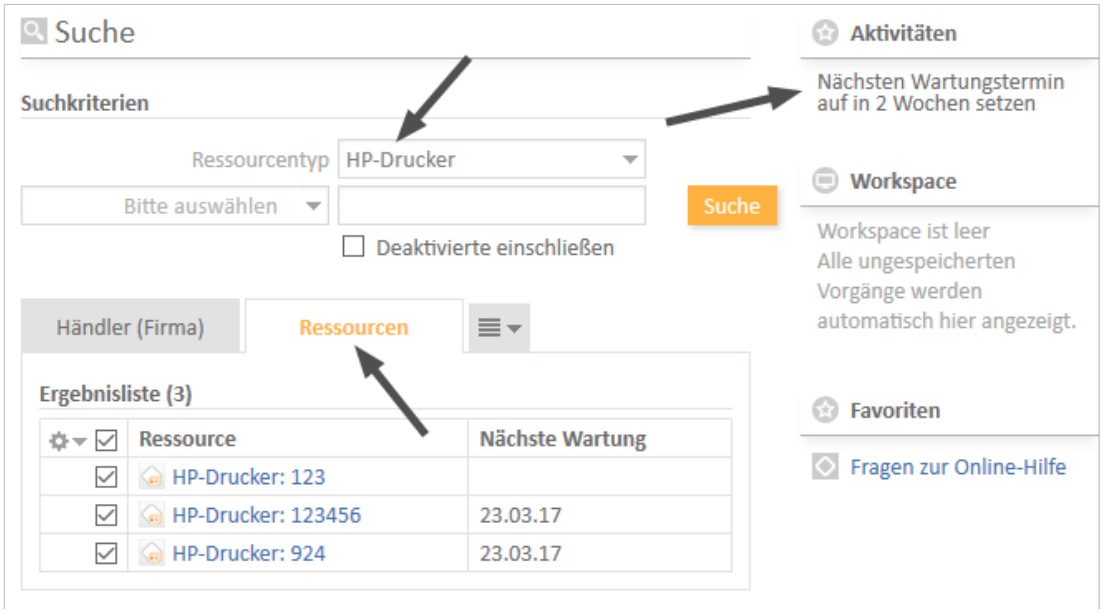


Abbildung 409: *ConSol CM Web Client - Suche-Aktion für Ressourcen, 1*

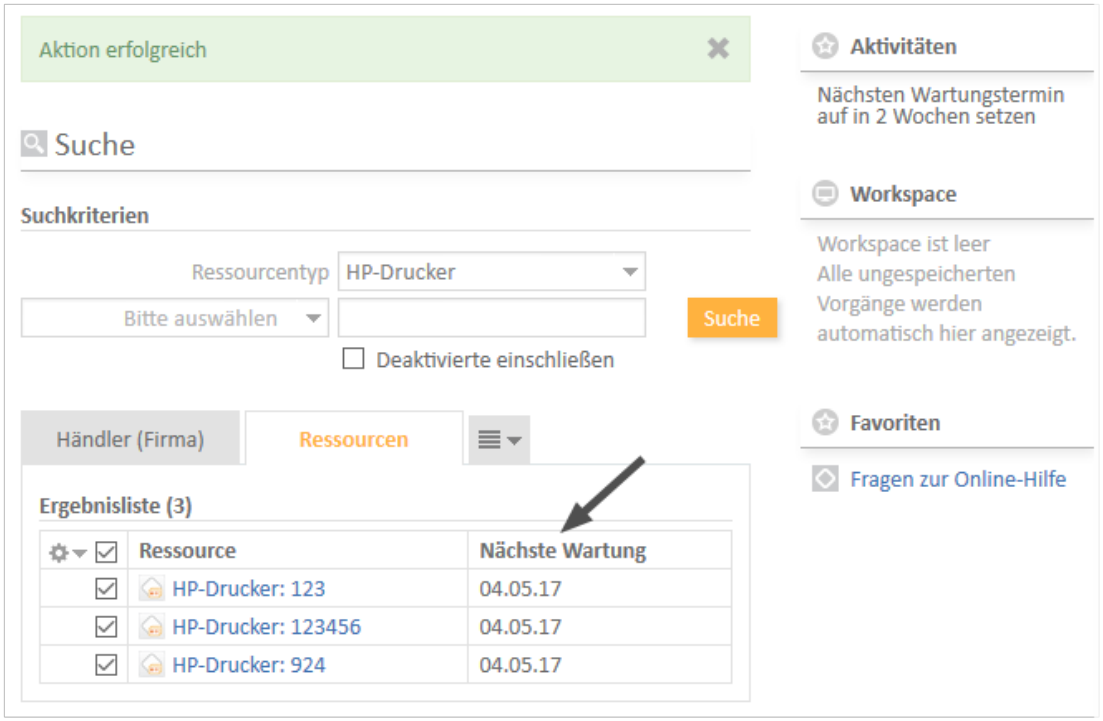


Abbildung 410: *ConSol CM Web Client - Suche-Aktion für Ressourcen, 2*

Für die Implementierung einer Suche-Aktion für Kunden (= Units) durchzuführende Schritte

1. Definieren und schreiben Sie das Admin-Tool-Skript
   - Obligatorisch: ein Skript des Typs *Suche-Aktion für Kunden*
   - Optional: ein Skript des Typs *Suche-Bedingung für Kunden* (in diesem Beispiel arbeiten wir mit einem Bedingungsskript)

   Beispiel für ein Aktionsskript, mit dem für jeden Kontakt in der Ergebnisliste ein Ticket erstellt wird:

```
// For all contacts in the result set, a new ticket in the queue Service Desk
 will be created
import com.consol.cmas.core.server.service.action.PostActionType
import com.consol.cmas.common.model.ticket.Ticket
import com.consol.cmas.common.model.customfield.Unit
import com.consol.cmas.common.model.resource.*
import com.consol.cmas.common.service.resource.*
import com.consol.cmas.common.model.ticket.Queue
import com.consol.cmas.common.model.resource.meta.*
import com.consol.cmas.core.server.service.action.*
import groovy.time.TimeCategory

println 'Search Result Action Script CreateAnSDTicketForAllSelectedContacts
 STARTED ... '
// deadline is a required field in Service Desk tickets!
def now = new Date()
def deadline
use(TimeCategory) {
   deadline = now + 2.weeks
}

Queue qu = queueService.getByName("ServiceDesk")
units.each{ cont ->
   def cont_name = cont.customer_name
   println 'Customer name is now : ' + cont_name
   Ticket newtic = new Ticket()
   newtic.setQueue(qu)
   newtic.set("serviceDesk_fields.desiredDeadline", deadline)
   newtic.setSubject("New Ticket due to Search Result for customer" + cont_
    name)
   newtic.set("helpdesk_standard.priority","low")
   ticketService.createWithUnit(newtic,cont)
   println 'New Ticket created for customer ' + cont_name
}
return actionScriptResultFactory.getPostAction("success",
 "cmweb.search.assigned").withRefreshContent();
```

Code-Beispiel 63: *Suche-Aktionsskript für Units*



Stellen Sie sicher, dass ...

- Sie im Skript nur Kundenfelder verwenden, die im Kundendatenmodell der Kundengruppe vorhanden sind, für die das Suche-Aktionsskript verwendet wird! Zum Beispiel: Um den Namen eines Kontakts zu referenzieren, muss der exakte Feldname verwendet werden. Im obigen Beispiel ist das *customer_name*)! Der erste Schritt vor der Implementierung des Suche-Aktionsskripts ist daher das Überprüfen der erforderlichen Felder im Kundendatenmodell.
- die Kundengruppe der Queue zugewiesen ist, in der die neuen Tickets erstellt werden sollen.

Beispiel für ein Bedingungsskript, mit dem die Suche-Aktion nur angezeigt wird, wenn die Liste mehr als fünf Einträge enthält.

```
if (units.size() >= 5){
   return true
} else {
   return false
}
```

Code-Beispiel 64: *Suche-Bedingungsskript für Units*



Eine detaillierte Erklärung über das Schreiben von Skripten für das ConSol CM Action Framework finden Sie in Skripte für das Action Framework.

2. Definieren Sie die Suche-Aktion im Navigationselement *Aktionen* der Navigationsgruppe *Kunden*. Dabei wird das Aktionsskript für die Suche-Aktion zugewiesen. Optional wird auch ein Bedingungsskript zugewiesen. Sie können den lokalisierten Begriff für die Kundenaktion über den Button *Lokalisieren* eingeben. Eine detaillierte Beschreibung des Lokalisierungsmechanismus finden Sie im Abschnitt Lokalisierung von Objekten allgemein, Typ 1.

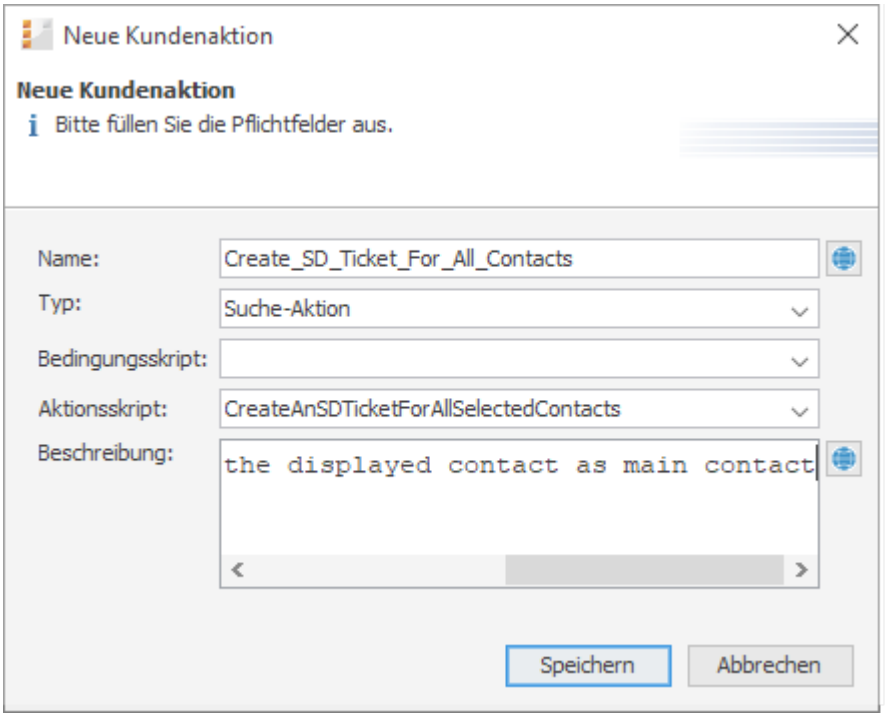


Abbildung 411: *ConSol CM Admin Tool - Kunden, Aktionen: Definieren einer Suche-Aktion für Kunden*

3. Fügen Sie das Bedingungsskript hinzu.

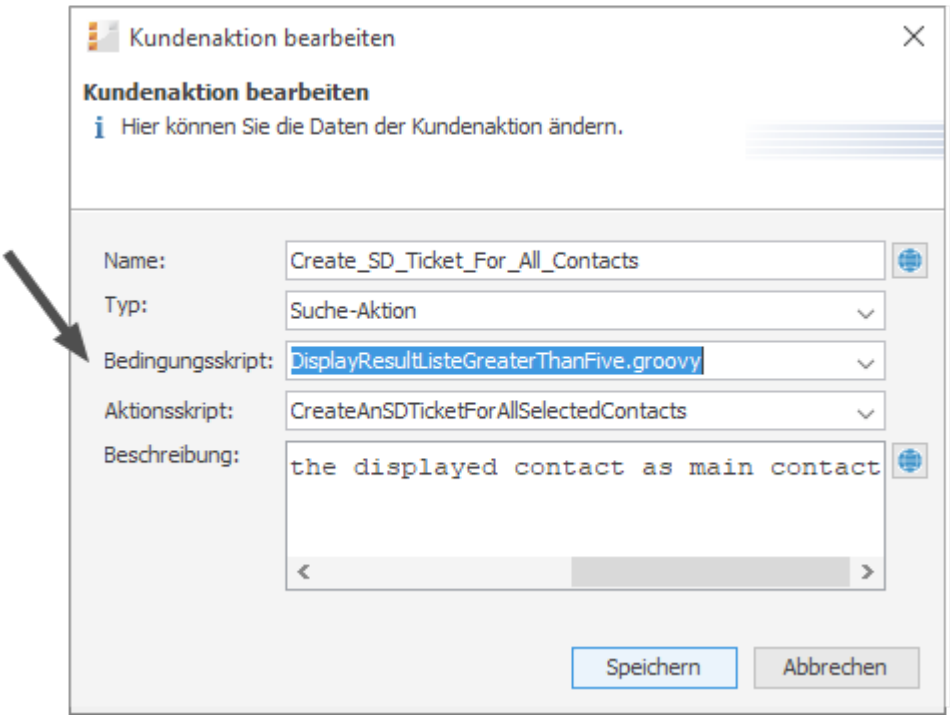


Abbildung 412: *ConSol CM Admin Tool - Kunden, Aktionen: Definieren einer Suche-Bedingung*

4. Weisen Sie die Aktion einer oder mehreren Kundengruppen zu (Navigationsgruppe *Kunden*, Navigationselement *Kundengruppen*, Tab *Suche-Aktionen* jeder einzelnen Kundengruppe). Weisen Sie die Aktionen den Kontakten (*Kontaktsuche-Aktionen*) und/oder Firmen (*Firmensuche-Aktionen*) dieser Kundengruppe zu.
   Beachten Sie, dass in der Tabelle sowohl die Aktions- als auch die Bedingungsskripte angeboten werden.

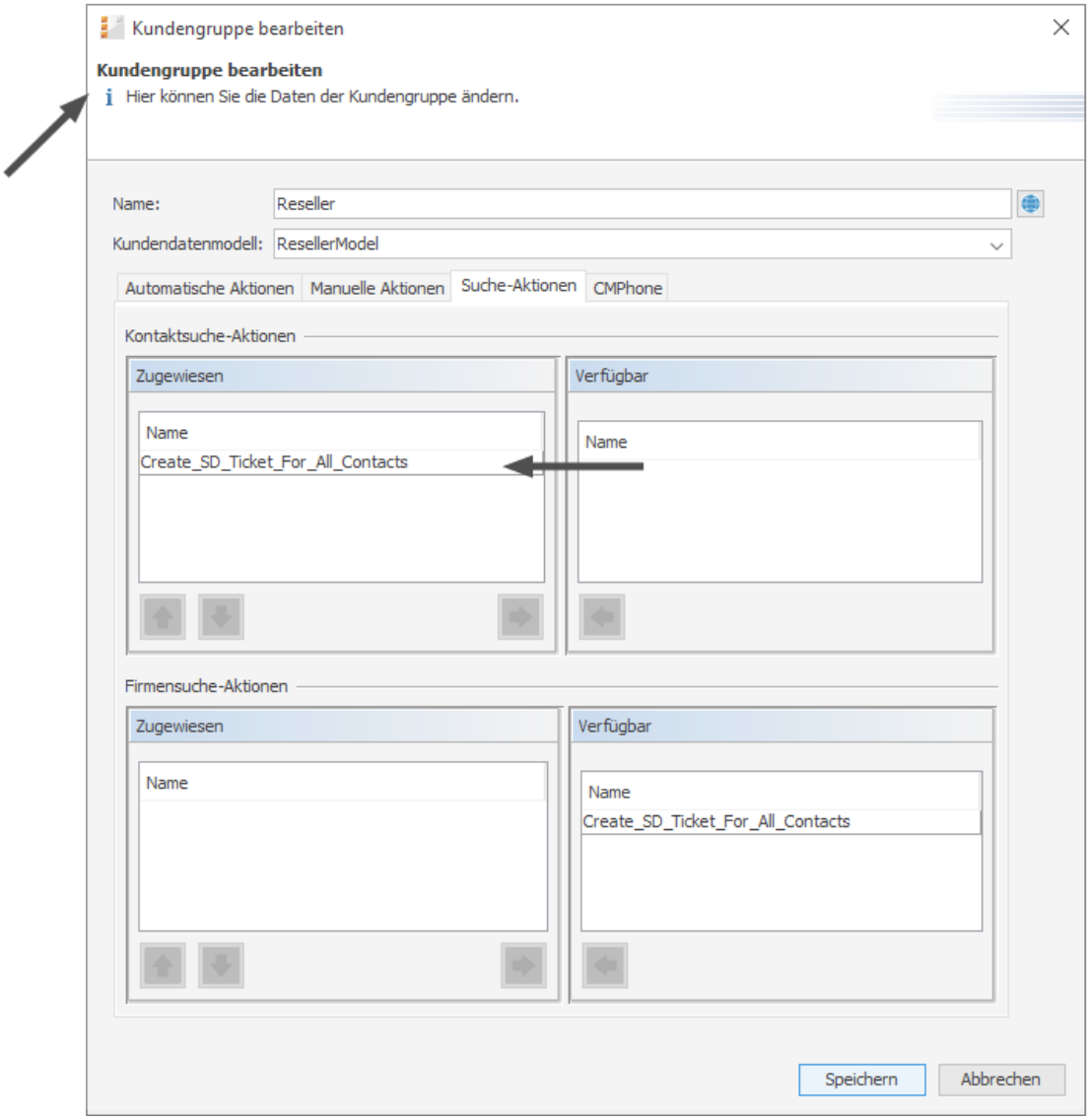


Abbildung 413: *ConSol CM Admin Tool - Kunden, Kundengruppen: Zuweisen einer Suche-Aktion zu einem Kontaktobjekt innerhalb einer Kundengruppe*

5. Überprüfen Sie im Web Client, dass die Aktion funktioniert.

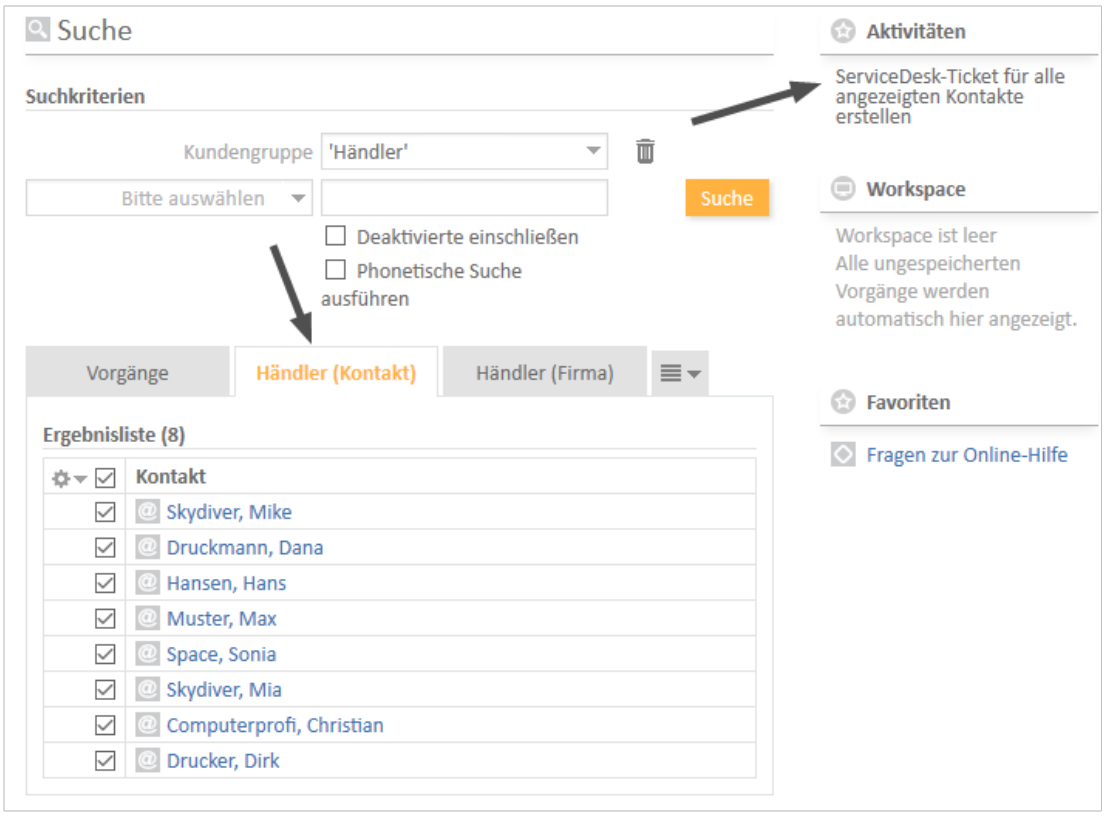


Abbildung 414: *ConSol CM Web Client - Suche-Aktion für Kunden (hier: Kontakte), 1*

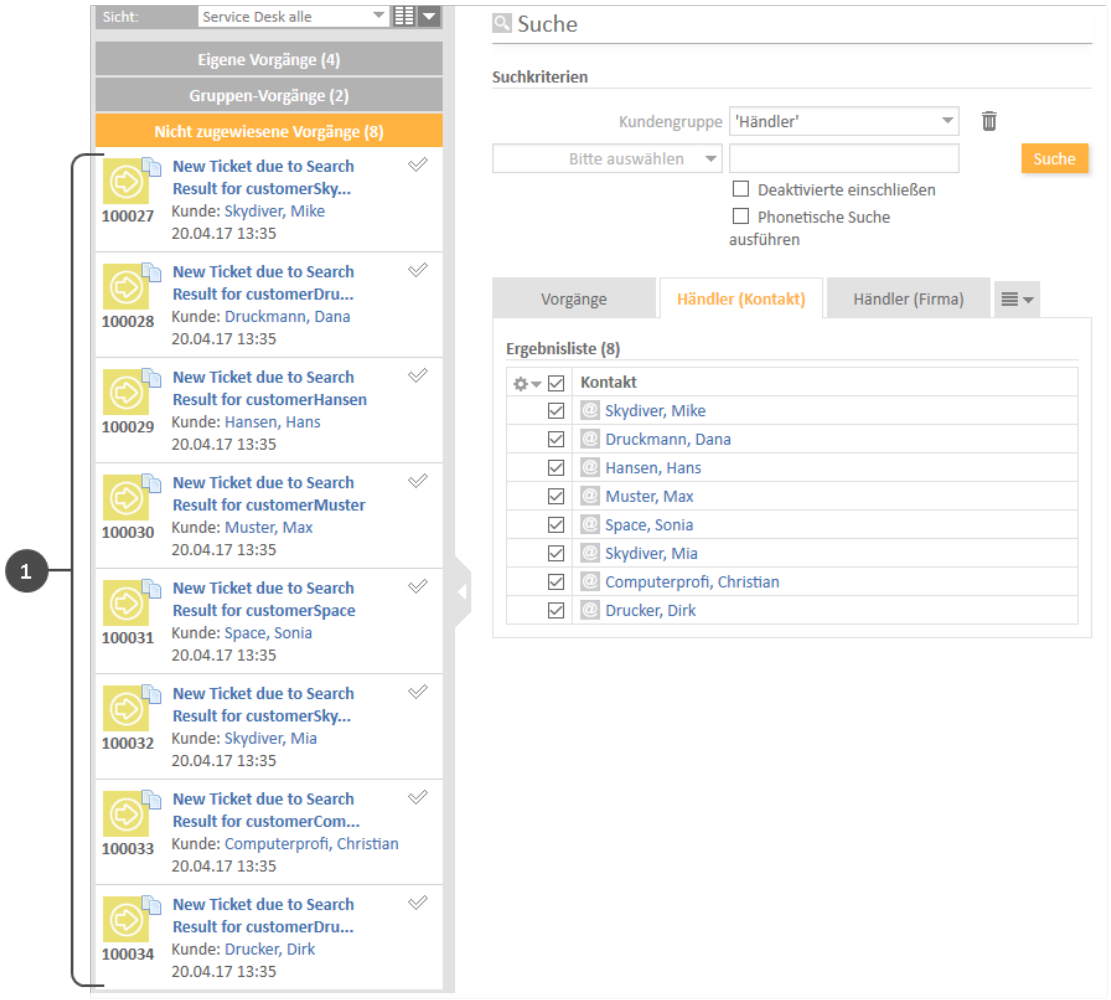


Abbildung 415: *ConSol CM Web Client - Suche-Aktion für Kunden (hier: Kontakte), 2*

Die Ticketliste enthält die neuen ServiceDesk-Tickets (1), die mit dem Aktionsskript erstellt wurden.

Die folgende Abbildung zeigt, dass die Aktivität nicht angezeigt wird, wenn die Ergebnisliste weniger als fünf Einträge enthält (wie im Bedingungsskript festgelegt).

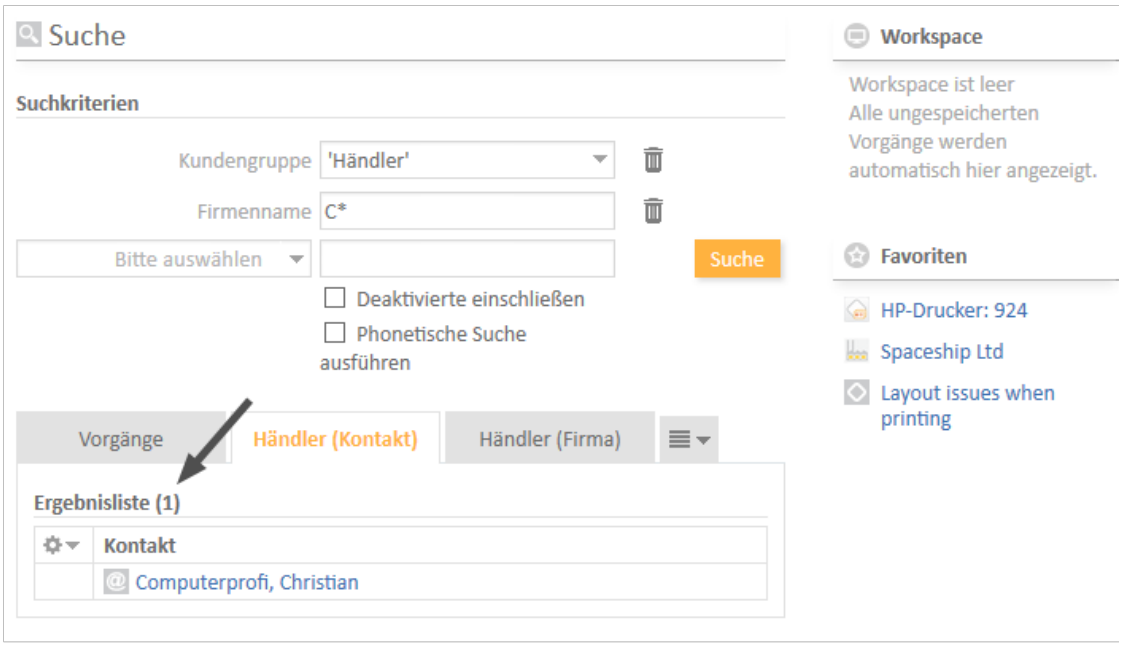


Abbildung 416: *ConSol CM Web Client - Suche-Aktion für Kunden (hier: Kontakte), 3*

F.4.3.3 Tipps und Tricks für Admin-Tool-Skripte mit Suche-Aktionen

Implizit verfügbare Objekte in Admin-Tool-Skripten für Suche-Aktionen

Einige Objekte sind implizit in Admin-Tool-Skripten verfügbar, d. h. Sie müssen weder bestimmte Groovy-Klassen importieren, noch das Objekt der Klasse instanziieren.

Implizit (ohne expliziten Import oder Instanziierung) verfügbare Objekte in Admin-Tool-Skripten:

Alle Suche-Aktionsskripte

- pageSize (Integer)
- pageNumber (Integer)

Skripttypen *Suche-Aktion für Tickets* und *Suche-Bedingung für Tickets*

- criteria (TicketCriteria)
- tickets (List<Ticket> für Suche-Aktion für Tickets)

Skripttypen *Suche-Aktion für Ressourcen* und *Suche-Bedingung für Ressourcen*

- criteria (ResourceCriteria)
- resources (List<Resource> für Suche-Aktion für Ressourcen)

Skripttypen *Suche-Aktion für Kunden* und *Suche-Bedingung für Kunden*

- criteria (UnitCriteria)
- units (List<Unit> für Suche-Aktion für Kunden)

Der Ergebnissatz in Admin-Tool-Skripten für Suche-Aktionen

Die Suche-Aktion kann für unterschiedliche Interpretationen des Ergebnissatzes ausgeführt werden. Das Verhalten der Suche-Aktion hängt von der Implementierung des Admin-Tool-Skripts ab.

Die Suche-Aktion kann durchgeführt werden:

a) nur für den momentan im Web Client angezeigten Teil der Ergebnisliste, z. B. für 20 von 100 Ergebnissen. Dies hängt also von der vom Bearbeiter ausgewählten Anzahl der Treffer pro Seite und, sofern die Zeilenauswahl aktiviert ist, von den vom Bearbeiter ausgewählten Zeilen ab (siehe Abschnitt Seitenanpassung, enableRowSelection).

Im Admin-Tool-Skript wird dies durch die Liste der implizit verfügbaren Objekte dargestellt, d. h. Ressourcen, Tickets oder Units

ODER

b) für die gesamte Ergebnisliste, z. B. alle 100 Ergebnisse Dies kann durch die Verwendung des Objekts *criteria* erreicht werden.

Der Rückgabecode in Admin-Tool-Skripten für Suche-Aktionen

Verwenden Sie die allgemeine Feedback-Komponente, um die neuen `PostActionTypes` zu verarbeiten: SUCCESS und FAILURE.

- SUCCESS - grün
- FAILURE - rot

Beispiel für positives Feedback:

```
return actionScriptResultFactory.getPostAction(PostActionType.SUCCESS,
 "cmweb.search.assigned").withRefreshContent();
```

Code-Beispiel 65: *Positives Feedback*

Eine detaillierte Erklärung über das Schreiben von Skripten für das ConSol CM Action Framework finden Sie in Skripte für das Action Framework.

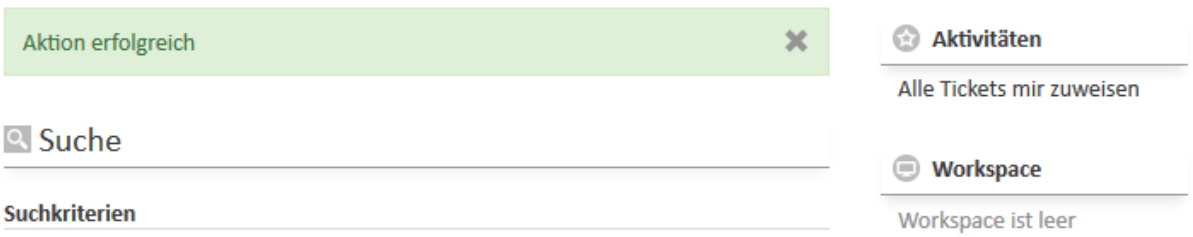


Abbildung 417: *ConSol CM Web Client - Erfolgsmeldung nach Suche-Aktion*

Beispiel für negatives Feedback:

```
return actionScriptResultFactory.getPostAction(PostActionType.FAILURE,
 "cmweb.search.assigned").withRefreshContent();
```

Code-Beispiel 66: *Negatives Feedback*

Eine detaillierte Erklärung über das Schreiben von Skripten für das ConSol CM Action Framework finden Sie in Skripte für das Action Framework.

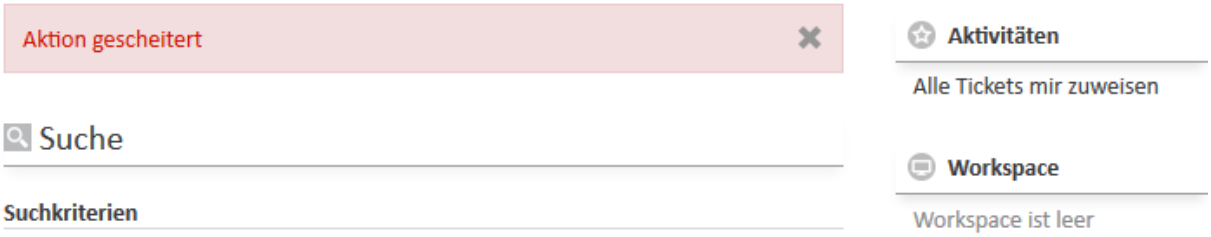


Abbildung 418: *ConSol CM Web Client - Fehlermeldung nach Suche-Aktion*

F.4.4 CSV-Export von Suchergebnissen

F.4.4.1 Einleitung

Ab ConSol CM-Version 6.10.1 können Listen mit Suchergebnissen im CSV-Format exportiert werden. Diese Funktion ist nicht standardmäßig aktiviert und muss in der Seitenanpassung aktiviert werden.

Wenn der CSV-Export aktiviert ist, kann der Bearbeiter die Option *Tabelle exportieren* verwenden, die auf den Seiten mit Suchergebnissen für folgende Objekte angeboten wird:

- Tickets
- Kontakte
- Firmen
- Ressourcen

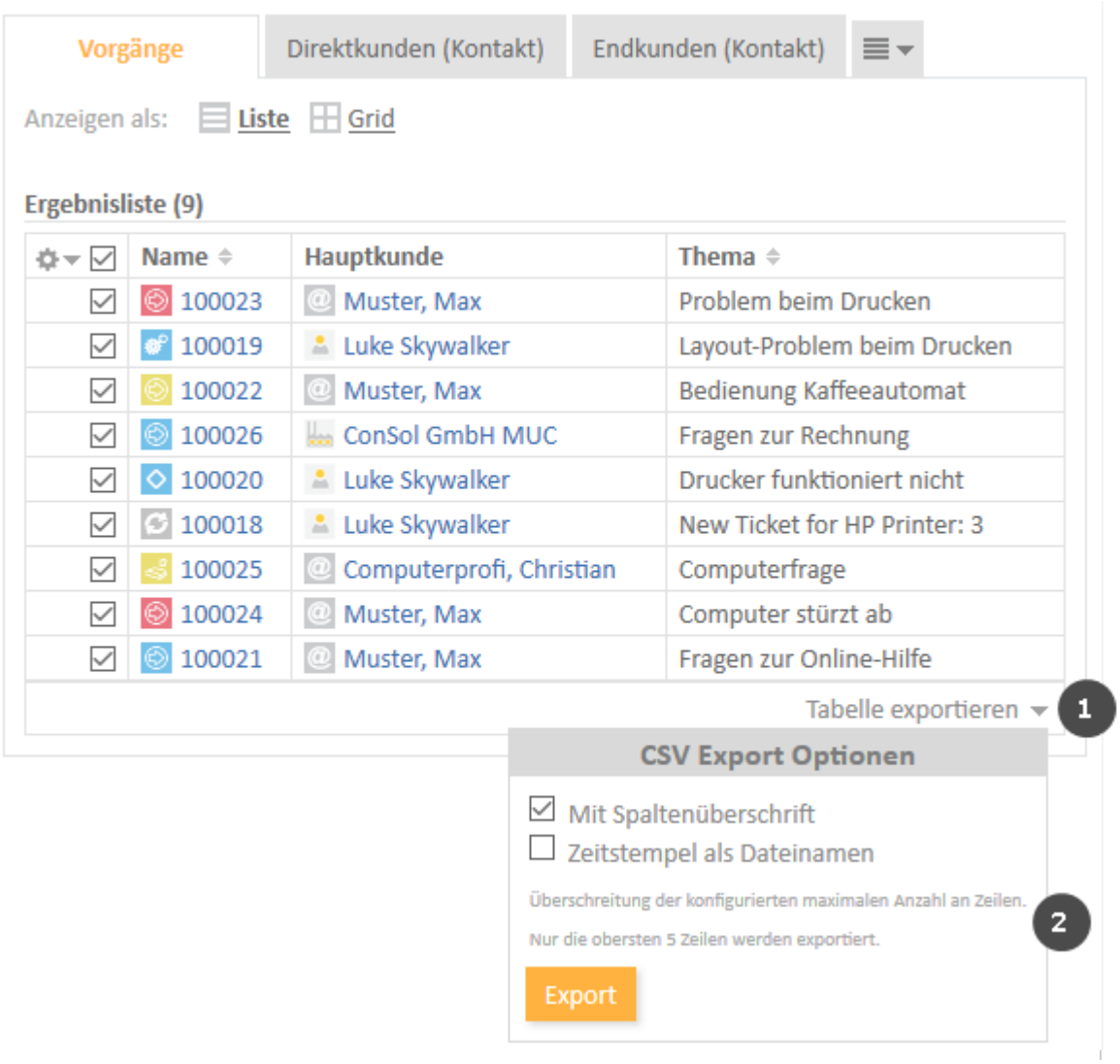


Abbildung 419: *ConSol CM Web Client - CSV-Exportoption für eine Liste mit Ergebnissen der Ticket-suche*

- CSV-Export ist über das Attribut der Seitenanpassung *enabled* aktiviert (1)
- Untermenü des CSV-Exports (2). Im Beispiel würde der Export mehr Zeilen enthalten, als im Attribut der Seitenanpassung *exportRowsLimit* konfiguriert.

Verfügbare CSV-Exportoptionen im Web Client:

- **Mit Spaltenüberschrift**
  Damit werden in der ersten Zeile die Spaltennamen der Ergebnistabelle als Beschreibung des Inhalts eingefügt.
- **Zeitstempel als Dateinamen**
  Wenn diese Option ausgewählt ist, enthält der Standarddateiname das aktuelle Datum und die Zeit, anstatt dem Standardnamen „export.csv“.

Durch Klicken auf *Export* erhalten die Bearbeiter automatisch die Möglichkeit, die neu erstellte CSV-Datei mit dem Standardprogramm zu öffnen, das auf dem Client-Rechner für CSV-Dateien konfiguriert ist. In dieser Datei gibt es keine Seitenumbrüche, es sind alle Ergebniszeilen enthalten (möglicherweise begrenzt durch die unten beschriebene Konfigurationsoption *exportRowsLimit*). In der Ergebnisanzeige vorhandene Icons werden nicht in die CSV-Datei geschrieben. String-Werte von Ticketfeldern werden zwischen doppelte Anführungszeichen gesetzt (ASCII-Wert 34, Dezimal 0x22 hex).

F.4.4.2 Aktivieren der CSV-Exportfunktion

Die CSV-Exportfunktion muss für jede Suchergebnisseite gesondert in der Seitenanpassung aktiviert werden.

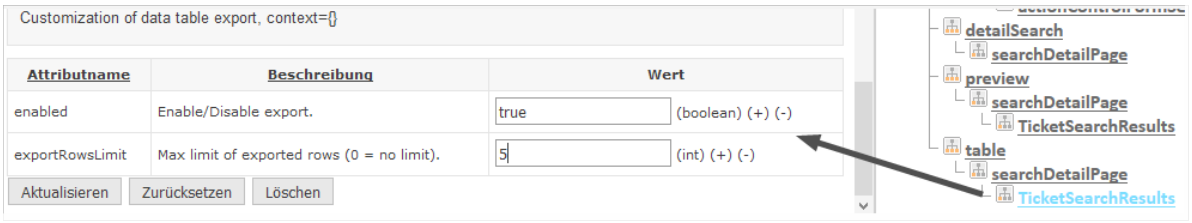


Abbildung 420: *ConSol CM Web Client - Seitenanpassung zum Aktivieren der CSV-Exportfunktion für die Ergebnislisten der Ticketsuche*

Verwenden Sie folgende Unterbereiche:

- **für Ergebnisse der Suche nach Tickets** (siehe obige Abbildung)
  table/searchDetailPage/TicketSearchResults
- **für Ergebnisse der Suche nach Kontakten**
  table/searchDetailPage/<Name der Kundengruppe>
  Dies muss für jede Kundengruppe definiert werden!
- **für Ergebnisse der Suche nach Firmen**
  table/searchDetailPage/<Name der Kundengruppe>
  Dies muss für jede Kundengruppe definiert werden!
- **für Ergebnisse der Suche nach Ressourcen**
  table/searchDetailPage/ResourceSearchResults

Beachten Sie, dass Sie zuerst eine Suche ausführen müssen, damit eine Ergebnistabelle angezeigt wird. Der Bereich *table* ist in der Baumstruktur der Seitenanpassung nur dann verfügbar, wenn mindestens einmal eine Ergebnistabelle erzeugt wurde.

Es sind folgende Attribute verfügbar:

- **enabled**
  Die CSV-Exportfunktion für die Suchergebnisseite aktivieren/deaktivieren, Boolean. Der Standardwert ist „false“.
- **exportRowsLimit**
  Die maximale Anzahl der Zeilen, die mit dem CSV-Export exportiert werden. 0 bedeutet, dass es keine Begrenzung gibt. Der Standardwert ist „0“.

F.5 Das Task Execution Framework (TEF)

In diesem Kapitel werden folgende Themen behandelt:

F.5.1 Einleitung 640

F.5.2 Admin-Tool-Skripte des Typs Task 643

F.5.3 Programmierung mit Tasks 646

F.5.4 Für das TEF relevante System-Properties 652

F.5.1 Einleitung

Mit dem ***Task Execution Framework*** (TEF) kann ConSol CM verschiedene Tasks ausführen, die nicht direkt mit einem anderen Skript (wie einem Workflow-Skript, einer Kundenaktion, einer Ressourcenaktion, einer Suche-Aktion oder einer anderen Art von Admin-Tool-Skript) verbunden sind bzw. nicht in einem anderen Skript eingebettet sind, und die asynchron ausgeführt werden können. Es kann zum Beispiel für lang dauernde System-Tasks verwendet werden, die ein Timeout hervorrufen würden, wenn sie innerhalb eines regulären ConSol CM-Skripts gestartet würden. Die TEF-Tasks können asynchron ausgeführt werden. In Version 6.9.4.0 erhielt ConSol CM eine API-Erweiterung zur Bereitstellung von TEF-Funktionen. TEF-Skripte können folgendermaßen gestartet werden (d. h. die TEF-API ist dort verfügbar):

- manuell im Admin Tool
- aus Workflow-Aktivitätsskripten
- aus E-Mail-Skripten
- aus Aktionsskripten (Suche-Aktionen, Kundenaktionen und Ressourcenaktionen)

Ein Task wird als Admin-Tool-Skript des Typs *Task* gespeichert (siehe Admin-Tool-Skripte). Dieser Skripttyp wird im nachfolgenden Abschnitt beschrieben.

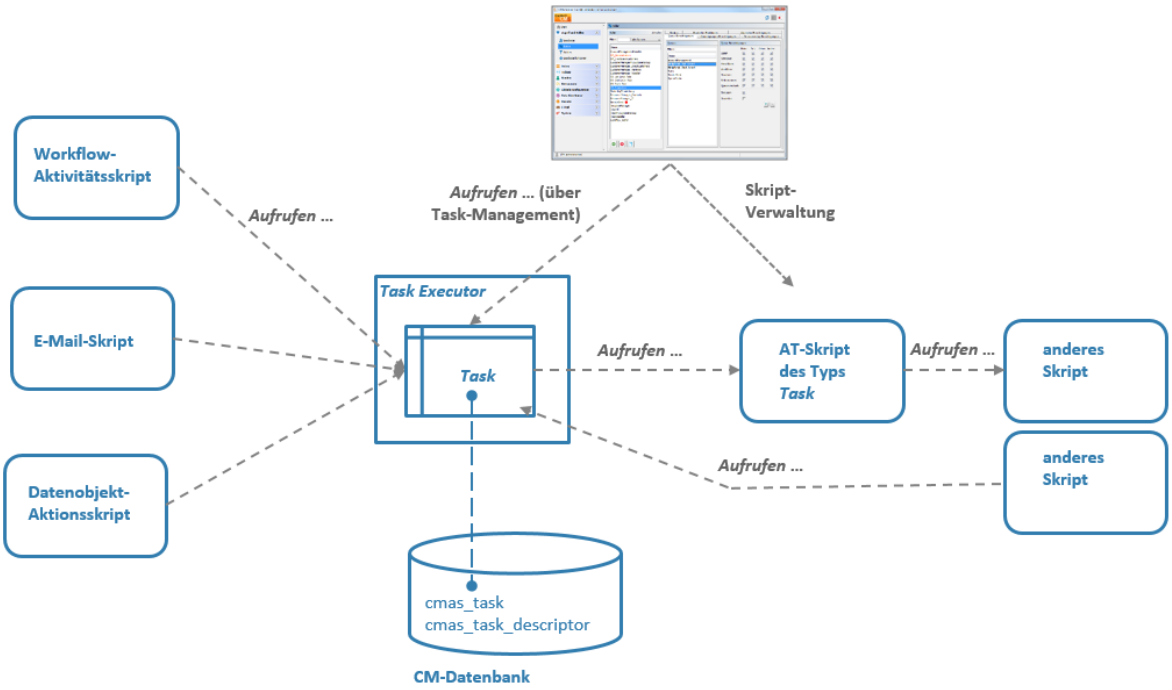


Abbildung 421: *ConSol CM Task Execution Framework*

Der **Task Executor** ist ein ConSol CM-Modul (ein Singleton mit Watchdog-Funktionen), das die Task-Ausführung steuert. Der Task Executor durchsucht die Datenbank nach neuen geplanten Tasks und verwendet einen Thread-Pool für die Ausführung der Tasks, die zu einem bestimmten Zeitpunkt ausgeführt werden müssen.

Die **Task-Definition** ist in einem Admin-Tool-Skript gespeichert. Für die Definition eines Tasks wird also normalerweise ein Admin-Tool-Skript verwendet.

Ein (geplanter) Task, d. h. eine einzige Ausführung des Tasks, kann folgendermaßen gestartet werden:

- Mit dem Admin Tool, Navigationselement *Task-Ausführung*, siehe Abschnitt Ausführung eines Tasks mit dem Admin Tool. Hier kann der Task sofort ausgeführt werden. Er kann nicht für einen anderen Zeitpunkt geplant werden.
- Durch eine Implementierung, in der das Task-Skript in einem anderen Skript eines der folgenden Typen aufgerufen wird. Auf diese Weise kann ein Skript sofort ausgeführt werden oder für einen späteren Zeitpunkt geplant werden.
    - Workflow-Aktivitätsskripte
    - E-Mail-Skripte
    - Aktionsskripte (als Teil des Action Frameworks, d. h. Unit-Aktionsskripte, Ressourcenaktionsskripte oder Suche-Aktionsskripte)

In den Skripten ist für jede Ausführung eines Tasks der sogenannte **Task Descriptor**, d. h. ein Objekt der Klasse `TaskDescriptor`, verfügbar. Dieser Task Descriptor liefert Informationen wie den Fortschritt des Tasks oder die Startzeit der Ausführung. Mit dem Task Descriptor wird ein neu definierter

Task entweder sofort ausgeführt oder für einen späteren Ausführungszeitpunkt geplant. Dies kann in Skripten angewendet werden. Details zur Programmierung finden Sie im Abschnitt Programmierung mit Tasks.

F.5.2 Admin-Tool-Skripte des Typs *Task*

F.5.2.1 Einführung in Admin-Tool-Skripte des Typs *Task*

Jedes Admin-Tool-Skript des Typs *Task* muss folgende Methoden implementieren. Die Methodensignaturen werden automatisch bei der Erstellung eines derartigen Skripts eingefügt.

```
def onInitialize(taskDescriptor) {}
def onExecute(taskDescriptor) {}
def onError(taskDescriptor) {}
def onCancel(taskDescriptor) {}
```

F.5.2.2 Beispiel für ein Admin-Tool-Skript des Typs *Task*

```
//Test
def onInitialize(taskDescriptor) {
   log.info("MyFirstTaskScript has been initialized!")
}

def onExecute(taskDescriptor) {
   log.info("MyFirstTaskScript is executed")
   try {
      Thread.Sleep(300000)
   } catch (Exception ex) {
      log.info("ztztzt ...")
   }
}

def onError(taskDescriptor) {
   log.info("MyFirstTaskScript has thrown an error!")
}

def onCancel(taskDescriptor) {
   log.info("MyFirstTaskScript has been cancelled!")
}
```

Code-Beispiel 67: *Beispiel für ein Admin-Tool-Skript des Typs Task*

F.5.2.3 Ausführung eines Tasks mit dem Admin Tool

Sie können die Tasks, die zuvor als Admin-Tool-Skripte definiert wurden, im Admin Tool in der Navigationsgruppe *Dienste*, Navigationselement *Task-Ausführung* starten.

Damit die Tasks ausgeführt werden können, muss die System-Property cmas-app-admin-tool, start.groovy.task.enabled im Modul auf „true“ gesetzt sein. Diese Property ist in einer Standardinstallation nicht vorhanden und muss manuell hinzugefügt werden.

Bedenken Sie, dass Sie absolut sicher sein müssen, was die ausgeführten Tasks machen, wenn Sie die System-Property cmas-app-admin-tool, start.groovy.task.enabled auf „true“ setzen, sodass Sie als Administrator Tasks ausführen können! Seien Sie sich des Risikos bewusst, das Tasks darstellen können, die zum Beispiel Kundendaten oder Tickets löschen und nur durch ein Workflow- oder Admin-Tool-Skript ausgeführt werden sollten.

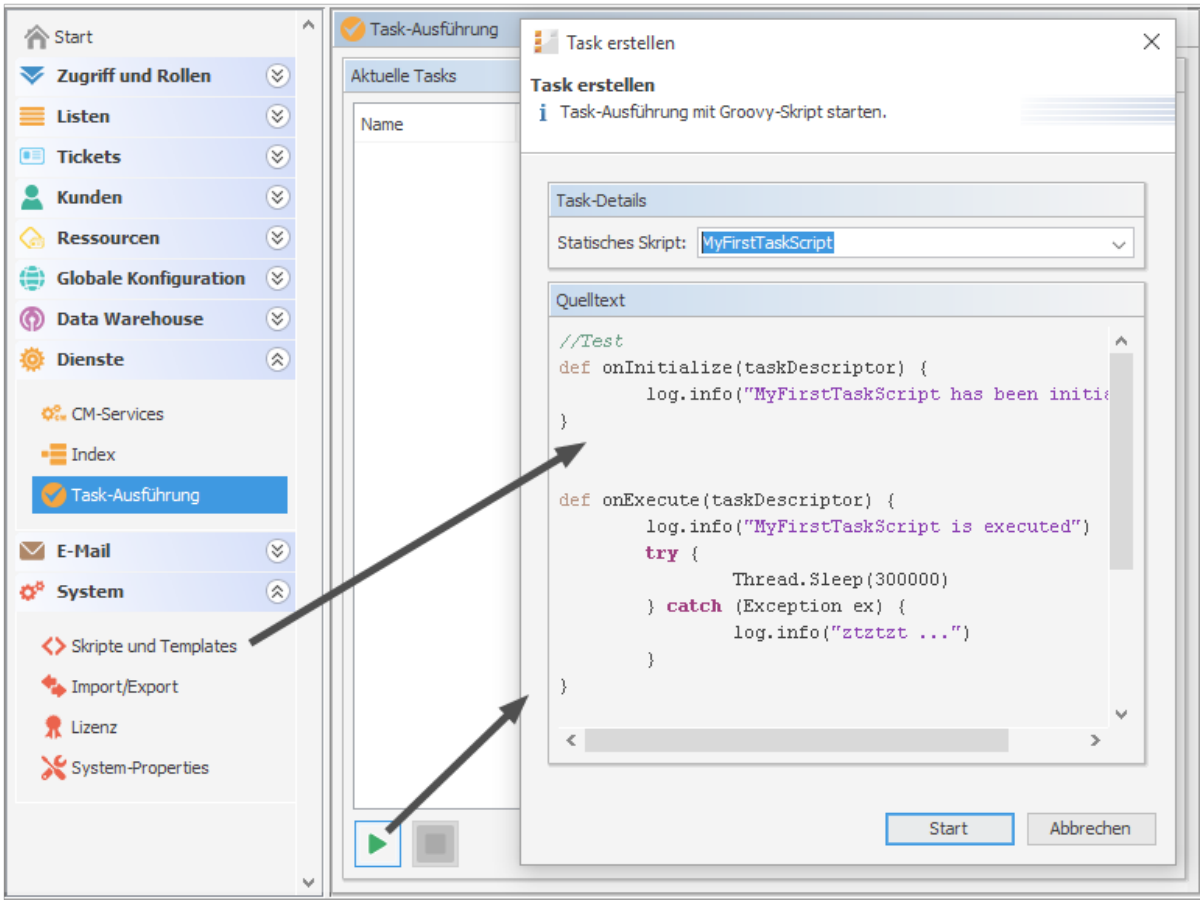


Abbildung 422: *ConSol CM Admin Tool - Dienste, Task-Ausführung: Erstellen eines Tasks*

Um ein Skript zu starten, klicken Sie auf den Button *Start* und wählen Sie den Namen des Admin-Tool-Skripts im Drop-down-Menü *Statisches Skript*. Hier sind alle Admin-Tool-Skripte des Typs *Task* aufgeführt. Klicken Sie auf *Start*, um das Skript sofort auszuführen. Es ist nicht möglich, einen Task über die GUI des Admin Tools zu planen. Wenn der Start verzögert werden soll, muss dies innerhalb des Skripts implementiert werden, siehe Definieren des (ersten) Ausführungsdatums.

Wenn ein Task ausgeführt wird, wird ein Fortschrittsbalken angezeigt. Sie können den Task über den Button *Stop* anhalten (abbrechen).

Das Aktualisierungsintervall der Task-Liste wird mit der CM-System-Property cmas-app-admin-tool, task.panel.refresh.interval.seconds definiert. Die Property ist standardmäßig nicht vorhanden und muss manuell hinzugefügt werden.

Die folgende Abbildung zeigt einen gerade ausgeführten Task. Sie gelangen auf diese Seite, indem Sie das Navigationselement *Task-Ausführung* der Navigationsgruppe *Dienste* öffnen.

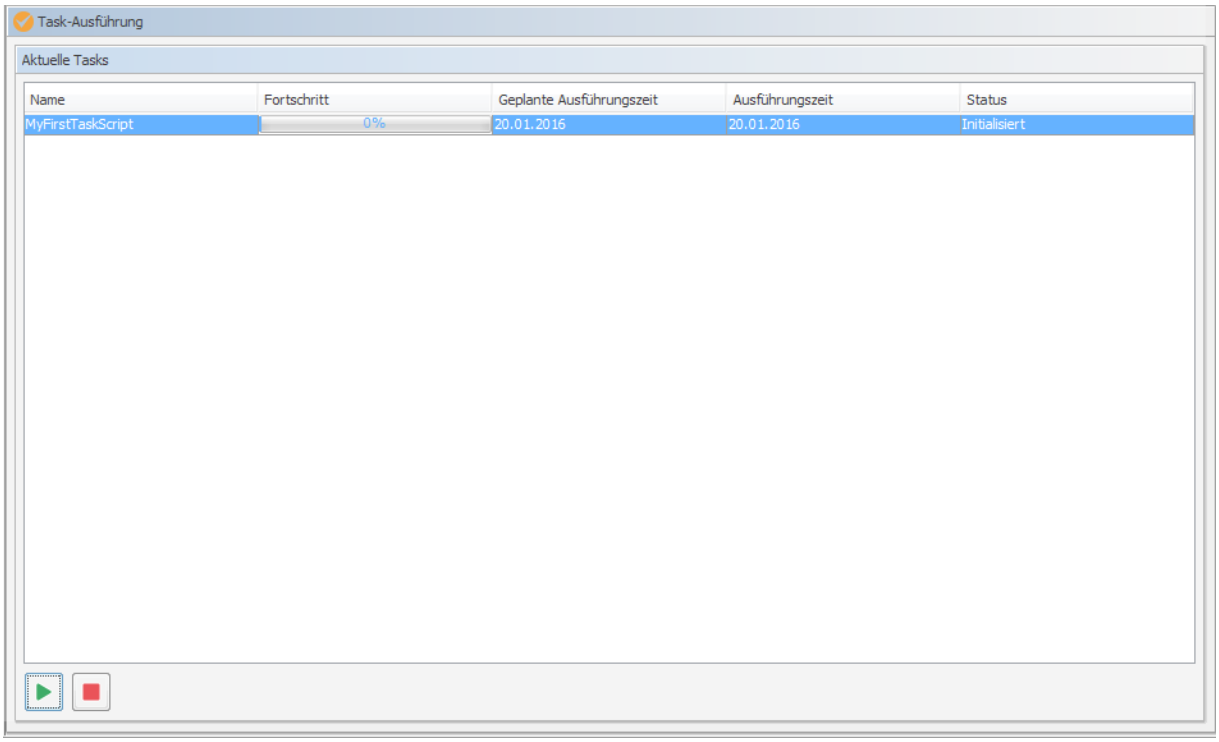


Abbildung 423: *ConSol CM Admin Tool - Services, Task-Ausführung: Ausführen von Task*

Wenn ein TEF-Skript manuell mit dem Admin Tool gestartet wird, wird es auf dem Rechner ausgeführt, auf dem das Admin Tool läuft. Wenn Sie in einer geclusterten Umgebung arbeiten, in der Skripte automatisch ausgeführt werden, können Sie die ID des Cluster-Nodes, auf dem TEF-Skripte ausgeführt werden sollen, mit der CM-System-Property cmas-core-server, task.execution.node.id definieren.

F.5.3 Programmierung mit Tasks

F.5.3.1 Einleitung

In der aktuellen ConSol CM-Version steht ein Task-Typ zur Verfügung, der **Groovy-Task mit einem statischen Skript**. Dies bezieht sich auf das Admin-Tool-Skript, mit dem der Task definiert wird, siehe vorherigen Abschnitt.

Der **Task Execution Service** (Groovy-Klasse `TaskExecutionService`, ein Singleton) läuft im Hintergrund und durchsucht die ConSol CM-Datenbank nach Tasks (Datenbank-Tabelle `cmas_task_descriptor`) mit dem Status INITIALIZED. Wie alle ConSol CM-Dienste ist er implizit als Objekt mit dem Namen `taskExecutionService` (siehe folgende Beispiele) verfügbar. Wenn die Startzeit des Tasks erreicht wurde, wird der Task gestartet.

Alle Parameter für den neuen Task müssen mit dem **Task Descriptor** (Groovy-Klasse `TaskDescriptor`) gesetzt werden, z. B. das Startdatum des Tasks. Der Task Descriptor liefert zudem Informationen über den laufenden Task, wie seinen Fortschritt.

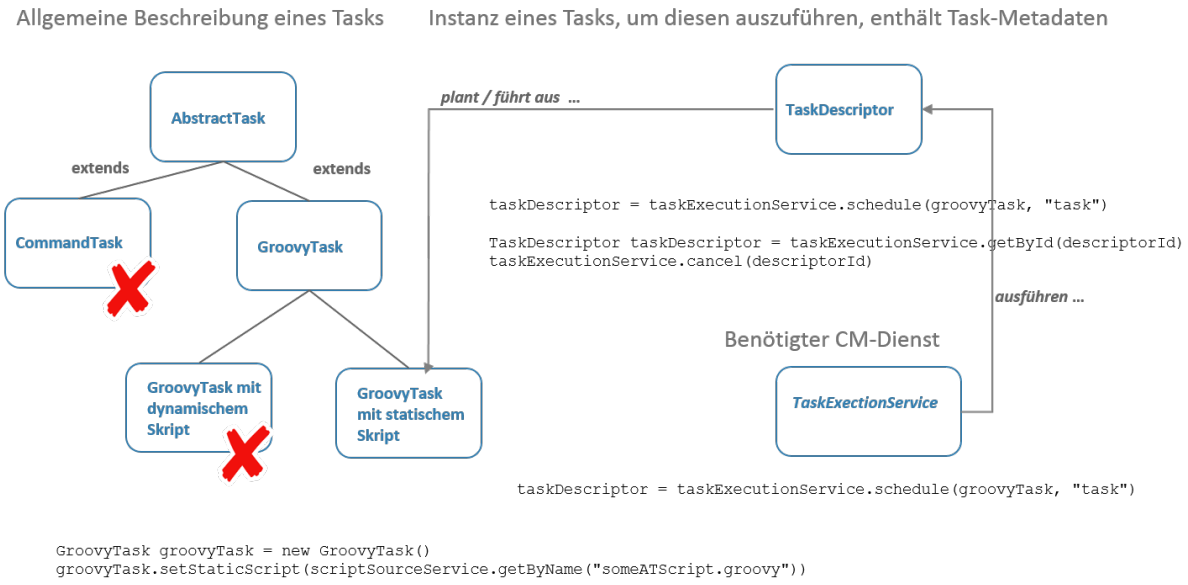


Abbildung 424: *Einige TEF-Groovy-Klassen*

F.5.3.2 Coding-Beispiele

Erstellen eines Tasks

```
GroovyTask groovyTask = new GroovyTask();
groovyTask.setStaticScript(scriptSourceService.getByName("someATScript.groovy"));
taskDescriptor = taskExecutionService.schedule(groovyTask, "task");
```

Code-Beispiel 68: *Erstellen eines Task Descriptors*

Abbrechen (Beenden) eines Tasks

Teil 1: Erstellen Sie den Task Descriptor und speichern Sie die ID.

```
GroovyTask groovyTask = new GroovyTask();
groovyTask.setStaticScript(scriptSourceService.getByName("someATScript.groovy"));
taskDescriptor = taskExecutionService.schedule(groovyTask, "task");
def myTaskDescriptorId = groovyTask.getId()
//save this Id wherever it will be needed, e.g., in a different script which might
 be used to kill the task
```

Code-Beispiel 69: *Abbrechen eines Tasks*

Teil 2: Möglicherweise während der Ausführung des Tasks verwendet.

```
taskExecutionService.cancel(myTaskDescriptorId)
```

Wiederholen eines Tasks

Wenn Sie nach der Ausführung eines Tasks ein anderes Ausführungsdatum setzen, wird der Task neu geplant. Dies erfolgt, wie hier gezeigt, innerhalb des Admin-Tool-Task-Skripts.

```
def onInitialize(taskDescriptor) {}

def onExecute(taskDescriptor) {
   //some code to execute
   ........................
   //here, we set the new future execution date for the task, we also need to
    return a special steering object
   taskDescriptor.setExecutionDate(new Date(new Date().getTime() + 15000));
   return new ExecutionSpecification().setRetryRequested(true);
}

def onError(taskDescriptor) {}
def onCancel(taskDescriptor) {}
```

Code-Beispiel 70: *Wiederholen eines Tasks*

Definieren des (ersten) Ausführungsdatums

Für ein Skript, das nicht sofort gestartet werden soll, können Sie die Startzeit in der Methode `onInitialize()` definieren.

```
def onInitialize(taskDescriptor) {
   taskDescriptor.setExecutionDate(yourDate)
}
```

Code-Beispiel 71: *Planen eines Tasks*

Wiederholen eines Tasks nach einem Fehler

```
def onInitialize(taskDescriptor) {}

def onExecute(taskDescriptor) {}

def onError(taskDescriptor) {
return new ExecutionSpecification().setRetryRequested(true);
// this will reschedule the task for immediate re-execution, in case a future date
 is needed, this can be set as explained in the example above

def onCancel(taskDescriptor) {}
```

Code-Beispiel 72: *Wiederholen eines Tasks nach einem Fehler*

Arbeiten mit ContextReference

Mit `ContextReference` können Sie feststellen, in welchem Kontext ein Skript aufgerufen wurde. So kann es zum Beispiel erforderlich sein, dass ein TEF-Skript sich unterschiedlich verhält, je nachdem, ob es aus dem Workflow-Skript einer bestimmten Aktivität aufgerufen wurde oder im Prozess einer anderen Aktivität. In solchen Fällen hilft Ihnen die `ContextReference` dabei, Rückschlüsse auf das Workflow-Skript zu ziehen, aus dem das TEF-Skript aufgerufen wurde. Sie setzen einfach in jedem Workflow-Skript eine andere `ContextReference` (ein einfacher String als Bezeichner). Dann können Sie die `ContextReference` im Task abrufen und der Task weiß, von wo er aufgerufen wurde.

Verwenden Sie folgende Methoden:

- `void GroovyTask.setContextReference(String ContRef)`
- `String TaskDescriptor.getContextReference()`

Siehe auch Beispiel 2 unten.

F.5.3.3 Beispiele für die Verwendung von Task-Skripten

Beispiel 1: Verwendung eines einfachen Task-Skripts

In diesem Beispiel wird ein Task-Skript mit einer Workflow-Aktivität ausgeführt. Es wird keine Verzögerung gesetzt, d. h. der Task wird sofort ausgeführt, wenn der Bearbeiter die Workflow-Aktivität im Web Client ausführt. Das Skript kann dann im Hintergrund laufen und der Bearbeiter sieht die Ergebnisse (wie die neuen Ticketeinträge oder die neuen Kundendaten) erst, wenn das Skript fertig ist. Ein Aktion des Bearbeiters ist zwischen dem Start und dem Ende der Skriptausführung nicht erforderlich.

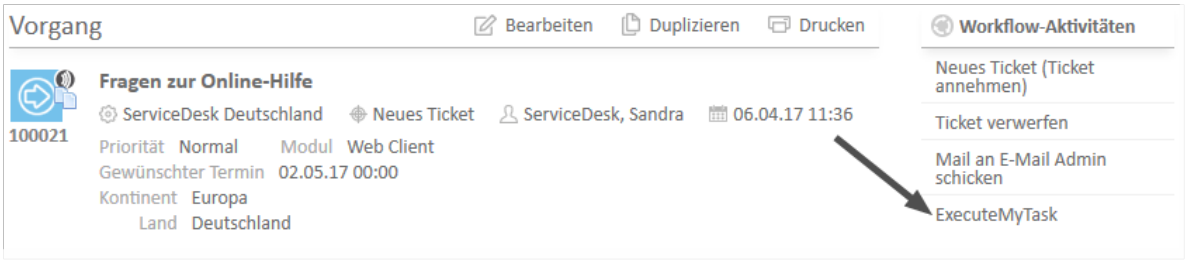


Abbildung 425: *ConSol CM Web Client - Workflow-Aktivität für Task-Ausführung*



Abbildung 426: *ConSol CM Process Designer - Workflow-Aktivität für Task-Ausführung*

```
def myNewTask = new GroovyTask()
myNewTask.setStaticScript(scriptSourceService.getByName("MyFirstTaskScript"))
def myTaskDescriptor = taskExecutionService.schedule(myNewTask, "myTaskGroup")
myTaskDescriptor.setExecutionDate(new Date())
```

Code-Beispiel 73: *Workflow-Aktivitätsskript für Task-Ausführung*

```
2015-02-20 11:54:24,742 INFO [rver.service.task.TaskExecutor] [task-executor-task-
 executor:10.0.6.200:0-] Task Executor task-executor:10.0.6.200:0 is executing
 task: TaskDesc-02-20 11:54:19.0, transactionTimeout (sec)=0, type=class
 com.consol.cmas.common.model.task.GroovyTask}
2015-02-20 11:54:24,747 INFO [ database_MyFirstTaskScript] [task-executor-task-
 executor:10.0.6.200:0-] MyFirstTaskScript is executed
2015-02-20 11:54:24,747 INFO [ database_MyFirstTaskScript] [task-executor-task-
 executor:10.0.6.200:0-] ztztzt ...
2015-02-20 11:54:24,748 INFO [rver.service.task.TaskExecutor] [task-executor-task-
 executor:10.0.6.200:0-] Task execution successful removing task : TaskDescriptor
 {group='myT, transactionTimeout (sec)=0, type=class
 com.consol.cmas.common.model.task.GroovyTask}
```

Beispiel 2: Arbeiten mit der ContextReference, einfaches Beispiel mit Log-Ausgabe

Das Task-Skript (Admin-Tool-Skript des Typs *Task*) zeigt die ContextReference an, aus der es aufgerufen wurde.

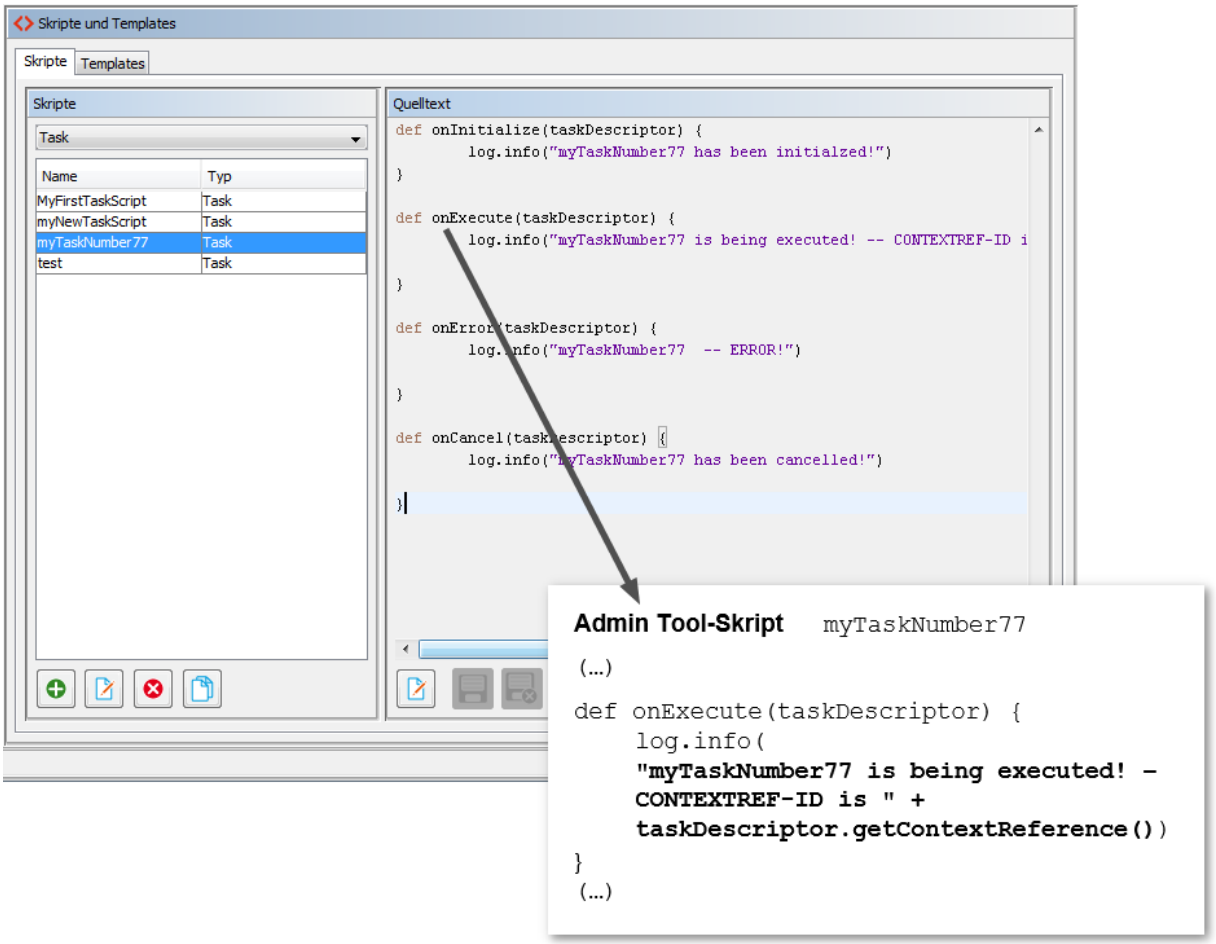


Abbildung 427: *ConSol CM Admin Tool - System, Skripte und Templates: Admin Tool-Skript für einen TEF-Task*

In anderen Workflow-Aktivitäten werden andere ContextReferences als eindeutige IDs gesetzt. Wenn das TEF-Skript aufgerufen wird, wird (in der Log-Ausgabe) immer die ContextReference der aufrufenden Workflow-Aktivität angezeigt.

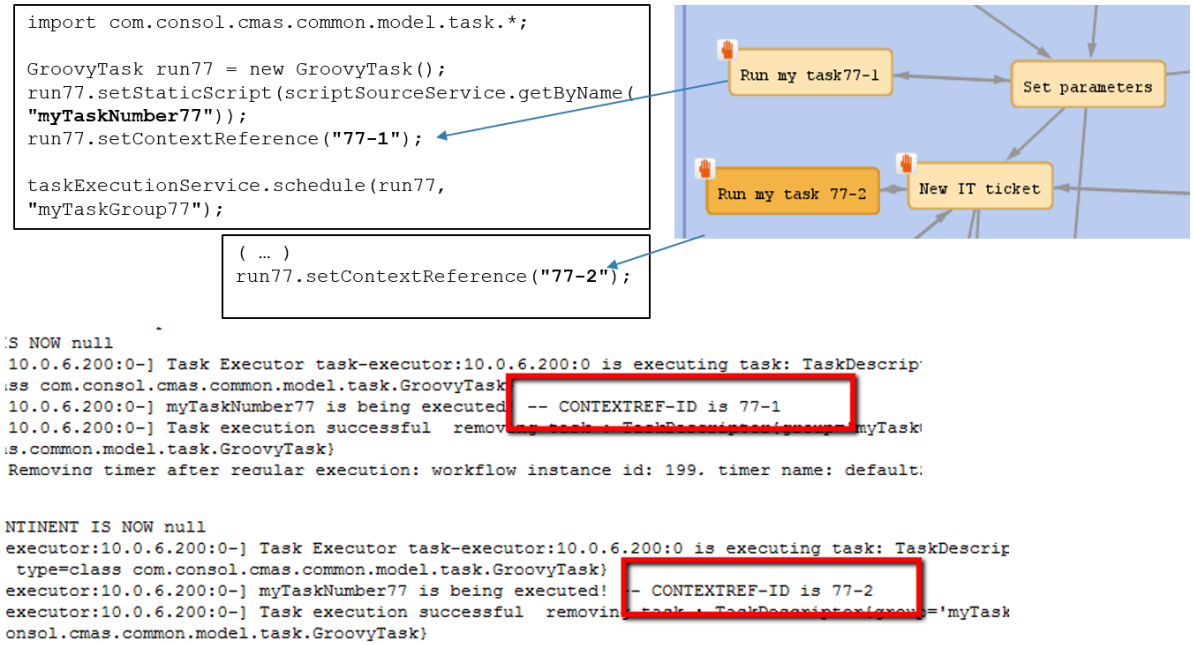


Abbildung 428: *Workflow-Aktivitäten und Log-Ausgabe, wenn eine der Aktivitäten das TEF-Skript aufgerufen hat*

F.5.4 Für das TEF relevante System-Properties

Modul	Parameter	Standardwert	Beschreibung
cmas-app-admin-tool	start.groovy.task.enabled	false	Aktiviert den Button zum Starten des Tasks im Admin Tool
cmas-core-server	transaction.timeout.minutes	60	Setzt den Timeout für den Task Execution Service des TEF, d. h. ein Durchlauf eines Tasks muss vor dem Ablauf dieser Zeitspanne abgeschlossen sein.
cmas-core-server	number.of.tasks	1	Größe des Thread-Pools, d. h. Anzahl der parallel ausgeführten Tasks
cmas-core-server	task.execution.interval.seconds	5	Wartezeit zwischen der Ausführung von zwei Tasks

F.6 Das ConSol CM Action Framework

Das ConSol CM Action Framework umfasst mehrere Arten von Aktionen, die KEINE Workflow-Aktionen sind. Workflow-Aktionen werden entweder manuell oder automatisch für Tickets ausgeführt, wohingegen das Action Framework Aktionen für andere Arten von Objekten bereitstellt. Zusätzliche Komponenten wie Aktionsformulare erweitern den Umfang des Frameworks.

Die folgende Übersicht bietet einen ersten Überblick über alle Komponenten des Action Frameworks. Detaillierte Informationen finden Sie in den entsprechenden Abschnitten.

F.6.1 Aktionen

- **Suche-Aktionen**
  Sind immer manuelle Aktionen, verfügbar für:
  - Tickets
  - Kunden (= Units)
  - Ressourcen

  Erklärt in Action Framework - Suche-Aktionen
- **Kundenaktionen**
  Können manuelle oder automatische Aktionen sein, verfügbar für:
  - Firmen
  - Kontakte

  Erklärt in Action Framework - Kundenaktionen
- **Ressourcenaktionen**
  Können manuelle oder automatische Aktionen sein, verfügbar für:
  - Ressourcen

  Erklärt in CM/Resource Pool - Ressourcenaktionen

F.6.2 Zusätzliche Komponenten

- **Aktionsformulare**
  Formulare, ähnlich wie Aktivitäts-Formulare, die es ermöglichen, Daten zu sammeln, während eine Aktion des Action Frameworks ausgeführt wird. Sie sind im Abschnitt Aktionsformulare erklärt.
- **Skripte für das Action Framework**
  Implementieren verschiedene Funktionalitäten für das Action Framework. Die Skripte sind im Abschnitt Skripte für das Action Framework erklärt.

F.6.3 Skripte für das Action Framework

F.6.3.1 Einleitung

Das ConSol CM Action Framework bietet die Möglichkeit, Aktionen zu starten, die sich nicht auf Workflow-Aktivitäten beziehen, d. h. Aktionen können in einem anderen Kontext als im Workflow-Kontext gestartet oder angestoßen werden.

Das Action Framework besteht (in ConSol CM-Version 6.10) aus drei Komponenten, die in den entsprechenden Handbuchabschnitten detailliert beschrieben sind.

- **Kundenaktionen (Unit-Aktionen)**
  Aktionen für Kontakt- oder Firmenobjekte, siehe Abschnitt Action Framework - Kundenaktionen.
- **Ressourcenaktionen**
  Aktionen für Ressourcenobjekte, siehe Abschnitt CM/Resource Pool - Ressourcenaktionen.
- **Suche-Aktionen**
  Aktionen für Suchergebnisse, siehe Abschnitt Action Framework - Suche-Aktionen.

Im vorliegenden Abschnitt erfahren Sie mehr über die Programmierung mit dem Action Framework.

F.6.3.2 Admin-Tool-Skripte für das Action Framework

Skripttypen

Jede Aktion basiert auf einem Admin-Tool-Skript, wie in den oben erwähnten Abschnitten beschrieben. Für jedes Admin-Tool-Skript muss der richtige Skripttyp gesetzt werden.



Beachten Sie, dass für jedes **Aktionsskript** ein Bedingungsskript definiert werden **kann**. Es muss im Admin Tool in der Definition der Aktion mit dem Aktionsskript verknüpft werden. Das **Bedingungsskript** wird vor dem Aktionsskript ausgeführt.

Daher wird das Aktionsskript nur ausgeführt, wenn entweder

- kein Bedingungsskript definiert wurde

oder

- das Bedingungsskript „true“ zurückgegeben hat.

Es stehen folgende Skripttypen für das Action Framework zur Verfügung:

- Für **Kundenaktionen (= Unit-Aktionen)**:
  - Kundenbedingung
  - Kundenaktion
- Für **Ressourcenaktionen**:
  - Ressourcen-Bedingung
  - Ressourcen-Aktion

- Für **Suche-Aktionen**:
  - Für Aktionen, die auf Ergebnislisten mit Tickets basieren:
    - Suche-Bedingung für Tickets
    - Suche-Aktion für Tickets
  - Für Aktionen, die auf Ergebnislisten mit Kunden (= Units) basieren, d. h. Ergebnislisten mit Kontakten oder Firmen:
    - Suche-Bedingung für Kunden
    - Suche-Aktion für Kunden
  - Für Aktionen, die auf Ergebnislisten mit Ressourcen basieren:
    - Suche-Bedingung für Ressourcen
    - Suche-Aktion für Ressourcen
- Für **Aktionsformulare** (siehe Abschnitt Aktionsformulare):
  - Bedingung für Aktionsformular
  - Vorbelegung für Aktionsformular

Aktionstypen

Eine Aktion kann von einem der folgenden Typen sein:

- **Automatisch**
  - Erzeugen
  - Aktualisieren
  - Löschen
  - Relation
  - Suche
- **Manuell**

F.6.3.3 Programmieren mit dem Action Framework

Verfügbare und veraltete Objekte

In allen verfügbaren Arten von Skripten (siehe Abschnitt Skripttypen) ist eine Klasse verfügbar, in der das Verhalten des Systems nach der Ausführung des Skripts implementiert wird. Das ist die Klasse `OperationResponseBuilder`. Ein Objekt dieser Klasse mit dem Namen `client` ist implizit in allen Aktionsskripten verfügbar. Das gilt sowohl für automatische als auch für manuelle Skripte. Dieses Objekt ist auch in Workflow-Skripten verfügbar, dieses Thema wird detailliert im *ConSol CM Process Designer Handbuch* beschrieben.

Das Objekt `client` bietet Methoden für Operationen, wie z. B.:

- Öffnen einer bestimmten URL
- Öffnen der Seite zum Erstellen einer Ressource

- Öffnen der Seite zum Erstellen eines Tickets mit bereits zugewiesenem Kunden

Neben diesen Aktionen bietet das Objekt `client` auch Methoden zum Anzeigen von Meldungen im Web Client, z. B.:

- Anzeigen der Information, dass die Kundenaktion erfolgreich durchgeführt wurde
- Anzeigen der Information, dass die Ressourcenaktion fehlgeschlagen ist
- Anzeigen einer Warnmeldung, wenn die eingegebenen Daten bestimmte Kriterien nicht erfüllen

Außerdem bietet das Objekt `client` die Methode `rollback()`.



Bitte beachten:

- Die Klassen und Objekte von `ActionScriptResult`, `ActionScriptResultFactory`, `PostActionScriptResult`, `PostActionType` und `postActivityExecutionHandler` sind in CM-Versionen 6.11.1.0 und höher veraltet! Verwenden Sie die neue Implementierung `client` (= `OperationResponseBuilder`).
- Die Methode `workflowApi.addValidationError()`, die zum Anzeigen von Meldungen im Web Client verwendet werden kann, ist noch gültig.

Überblick über alle Methoden von `client` (Vergleich mit alten Methoden)

Die folgenden Methoden sind für das Objekt `client` verfügbar, einer Implementierung der Klasse `OperationResponseBuilder`. Weitere Details finden Sie in der ConSol CM Java / Groovy API Dokumentation. Eine Zuordnung der veralteten Objekte und Methoden finden Sie unten. Diese könnte es Ihnen erleichtern, den Code in Ihren Skripten auszutauschen.

Neue Klasse/Methode	Veraltete Klasse/Methode	Hinweis
client.failure()	PostActionType.FAILURE	Setzt OperationType = FAILURE.
client.goToCreateResource (resource)	actionScriptResultFactory.getPostAction (PostActionType.CREATE_RESOURCE, resource)	Öffnet die Seite zum Erstellen einer Ressource, die Daten aus dem Objekt `resource` sind vorausgefüllt.
client.goToCreateTicket (ticket)	actionScriptResultFactory.getPostAction (PostActionType.CREATE_TICKET, ticket)	Öffnet die Seite zum Erstellen eines Tickets, die Daten aus dem Objekt `ticket` sind vorausgefüllt.

Neue Klasse/Methode	Veraltete Klasse/Methode	Hinweis
client.goToCreateTicket (ticket).withCustomer(unit);	actionScriptResultFactory.getPostAction (PostActionType.CREATE_TICKET, ticket, unit)	Öffnet die Seite zum Erstellen eines Tickets, die Daten aus den Objekten `ticket` und `unit` sind vorausgefüllt.
client.goToCreateUnit(unit)	actionScriptResultFactory.getPostAction (PostActionType.CREATE_UNIT, unit)	Öffnet die Seite zum Erstellen eines Kunden, die Daten aus dem Objekt `unit` sind vorausgefüllt.
client.goToPage("url")	actionScriptResultFactory.getPostAction (PostActionType.GOTO_PAGE, "url");	Öffnet die angegebene URL.
client.goToResource (resource)	actionScriptResultFactory.getPostAction (PostActionType.GOTO_RESOURCE, resource);	Öffnet die Ressourcenseite des angegebenen Objekts `resource`.
client.goToTicket(ticket)	actionScriptResultFactory.getPostAction (PostActionType.GOTO_TICKET, ticket)	Öffnet die Ticketseite des angegebenen Objekts `ticket`.
client.goToUnit(unit)	actionScriptResultFactory.getPostAction (PostActionType.GOTO_UNIT, unit)	Öffnet die Kundenseite des angegebenen Objekts `unit`.
client.goToResource(resource).openActionForm("actionformname");	-	Öffnet das angegebene Aktionsformular der Ressource.
client.goToUnit(unit).openActionForm("actionformname")	-	Öffnet das angegebene Aktionsformular des Kunden.
client.goToTicket(ticket) .openActivityForm("acfpath")	actionScriptResultFactory.getPostAction (PostActionType.GOTO_TICKET, ticket, acfExecutionContext)	Öffnet das ACF des aktuellen Tickets.
client.rollback()	-	Macht die Aktion rückgängig. Optional kann eine Fehlermeldung angegeben werden.

Neue Klasse/Methode	Veraltete Klasse/Methode	Hinweis
client.showDebugMessage ("messageOrLabel")	actionScriptResultFactory.getPostAction (PostActionType.FAILURE, "mes-sageOrLabel")	Zeigt eine rote Meldung, der Meldungstext kann als String oder Bezeich-nung angegeben werden.
client.showErrorMessage()	actionScriptResultFactory.getPostAction (PostActionType.FAILURE, "mes-sageOrLabel")	Zeigt eine rote Meldung, der Meldungstext kann als String oder Bezeich-nung angegeben werden.
client.showInfoMessage()	actionScriptResultFactory.getPostAction (PostActionType.SUCCESS, "mes-sageOrLabel")	Zeigt eine grüne Mel-dung, der Meldungstext kann als String oder Bezeichnung angegeben werden.
client.showMessage("mes-sageOrLabel")	actionScriptResultFactory.getPostAction (PostActionType.SUCCESS, "mes-sageOrLabel")	Zeigt eine grüne Mel-dung, der Meldungstext kann als String oder Bezeichnung angegeben werden.
client.showWarningMessage ()	actionScriptResultFactory.getPostAction (PostActionType.SUCCESS, "mes-sageOrLabel")	Zeigt eine grüne Mel-dung, der Meldungstext kann als String oder Bezeichnung angegeben werden.
client.success()	PostActionType.SUCCESS	Setzt OperationType = SUCCESS.

Beispiele

Seite *Neues Ticket* öffnen

- client.goToCreateTicket(ticket)
- client.goToCreateTicket(ticket).withCustomer(unit)

Beispiel:

```
import com.consol.cmas.common.model.ticket.Ticket

Ticket ticket = new Ticket();
ticket.setQueue(queueService.getByName("Helpdesk"))
ticket.setSubject("sample subject")
ticket.set("queue_fields.string", "test")
ticket.set("queue_fields", "boolean", "true")
client.goToCreateTicket(ticket)
//to additionally set main contact use unit , has to be retrieved in code
//client.goToCreateTicket(ticket).withCustomer(unit)
```

Code-Beispiel 74: *Öffnen der Seite Neues Ticket*

Ticketseite im Anzeigemodus öffnen

- client.goToTicket(ticket)

Beispiel:

```
client.goToTicket(newtic)
```

Code-Beispiel 75: *Öffnen der Ticketseite im Anzeigemodus*

Ticketseite im Anzeigemodus öffnen und vorher ACF anzeigen

- `client.goToTicket(ticket).openActivityForm(executionContext)` Leitet auf die Ticketseite des angegebenen Tickets weiter und öffnet ein über `controlFormDefinitionService.getExecutionContext(Ticket,ACF-path)`. aufgerufenes Aktivitätsformular.

Beispiel:

```
(...)
def executionContext = controlFormDefinitionService.getExecutionContext(newtic,
 "defaultScope/TaskInProgress/AcceptTask")
if (!executionContext) {
   client.showErrorMessage("action.fail.wrong.activity")
}

// Modify entities from the execution context - not the original ones
// - since the user may still press cancel.

executionContext.ticket.add("SpecialTasks_Fields","Deadline", new Date());

client.goToTicket(newtic).openActivityForm(executionContext)
```

Code-Beispiel 76: *Öffnen der Ticketseite im Anzeigemodus, nach Anzeige eines ACFs*

Seite *Neuen Kunden* (Kontakt oder Firma) öffnen

- `client.goToCreateUnit(unit)`
  Leitet auf die Seite *Neuer Kunde* weiter, auf der Felder mit Kundendaten gefüllt sind.

Beispiel:

```
// used for companies in MyCustomerGroup to create new contacts easily
import com.consol.cmas.common.model.customfield.meta.*
import com.consol.cmas.common.model.customfield.*

def isAutoBindingEnabled() {
   return true;
}
def myunit = new Unit()
def mycustomergroup = customerGroupService.getByName("MyCustomerGroup")

myunit.setCustomerGroup(mycustomergroup)

def mycustomerdefinition = unitDefinitionService.getByName("customer")
myunit.setDefinition(mycustomerdefinition)

myunit.set("company()", unit)

def fkey_newFirstname = new FieldKey("customer","firstname")
def newFirstname = formFields.get(fkey_newFirstname).value
myunit.set("customer.firstname", newFirstname)
def fkey_newName = new FieldKey("customer","name")
def newName = formFields.get(fkey_newName).value
myunit.set("customer.name", newName)

client.goToCreateUnit(myunit)
```

Code-Beispiel 77: *Kundenaktionsskript (hier: Firmenskript -> Firma implizit verfügbar), um die Seite Neuer Kunde zu öffnen*

Eine URL öffnen

- `client.goToPage(url)`
  Leitet auf die Seite einer angegebenen URL weiter.

Beispiel:

```
// opens company's web site
def url = unit.get("company.www")

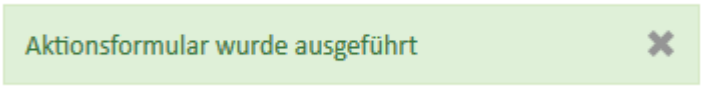
if (!url) {
   client.showErrorMessage("error.script.no.url")
} else {
   client.goToPage(url)
}
```

Code-Beispiel 78: *Öffnen einer URL*

Ergebnismeldung im Web Client anzeigen: SUCCESS, grüne Meldungen

- `client.success()`
  Zeigt die Meldung „Aktion erfolgreich“ mit grünem Hintergrund an. Setzt den Operationstyp auf OperationType.SUCCESS.
- `client.showInfoMessage(string or label)`
  Zeigt den String oder die in der Bezeichnung codierte Meldung mit grünem Hintergrund an.
- `client.showMessage(string or label)`
  Zeigt den String oder die in der Bezeichnung codierte Meldung mit grünem Hintergrund an.
- `showMessage(String pMessageKey, OperationMessage.Severity.INFO)`
  Zeigt den String oder die in der Bezeichnung codierte Meldung mit grünem Hintergrund an. Ein anderer Schweregrad würde den Meldungshintergrund in einer anderen Farbe anzeigen. Im folgenden Beispiel wird eine Bezeichnung verwendet.
- **Beispiel:**

```
client.showMessage("controlForm.info.executionOK",
 OperationMessage.Severity.INFO)
```



Ergebnismeldung im Web Client anzeigen: FAILURE, ERROR, rote Meldungen

- `client.failure()`
  Zeigt die Meldung „Aktion gescheitert“ mit rotem Hintergrund an. Setzt den Operationstyp auf OperationType.FAILURE.
- `client.showDebugMessage(string or label)`
  Zeigt den String oder die in der Bezeichnung codierte Meldung mit rotem Hintergrund an.
- `showMessage(String pMessageKey, OperationMessage.Severity.WARN)`
  Zeigt den String oder die in der Bezeichnung codierte Meldung mit rotem Hintergrund an.

Arbeiten mit Relationsaktionen

Relationsaktionen sind für die drei wichtigsten Objekttypen verfügbar und werden im Admin Tool in den entsprechenden Navigationselementen definiert. Detaillierte Erklärungen finden Sie in den entsprechenden Abschnitten dieses Handbuchs.

- **Units / Kunden**
  erklärt im Abschnitt Action Framework - Kundenaktionen
- **Tickets**
- **Ressourcen**
  erklärt im Abschnitt CM/Resource Pool - Ressourcenaktionen

Eine Relationsaktion wird ausgeführt, wenn die Relation

- erstellt
- gelöscht

wird. Sie wird nicht ausgeführt, wenn der Kommentar einer Relation geändert wird.



Bedenken Sie, dass eine Relationsaktion immer ausgeführt wird, wenn der Objekttyp, für den die Relation definiert ist, beteiligt ist. Das bedeutet, dass zwei Aktionen ausgeführt werden, wenn sowohl das Quell- als auch das Zielobjekt zu einem Typ mit einer Relationsaktion gehören.

In Skripten, die Relationsaktionen definieren, sind folgende Objekte verfügbar:

- Unit-Unit-Relation:
  - Unit-Objekt
  - Relationsobjekt (`UnitRelation`)
- Unit-Ticket-Relation:
  - Unit-Objekt
  - Ticketobjekt
  - Rollenobjekt (`ContactTicketRole`)
- Ressource-Unit-Relation:
  - Ressourcenobjekt
  - Unit-Objekt
  - Relationsobjekt (`ResourceUnitRelation`)
- Ressource-Ticket-Relation:
  - Ressourcenobjekt
  - Ticketobjekt
  - Relationsobjekt (`ResourceTicketRelation`)
- Ressource-Ressource-Relation:
  - Ressourcenobjekt
  - Relationsobjekt (`ResourceResourceRelation`)

Ab ConSol CM-Version 6.11.1.1 ist es möglich, zwischen Operationen, in denen eine Relation hinzugefügt wurde, und Operationen, in denen eine Relation entfernt wurde, zu unterscheiden. Zu diesem Zweck können Sie den Kontextparameter `actionType` verwenden. Er kann in einem Aktions-Bedingungsskript verwendet werden, z. B. um zu steuern, ob das Aktionsskript ausgeführt werden soll oder nicht, oder im Aktionsskript selber eingesetzt werden. `actionType` kann einen der beiden folgenden Werte haben:

- ADD
- REMOVE

Im folgenden Beispiel soll die Person, die der Ansprechpartner für eine SLA ist, eine E-Mail erhalten, wenn eine neue Relation zwischen der SLA und einem Kunden erstellt wird oder eine vorhandene Relation gelöscht wird.

Gehen Sie folgendermaßen vor:

1. Definieren Sie eine Relation, in diesem Beispiel *SLA_to_Company_Relation*, bei der Quelle = Ressourcentyp *SLAs* und Ziel = *Firma* ist, für mehrere Kundengruppen.
2. Schreiben Sie das Ressourcenaktions-Skript, in diesem Beispiel: `SLA_CompanyRelationAction.groovy`
3. Definieren Sie eine Ressourcenaktion (in diesem Beispiel: *SLA_CompanyRelationAction*) des Typs *Relation*, die das Skript als Aktionsskript verwendet.
4. Weisen Sie die Ressourcenaktion einem Ressourcentyp zu, in diesem Beispiel dem Typ *SLAs*.
5. Testen Sie den gewünschten Anwendungsfall im Web Client

Beispiel für Relationsaktions-Skript `SLA_CompanyRelationAction.groovy`:

```
import com.consol.cmas.common.model.mail.MailSendHolder
log.info 'SLA to company relation action has been triggered!'
def myUnit = relation.targetUnit
def myUnitDef = relation.targetUnit.definition.name
log.info 'myUnitDef is ' + myUnitDef
def nameField
def groupField
switch(myUnitDef) {
   case "ResellerCompany": nameField = "company_name";
      break;
   case "company": nameField = "name1";
      break;
   case "DirCustCompany": nameField = "dir_cust_company_name"
      break;
}
def myCustName = myUnit.get(nameField)
log.info 'customer name is ' + myCustName
def contPersMail = resource.get("SLA_Fields_basic.responsible_person_email")
   if (contPersMail) {
   log.info ' Email about SLA will be sent to ' + contPersMail

   // Send an email asynchronuously. Mail object is not available outside workflow
    context!
   def pResName = resource.get("SLA_Fields_basic.SLA_Name")

   // alternative to fixed text: use text template, not shown here

// distinguish between ADD and REMOVE of relation
def relOp
if (actionType == "ADD"){
   relOp = "added to "
} else if(actionType == "REMOVE") {
   relOp = "removed from "
}
```

```
 def pText = "A relation has been " + relOp + "from resource " + pResName + " to
 company " + myCustName

  def pTo = contPersMail
  def pSubject = " new or deleted SLA relation - please take care"
  def pHtml = false
  def pFromEmail = configurationService.getValue("cmweb-server-
   adapter","mail.reply.to")

  def holder = MailSendHolder.createSelfSendHolder(pText, pHtml, null, null)
  holder.setSubject(pSubject)
  holder.setTo(pTo)
  holder.setFrom(pFromEmail)
  mailService.sendMailAsynchronous(holder)
} else {
  log.info 'No mail sent to SLA contact person, no email address found'
}
```

Code-Beispiel 79: *Relationsaktions-Skript, mit dem eine E-Mail gesendet wird, wenn eine Relation zwischen Ressource und Firma erstellt oder gelöscht wird*

Arbeiten mit Daten aus Aktionsformularen

Spezifische Objekte von Aktionsformularen

In Skripten im Zusammenhang mit Aktionsformularen (siehe Abschnitt Aktionsformulare), d. h. in folgenden Skripten, sind zwei besondere Objekte verfügbar:

- Aktionsskripte (der Aktion, die ausgeführt wird, wenn das Aktionsformular ausgefüllt wurde)
- Bedingungsskripte für Aktionsformulare
- Vorbelegungsskripte für Aktionsformulare

Die beiden Objekte sind:

- **formFields** (Map<FieldKey, AbstractField>)
  Die Einträge in der Map stellen die Datenfelder und die entsprechenden Werte im Formular dar. In Bedingungsskripten für Aktionsformulare sind `formFields` erst verfügbar, nachdem das Aktionsformular zum ersten Mal ausgefüllt wurde.
- **controlForm** (Klasse `TicketActionControlForm`, `UnitActionControlForm` oder `ResourceActionControlForm`, je nach Kontext)
  Stellt das Aktionsformular dar. Kann „null“ sein, wenn kein Aktionsformular verwendet wird. In Bedingungsskripten für Aktionsformulare ist das Objekt `controlForm` erst verfügbar, nachdem das Aktionsformular zum ersten Mal ausgefüllt wurde.

Außerdem wurden neue Methoden zur Klasse `ControlFormService` hinzugefügt, mit denen Aktionsformulare mit unterschiedlichen Parametern als Eingabewert abgerufen werden können:



Abbildung 429: *ConSol CM API Doc - Methoden der Klasse ControlFormService speziell für die Arbeit mit Aktionsformularen*

AutoBinding

Die Einstellung (eingeschaltet/ausgeschaltet) von AutoBinding, zum Beispiel in einem Aktionsskript, das ausgeführt wird, wenn ein Aktionsformular ausgefüllt wurde, definiert das Verhalten des Systems bezüglich der Werte des Formulars:

- **AutoBinding ausgeschaltet**
  Die Werte des Formulars sind nur im Formular verfügbar. Sie können über das Objekt `formFields` abgerufen werden.
- **AutoBinding eingeschaltet**
  Die Werte aus dem Formular werden zum Ändern der Werte (Eigenschaften) des Originalobjekts verwendet, da eine Verknüpfung zwischen den Datenfeldern und den Objekteigenschaften besteht. Die Formularwerte sind auch im Objekt `formFields` verfügbar, aber das Originalobjekt wird ebenfalls geändert.

Das folgende Beispiel erklärt den Unterschied zwischen den beiden Werten.

Das Skript `CheckAndUpdateCompanyServiceStatus` ist ein Kundenaktionsskript.

Beispiel AutoBinding ausgeschaltet:

Skript:

```
def isAutoBindingEnabled() {
   return false;
}

log.info ' Getting FormFields ...'
log.info ' FormFields are ' + formFields

client.showMessage("controlForm.info.executionOK", OperationMessage.Severity.INFO)
```

Web Client:

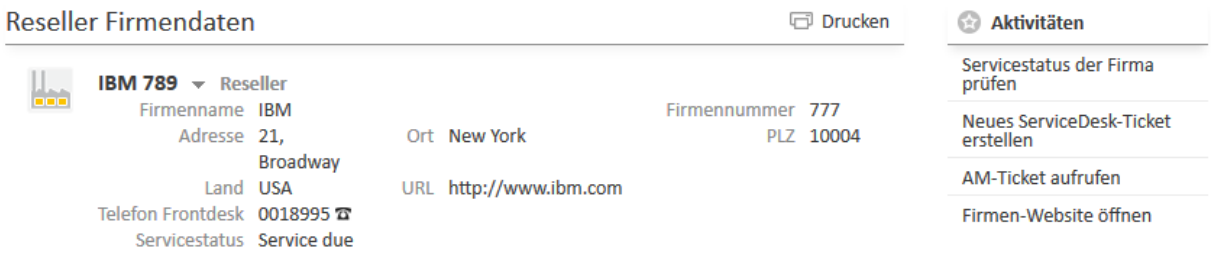


Abbildung 430: *Firmenseite vor der Ausführung der Kundenaktion*



Abbildung 431: *Aktionsformular der Kundenaktion, Wert ist im Formular geändert*

Ausgabe in `server.log`, geänderter Wert im Objekt `formFields` verfügbar:

```
2018-01-04 14:46:31,965 INFO [heckServiceStatus1515073591944] [Susan-c4258118-
f127-11e7-af12-858479a7cfd7] Getting FormFields ...

2018-01-04 14:46:31,965 INFO [heckServiceStatus1515073591944] [Susan-c4258118-
f127-11e7-af12-858479a7cfd7] FORM FIELDS are a null

2018-01-04 14:46:31,966 INFO [heckServiceStatus1515073591944] [Susan-c4258118-
f127-11e7-af12-858479a7cfd7] FormFields are
[(address,ResellerCompanyData):AbstractField{key=(address,ResellerCompanyData),
value=21, Broadway}, (company_name,ResellerCompanyData):AbstractField{key=
(company_name,ResellerCompanyData), value=IBM}, (company_
number,ResellerCompanyData):AbstractField{key=(company_
number,ResellerCompanyData), value=789}, (city,ResellerCompanyData):AbstractField
{key=(city,ResellerCompanyData), value=New York},
(zip,ResellerCompanyData):AbstractField{key=(zip,ResellerCompanyData),
value=10004}, (service_status,ResellerCompanyData):AbstractField{key=(service_
status,ResellerCompanyData), value=EnumValue[name=serviceDue,orderIndex=2]}]
```

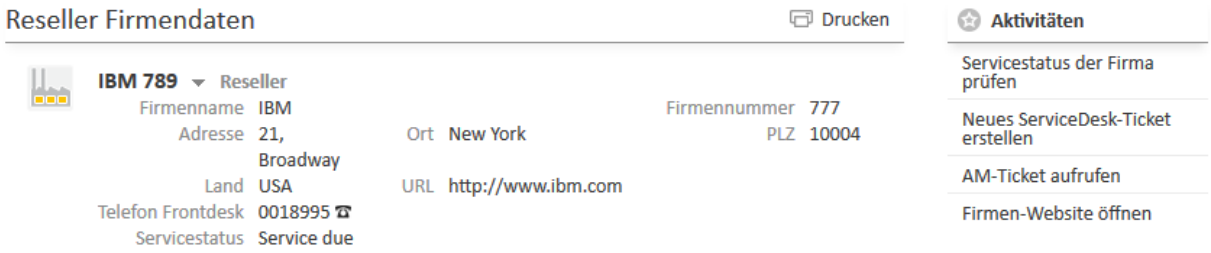


Abbildung 432: *Firmenseite nach der Ausführung der Kundenaktion. Es hat sich nichts geändert, da AutoBinding ausgeschaltet ist*

Beispiel AutoBinding eingeschaltet:

Skript:

```
def isAutoBindingEnabled() {
   return true;
}

log.info ' Getting FormFields ...'
log.info ' FormFields are ' + formFields

client.showMessage("controlForm.info.executionOK", OperationMessage.Severity.INFO)
```

Web Client:

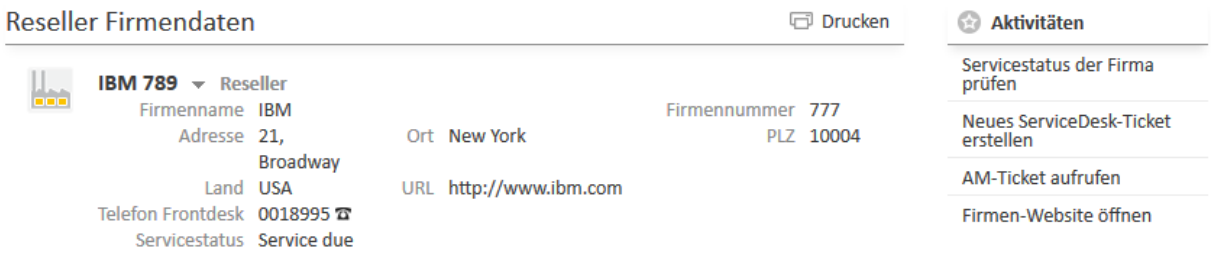


Abbildung 433: *Firmenseite vor der Ausführung der Kundenaktion*



Abbildung 434: *Aktionsformular der Kundenaktion, Wert ist im Formular geändert*

Ausgabe in `server.log`, geänderter Wert im Objekt `formFields` verfügbar:

```
2018-01-04 14:46:31,965 INFO [heckServiceStatus1515073591944] [Susan-c4258118-
f127-11e7-af12-858479a7cfd7] Getting FormFields ...

2018-01-04 14:46:31,965 INFO [heckServiceStatus1515073591944] [Susan-c4258118-
f127-11e7-af12-858479a7cfd7] FORM FIELDS are a null

2018-01-04 14:46:31,966 INFO [heckServiceStatus1515073591944] [Susan-c4258118-
f127-11e7-af12-858479a7cfd7] FormFields are
[(address,ResellerCompanyData):AbstractField{key=(address,ResellerCompanyData),
value=21, Broadway}, (company_name,ResellerCompanyData):AbstractField{key=
(company_name,ResellerCompanyData), value=IBM}, (company_
number,ResellerCompanyData):AbstractField{key=(company_
number,ResellerCompanyData), value=789}, (city,ResellerCompanyData):AbstractField
{key=(city,ResellerCompanyData), value=New York},
(zip,ResellerCompanyData):AbstractField{key=(zip,ResellerCompanyData),
value=10004}, (service_status,ResellerCompanyData):AbstractField{key=(service_
status,ResellerCompanyData), value=EnumValue[name=serviceDue,orderIndex=2]}]
```

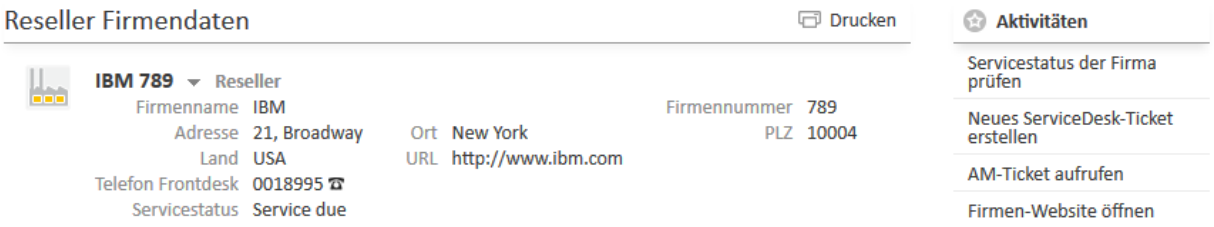


Abbildung 435: *Firmenseite nach der Ausführung der Firmenaktion. Der Wert der Firmennummer hat sich geändert, da AutoBinding eingeschaltet ist. Der neue Wert wird nach einem Neuladen der Seite angezeigt.*

F.6.4 Aktionsformulare

F.6.4.1 Einführung

Was sind Aktionsformulare?

Aktionsformulare bieten die Möglichkeit, Daten zu sammeln, bevor eine bestimmte Aktion ausgeführt wird. So wie ACFs (Aktivitäts-Formulare, siehe Tab Aktivitätsformulare) Daten für Workflow-Aktivitäten sammeln, sammeln Aktionsformulare Daten für das ConSol CM Action Framework, d. h. für folgende Aktionstypen:

- Suche-Aktionen für Tickets
- Kundenaktionen
    - Manuelle Aktionen
    - Suche-Aktionen
- Ressourcenaktionen
    - Manuelle Aktionen
    - Suche-Aktionen

Das Aktionsformular wird geöffnet und muss vor der Aktion ausgefüllt werden, d. h. bevor das Aktionsskript ausgeführt wird. Eine detaillierte Einführung in Skripte für Aktionen des Action Framework finden Sie im Abschnitt Skripte für das Action Framework.

Wie ACFs können auch Aktionsformulare mit Skripten angepasst werden. Jedes Aktionsformular kann folgende Skripte haben. Beide Skripte sind optional, Aktionsformulare funktionieren auch ohne Skripte.

- Aktionsformular Bedingungsskript
  Nur wenn die Bedingung erfüllt ist („true“), wird das Aktionsformular angezeigt. Siehe AbschnittAktionsformular Bedingungsskript, optional
- Aktionsformular Vorbelegungsskript
  Kann Werte im Aktionsformular vorbelegen, d. h. wenn das Formular angezeigt wird, sind die entsprechenden Felder ausgefüllt. Siehe Abschnitt Aktionsformular Vorbelegungsskript, optional.

Folgende Anwendungsfälle können beispielsweise mit Aktionsformularen abgedeckt werden:

- **Firmenaktion**
  Geben Sie neue Kontaktdaten in das Formular ein - erstellen Sie mit einem Klick einen neuen Kontakt für die Firma.
- **Suche-Aktion für Tickets**
  Geben Sie mit einem Klick ein neues Ablaufdatum für den Vertrag für jedes Asset einer Liste an IT-Assets ein, das in einer Suche gefunden wurde.
- **Suche-Aktion für Firmen**
  Geben Sie alle Daten über eine anstehende Konferenz in jeder Firma einer Liste an Firmen ein, die Sie einladen möchten.

- **Suche-Aktion für Kontakte**
  Geben Sie einfach den Namen der E-Mail-Vorlage ein, die Sie verwenden möchten, und senden Sie eine E-Mail an eine Liste an Empfängern:
  - Infomieren Sie Bewerber über das Datum des Assessment Centers.
  - Informieren Sie alle Mitarbeiter, die eine bestimmte Art von Software-Lizenz haben, über die Notwendigkeit, die Lizenz zu aktualisieren.

Wenn ein Aktionsformular angezeigt wurde und die entsprechenden Daten ausgefüllt oder geändert wurden, wird das Aktionsskript, in dem die Ticket-/Kunden-/Ressourcenaktion implementiert ist, ausgeführt. Es sind spezielle Methoden verfügbar, um die Daten aus dem Aktionsformular zu verarbeiten. Dies ist im Abschnitt Arbeiten mit Daten aus Aktionsformularen detailliert beschrieben.

Das folgende Beispiel vermittelt Ihnen einen ersten Eindruck über die Funktionsweise von Aktionsformularen.

Erster Eindruck von Aktionsformularen - Aktionsformular für Ressourcenaktion

Die Ressourcenseite eines HP-Druckers ist geöffnet und der Bearbeiter führt die Ressourcenaktivität *Wartungsticket für HP-Drucker erstellen* aus. Es wird ein Formular geöffnet, in dem die Deadline und die Priorität des neuen Tickets eingegeben werden müssen. Wenn der Bearbeiter auf *Speichern und weiter* geklickt hat, werden die Formulardaten gespeichert und die Ressourcenaktion wird ausgeführt. Im aktuellen Beispiel wäre es nicht möglich, nichts einzugeben, da das Formular Pflichtfelder (mit rotem Sternchen markiert) enthält.



Abbildung 436: *ConSol CM Web Client - Aktionsformular einer Ressourcenaktion*

F.6.4.2 Definieren von Aktionsformularen mit dem Admin Tool

Alle Komponenten von Aktionsformularen werden im Admin Tool definiert.

Ein Aktionsformular enthält mindestens eine und bis zu drei Komponenten:

- Das Aktionsformular selber, Pflichtangabe
- Aktionsformular Bedingungsskript, optional
- Aktionsformular Vorbelegungsskript, optional

Das Aktionsformular selber, Pflichtangabe

Es hängt vom Typ der Aktion ab, zu der das Formular zugewiesen werden soll, an welcher Stelle des Admin Tools das Aktionsformular definiert werden muss:

- Suche-Aktionen für Tickets:
  Navigationsgruppe *Tickets*, Navigationselement *Suche-Aktionen*, Tab *Aktions-Formulare*
- Für Kundenaktionen
  Navigationsgruppe *Kunden*, Navigationselement *Aktionen*, Tab *Aktions-Formulare*
- Für Ressourcenaktionen:
  Navigationsgruppe *Ressourcen*, Navigationselement *Aktionen*, Tab *Aktions-Formulare*

Im folgenden Beispiel erklären wir eine Kundenaktion. Das Prinzip gilt für alle Aktionsformulare.

Führen Sie folgende Schritte durch:

- Definieren der Datenfelder für das neue Aktionsformular
- Zuweisen des Aktionsformulars zur Aktion, in der es geöffnet werden soll
- Testen des neuen Aktionsformulars

Definieren der Datenfelder für das neue Aktionsformular

Um ein neues Aktionsformular für eine Kundenaktion zu erstellen, öffnen Sie die Navigationsgruppe *Kunden*, Navigationselement *Aktionen*, Tab *Aktions-Formulare*. Klicken Sie auf *Hinzufügen* und wählen Sie die Datenfelder, die zum neuen Aktionsformular gehören sollen, im Abschnitt *Verfügbar* aus und verschieben Sie sie in den Abschnitt *Zugewiesen*.



Beachten Sie, dass anders als in ACFs für Tickets, in denen nur Ticketfelder verfügbar sind, in einem Aktionsformular alle Arten von Datenfeldern verfügbar sind:

- Ticketfelder
- Kundenfelder:
  - Firmenfelder
  - Kontaktfelder
- Ressourcenfelder

Dies bedeutet, dass ein Aktionsformular andere Daten, als die des Objektes der Aktion, enthalten kann. Zum Beispiel: In einer Kontaktaktion können nicht nur die Daten des Kontakts, wie E-Mail oder Telefonnummer angefordert werden, sondern auch jedes andere Datenfeld, wie Daten des Tickets oder der Firma. Dafür sind zwei Aspekte wichtig:

1. Das Aktionsskript entscheidet, was mit den Daten des Formulars geschieht. Der Entwickler des Skripts ist für die Datenverarbeitung verantwortlich!
2. Das Verhalten des Aktionsskripts bezüglich „Auto-binding“ ist entscheidend. Siehe Abschnitt AutoBinding.

In den Drop-down-Menüs über der Liste der Datenfelder, die als Filter dienen, können Sie auswählen, welche Art von Datenfeldern angezeigt werden sollen:

- Ticket
- Kunde
- Ressource

Im zweiten Drop-down-Menü / Filter können Sie die Datenfeldgruppe, deren Felder in der Liste *Verfügbar* angezeigt werden sollen, auswählen. Wenn Sie einen bestimmten Datenfeldtyp im ersten Drop-down-Menü / Filter ausgewählt haben, werden nur die passenden Datenfeldgruppen angezeigt, z. B. nur Felder der Kundenfeldgruppe *ResellerCompanyData* (in unserem Beispiel).

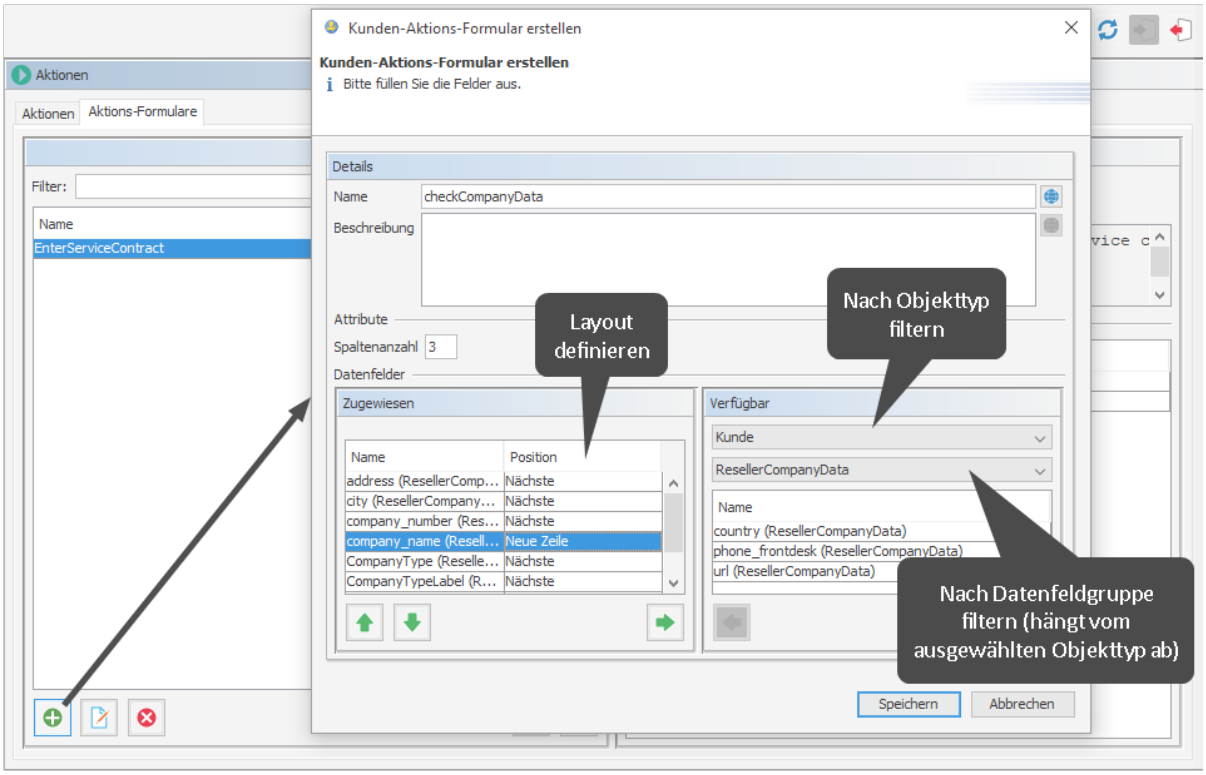


Abbildung 437: *ConSol CM Admin Tool - Kunden, Aktionen: Erstellen eines neuen Aktionsformulars für eine Kundenaktion*

In diesem Beispiel wurden einige Firmenfelder der Kundenfeldgruppe *ResellerCompanyData* ausgewählt.

Zuweisen des Aktionsformulars zur Aktion, in der es geöffnet werden soll

Nach der Definition des neuen Aktionsformulars müssen Sie das Formular einer Aktion des Action Framework zuweisen. In unserem Beispiel einer Firmenaktion.

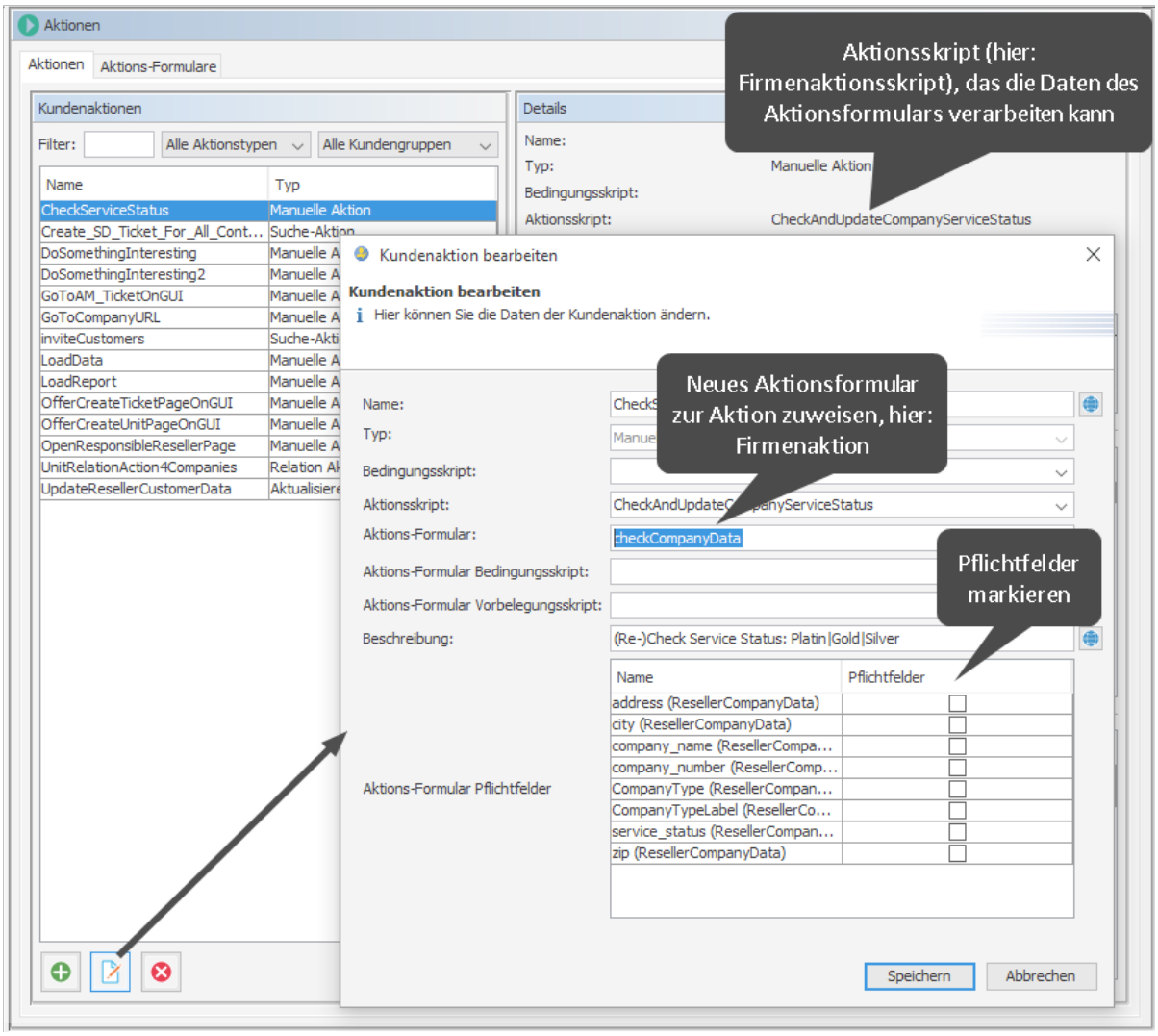


Abbildung 438: *ConSol CM Admin Tool - Kunden, Aktionen: Zuweisen eines neuen Aktionsformulars zu einer Firmenaktion*

Testen des neuen Aktionsformulars

Öffnen Sie eine Firmenseite in der Kundengruppe, der die manuelle Aktion zugewiesen ist, und führen Sie die Firmenaktion aus. Das Aktionsformular wird geöffnet und kann geändert werden.

Im aktuellen Beispiel sind die Felder bereits gefüllt, da die Firmenfelder bereits ausgefüllt sind und die Daten einfach angezeigt werden. Sie können im Formular geändert werden.

> Beachten Sie das unterschiedliche Systemverhalten bezüglich des Speicherns von Formulardaten je nach Einstellung von „Auto-binding“. Dies ist im Abschnitt AutoBinding näher erklärt.

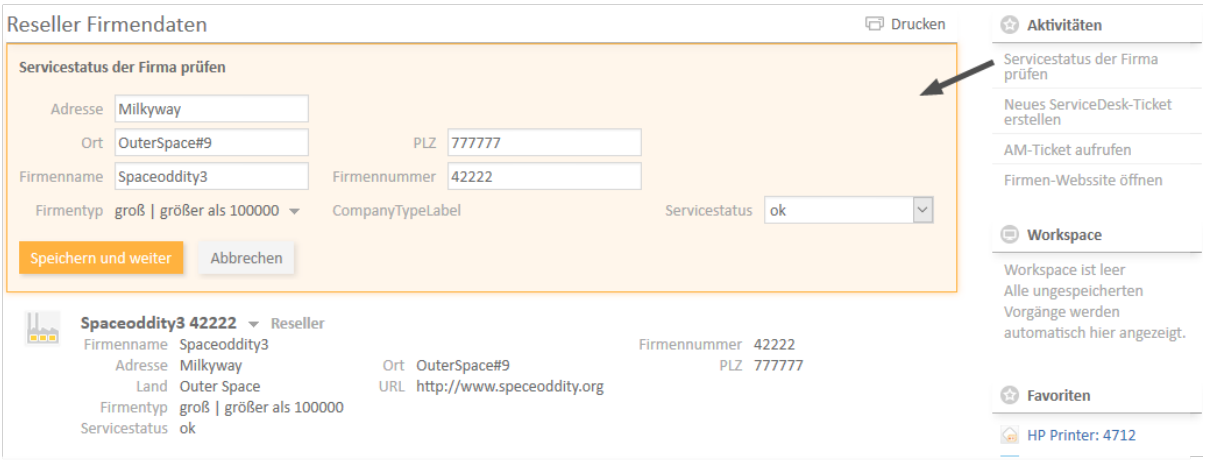


Abbildung 439: *ConSol CM Web Client - Aktionsformular einer Firmenaktion*

Aktionsformular Bedingungsskript, optional

Im Web Client wird ein Aktionsformular nur angezeigt, wenn

- es kein Bedingungsskript gibt

oder

- es ein Bedingungsskript gibt, das „true“ zurückgegeben hat

Das Aktionsformular Bedingungsskript wird im Admin Tool im Abschnitt *Skripte* gespeichert und muss den Typ *Bedingung für Aktions-Formular* haben. Es muss entweder „true“ oder „false“ zurückgeben. Führen Sie zum Implementieren eines Aktionsformular Bedingungsskripts folgende Schritte aus:

1. Schreiben Sie das Aktionsformular Bedingungsskript.
2. Weisen Sie das Skript der Aktion als *Aktions-Formular Bedingungsskript* zu.

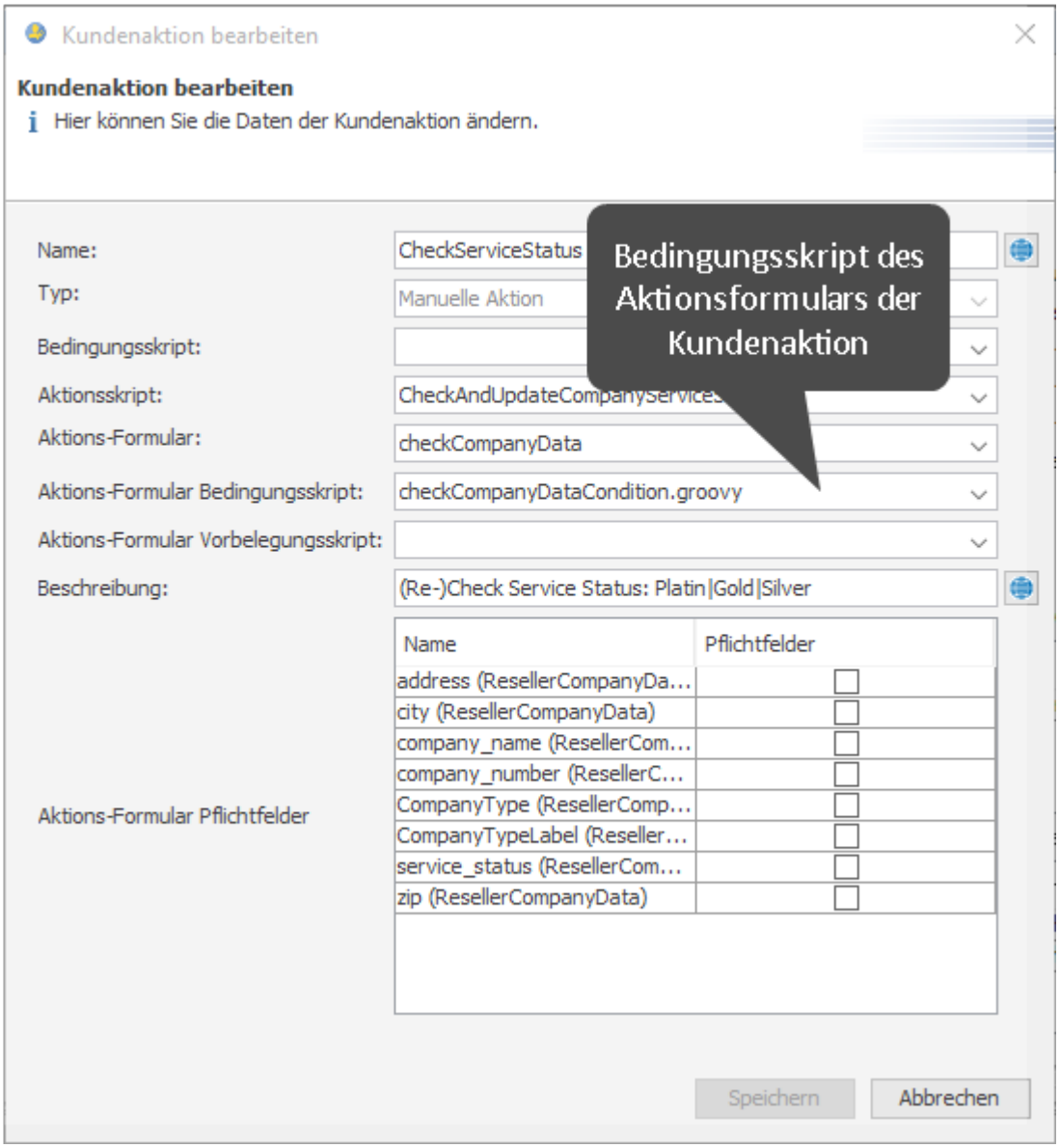


Abbildung 440: *ConSol CM Admin Tool - Kunden, Aktionen: Zuweisen eines Bedingungsskripts zu einem Aktionsformular einer Firmenaktion*

Wenn das Skript „true“ zurückgibt, wird das Aktionsformular angezeigt. Wenn es „false“ zurückgibt, wird das Aktionsformular nicht angezeigt, die Aktion wird aber ausgeführt.

Das folgende Beispiel zeigt ein Aktionsformular Bedingungsskript, das überprüft, ob eine Ressource leere Felder hat. Nur dann wird das Formular angezeigt, um die fehlenden Daten zu sammeln. Wenn alle Felder bereits ausgefüllt sind, wird kein Formular angezeigt.

```
if(resource.get("product.quantity") == null || resource.get("product.price") ==
 null) {
   return true
}
return false
```

Code-Beispiel 80: *Aktionsformular Bedingungsskript, das überprüft, ob Felder ausgefüllt sind*

Aktionsformular Vorbelegungsskript, optional

Wenn in einem Formular beim Öffnen bereits einige Felder mit bestimmten Werten vorbelegt sein sollen, müssen Sie der entsprechenden Kunden-, Ticket- oder Ressourcenaktion ein Vorbelegungsskript zuweisen. In diesem Skript können die Felder ausgefüllt werden.

Das folgende Aktionsformular Vorbelegungsskript könnte zum Beispiel sicherstellen, dass der Wert für den Servicestatus vom Bearbeiter auf den aktuellen Wert gesetzt wird.

```
company.set("ResellerCompanyData.service_status","unknown")
```

Im Web Client wird das entsprechende Feld mit dem vorgegebenen Wert ausgefüllt:



Abbildung 441: *ConSol CM Web Client - Kundenaktion mit Aktionsformular, in dem ein Feld von einem Skript vorbelegt wurde*

> Beachten Sie das unterschiedliche Systemverhalten bezüglich des Speicherns von Formulardaten je nach Einstellung von „Auto-binding“. Dies ist im Abschnitt AutoBinding näher erklärt.

Wenn das Formular geöffnet wird:

- Es hängt von der Auto-binding-Definition (siehe Infobox) ab, ob die Felder, die zum Kontextobjekt gehören, ausgefüllt sind oder nicht.
- Werte aus dem Vorbelegungsskript überschreiben die ursprünglichen Werte des Objekts.

Wenn das Formular gespeichert wird:

Die Werte, die innerhalb des Vorbelegungsskripts gesetzt werden, werden nur zum Ausfüllen des Formulars verwendet. Die Daten werden nicht gespeichert. Ob die Daten in den Objekten tatsächlich gespeichert werden, hängt vom Aktionsskript ab, nämlich von:

- der Definition von Auto-binding (siehe Infobox)
- der Verwendung des Inhalts von `formFields` (siehe Arbeiten mit Daten aus Aktionsformularen über `formFields`)

F.7 E-Mail-Konfiguration

ConSol CM verfügt über eine ausgereifte E-Mail-Schnittstelle. CM ruft E-Mails von einem oder mehreren Mail-Servern ab und kann E-Mails versenden. Sowohl eingehende als auch ausgehende E-Mails werden in die entsprechenden Tickets eingefügt, sodass Sie immer Zugang zu allen Informationen über einen Vorgang haben.

Die grundlegenden E-Mail-Funktionen von CM werden in den folgenden Abschnitten erklärt:

- E-Mail
- E-Mail-Backups

Falls Ihr Unternehmen die E-Mail-Verschlüsselung verwenden möchte, können Sie Server- und/oder Client-Zertifikate einsetzen, siehe Abschnitt:

- E-Mail-Verschlüsselung

F.7.1 E-Mail

F.7.1.1 Einführung in die E-Mail-Funktionen in ConSol CM

Das Senden und Empfangen von E-Mails ist eine der Kernfunktionen von ConSol CM. Die Applikation interagiert mit einem oder mehreren Mail-Servern um E-Mails abzuholen und zu senden.

Die folgenden Abschnitte enthalten Informationen über folgende Themen im Zusammenhang mit E-Mails:

- E-Mail-Funktionen in ConSol CM (Senden und Empfangen von E-Mails, E-Mail-Duplizierung)
- Details zur E-Mail-Konfiguration
- E-Mail-Konfiguration mit dem Admin Tool

Informationen über die Einrichtung der E-Mail-Funktionen in einem Cluster finden Sie im *ConSol CM Cluster Manual*, Abschnitt *Email*. Hinweise zur Verwaltung von E-Mails beim Systembetrieb stehen im *ConSol CM Betriebshandbuch*, Abschnitt *Email*.

F.7.1.2 E-Mail-Funktionen in ConSol CM

Dieser Abschnitt enthält eine kurze Einführung in das Thema „E-Mails in Consol CM“. Das Senden und Empfangen von E-Mails ist eine Kernfunktion der Applikation.

Senden von E-Mails aus ConSol CM

Manuelle E-Mails

E-Mails können entweder manuell von einem Bearbeiter oder automatisch vom System gesendet werden. **Manuelle** E-Mails werden mit dem **Ticket-E-Mail-Editor** gesendet. In den meisten Systemen ist standardmäßig der Hauptkunde des Tickets der Empfänger der E-Mail. Der Bearbeiter kann aber eine andere E-Mail-Adresse auswählen oder eingeben. Der systemweite Standardwert kann durch einen Administrator in der Seitenanpassung geändert werden. Dazu wird mailToSelection der Seitenanpassung verwendet. Außerdem kann der Bearbeiter E-Mail-Vorlagen verwenden und/oder Text aus dem Ticket zitieren. Im *ConSol CM Benutzerhandbuch* finden Sie eine detaillierte Einführung in die Arbeit mit dem Ticket-E-Mail-Editor.

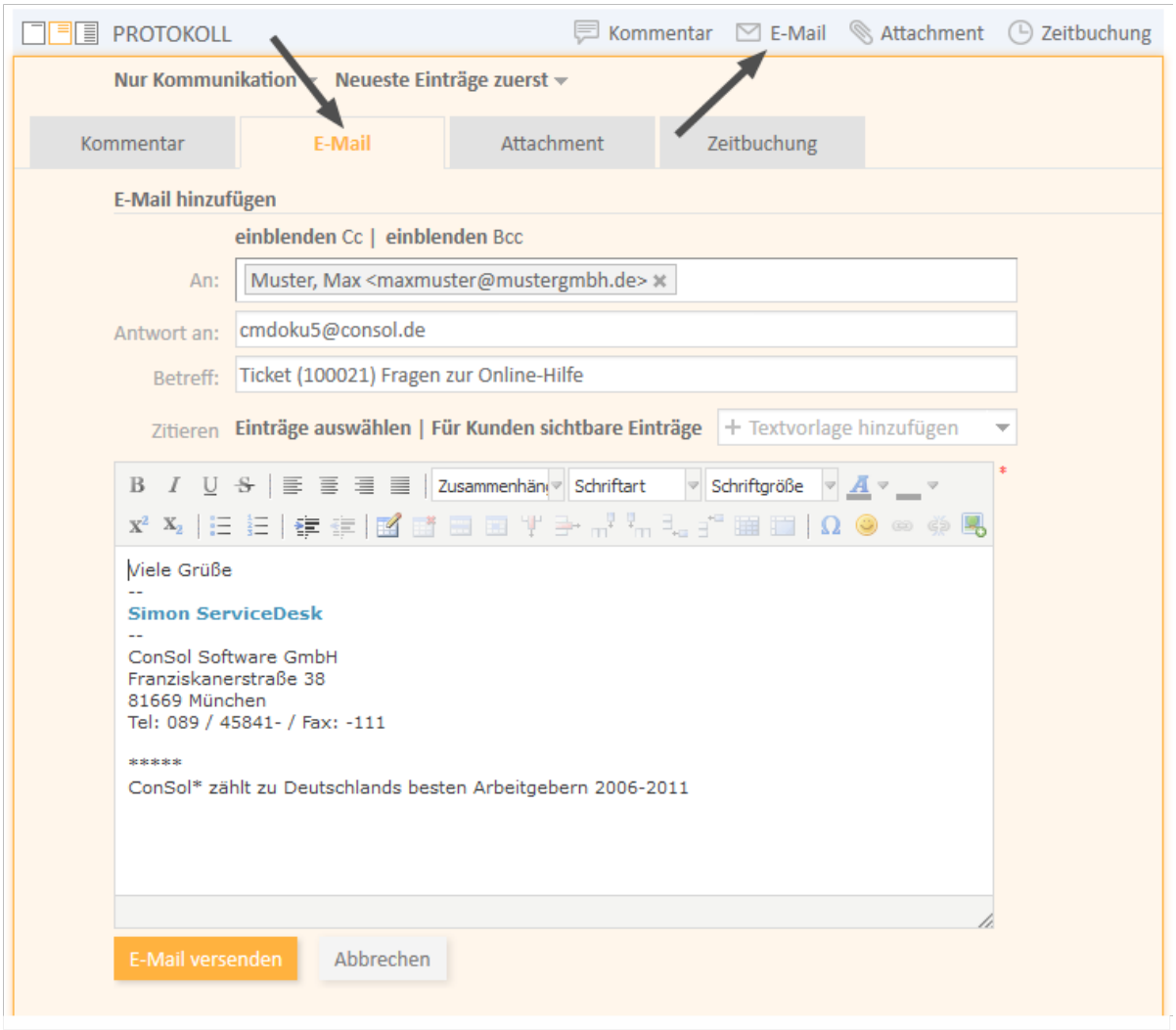


Abbildung 442: *ConSol CM Web Client - Ticket-E-Mail-Editor*

Automatische E-Mails

Automatische E-Mails können von ConSol CM beispielsweise in folgenden Situationen gesendet werden:

1. Durch den Workflow ausgelöst, z. B.
    a. wenn der Bearbeiter, dem das Ticket zugewiesen ist, daran erinnert werden soll, an dem Ticket weiterzuarbeiten.
    b. wenn der Kunde eine automatische Bestätigung erhalten soll, dass für ihn ein Ticket geöffnet wurde.
    c. wenn der Kunde eine automatische Bestätigung erhalten soll, dass sein Ticket geschlos-

sen wurde.

d. wenn ein Vorgesetzter oder Genehmiger eine Nachricht erhalten soll, dass ein neuer Vorgang genehmigt werden muss.

In einer Workflow-Aktivität kann eine E-Mail an jede beliebige gültige E-Mail-Adresse gesendet werden. Eine detaillierte Beschreibung der zu verwendenden Methoden finden Sie im *ConSol CM Process Designer Handbuch*.

2. Durch das System im Fall einer Fehler- oder Erfolgsmeldung ausgelöst, z. B.
   a. Systemfehler
   b. E-Mail-Fehler
   c. DWH-Synchronisierung (Fehler oder Erfolg)
   Normalerweise werden diese E-Mails an den ConSol CM-Administrator gesendet. Für die meisten speziellen Fehlermeldungen kann aber in den System-Properties eine spezielle Empfänger-E-Mail-Adresse konfiguriert werden. Details dazu finden Sie im Abschnitt System-Properties.
3. Ausgelöst durch das ConSol CM-System, um Erinnerungen an Bearbeiter zu schicken
   a. Wenn ein Bearbeiter ein Ticket erhält oder ein Ticket dem Bearbeiter entzogen wird, kann das System eine E-Mail an diesen Bearbeiter senden. Dies kann pro Queue konfiguriert werden, siehe Abschnitt Queue-Verwaltung.

Empfangen von E-Mails mit ConSol CM

Das ConSol CM-System kann E-Mails aus einem oder mehreren Postfächern (= E-Mail-Konten) auf einem oder mehreren Mail-Servern abrufen. Die Postfächer werden im Admin Tool konfiguriert (E-Mail-Funktionen in ConSol CM) Denken Sie daran, dass ConSol CM hier mit Postfächern arbeitet. Jedes Postfach kann von mindestens einer E-Mail-Adresse erreicht werden. In bestimmten Fällen kann ein Postfach für mehr als eine E-Mail-Adresse verwendet werden. Dies kann beim Schreiben von Skripten des Typs E-Mail wichtig sein.

ConSol CM verhält sich gegenüber dem Mail-Server genauso wie ein normaler E-Mail-Client, der E-Mails über ein Standard-E-Mail-Protokoll abruft: IMAP(s), POP3(s). Abhängig von der Konfiguration des Mail-Servers und der ConSol CM-System-Property cmas-nimh, mailbox.default.task.delete.read.messages werden die E-Mails nach dem Abruf durch ConSol CM aus dem Postfach des Mail-Servers gelöscht. Die Standardeinstellung ist, dass **E-Mails nach dem Abruf nicht gelöscht werden**.

⚠ Falls Sie nicht möchten, dass ConSol CM die E-Mails vom Mail-Server löscht, achten Sie darauf, die Postfächer zu überwachen, um einen Datenüberlauf und Server- oder Leistungsprobleme zu vermeiden.

Alle eingehenden E-Mails werden zuerst im E-Mail-Eingangspool in ConSol CM gespeichert und danach in einer Kette von E-Mail-Skripten verarbeitet. Eine detaillierte Beschreibung dieser Skripte finden Sie im Abschnitt Skripte des Typs E-Mail. Wenn eine E-Mail nicht verarbeitet werden kann, erhält der Administrator eine Benachrichtigungs-E-Mail. Die nicht verarbeitete E-Mail ist unter E-Mail-Backups aufgeführt.

Das Standardverhalten des Systems bei einer eingehenden E-Mail ist für folgende Fälle definiert:

- Der Betreff einer E-Mail enthält keine Ticketnummer in einer gültigen Syntax (d. h. er entspricht nicht dem als regulärer Ausdruck (RegEx) für den Betreff definierten Muster):
  Es wird ein neues Ticket erstellt.
- Der Betreff der E-Mail enthält eine Ticketnummer in einer gültigen Syntax (RegEx) und das Ticket ist noch offen:
  Die E-Mail wird an das vorhandene Ticket angehängt.
- Der Betreff der E-Mail enthält eine Ticketnummer in einer gültigen Syntax (RegEx) aber das Ticket ist geschlossen:
  Es wird ein neues Ticket erstellt und dabei eine Referenz zum alten Ticket erzeugt.

Das Standardverhalten des Systems kann durch eine Änderung der E-Mail-Skripte verändert werden (siehe Abschnitt Skripte des Typs E-Mail). Da dadurch allerdings Schlüsselfunktionen des Systems beschädigt werden können, sollten diese Skripte entweder gar nicht oder nur von einem erfahrenen ConSol-Consultant geändert werden!

E-Mail-Duplizierung im ConSol CM Web Client

Eine Beschreibung dazu finden Sie auf der Seite *Seitenanpassung* in showCloneOption und appendOrReplaceOnClone.

F.7.1.3 E-Mail-Konfiguration mit dem Admin Tool

In diesem Abschnitt ist das Navigationselement *E-Mail* der Navigationsgruppe *E-Mail* im Admin Tool erklärt.

WICHTIGE INFORMATION

Seit ConSol CM-Version 6.9.4 gibt es zwei Modi für eingehende E-Mails:

- **Mule/ESB** - Dieser Modus ist seit Version 6 in ConSol CM verfügbar. Er ist in CM-Versionen 6.11 und höher veraltet. Deshalb wird dieser Modus nicht mehr in den ConSol CM-Handbüchern behandelt. Wenn Sie ein älteres CM-System mit Mule/ESB betreiben, finden Sie die nötigen Informationen in den jeweiligen Handbüchern. Alle ConSol CM 6.11-Systeme müssen im NIMH-Modus betrieben werden. Informationen über den Wechsel von Mule/ESB zu NIMH finden Sie in den älteren Administratorhandbüchern.
- **NIMH** - *New Incoming Mail Handler*, seit Version 6.9.4 verfügbar. Dies ist der einzige Modus für eingehende E-Mails in ConSol CM-Versionen 6.11 und höher.

Der aktive E-Mail-Modus wird in der System-Property cmas-core-server, nimh.enabled gesetzt:

- **true** - NIMH-Modus
- **false** - Mule/ESB-Modus

Das Senden von E-Mails, d. h. die Konfiguration des SMTP-Servers, wird vom Modus für eingehende E-Mails nicht beeinflusst.

Allgemeine E-Mail-Konfiguration (Navigationselement *E-Mail*)

In diesem Navigationselement können Sie die Parameter für die E-Mail-Anbindung konfigurieren.

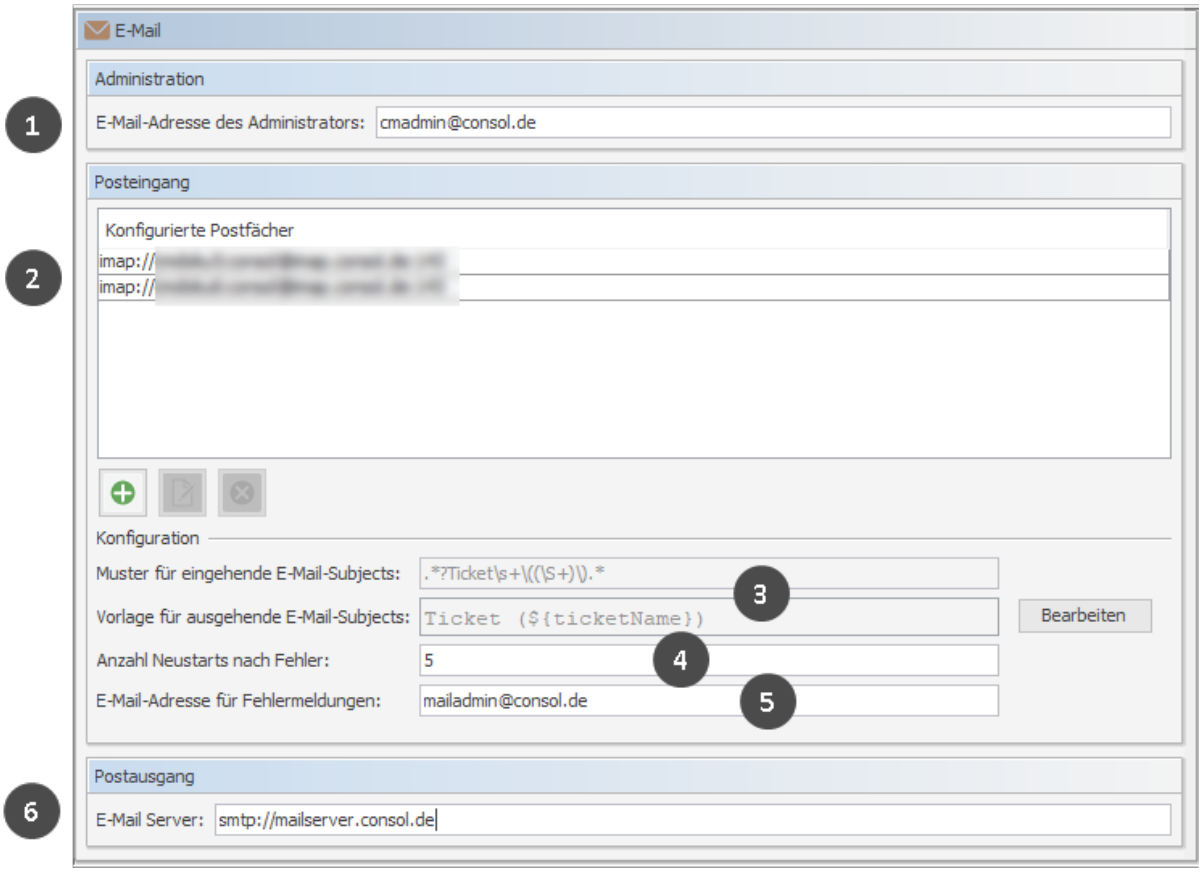


Abbildung 443: *ConSol CM Admin Tool - E-Mail*

Die Seite *E-Mail* enthält folgende Informationen:

- Allgemeine E-Mail-Adresse für Fehlermeldungen (1)
- E-Mail-Postfächer (nicht E-Mail-Adressen!) auf einem oder mehreren E-Mail-Servern, von denen ConSol CM E-Mails abruft (2)
- Konfiguration des RegEx-Musters zum Erkennen von vorhandenen Tickets (eingehende E-Mails) (3)
- Anzahl an erneuten Versuchen nach einem Fehler bei der E-Mail-Zustellung (4)
- E-Mail-Adresse für Fehlermeldungen im Zusammenhang mit E-Mails (5)
- Adresse des E-Mail-Servers für ausgehende E-Mails (6)

⚠ Bei der Verwendung von NIMH muss der NIMH-Dienst angehalten werden, um die Konfiguration von vorhandenen Postfächern (*Konfigurierte Postfächer*) editieren und neue Postfächer hinzufügen zu können.

Administration

E-Mail-Adresse des Administrators:

Geben Sie die E-Mail-Adresse ein, die allgemeine Nachrichten oder Warnungen vom System erhalten soll. Über die GUI können Sie nur eine E-Mail-Adresse für den Administrator eingeben. Mehrere durch Kommas getrennte Adressen sind möglich, müssen aber direkt im Navigationselement System-Properties eingegeben werden. Verwenden Sie dazu die System-Property cmas-core-security, admin.e-mail. Die Gesamtanzahl der Zeichen (sowohl GUI als auch Property) darf 72 nicht überschreiten. Wenn es viele Empfänger gibt, empfehlen wir die Verwendung einer Mailing-Liste auf dem Mail-Server.

Posteingang

Die Konfiguration des Posteingangs ist in zwei Bereiche aufgeteilt:

- **Konfigurierte Postfächer:**
  Hier können Sie die Postfächer, aus denen E-Mails abgerufen werden, in einem Pop-up-Fenster hinzufügen oder editieren. Die Verbindung zum Postfach wird während der Einrichtung geprüft. Es ist also nicht möglich, ein Postfach zu konfigurieren, das nicht verwendet werden kann, wenn das System in Betrieb geht (sofern der Mail-Server nicht geändert wurde o ä.). Die Werte werden als System-Properties im Modul `cmas-nimh` gespeichert. Detaillierte Informationen dazu finden Sie im Abschnitt mit den E-Mail-Properties in System-Properties. Erforderliche Werte sind:
  - **Protokoll**
    Das Protokoll, über das E-Mails vom Server abgerufen werden. Die unterstützten Protokolle sind IMAP4, IMAP4s, POP3 und POP3s. Denken Sie daran, dass sich ConSol CM genauso verhält wie ein normaler E-Mail-Client. Wenn die sichere Version eines Protokolls verwendet wird, ist das entsprechende Zertifikat erforderlich! Es muss im Security Store des Application Servers gespeichert werden.
  - **Server**
    Der über DNS auflösbare Name oder die IP-Adresse des Mail-Servers.
  - **Port**
    Der Port des Mail-Servers, den der Mail Daemon/Service abhört.
  - **Benutzername**
    Der Benutzername des E-Mail-Kontos.
  - **Passwort**
    Das Passwort des E-Mail-Kontos.

    ⚠ Denken Sie daran, dass ein E-Mail-Konto mehr als eine E-Mail-Adresse haben kann. An dieser Stelle geht es um den Kontonamen, d. h. den Namen des Postfaches. Wenn Sie ein Admin-Tool-Skript editieren, das eingehende E-Mails verarbeitet, ist es unter Umständen erforderlich, die E-Mail-Adresse zu benutzen. Die E-Mail-Adresse wird auch benötigt, um die Reply-To-Adresse, die From-Adresse und die queue-spezifischen E-Mail-Adressen zu konfigurieren! Stellen Sie also sicher, dass Sie den richtigen Parameter verwenden: Postfach oder E-Mail-Adresse!

- **Konfiguration:**
  - **Muster für eingehende E-Mail-Subjects:**
    Beschreibt die Elemente, die der Betreff einer eingehenden E-Mail enthalten muss, damit diese E-Mail einem bestimmten Ticket zugewiesen werden kann. Das Muster wird in einem regulären Ausdruck definiert (RegEx).
    Beispiel: *.*?Ticket\s+\((\S+)\).** passt zu allen Betreffzeilen, die *Ticket (<Ticketnummer>)* enthalten.
  - **Vorlage für ausgehende E-Mail-Subjects:**
    Beschreibt das Muster, das zur Erzeugung der Ticket-ID im Betreff einer ausgehenden E-Mail verwendet wird. Die Vorlage sollte mit dem Muster für eingehende E-Mail-Subjects zusammenpassen. Über den Button *Bearbeiten* auf der rechten Seite können Sie das Muster für eingehende E-Mail-Subjects und die Vorlage für ausgehende E-Mail-Subjects ändern und prüfen, ob sie zueinander passen.
    Beispiel: *Ticket (${ticketName})* passt zum obigen RegEx-Beispiel.

    > ⚠ Mit dem Button *Bearbeiten* können Sie im offenen Editorfenster überprüfen, ob das Muster für eingehende E-Mail-Subjects und die Vorlage für ausgehende E-Mail-Subjects zueinander passen. Stellen Sie sicher, dass das E-Mail-Subject an **allen** Stellen korrekt gesetzt wird, d. h. auch in allen Workflow-Skripten und Admin-Tool-Skripten!

  - **Anzahl Neustarts nach Fehler:**
    Zeigt die maximale Anzahl der Neustarts nach einem Fehler beim Abrufen der E-Mails durch ConSol CM. Gilt für alle E-Mail-Poller.
  - **E-Mail-Adresse für Fehlermeldungen:**
    E-Mail-Adresse, an die Nachrichten und Warnungen vom Mail-System gesendet werden. Normalerweise stimmt sie mit der allgemeinen Administrator-E-Mail-Adresse überein.

Für die Konfiguration der Postfächer können Sie auch die System-Properties für die E-Mail-Funktion verwenden, siehe System-Properties. Besonders das Abrufintervall (Zeitintervall für den Abruf von E-Mails vom Mail-Server, System-Property Alphabetische Liste der System-Properties (Modul `cmas-nimh`) oder die entsprechende Property eines anderen Postfaches (z. B. `mailbox.1.task.interval.seconds`) könnte von Interesse sein.

Postausgang

Hier werden die Verbindungsdaten für ausgehende E-Mails konfiguriert:

Connector für ausgehende E-Mails

Verwenden Sie folgendes Format:

```
smtp://<IP address of mail server>:<port>
```

Beispiel mit Standard-Port:

```
smtp://123.123.123.123:25
```

Beispiel für SMTP-Server mit Authentifizierung:

```
smtp://test:testpassword@ConSolMailServer2:25
```

Im obigen Beispiel werden folgende Parameter verwendet:

- Protokoll: SMTP
- Benutzername: test
- Passwort: testpassword
- Host/Mail-Server: ConSolMailServer2
- Port: 25

Beachten Sie: Wenn der anzugebende Benutzername und/oder das Passwort das Zeichen „@“ enthält, muss das „@“ als „%40“ codiert werden.

Um SMTPS für ausgehende E-Mails zu aktivieren, müssen Sie die System-Property cmas-core-server, mail.smtp.tls.enabled auf „true“ setzen.

F.7.1.4 Details zur E-Mail-Konfiguration

Dieser Abschnitt beschreibt die E-Mail-Funktionen in einer Umgebung mit einem Server. Informationen über Umgebungen mit mehreren Servern finden Sie im *ConSol CM Cluster Manual*.

WICHTIGE INFORMATION

Seit ConSol CM-Version 6.9.4 gibt es zwei Modi für eingehende E-Mails:

- **Mule/ESB** - Dieser Modus ist seit Version 6 in ConSol CM verfügbar. Er ist in CM-Versionen 6.11 und höher veraltet. Deshalb wird dieser Modus nicht mehr in den ConSol CM-Handbüchern behandelt. Wenn Sie ein älteres CM-System mit Mule/ESB betreiben, finden Sie die nötigen Informationen in den jeweiligen Handbüchern. Alle ConSol CM 6.11-Systeme müssen im NIMH-Modus betrieben werden. Informationen über den Wechsel von Mule/ESB zu NIMH finden Sie in den älteren Administratorhandbüchern.
- **NIMH** - *New Incoming Mail Handler*, seit Version 6.9.4 verfügbar. Dies ist der einzige Modus für eingehende E-Mails in ConSol CM-Versionen 6.11 und höher.

Der aktive E-Mail-Modus wird in der System-Property cmas-core-server, nimh.enabled gesetzt:

- **true** - NIMH-Modus
- **false** - Mule/ESB-Modus

Das Senden von E-Mails, d. h. die Konfiguration des SMTP-Servers, wird vom Modus für eingehende E-Mails nicht beeinflusst.

E-Mail-Konfiguration mit NIMH

NIMH, der ***New Incoming Mail Handler***, ist ein proprietäres Modul von ConSol CM. Die folgende Abbildung zeigt eine Übersicht über alle Komponenten.

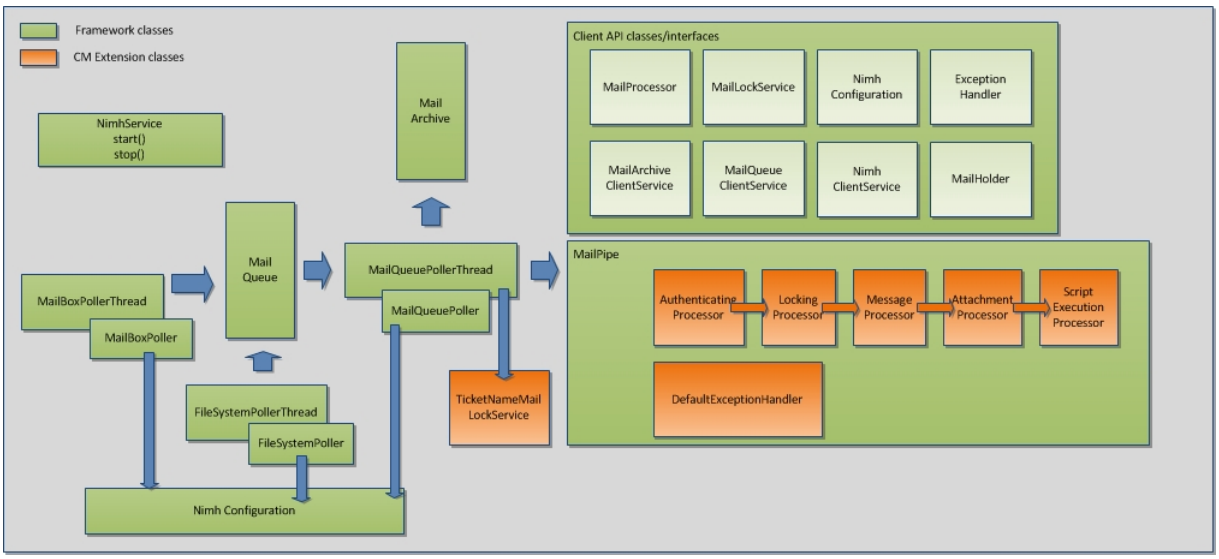


Abbildung 444: *NIMH-Komponenten*

Wenn NIMH aktiviert ist, gilt Folgendes:

- NIMH läuft als (einzelner) ConSol CM-Dienst, siehe Abschnitt CM-Services.
- E-Mails aus Postfächern auf einem Mail-Server werden von ConSol CM mit dem MailBoxPoller abgerufen.
- E-Mails können mit dem FileSystemPoller aus einem Datenverzeichnis abgerufen werden.
- Alle ConSol CM-E-Mails werden in der ConSol CM-Datenbank gespeichert (es wird nichts im Dateisystem gespeichert).
- Der MailQueuePoller ruft die E-Mails aus der Datenbank ab und leitet sie an das Kernsystem von ConSol CM weiter, wo die E-Mails durch die hintereinander geschalteten E-Mail-Skripte laufen.
- E-Mails, die nicht verarbeitet werden konnten, werden in einer eigenen Datenbanktabelle gespeichert und können erneut an das ConSol CM-System gesendet werden, siehe Abschnitt Verwaltung von E-Mail-Backups mit NIMH.
- Es werden NIMH-spezifische System-Properties verwendet (diese werden bei einem Update auf Version 6.9.4 und höher automatisch zur Systemkonfiguration hinzugefügt):
    - Allgemeine NIMH-Properties
    - Standardpostfach-Properties, die verwendet werden, wenn die Property nicht als postfachspezifische Property gesetzt ist
    - Postfachspezifische Properties für jedes Postfach, das abgerufen werden soll
    - FileSystemPoller-Properties
    - MailQueuePoller-Properties

Die folgende Abbildung zeigt ein Beispiel für NIMH-Properties. Eine detaillierte Beschreibung aller System-Properties für NIMH finden Sie in Liste der System-Properties nach Bereich.

cmas-core-server	nimh.enabled	true
cmas-nimh	filesystem.polling.threads.number	
cmas-nimh	filesystem.polling.threads.shutdown.timeout.seconds	
cmas-nimh	filesystem.polling.threads.watchdog.interval.seconds	
cmas-nimh	filesystem.task.enabled	
cmas-nimh	filesystem.task.interval.seconds	
cmas-nimh	filesystem.task.polling.folder	
cmas-nimh	filesystem.task.timeout.seconds	
cmas-nimh	filesystem.task.transaction.timeout.seconds	
cmas-nimh	mailbox.1.connection.host	
cmas-nimh	mailbox.1.connection.password	
cmas-nimh	mailbox.1.connection.port	
cmas-nimh	mailbox.1.connection.protocol	
cmas-nimh	mailbox.1.connection.username	
cmas-nimh	mailbox.1.task.enabled	
cmas-nimh	mailbox.2.connection.host	
cmas-nimh	mailbox.2.connection.password	
cmas-nimh	mailbox.2.connection.port	
cmas-nimh	mailbox.2.connection.protocol	
cmas-nimh	mailbox.2.connection.username	
cmas-nimh	mailbox.2.task.enabled	
cmas-nimh	mailbox.default.connection.host	
cmas-nimh	mailbox.default.connection.password	
cmas-nimh	mailbox.default.connection.port	
cmas-nimh	mailbox.default.connection.protocol	
cmas-nimh	mailbox.default.connection.username	
cmas-nimh	mailbox.default.session.mail.debug	
cmas-nimh	mailbox.default.session.mail.imap.connectiontimeout	
cmas-nimh	mailbox.default.session.mail.imap.timeout	
cmas-nimh	mailbox.default.session.mail.mime.address.strict	
cmas-nimh	mailbox.default.session.mail.pop3.connectiontimeout	
cmas-nimh	mailbox.default.session.mail.pop3.timeout	
cmas-nimh	mailbox.default.task.delete.read.messages	
cmas-nimh	mailbox.default.task.enabled	
cmas-nimh	mailbox.default.task.interval.seconds	
cmas-nimh	mailbox.default.task.max.message.size	
cmas-nimh	mailbox.default.task.max.messages.per.run	
cmas-nimh	mailbox.default.task.timeout.seconds	
cmas-nimh	mailbox.polling.threads.mail.log.enabled	
cmas-nimh	mailbox.polling.threads.number	
cmas-nimh	mailbox.polling.threads.shutdown.timeout.seconds	
cmas-nimh	mailbox.polling.threads.watchdog.interval.seconds	
cmas-nimh	queue.polling.threads.number	
cmas-nimh	queue.polling.threads.shutdown.timeout.seconds	
cmas-nimh	queue.polling.threads.watchdog.interval.seconds	
cmas-nimh	queue.task.error.pause.seconds	
cmas-nimh	queue.task.interval.seconds	
cmas-nimh	queue.task.max.retries	
cmas-nimh	queue.task.timeout.seconds	
cmas-nimh	queue.task.transaction.timeout.seconds	
cmas-nimh-extension	mail.attachments.validation.info.sender	
cmas-nimh-extension	mail.attachments.validation.info.subject	
cmas-nimh-extension	mail.db.archive	
cmas-nimh-extension	mail.error.from.address	
cmas-nimh-extension	mail.error.to.address	
cmas-nimh-extension	mail.on.error	

FileSystemPoller-Einstellungen

Kundenspezifisches Postfach Nr. 1

Kundenspezifisches Postfach Nr. 2

Standardwerte, die verwendet werden, wenn kein postfachspezifischer Wert gesetzt ist

Allgemeine NIMH-Einstellungen

MailQueuePoller-Einstellungen

Erweiterte NIMH-Einstellungen

Abbildung 445: *System-Properties für NIMH*

F.7.2 E-Mail-Backups

F.7.2.1 Einleitung

Eingehende E-Mails, die nicht verarbeitet werden konnten, werden in der CM-Datenbank gesichert.

Als Administrator können Sie versuchen, die E-Mails manuell erneut an das System zu senden. Die hier gespeicherten E-Mails können auch gelöscht werden, z. B. wenn es sich um Spam-E-Mails handelt.

WICHTIGE INFORMATION

Seit ConSol CM-Version 6.9.4 gibt es zwei Modi für eingehende E-Mails:

- **Mule/ESB** - Dieser Modus ist seit Version 6 in ConSol CM verfügbar. Er ist in CM-Versionen 6.11 und höher veraltet. Deshalb wird dieser Modus nicht mehr in den ConSol CM-Handbüchern behandelt. Wenn Sie ein älteres CM-System mit Mule/ESB betreiben, finden Sie die nötigen Informationen in den jeweiligen Handbüchern. Alle ConSol CM 6.11-Systeme müssen im NIMH-Modus betrieben werden. Informationen über den Wechsel von Mule/ESB zu NIMH finden Sie in den älteren Administratorhandbüchern.
- **NIMH** - *New Incoming Mail Handler*, seit Version 6.9.4 verfügbar. Dies ist der einzige Modus für eingehende E-Mails in ConSol CM-Versionen 6.11 und höher.

Der aktive E-Mail-Modus wird in der System-Property cmas-core-server, nimh.enabled gesetzt:

- **true** - NIMH-Modus
- **false** - Mule/ESB-Modus

Das Senden von E-Mails, d. h. die Konfiguration des SMTP-Servers, wird vom Modus für eingehende E-Mails nicht beeinflusst.

F.7.2.2 E-Mail-Backups im Admin Tool

Alle E-Mails, die nicht verarbeitet werden konnten, sind im Navigationselement *E-Mail-Backups* der Navigationsgruppe *E-Mail* aufgeführt.

Normalerweise hat es einen der folgenden Gründe, wenn eine E-Mail nicht verarbeitet werden kann:

- Eines der E-Mail-Skripte (siehe Abschnitt Skripte des Typs E-Mail) hat einen Fehler und daher kann die vom Mail-Server abgerufene E-Mail in CM nicht weiter verarbeitet werden.
- Das Attachment der E-Mail ist zu groß. Für NIMH wird die Größenbeschränkung durch die System-Property cmas-nimh, mailbox.default.task.max.message.size oder durch einen postfachspezifischen Wert definiert, z. B. `cmas-nimh`, `mailbox.1.task.max.message.size`.

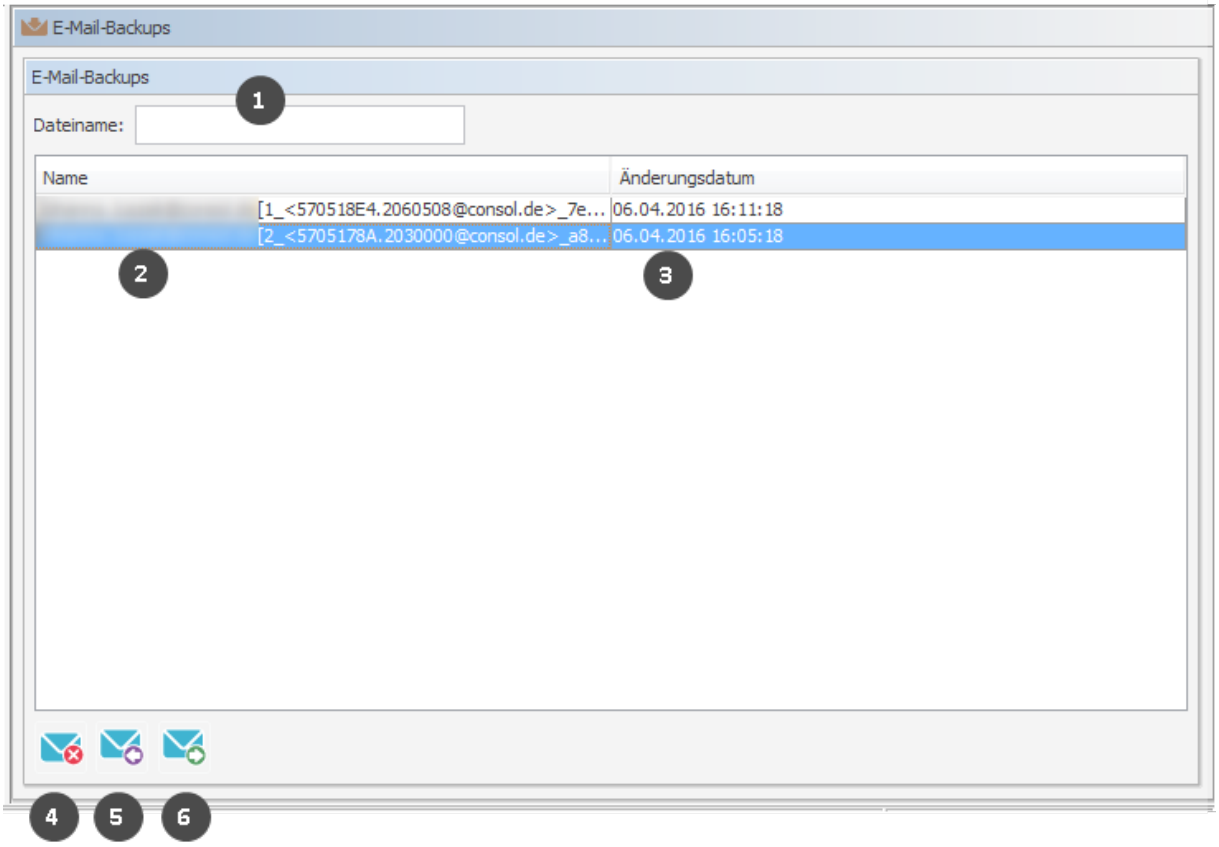


Abbildung 446: *ConSol CM Admin Tool - E-Mail, E-Mail-Backups*

Das Listenfeld mit den E-Mails, die nicht verarbeitet konnten, enthält folgende Elemente:

- **Dateiname (1)**
  Dieses Feld enthält einen Filter. Wenn Sie den Namen oder einen Teil des Namens der E-Mail-Dateien eingeben, werden nur die Dateien mit passendem Namen in der Liste angezeigt.
- **Name (2)**
  Der Name der E-Mail-Datei (normalerweise mit der Erweiterung `.eml`).
- **Änderungsdatum (3)**
  Das letzte Datum, an dem die Datei geändert wurde. Dies ist normalerweise das Datum, an dem die E-Mail auf dem ConSol CM-Server gespeichert wurde.

In einem fehlerfrei laufenden ConSol CM-System sollte die Liste der E-Mails, die nicht verarbeitet werden konnten, leer sein. Falls dort E-Mails aufgeführt sind, ist ein Fehler bei der Verarbeitung der eingehenden E-Mails aufgetreten. Eine detaillierte Beschreibung Verarbeitungs-Pipeline finden Sie im Abschnitt Skripte des Typs E-Mail.

Sie können mit den E-Mail-Backups folgende Aktionen ausführen:

- **Löschen (4)**
  Um eine E-Mail zu löschen, wählen Sie sie in der Liste aus und klicken Sie auf den Button *Löschen*. Beachten Sie, dass die Informationen dabei verloren gehen! Sie werden nicht

gespeichert oder auf einem anderen Weg an ConSol CM übertragen!

- **Erneut senden (importieren) (5)**
  Wählen Sie die E-Mail in der Liste aus und klicken Sie auf den Button *Erneut senden*, um die E-Mail erneut an die Verarbeitungs-Pipeline zu senden (importieren), z. B. wenn ein Skript vorher nicht richtig funktioniert hat und das Problem behoben wurde.
- **Ins Dateisystem exportieren (6)**
  Sie können eine E-Mail als .eml-Datei ins Dateisystem des Rechners exportieren, auf dem das Admin Tool ausgeführt wird. z. B. für weitere Analysen einer fehlerhaften E-Mail.

F.7.2.3 Verwaltung von E-Mail-Backups mit NIMH

Eingehende E-Mails, die nicht verarbeitet werden können, werden in der ConSol CM-Datenbank (als .eml-Dateien) in folgender Tabelle gespeichert:

```
cmas_nimh_archived_mail
```

Das Systemverhalten von E-Mail-Backups bei NIMH wird von der System-Property cmas-nimh-extension, mail.db.archive gesteuert.

Wenn die Property auf „true“ gesetzt ist, werden alle eingehenden E-Mails in der Datenbank archiviert. Wenn die Property auf „false“ gesetzt ist, werden die E-Mails, die verarbeitet werden können, nach der Verarbeitung gelöscht. Die E-Mails, die nicht verarbeitet werden können, werden so lange in der Datenbank gespeichert, bis sie entweder verarbeitet oder gelöscht werden (dies kann entweder durch erneutes Verarbeiten oder durch Löschen erreicht werden, siehe obigen Abschnitt). Der E-Mail-Eintrag in der Datenbank wird automatisch nach der erfolgreichen Verarbeitung gelöscht.

F.7.3 E-Mail-Verschlüsselung

F.7.3.1 Einleitung

Aufgrund von Sicherheitsrichtlinien ist es möglicherweise erforderlich, den E-Mail-Verkehr (einschließlich der E-Mails, die das ConSol CM-System sendet und erhält) mit Standard-S/MIME-Verschlüsselung zu verschlüsseln. Falls Sie mit dem Thema nicht vertraut sind und sich gerne informieren möchten, könnten Sie z. B. den Wikipedia-Artikel über Public-Key-Verschlüsselungsverfahren lesen.

Damit ConSol CM verschlüsselte E-Mails verwenden kann, müssen Sie zuerst die E-Mail-Verschlüsselung im System aktivieren:

1. **Obligatorisch:**
   Setzen Sie die System-Property cmas-core-server, mail.encryption auf „true“. Sie hat standardmäßig den Wert „false“. Dies ist die grundlegende Konfiguration, um die E-Mail-Verschlüsselung im gesamten System zu aktivieren.
2. **Optional:**
   Setzen Sie das Attribut der Seitenanpassung mailEncryptionAvailable auf „true“. Damit wird im Web Client eine Option aktiviert, mit der man aussuchen kann, ob die E-Mail verschlüsselt werden soll.

Allgemeine Erklärung über die E-Mail-Verschlüsselung in ConSol CM

Es gibt zwei Arten von Zertifikaten:

- **Server-Zertifikate, hauptsächlich genutzt für eingehende E-Mails**
  Ein Zertifikat mit dem Public Key und dem Private Key der für den Empfang von E-Mails konfigurierten E-Mail-Adresse kann aus einer PKCS12-Datei (.p12 bzw. .pfx) manuell ins CM-System importiert werden. Wenn das Zertifikat durch ein Passwort geschützt ist, muss der Administrator dieses beim Importieren des Zertifikats eingeben.
- **Client-Zertifikate, genutzt für ausgehende E-Mails**
  Für ausgehende E-Mails werden die Zertifikate (Public Keys) der jeweiligen Empfänger benötigt. Diese Zertifikate können auf zwei Wegen ins CM-System importiert werden
  - **Manuell**
    Durch Auswählen (Hochladen) der X.509-Datei (.cer oder .crt).
  - **Automatisch**
    Aus dem LDAP-Repository, in dem es im richtigen Format gespeichert ist (d. h. dem gleichen Format wie für den Datei-Import). Dieser automatische Abruf des angeforderten Zertifikats kann „on-the-fly“ beim Senden der E-Mail durchgeführt werden.

⚠ Die hier beschriebenen Zertifikate werden nur für die Verschlüsselung von E-Mails verwendet und gelten **nicht** für den Zugang von ConSol CM (als E-Mail-Client) zum E-Mail-Server! Dieser muss mithilfe von Zertifikaten verwaltet werden, die im Security Store des Application Servers gespeichert sind.

Die folgende Abbildung zeigt das Grundprinzip der E-Mail Verschlüsselung für ein- und ausgehende E-Mails in ConSol CM.

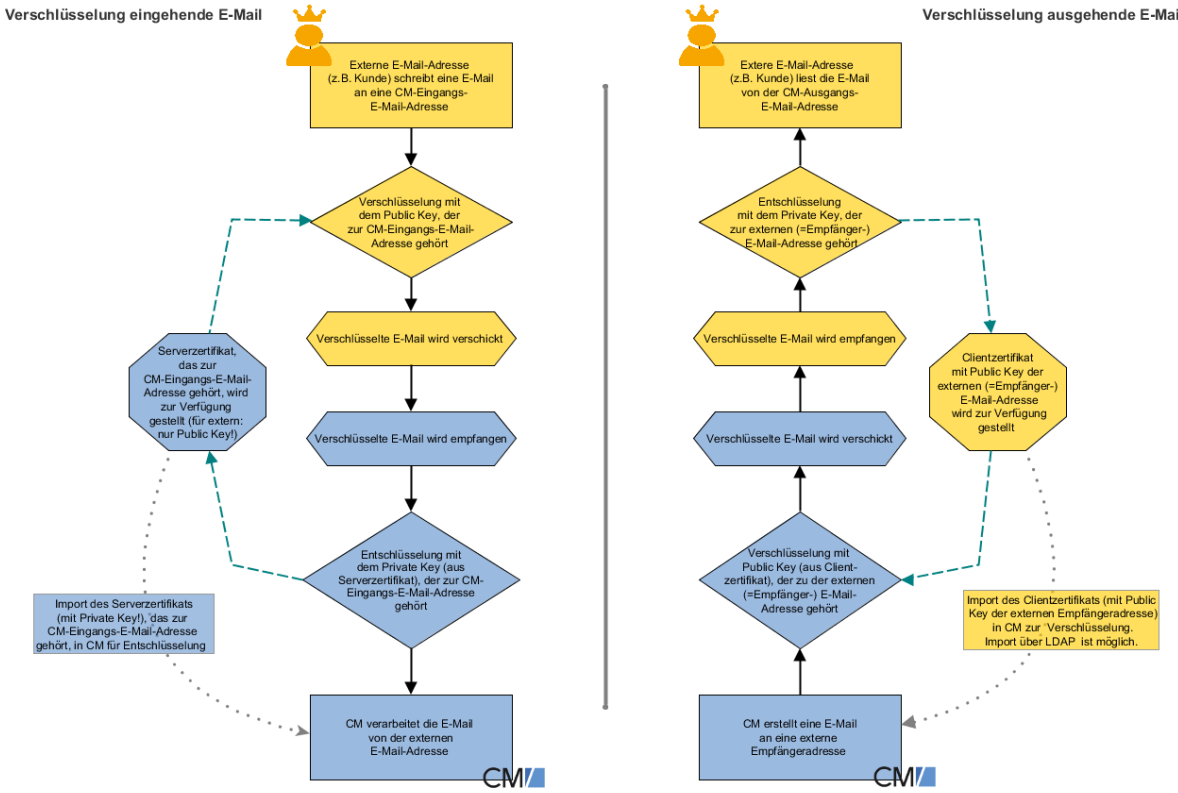


Abbildung 447: *Grundprinzip der E-Mail-Verschlüsselung in ConSol CM*

Anforderungen

- Das Client-Zertifikat muss die E-Mail-Adresse des Kunden (Empfänger der E-Mail) entweder im Attribut `SubjectDN` (`E=` oder `EMAILADDRESS=`) oder im Element `X509v3 Subject Alternative Name` des Abschnitts `Extensions` enthalten.
- *Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files* muss auf dem Server und auf dem Rechner, auf dem das Admin Tool gestartet wird, installiert sein. Dies ist erforderlich, damit das Admin Tool Zertifikate importieren kann.
- Es werden in ***X.509 Base64*** codierte Zertifikate unterstützt.

Importieren von Zertifikaten aus LDAP (verfügbar für Client-Zertifikate)

Wenn LDAP konfiguriert ist, sucht ConSol CM das Client-Zertifikat für den angeforderten Kontakt im LDAP-Repository. Dies läuft folgendermaßen ab:

1. Jemand versucht, eine verschlüsselte E-Mail zu senden.
2. Der Verschlüsselungsdienst sucht nach dem Client-Zertifikat (mit dem Public Key) des Empfängers.

3. Wenn ein Client-Zertifikat gefunden wird, wird die E-Mail mit dem Public Key des Empfängers verschlüsselt und gesendet.
4. Wenn kein Client-Zertifikat im Admin Tool gefunden wird oder das Client-Zertifikat abgelaufen ist, wird es im LDAP-Repository gesucht.
5. Wenn es gefunden wird, wird es nach ConSol CM importiert, und die E-Mail wird verschlüsselt und gesendet.
6. Wenn es nicht gefunden wird, wird die E-Mail unverschlüsselt gesendet.

Die folgenden Konfigurations-Properties müssen gesetzt werden, damit über LDAP nach Zertifikaten gesucht werden kann:

- ldap.certificate.basedn
- ldap.certificate.searchattr
- ldap.certificate.content.attribute

Details dazu finden Sie im Abschnitt *LDAP-Zertifikatparameter* in den System-Properties.

F.7.3.2 Zertifikatsverwaltung mit dem Admin Tool

Im Admin Tool wird die CM-Umgebung in den Navigationselementen *Server-Zertifikate* und *Client-Zertifikate* der Navigationsgruppe *E-Mail* für die E-Mail-Verschlüsselung konfiguriert.

Server-Zertifikate

Server-Zertifikate werden verwendet, um **eingehende** E-Mail-Nachrichten zu entschlüsseln. In Ausnahmefällen werden Server-Zertifikate auch genutzt, um ausgehende E-Mails zu verschlüsseln: Wenn eine E-Mail an eine der als Eingangs-E-Mails konfigurierten E-Mail-Adressen gesendet wird, wird der Public Key des Server-Zertifikates genutzt, um diese E-Mail zu verschlüsseln. Server-Zertifikate enthalten jeweils den Public Key und den Private Key für die angegebene E-Mail-Adresse. Wenn Sie ein Postfach für eingehende E-Mails definieren (siehe obiges Kapitel) müssen Sie ein Server-Zertifikat für diese E-Mail-Adresse (oder für alle E-Mail-Adressen dieses Postfachs) hochladen, um verschlüsselte Nachrichten erhalten zu können (da das Server-Zertifikat den entsprechenden Private Key enthält). Wenn Sie mehrere Eingangspostfächer haben, müssen Sie entweder ein Server-Zertifikat für jedes der Postfächer oder ein Zertifikat mit allen erforderlichen E-Mail-Adressen hochladen.

Wenn Sie das Navigationselement *Server-Zertifikate* öffnen, wird eine Liste aller vorhandenen Server-Zertifikate angezeigt. Um ein neues Server-Zertifikat hinzuzufügen, klicken Sie auf den Button *Hinzufügen* und suchen Sie das gewünschte Zertifikat im Dateibrowser. Das Zertifikat wird vor dem Import validiert. Wenn Inkompatibilitäten festgestellt werden, wird eine Fehlermeldung angezeigt und das Zertifikat wird nicht importiert.

Folgende Formate werden für Server-Zertifikate unterstützt:

- Archivdatei `PKCS #12`, die das Zertifikat (öffentlich) und den privaten Schlüssel (durch ein Passwort geschützt) enthält.
  Die für PKCS #12-Dateien unterstützten Dateinamenerweiterungen sind:
  - .p12
  - .pfx

Client-Zertifikate

Ein Client-Zertifikat enthält nur den Public Key eines externen Empfängers (z. B. eines Kunden). Es ermöglicht die Verschlüsselung von Nachrichten, die an diesen Benutzer gesendet werden, wird also für **ausgehende** E-Mails verwendet.

Wenn Sie das Navigationselement *Client-Zertifikate* öffnen, wird eine Liste aller vorhandenen Client-Zertifikate angezeigt. Um ein neues Client-Zertifikat hinzuzufügen, klicken Sie auf den Button *Hinzufügen* und suchen Sie das gewünschte Zertifikat im Dateibrowser. Das Zertifikat wird vor dem Import validiert. Wenn Inkompatibilitäten festgestellt werden, wird eine Fehlermeldung angezeigt und das Zertifikat wird nicht importiert.

Folgende Formate werden für Client-Zertifikate unterstützt:

- Standardformat `X509`.
  Die für X.509-Zertifikate unterstützten Dateinamenerweiterungen sind:
  - .cer
  - .crt
  - .der
  - .pem

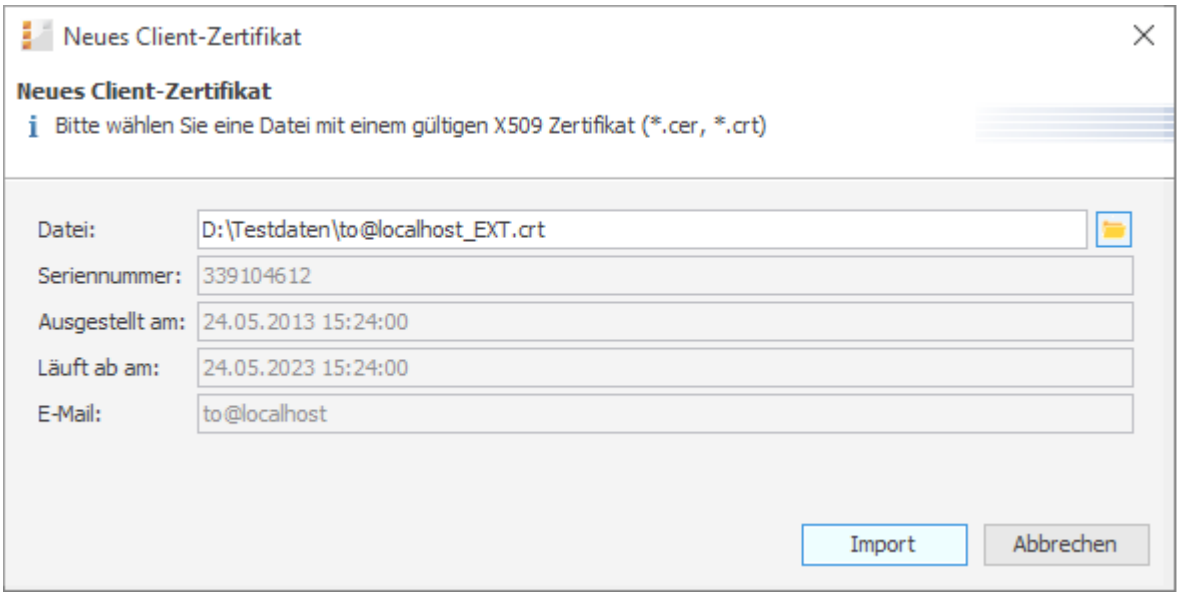


Abbildung 448: *ConSol CM Admin Tool - E-Mail, Client-Zertifikate: Pop-up-Fenster zum Hinzufügen eines Client-Zertifikats*

Im Folgenden sind einige beispielhafte Anwendungsfälle beschrieben:

1. Ein Bearbeiter arbeitet im ConSol CM Web Client und schreibt mit dem Ticket-E-Mail-Editor eine verschlüsselte E-Mail. Wenn er auf den Button *E-Mail versenden* klickt, sucht das ConSol CM-System in der Liste der E-Mail-Adressen unter *Client-Zertifikate* nach der Empfängeradresse und verwendet den Public Key des Empfängers zur Verschlüsselung der ausgehenden E-Mail.

Wenn ConSol CM kein passendes Zertifikat findet (d. h. die E-Mail-Adresse steht nicht in der Zertifikatsliste), wird das Zertifikat für die E-Mail aus LDAP geladen. Wenn auch dort kein kompatibles Zertifikat gefunden wird, wird die E-Mail unverschlüsselt gesendet.
Wenn einer der Empfänger mit einem der Postfächer für eingehende E-Mails übereinstimmt, wird auch das Server-Zertifikat (Public Key) zum Verschlüsseln der Nachricht verwendet.

2. ConSol CM erhält eine E-Mail und überprüft die To-Adresse. Wenn diese in der Liste unter *Server-Zertifikate* gefunden wird, verwendet ConSol CM den im Zertifikat angegebenen Private Key zur Entschlüsselung der Nachricht, um entweder ein neues Ticket zu erstellen oder die Nachricht an ein vorhandenes Ticket anzuhängen.

Senden von verschlüsselten E-Mails

Auswählen, ob E-Mails aus dem Web Client verschlüsselt gesendet werden sollen

Wenn das Attribut der Seitenanpassung mailEncryptionAvailable auf „true“ gesetzt ist, ist im Ticket-E-Mail-Editor des Web Clients eine Checkbox *Verschlüsselt senden* verfügbar. Der Benutzer kann damit auswählen, ob die E-Mail verschlüsselt gesendet werden soll.

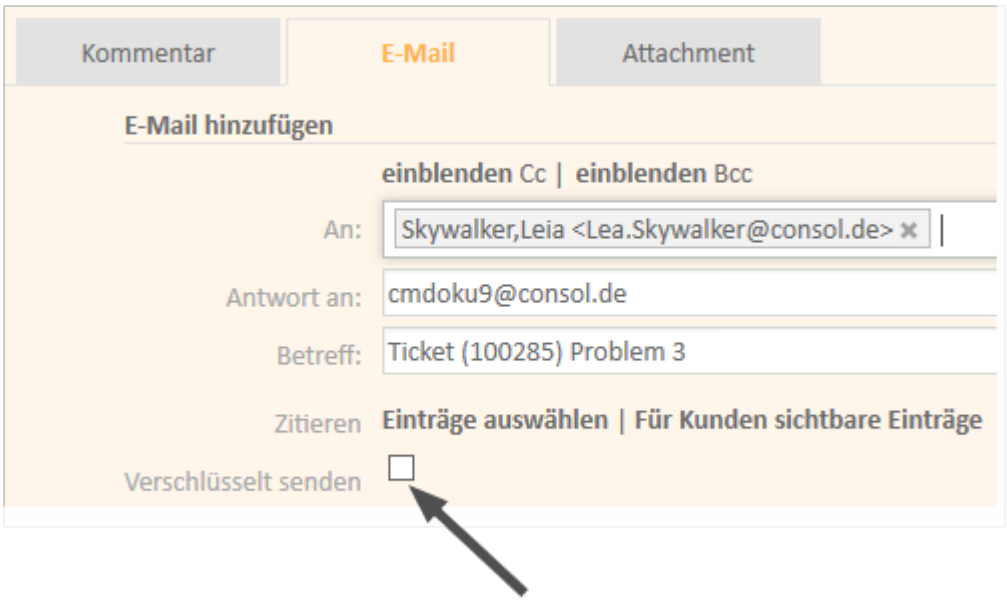


Abbildung 449: *ConSol CM Web Client - Verschlüsselte E-Mail senden*

Senden einer verschlüsselten E-Mail aus dem Workflow

Mit der Methode `enableEncryption()` kann eine verschlüsselte E-Mail gesendet werden. Eine detaillierte Beschreibung dazu finden Sie im *ConSol CM Process Designer Handbuch*.

Standardmäßiges Senden von verschlüsselten E-Mails

Wenn die System-Property cmas-core-server, mail.encryption auf „true“ gesetzt ist, werden alle ausgehenden E-Mails aus dem Workflow und Web Client standardmäßig verschlüsselt.

Wenn Benutzer eine bestimmte E-Mail unverschlüsselt senden möchten, können sie im Web Client die Checkbox *Verschlüsselt senden* deaktivieren. Für aus dem Workflow gesendete E-Mails kann die Methode `disableEncryption()` verwendet werden.

F.8 Verwaltung von Admin-Tool-Skripten und -Templates

In diesem Kapitel erfahren Sie, wie man mit den Skripten und Templates arbeitet, die im Admin Tool gespeichert sind und dort verwaltet werden.

Skripte werden in verschiedenen Kontexten in ConSol CM verwendet, insbesondere in den Workflows im Process Designer. Eine detaillierte Beschreibung dazu finden Sie im *ConSol CM Process Designer Handbuch*. Im Bereich *Skripte* des Admin Tools sind ebenfalls verschiedenartige Skripte gespeichert. Dies ist im Abschnitt Admin-Tool-Skripte erklärt.

Templates bzw. Vorlagen sind ebenfalls an verschiedenen Orten gespeichert:

- Zugang über den Web Client:
  - Im Textvorlagen-Manager (E-Mail-Vorlagen)
  - Im Dokumentvorlagen-Manager (Microsoft Word- oder OpenOffice-Vorlagen in CM/Doc)
- Zugang über das Admin Tool:
  - Im Bereich *Skripte und Templates*
    - Einige spezielle E-Mail-Templates
    - Templates für Kundendaten
    - Templates für Ressourcendaten
    - Template zum Zurücksetzen des Passworts
    - CM/Phone-Templates
    - <weitere Templates je nach eigener Anpassung>

Eine Beschreibung der Arbeit mit E-Mail-Vorlagen im Textvorlagen-Manager und der Konfiguration von CM/Doc finden Sie im Abschnitt Arbeiten mit Textvorlagen. Eine Beschreibung der Templates im Admin Tool finden Sie im Abschnitt Admin-Tool-Templates.

F.8.1 Admin-Tool-Skripte

F.8.1.1 Einführung in die Skripte im Admin Tool

Skripte werden im Bereich *Skripte und Templates* des Admin Tools gespeichert. Sie werden in Groovy geschrieben und sollten nur von erfahrenen ConSol CM-Consultants und -Administratoren editiert werden.

Öffnen Sie im Admin Tool das Navigationselement *Skripte und Templates* in der Navigationsgruppe *System*, um mit den Skripten zu arbeiten. Es wird der Tab *Skripte* angezeigt.

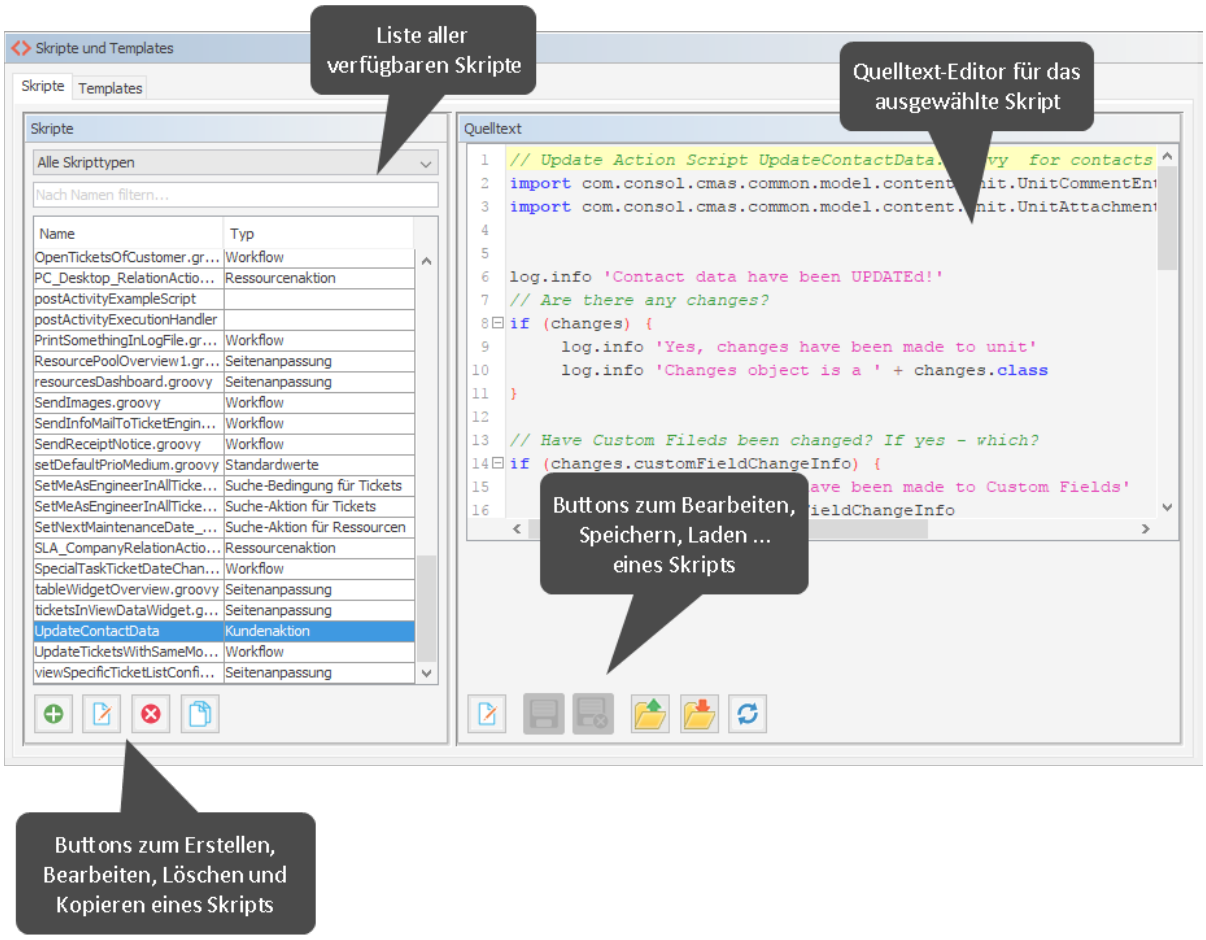


Abbildung 450: *ConSol CM Admin Tool - System, Skripte und Templates*

Auf der linken Seite sehen Sie eine Liste aller Skripte. Die Liste kann über das Drop-down-Menü folgendermaßen gefiltert werden:

- Im Drop-down-Menü kann der Skripttyp ausgewählt werden.
- Im Eingabefeld kann ein Textfilter für den Skriptnamen eingegeben werden.

Für jedes Skript müssen zwei Parameter gesetzt werden:

- **Name**
  Dies ist der Name, über den das Skript referenziert wird, z. B. im Workflow oder in anderen Objekten wie Queues.
- **Typ**
  Der Typ des Skripts. Es muss einer der folgenden Skripttypen ausgewählt werden:
  - **Abhängige sortierte Listen**
    Mit Skripten dieses Typs werden *abhängige sortierte Listen* definiert, Strukturen, mit denen hierarchische Listen abgebildet werden. Details dazu finden Sie im Abschnitt Skripte des Typs Abhängige sortierte Liste.
  - **Kundenaktion**
    Dieses Skript wird ausgeführt, wenn eine Kundenaktion stattfindet, siehe Abschnitt Action Framework - Kundenaktionen.
  - **Kundenbedingung**
    Dieses Skript wird ausgeführt, um zu prüfen, ob eine Kundenaktion im Web Client angeboten werden soll, siehe Abschnitt Action Framework - Kundenaktionen.
  - **Duplizieren**
    Dieses Skript wird ausgeführt, wenn die Option *Duplizieren* in einem Ticket ausgewählt wird. Es muss der Queue zugewiesen sein. Details dazu finden Sie im Abschnitt Skripte des Typs Duplizieren.
  - **E-Mail**
    Mit Skripten dieses Typs werden eingehende und ausgehende E-Mails verwaltet. Details dazu finden Sie im Abschnitt Skripte des Typs E-Mail.
  - **Feld-Visualisierung**
    Skripte dieses Typs werden zur Konfiguration der Visualisierung von Datenfeldern verwendet. Detaillierte Informationen finden Sie im Abschnitt Skripte des Typs Feld-Visualisierung.
  - **Integration**
    Skripte dieses Typs werden verwendet, um Services für die CM-Webhooks-Schnittstelle zu implementieren. Siehe Abschnitt Webhooks für Details.
  - **Kalenderintegration**
    Skripte dieses Typs liefern die Verbindungsdaten für die Integration von Microsoft Exchange-Kalendern. Siehe Abschnitt Integration von Microsoft Exchange-Kalendern.
  - **PostActivityExecutionScript** (kein bestimmter Typ angegeben)
    Das (optionale) Skript, das nach jeder manuellen Workflow-Aktivität ausgeführt wird. Details dazu finden Sie im Abschnitt PostActivityExecutionScript.
  - **Relationsgraph**
    Skripte dieses Typs werden zum Implementieren der Graphenanzeige von Relationen in den Standardbereichen oder dem Expertenbereich von Ticket-, Kunden- (Firma/Kontakt) und Ressourcenseiten verwendet. Eine detaillierte Erklärung finden Sie im Abschnitt Konfiguration der Graphendarstellung von Relationen.

- **Ressourcen-Aktion**
  Dieses Skript wird ausgeführt, wenn eine Ressourcenaktion stattfindet, siehe Abschnitt CM/Resource Pool - Ressourcenaktionen.
- **Ressourcen-Bedingung**
  Dieses Skript wird ausgeführt, um zu prüfen, ob eine Ressourcenaktion im Web Client angeboten werden soll. Eine detaillierte Erklärung dazu finden Sie im Abschnitt CM/Resource Pool - Ressourcenaktionen.
- **Seitenanpassung**
  Skripte dieses Typs werden in den Einstellungen der Seitenanpassung referenziert. Details dazu finden Sie in den Abschnitten Seitenanpassung für das Web Client Dashboard und CM/Resource Pool - Das Ressourcenpool-Dashboard. Der Skripttyp *Widget* kann statt dieses Typs verwendet werden. Die beiden Typen funktionieren gleich. Wenn Sie den Skripttyp *Seitenanpassung* für Skripte verwenden, die in den Bereichen und Typen der Seitenanpassung referenziert werden, und den Skripttyp *Widget* für Skripte, in denen Dashboard-Widgets implementiert sind, können Sie in der Skriptliste (durch Sortieren nach Typ) einfach zwischen den Skripten unterscheiden.
- **Standardwerte**
  Mit Skripten dieses Typs werden Standardwerte definiert, d. h. Werte, mit denen Datenfelder (vor)befüllt werden, wenn ein neues Ticket erstellt wird. Details dazu finden Sie im Abschnitt Skripte des Typs Standardwerte.
- **Suche-Aktion für Kunden**
  Dieser Skripttyp gehört zum Action Framework, genauer gesagt zu den Suche-Aktionen, und wird im Abschnitt Action Framework - Suche-Aktionen beschrieben.
- **Suche-Bedingung für Kunden**
  Dieser Skripttyp gehört zum Action Framework, genauer gesagt zu den Suche-Aktionen, und wird im Abschnitt Action Framework - Suche-Aktionen beschrieben.
- **Suche-Aktion für Ressourcen**
  Dieser Skripttyp gehört zum Action Framework, genauer gesagt zu den Suche-Aktionen, und wird im Abschnitt Action Framework - Suche-Aktionen beschrieben.
- **Suche-Bedingung für Ressourcen**
  Dieser Skripttyp gehört zum Action Framework, genauer gesagt zu den Suche-Aktionen, und wird im Abschnitt Action Framework - Suche-Aktionen beschrieben.
- **Suche-Aktion für Tickets**
  Dieser Skripttyp gehört zum Action Framework, genauer gesagt zu den Suche-Aktionen, und wird im Abschnitt Action Framework - Suche-Aktionen beschrieben.
- **Suche-Bedingung für Tickets**
  Dieser Skripttyp gehört zum Action Framework, genauer gesagt zu den Suche-Aktionen, und wird im Abschnitt Action Framework - Suche-Aktionen beschrieben.
- **Task**
  Skripte dieses Typs werden vom TEF (Task Execution Framework) verwendet, siehe Abschnitt Das Task Execution Framework (TEF).

- **Text-Autovervollständigung**
  Mit diesem Skripttyp werden Autocomplete-Listen mit Skript implementiert, siehe Abschnitt Skriptbasierte Autocomplete-Listen.
- **Widget**
  Dieses Skripttyp wird für Skripte verwendet, in denen Dashboards implementiert sind. Dabei kann es sich um das Dashboard auf der Hauptseite (siehe Abschnitt Seitenanpassung für das Web Client Dashboard) oder das Ressourcenpool-Dashboard (siehe CM/Resource Pool - Das Ressourcenpool-Dashboard) handeln. Der Skripttyp *Seitenanpassung*kann statt dieses Typs verwendet werden. Die beiden Typen funktionieren gleich. Wenn Sie den Skripttyp *Seitenanpassung* für Skripte verwenden, die in den Bereichen und Typen der Seitenanpassung referenziert werden, und den Skripttyp *Widget* für Skripte, in denen Dashboard-Widgets implementiert sind, können Sie in der Skriptliste (durch Sortieren nach Typ) einfach zwischen den Skripten unterscheiden.
- **Workflow**
  Skripte dieses Typs werden im Workflow referenziert. Details dazu finden Sie im Abschnitt Skripte des Typs Workflow.

Die Buttons unter der Liste stellen die Standardfunktionen des Admin Tools bereit:

- Skript hinzufügen
- Skript editieren
- Skript löschen
- Skript kopieren

Auf der rechten Seite sehen Sie den *Quelltext-Editor*. Hier wird das in der Liste auf der linken Seite ausgewählte Skript angezeigt. Wenn Sie sich im Editiermodus befinden, können Sie hier den Quelltext des Skripts schreiben.

F.8.1.2 Der Quelltext-Editor

Der Quelltext-Editor bietet einen Editierbereich mit:

- Syntaxhervorhebung
- Code-Vervollständigung (drücken Sie dazu STRG + Leertaste, es werden die Klassen im lokal ausgeführten Tool, d. h. im CLASSPATH, angeboten, nicht alle Klassen und Methoden in ConSol CM).
- Code-Validierung (siehe Anzeige im Feld *Ergebnis des Kompilierens*; da die Validierung auf der Server-Seite erfolgt, werden alle Klassen und Methoden von ConSol CM geprüft).
  - Das Intervall zwischen zwei Code-Prüfungen kann über die CM-System-Property cmas-app-admin-tool, script.validation.interval.seconds eingestellt werden.
  - Die Code-Validierung kann für eine Session mithilfe der Checkbox *Validierung deaktivieren* unter dem Editierbereich deaktiviert werden.

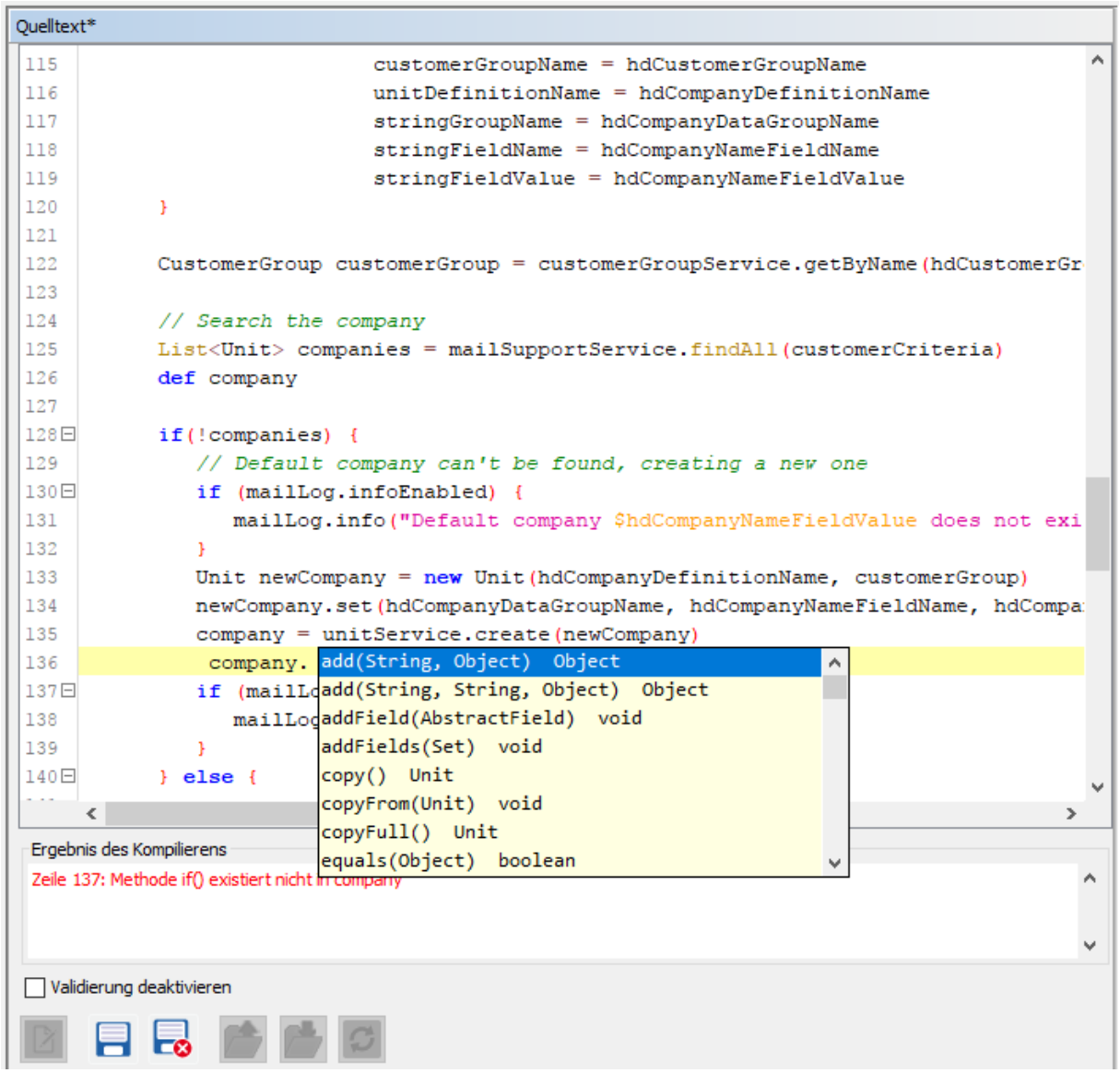


Abbildung 451: *ConSol CM Admin Tool - System, Skripte und Templates: Quelltext-Editor*

Im unteren Bereich des Quelltext-Editors finden Sie folgende Buttons:

- **Bearbeiten**
  Klicken Sie auf diesen Button, um im Quelltext -Editor in den Editiermodus zu wechseln. Wenn Sie im Admin Tool das Navigationselement *Skripte und Templates* öffnen, befinden sich alle Skripte im schreibgeschützten Modus, um zu verhindern, dass ein Administrator versehentlich etwas verändert.
- **Editiermodus verlassen und Änderungen speichern**
  Speichert das Skript und verlässt den Editiermodus, d. h. wechselt zurück in den schreibgeschützten Modus.

- **Editiermodus verlassen, ohne die Änderungen zu speichern**
  Wechselt zurück in den schreibgeschützten Modus, ohne die Änderungen zu speichern, die Sie am Quelltext vorgenommen haben.
- **Skript aus Datei öffnen**
  Mit dieser Option wird ein Dateibrowser geöffnet. Es sind zwei Fälle möglich:
  - Der Name der ausgewählten Datei und der Name des Skripts sind identisch.
    In diesem Fall wird der Code des Skripts (im Quelltext-Editor) komplett ersetzt.
  - Der Name der ausgewählten Datei und der Name des Skripts sind unterschiedlich.
    In diesem Fall wird ein neues Skript mit dem Namen der ausgewählten Datei erstellt. Das neue Skript wird im Editiermodus im Quelltext-Editor angezeigt.
- **Skript in Datei speichern**
  Hier können Sie den Text des Skripts als reine Textdatei im Dateisystem auf dem Rechner speichern, auf dem das Admin Tool ausgeführt wird.
- **Skript aus Datei aktualisieren**
  Mit dieser Option wird ein Dateibrowser geöffnet. Der Code im Quelltext-Editor wird durch den Text in der Datei ersetzt, unabhängig davon, ob Skript und Datei den gleichen Namen haben oder nicht.

F.8.1.3 Skripttypen

Im folgenden Abschnitt werden die möglichen Skripttypen erklärt. Die Beispiele sollen Ihnen einen Eindruck über die Verwendungsmöglichkeiten von Skripten vermitteln.

In diesem Abschnitt sind folgende Skripttypen beschrieben. Eine vollständige Übersicht über alle Skripttypen mit Links zu der entsprechenden Abschnitten finden Sie in der obigen Liste.

- Skriptbasierte Autocomplete-Listen
- Skripte des Typs Abhängige sortierte Liste
- Skripte des Typs Duplizieren
- Skripte des Typs E-Mail
- Skripte des Typs Feld-Visualisierung
- Skripte des Typs Standardwerte
- Skripte des Typs Workflow
- PostActivityExecutionScript

F.8.1.4 Skriptbasierte Autocomplete-Listen

Beachten Sie, dass die Implementierung von skriptbasierten Autocomplete-Listen zwischen ConSol CM-Version 6.11.1.0 und Version 6.11.1.1 geändert wurde. Lesen Sie für Version 6.11.1.0 bitte das *ConSol CM Administratorhandbuch* dieser Version. Der aktuelle Abschnitt behandelt nur skriptbasierte Autocomplete-Listen in CM-Version 6.11.1.1!

Einleitung

Skriptbasierte Autocomplete-Listen können für Drop-down-Menüs mit dynamisch erzeugtem Inhalt verwendet werden. Die folgenden Beispiele vermitteln Ihnen einen ersten Eindruck davon.

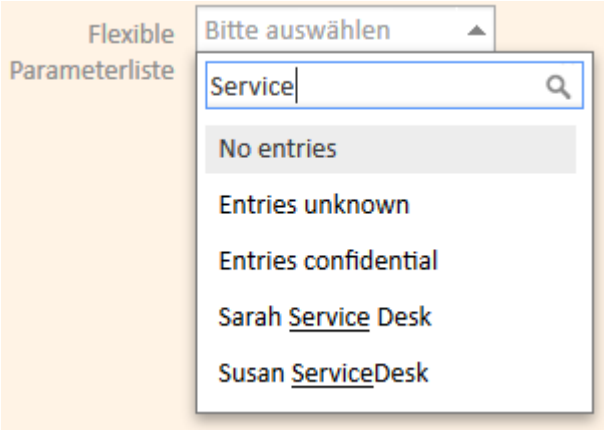


Abbildung 452: *ConSol CM Web Client - Skriptbasierte Autocomplete-Liste, die statische Werte und das Ergebnis einer Suche nach Bearbeitern enthält*

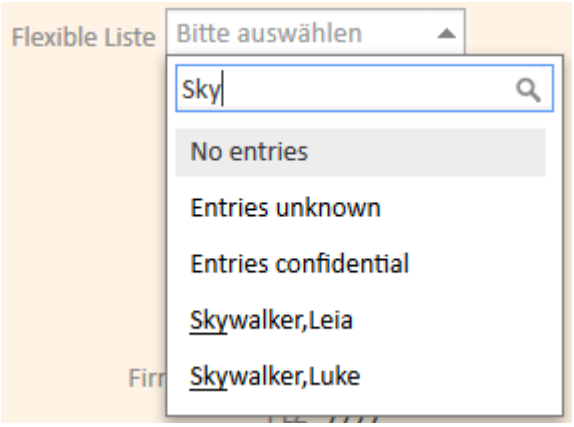


Abbildung 453: *ConSol CM Web Client - Skriptbasierte Autocomplete-Liste, die statische Werte und das Ergebnis einer Suche nach Kunden enthält*

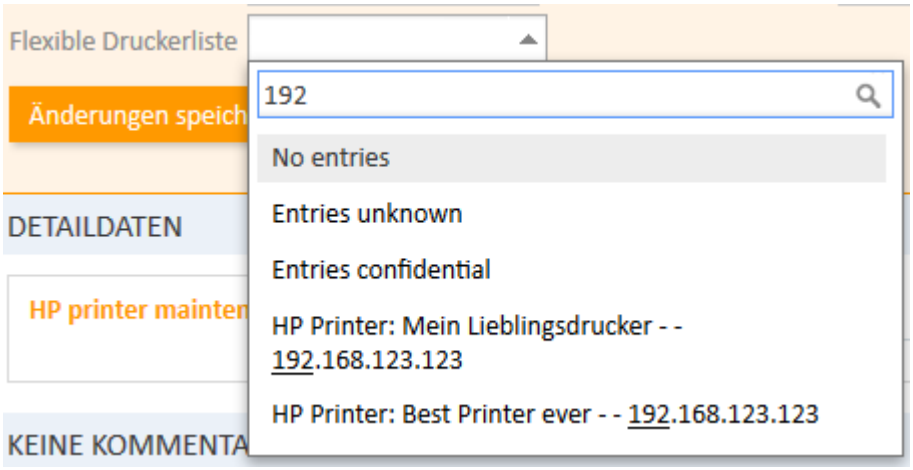


Abbildung 454: *ConSol CM Web Client - Skriptbasierte Autocomplete-Liste, die statische Werte und das Ergebnis einer Suche nach Ressourcen enthält*

Diese Art von Liste kann sehr hilfreich sein, um dem Bearbeiter statt einer unübersichtlichen langen Liste ein Teilmenge der Liste anzuzeigen. In den obigen Beispielen sind skriptbasierte Autocomplete-Listen für folgende Anwendungsfälle implementiert:

1. In einem Ticket soll ein Bearbeiter außerhalb des Standardfeldes *Bearbeiter* (im oberen Bereich des Tickets) gesetzt werden, z. B. um eine Informations-E-Mail an diesen Bearbeiter zu senden. Ein anderer Anwendungsfall für eine Bearbeiterliste wäre ein ACF, in dem der nächste Bearbeiter in einer Prozesskette ausgewählt werden kann.
2. Auf einer Kundenseite soll der verantwortliche Bearbeiter gesetzt werden.
3. Auf einer Kundenseite sollen die Ressourcen/Assets, die am Standort des Kunden in Betrieb sind, erfasst werden. Eine ähnliche Liste kann in einem Vorfallticket hilfreich sein, in dem die Ressource bzw. das Asset, das das Problem hervorgerufen hat, verknüpft wird.

Die folgenden **Objekttypen** können als Einträge für skriptbasierte Autocomplete-Listen verwendet werden:

- Statische Strings
- Bearbeiter (dies hat nichts mit der Zuweisung eines Tickets zu einem Bearbeiter zu tun!)
- Tickets
- Units (= Kunden, d.h. Kontakte und Firmen)
- Ressourcen

Die Einträge für Bearbeiter, Tickets, Kunden und Ressourcen können auf Suchergebnissen basieren. Auf diese Weise können Sie CM anpassen, um die Einträge in den Drop-down-Menüs dynamisch zu erzeugen.

Der Anzeigewert für das Objekt in der Liste hängt von der Art des definierten Labels (statisch vs. dynamisch) ab. Dies ist in den nachfolgenden Abschnitten erklärt.

Konfigurieren einer skriptbasierten Autocomplete-Liste mit dem Admin Tool

Konfiguration der Datenfelder

Um eine skriptbasierte Autocomplete-Liste umzusetzen, müssen Sie folgende Schritte durchführen:

Schritt 1: Definieren Sie das Autocomplete-Skript, d. h. erstellen Sie ein Skript des Typs *Text-Autovervollständigung*. Das Skript kann später geändert oder erweitert werden. Sie brauchen aber ein vorhandenes Skript für die Zuweisung in Schritt 2.

Schritt 2: Definieren Sie das Datenfeld, das die Liste enthalten soll, d. h. definieren Sie eines der folgenden Felder:

- Ticketfeld
- Kundenfeld
- Ressourcenfeld

Das Feld muss den Typ *autocomplete (Autovervollständigung)* haben und das Skript referenzieren, das die Autocomplete-Funktion implementiert.

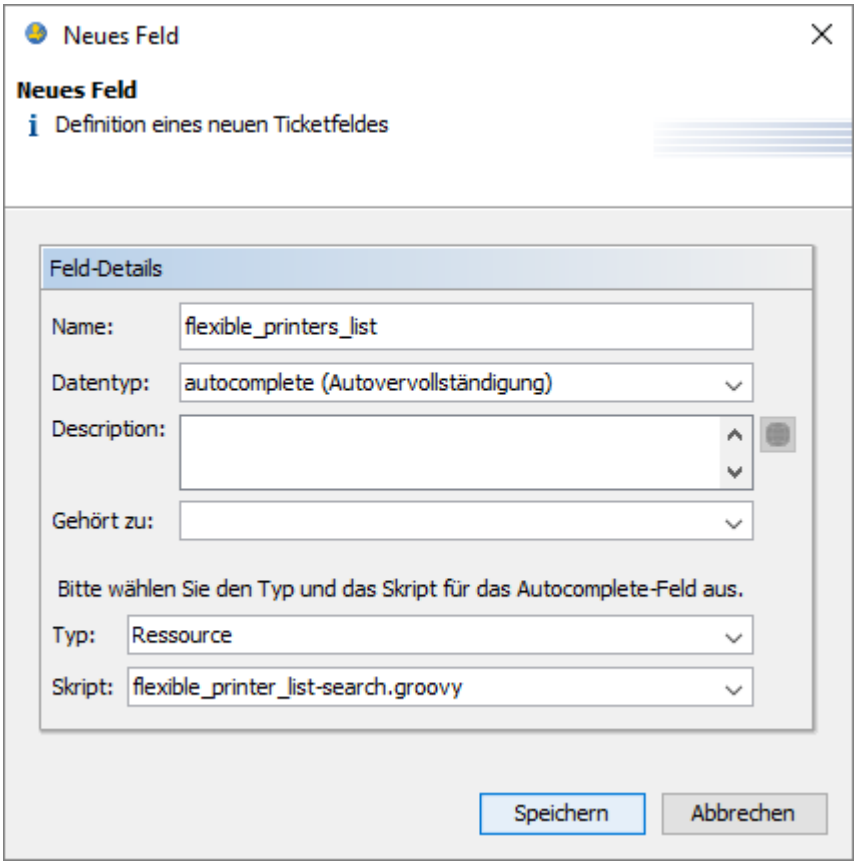


Abbildung 455: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Erstellen eines neuen Ressourcenfeldes des Typs autocomplete*

Die folgenden Werte sind für einen Eintrag in einer Autocomplete-Liste, d. h. für einen Listenwert, relevant:

- **Interner Wert**
  Für Felder des Typs *Ticket*, *Unit*, *Ressource* oder *Bearbeiter* ist dies die Referenz auf das entsprechende Objekt. Für Felder des Typs *String* ist es der String selber.
- **Anzeigewert**
  Dies ist der Wert, der dem Endbenutzer im Web Client oder CM/Track V2 angezeigt wird. Der Anzeigewert kann folgendermaßen konfiguriert sein:
  - **dynamisch**
    Der Anzeigewert wird aus dem Objekt selber abgerufen.
  - **statisch**
    Es wird ein String definiert, der als Anzeigewert verwendet wird.

Folgende **Templates** werden für dynamische Anzeigewerte zur Darstellung der Einträge im Drop-down-Menü verwendet:

- **Bearbeiter:**
  - *engineer description template name*
- **Kunden:**
  - *Kundensuchergebnis*
  - Wenn kein Suche-Template definiert ist: Standard-Template
- **Ressourcen:**
  - *Suche*
- **Tickets:**
  - Festes internes Template mit Ticketname und -betreff

Um einen dynamischen oder statischen Anzeigewert zu definieren, verwenden Sie zum Hinzufügen der Werte zum Resultset die entsprechende Methode.

Zum Definieren eines statischen Anzeigewerts, d. h. der Anzeigewert wird bei Änderungen am referenzierten Objekt **nicht** aktualisiert, verwenden Sie die Methodensignatur, die sowohl ein Objekt als auch einen String als Parameter hat, um das Objekt und einen Wert aus dem referenzierten Objekt oder einen beliebigen String zum Resultset hinzuzufügen. Zum Beispiel:

```
resultset.add(ticket, ticket.getSubject());
```

```
resultset.add(ticket, "some description");
```

Zum Definieren eines dynamischen Labels, d. h. der Anzeigewert wird aus dem referenzierten Objekt abgerufen und bei relevanten Änderungen am referenzierten Objekt aktualisiert, verwenden Sie die Methodensignatur, die nur das Objekt als Parameter hat, um nur das Objekt selber zum Resultset hinzufügen. Zum Beispiel:

```
resultset.add(ticket);
```

Das Skript Text-Autovervollständigung

Ein Skript des Typs *Text-Autovervollständigung* muss die Methode `onSearchInput` implementieren. Wenn Sie ein neues Skript des Typs *Text-Autovervollständigung* erstellen, wird die Methode automatisch eingefügt und Sie müssen nur die „Intelligenz“ des Skripts hinzufügen.

```
import com.consol.cmas.common.model.autocomplete.script.*
/**
* This method is called by web client when the user clicks or types into the
 autocomplete field
* @param pSearchStr the search String types or NULL if user clicked into field
 without typing
* @param pKey fieldKey
* @param pContext (Ticket/Unit/Resource) holder containing current values set on
 the form
*/
ScriptAutocompleteResult onSearchInput(String pSearchStr, FieldKey pKey, Context
 pContext) {
   return ScriptAutocompleteResult.noResults('No Results')
}
/**
* For custom-fields in ListFields row index is provided as well
*/
ScriptAutocompleteResult onSearchInput(String pSearchStr, FieldKey pKey, int
 pRowIndex, Context pContext) {
   return ScriptAutocompleteResult.noResults('No Results')
}
/**
* This method controls the value displayed when this field enters edit mode on
 screen. Can be used to obtain the beautified value based on the current internal
 value
```

```
* for display purposes.
* @param pValue the stored value of the field
* @param pKey fieldKey
* @param pContext (Ticket/Unit/Resource) holder containing current values set on
 the form
* @return - the string which contain beautified label or one of the domain objects:
 Ticket/Unit/Resource,
* for domain object beautified label will be taken from standard template of the
 object.
* @deprecated - For new AutocompleteField this method is ignored. Labels are stored
 in DB so this method is not needed any more
* The old scripted autocomplete fields still support this method, but it will be
 removed with next release
*/
Object onEditDisplayEntered(String pValue, FieldKey pKey, Context pContext) {
   if (!pValue) return null; // display nothing (current behaviour)
      return "Beautified: " + pValue;
}
/**
* @param pValue the stored value of the field
* @param pKey fieldKey
* @param pContext (Ticket/Unit/Resource) holder containing current values set on
 the form
* @return - the string which contain beautified label or one of the domain objects:
 Ticket/Unit/Resource,
* for domain object beautified label will be taken from standard template of the
 object.
* @deprecated - For new AutocompleteField this method is ignored. Labels are stored
 in DB so this method is not needed any more
* The old scripted autocomplete fields still support this method, but it will be
 removed with next release
*/
Object onEditDisplayEntered(String pValue, FieldKey pKey, int pRowIndex, Context
 pContext) {
   if (!pValue) return null;
      return "Beautified:" + pValue;
}
```

Code-Beispiel 81: *Template für Admin-Tool-Skripte des Typs Text-Autovervollständigung*

Sie müssen das Objekt `ScriptAutoCompleteResult` (in den folgenden Beispielen *Resultset* genannt) füllen, das zurückgegeben wird. Dieses Objekt bildet das Drop-down-Menü. Sie können Objekte eines Typs zum Resultset hinzufügen, wobei der Typ durch den bei der Definition im Admin Tool eingegebenen Resultset-Typ definiert wird. Im obigen Beispiel ist der Typ des Resultsets *Ressource*, sodass nur Ressourcen zum Resultset, das für die Liste zurückgegeben wird, hinzugefügt werden können. Wenn Sie andere Werte benötigen, zum Beispiel weil Sie eine Meldung (d. h. einen String) anzeigen möchten, für die eine Mindestanzahl an Zeichen eingegeben werden muss, muss ein eigenes Resultset zurückgegeben werden. Dies kann natürlich im gleichen Skript erfolgen.

Um eine Meldung anzuzeigen, dass keine Ergebnisse gefunden wurden, können Sie die Methode `noResults(StringMessage)` verwenden.

Eine detaillierte Übersicht über die Methoden der Java-Klasse `ScriptAutoCompleteResult` finden Sie in der ConSol CM API-Dokumentation.

Die Objekte, die zum Resultset hinzugefügt werden, können von internen CM-Daten (z.B. von einer Bearbeitersuche) abgeleitet werden oder aus einer Suche in externen Quellen stammen. Auf diese Weise können Sie zum Beispiel eine Suchliste mit Namen aus einer externen Datenbank anbieten.

Um mit der Eingabe des Bearbeiters arbeiten zu können, bevor diese gespeichert wurde (z.B. wenn der Bearbeiter mit der Eingabe eines Kundennamen in ein String-Feld beginnt), müssen Sie das Objekt `pContext` verwenden. Je nachdem, welche Seite geöffnet ist (d.h. je nach Kontext), enthält `pContext` unterschiedliche Daten und kann unterschiedliche Objekte zurückgeben:

- Unit
    - Nur auf Unit-Seite (Kontakt/Firma) verfügbar:
    - Beispiel: myunit = pContext.getUnit() oder myunit = pContext.unit
- Bearbeiter:
    - Verfügbar für den aktuellen Bearbeiter und den zugewiesenen Bearbeiter.
    - Aktueller Bearbeiter nur über engineerService. Beispiel: def myeng = engineerService.current
    - Zugewiesener Bearbeiter über das Ticket: def myeng = pContext.ticket?.engineer
- Ticket

  Verfügbar in
    - Seite zum Erstellen eines Tickets
    - Seite zum Bearbeiten eines Tickets
    - ACF
- Ressource
    - Nur auf Ressourcenseiten verfügbar
    - Beispiel: myres = pContext.resource
- Strings
    - Immer verfügbar, unabhängig von pContext
    - Beispiel: "p1"

Das folgende Beispielskript erstellt eine Liste (d. h. ein Drop-down-Menü), in der eines der folgenden Objekte angezeigt wird:

- Ein Resultset, das den Benutzer darüber informiert, dass mindestens drei Zeichen eingegeben werden müssen.
- Ein Resultset, das nur eine Meldung enthält, dass keine Ergebnisse gefunden wurden, d. h. dass anhand der bereits vom Benutzer eingegebenen Zeichen keine Treffer abgerufen werden konnten.
- Ein Resultset mit den Suchergebnissen in den Ressourcen, die mit den vom Benutzer bereits eingegebenen Zeichen gefunden wurden. Das ist das hauptsächliche Resultset des Skripts.

```
import com.consol.cmas.common.model.autocomplete.script.*
import com.consol.cmas.common.model.resource.ResourceCriteria
/**
* This method is called by web client when the user clicks or types into the
 autocomplete field
* @param pSearchStr the search String types or NULL if user clicked into field
 without typing
* @param pKey fieldKey
* @param pContext (Ticket/Unit/Resource) holder. More about this later in the
 specification (Context section)
*/
log.info '##### Starting Autocomplete script ... #####'
ScriptAutocompleteResult onSearchInput(String pSearchStr, FieldKey pKey, Context
 pContext) {
log.info '##### Class of pContext is now: ' + pContext.class
log.info '##### Resource of pContext is now: ' + pContext.resource
ScriptAutocompleteResult resultset = new ScriptAutocompleteResult();

// Add matching customer when entering more than 2 characters
if (pSearchStr.length() < 3) {
   return new ScriptAutocompleteResult("NOTE: Please type at least 3 characters!");
} else
   log.info ' Building RESOURCE criteria ...'
   ResourceCriteria criteria = new ResourceCriteria();
   criteria.setPattern("*"+pSearchStr+"*")
   def resources = resourceService.getByCriteria(criteria)
   if(resources.size()<1){
      return ScriptAutocompleteResult.noResults('No matching resources found')
   } else {
      resultset.add(resources);
   }
      return resultset;
}
```

Code-Beispiel 82: *Skript einer Autocomplete-Liste für ein Ressourcenfeld*

Im obigen Skript werden die Ressourcenobjekte zum Resultset hinzugefügt, d. h. wir arbeiten mit dynamischen Anzeigewerten. Das bedeutet, dass der Anzeigewert eines Ressourcenobjekts in der Liste immer die Daten des Originalobjekts widerspiegelt. Diese werden mit dem entsprechenden Template dargestellt.

Die Liste, die mit dem oben gezeigten Skript implementiert ist, wird im Web Client wie folgt angezeigt:

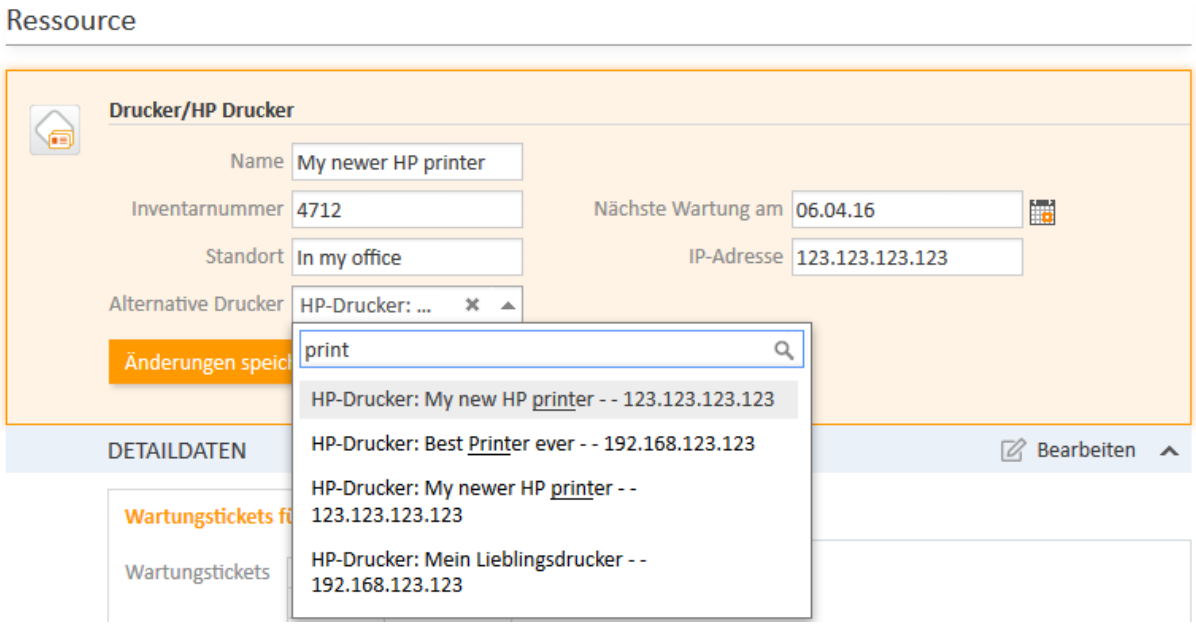


Abbildung 456: *ConSol CM Web Client - Skriptbasierte Autocomplete-Liste des Typs Ressource*

Sie können die Anzahl der Zeichen, die der Bearbeiter eingeben muss, bevor die Suche beginnt, steuern. Dies erfolgt in der folgenden If-Anweisung, die auch im obigen Beispiel enthalten ist.

```
if (pSearchStr.length() < 3) { }
```

In diesem Beispiel muss der Bearbeiter mindestens drei Zeichen in das String-Feld eingeben, bevor die Suche beginnt.

Abrufen und Setzen von Werten von Autocomplete-Feldern

Auf den Wert eines Autocomplete-Feldes kann später mit der neuen Klasse `AutocompleteValue` zugegriffen werden:

```
AutocompleteValue myAutocompleteValue = <object>.get("<field group>.<field>");
myAutocompleteValue.getInternalValue();
myAutocompleteValue.getDisplayValue();
```

Beispielskript, das zu Testzwecken verwendet werden kann. Das Skript schreibt lediglich den Wert eines Autocomplete-Ticketfeldes mit Bearbeiterdaten in die Log-Datei.

```
// displayFieldValues.groovy
// Test display of filed values in log file ... for test purposes only!
// display engineer of ticket field (not engineer field in header!)
def ticket = workflowApi.ticket

AutocompleteValue myAutocompleteValue = ticket.get("serviceDesk_fields.flexible_
 engineers_list")
def intValue = myAutocompleteValue.getInternalValue()
log.info 'INT VALUE IS NOW ' + intValue
def dispValue = myAutocompleteValue.getDisplayValue()
log.info 'DISP VALUE IS NOW ' + dispValue
```

Code-Beispiel 83: *Skript, das die Werte eines Autocomplete-Ticketfeldes mit Bearbeiterdaten in die Datei server.log schreibt*

Es ist auch möglich, die Werte eines Autocomplete-Feldes über die API zu setzen. Das folgende Beispiel setzt ein dynamisches Label. Um ein statisches Label zu setzen, lassen Sie einfach die zweite Zeile des Codes aus.

```
AutocompleteValue myAutocompleteValue = new AutocompleteValue("<internal
 value>","<display value>");
myAutocompleteValue.setDynamicDisplayValue(true);
<object>.set("<field group>.<field>", myAutocompleteValue);
```

Zugriff auf die Werte von skriptbasierten Autocomplete-Listen in CM/Track

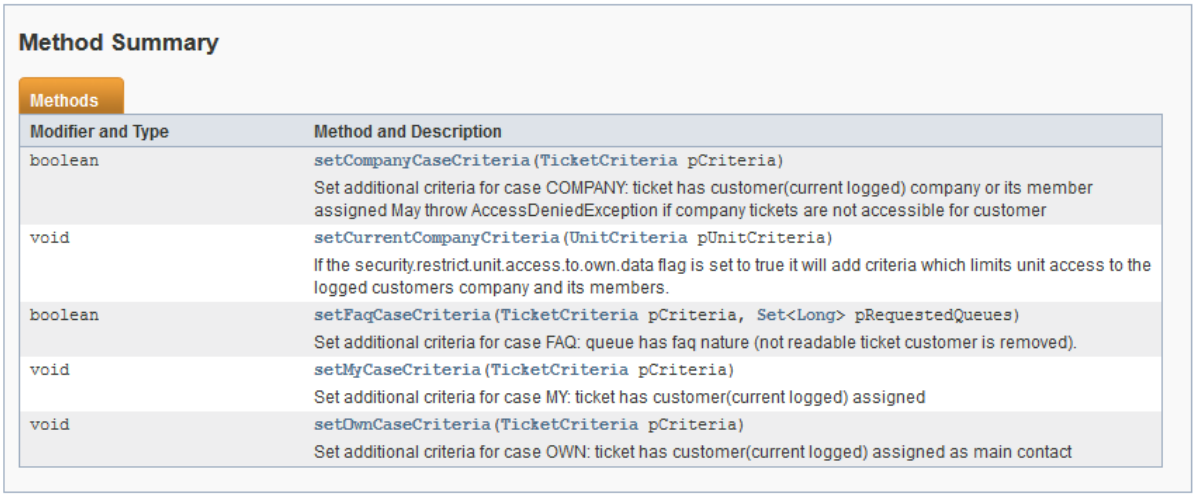
Die Berechtigungen werden während der Ausführung des Autocomplete-Skripts beim Editieren des Feldes im Web Client oder CM/Track V2 überprüft, d. h. der Benutzer kann nur Einträge sehen und auswählen, die zu Objekten gehören, die er mit den ihm zugewiesenen Rollen sehen darf. Beim Anzeigen von Autocomplete-Feldern findet keine Überprüfung der Berechtigungen statt. Das bedeutet, dass der Benutzer in Autocomplete-Feldern und Protokolleinträgen Label sehen könnte, die sich auf Objekte beziehen, für die er keine Berechtigung hat.

Hinweis zu Berechtigungen in CM/Track V2:

Berechtigungen für skriptbasierte Autocomplete-Felder funktionieren wie im Web Client. Spezifische Einschränkungen für CM/Track V2, insbesondere die Einschränkung, dass der Benutzer nur seine eigenen Tickets (oder Tickets anderer Kontakte seiner Firma, sofern konfiguriert) und Kontakte seiner Firma sieht, gehören nicht zur Standardkonfiguration. Dies bedeutet, dass der Benutzer standardmäßig die Berechtigung hat, Einträge zu sehen und auszuwählen, die sich auf Tickets beziehen, die sich in Queues befinden, für die der Track-Benutzer mindestens Leseberechtigungen hat, und die sich auf Kunden beziehen, die Kundengruppen angehören, für die der Track-Benutzer mindestens Leseberechtigungen hat. Zusammengefasst, der Kunde würde Daten sehen, die zu anderen Kunden gehören.

Wenn dies nicht erwünscht ist, muss das Autocomplete-Skript mithilfe der Methoden der neuen Klasse `customerSecurityCriteriaBuilder` angepasst werden. Weitere Details dazu im Beispielskript unten.

Die Klasse `customerSecurityCriteriaBuilder` wurde zur ConSol CM-API hinzugefügt. Sie enthält die folgenden Methoden zur Kontrolle der Berechtigungen für Autocomplete-Ergebnisse in CM/Track V2:



Method Summary

Methods

Modifier and Type	Method and Description
boolean	`setCompanyCaseCriteria(TicketCriteria pCriteria)` Set additional criteria for case COMPANY: ticket has customer(current logged) company or its member assigned May throw AccessDeniedException if company tickets are not accessible for customer
void	`setCurrentCompanyCriteria(UnitCriteria pUnitCriteria)` If the security.restrict.unit.access.to.own.data flag is set to true it will add criteria which limits unit access to the logged customers company and its members.
boolean	`setFaqCaseCriteria(TicketCriteria pCriteria, Set<Long> pRequestedQueues)` Set additional criteria for case FAQ: queue has faq nature (not readable ticket customer is removed).
void	`setMyCaseCriteria(TicketCriteria pCriteria)` Set additional criteria for case MY: ticket has customer(current logged) assigned
void	`setOwnCaseCriteria(TicketCriteria pCriteria)` Set additional criteria for case OWN: ticket has customer(current logged) assigned as main contact

Abbildung 457: *ConSol CM API-Dokumentation der Klasse customerSecurityCriteriaBuilder*

Beispielskript: Das folgende Autocomplete-Skript beschränkt die Suchergebnisse eines in CM/Track V2 verwendeten Autocomplete-Feldes auf die Tickets, die dem Benutzer gehören.

```
ScriptAutocompleteResult onSearchInput(String pSearchStr, FieldKey pKey, Context
 pContext) {
def result = new ScriptAutocompleteResult();
def criteria= new TicketCriteria();
criteria.setPattern(pSearchStr+ "*");
customerSecurityCriteriaBuilder.setMyCaseCriteria(criteria);
for (Ticket ticket : ticketService.getByCriteria(criteria)) {
   result.add(ticket);
}
return result;
}
```

Code-Beispiel 84: *Skript einer Autocomplete-Liste, die in CM/Track V2 angezeigt werden kann*

Allgemeine Informationen über skriptbasierte Autocomplete-Felder in CM-Version 6.11.1.1

Bei der Verwendung der neuen Methode zum Konfigurieren von skriptbasierten Autocomplete-Feldern gibt es folgende Einschränkungen:

- Keine Unterstützung von CM/Track V1. Skriptbasierte Autocomplete-Felder werden in CM/Track V1 nicht unterstützt.
- Keine Übertragung an das DWH. Skriptbasierte Autocomplete-Felder werden nicht an das DWH übertragen und können nicht in Berichten verwendet werden.

- Keine Unterstützung von ETL. Der Import und Export von skriptbasierten Autocomplete-Feldern mittels ETL wird nicht unterstützt.
- Skriptbasierte Autocomplete-Felder können nicht indiziert werden. Sie stehen daher in Suchen nicht zur Verfügung.
- Wenn ein Objekt, das in einem Autocomplete-Feld referenziert wird, gelöscht wird, kann das Feld auf ein nicht existierendes Objekt zeigen. Dies muss berücksichtigt werden, wenn der Feldwert später in Skripten verwendet wird.
- Skriptbasierte Autocomplete-Felder können nicht in Textvorlagen und Dokumentvorlagen verwendet werden.

Hinweis zu skriptbasierten Autocomplete-Feldern in der aktuellen CM-Version vs. vorherigen Versionen

Diese neue Methode der Konfiguration von Autocomplete-Feldern ersetzt die alte Methode, die die Annotation *text-type* = „autocomplete“ verwendet hat. Vorhandene Felder, die die alte Methode verwenden, bleiben in Version 6.11.1.1 funktionsfähig. Diese Felder werden bei einer Aktualisierung nicht verändert. Die vorher verwendete Methode `onEditDisplayEntered` funktioniert für vorhandene Felder weiterhin. Sie wird allerdings nicht mehr benötigt und in einer zukünftigen Version von ConSol CM entfernt.

F.8.1.5 Skripte des Typs *Abhängige sortierte Liste*

Abhängige sortierte Listen sind hierarchische Listen, die eine ähnliche Datenstruktur abbilden wie MLAs (siehe Abschnitt MLA-Verwaltung). Im Gegensatz zu MLAs wird bei abhängigen sortierten Listen immer nur eine Ebene gleichzeitig angezeigt. Abhängig von dem Wert, den der Benutzer auf der aktuellen Listenebene auswählt, wird eine andere Liste der untergeordneten Ebene geöffnet. Es muss nicht für jeden Listeneintrag eine untergeordnete Liste geben. In der Graphenterminologie kann eine Liste also eine Kombination aus Knoten und Blättern sein. Ein Skript für abhängige sortierte Listen wird der Ticketfeldgruppe, Kundenfeldgruppe oder Ressourcenfeldgruppe zugewiesen, für die es benötigt wird.

Siehe folgendes Beispiel:

In einem *Helpdesk-Ticket* kann ein Ort ausgewählt werden. Zuerst wählt der Benutzer den *Kontinent* aus. Je nach ausgewähltem Kontinent wird eine Unterliste mit Subkontinenten oder mit Ländern angezeigt. Alle Ticketfelder müssen zuerst als normale Felder des Typs *enum (Sortierte Liste)* definiert werden. Im Skript wird der Wert der ersten Listenebene überprüft und je nachdem, welcher Wert ausgewählt wurde, wird eine andere Liste auf der zweiten Ebene angezeigt. Sie können so viele Ebenen anlegen, wie Sie möchten. Bedenken Sie aber, dass das Editieren des Skripts immer komplexer wird.

Das *Skript für abhängige sortierte Listen* wird im Admin Tool gespeichert. Lassen Sie sich beim Erstellen bzw. Editieren des Skripts von einem ConSol CM-Consultant unterstützen, da es sich um eine ziemlich komplexe Aufgabe handelt.

Die folgende Abbildung zeigt ein Skript für abhängige sortierte Listen. Sie gelangen auf diese Seite, indem Sie das Navigationselement *Skripte und Templates* der Navigationsgruppe *System* öffnen.

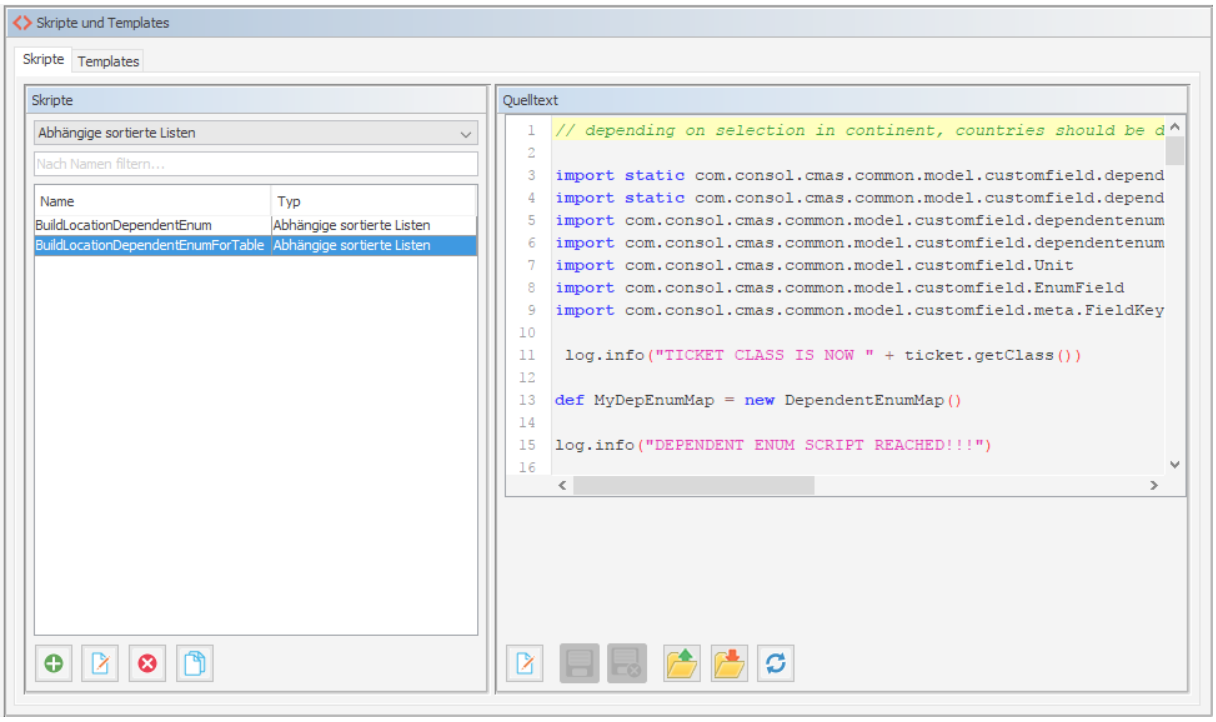


Abbildung 458: *ConSol CM Admin Tool - System, Skripte und Templates: Skript für abhängige sortierte Listen*

Das *Skript für abhängige sortierte Listen* wird der Ticketfeldgruppe, Kundenfeldgruppe oder Ressourcenfeldgruppe zugewiesen, für die es benötigt wird.

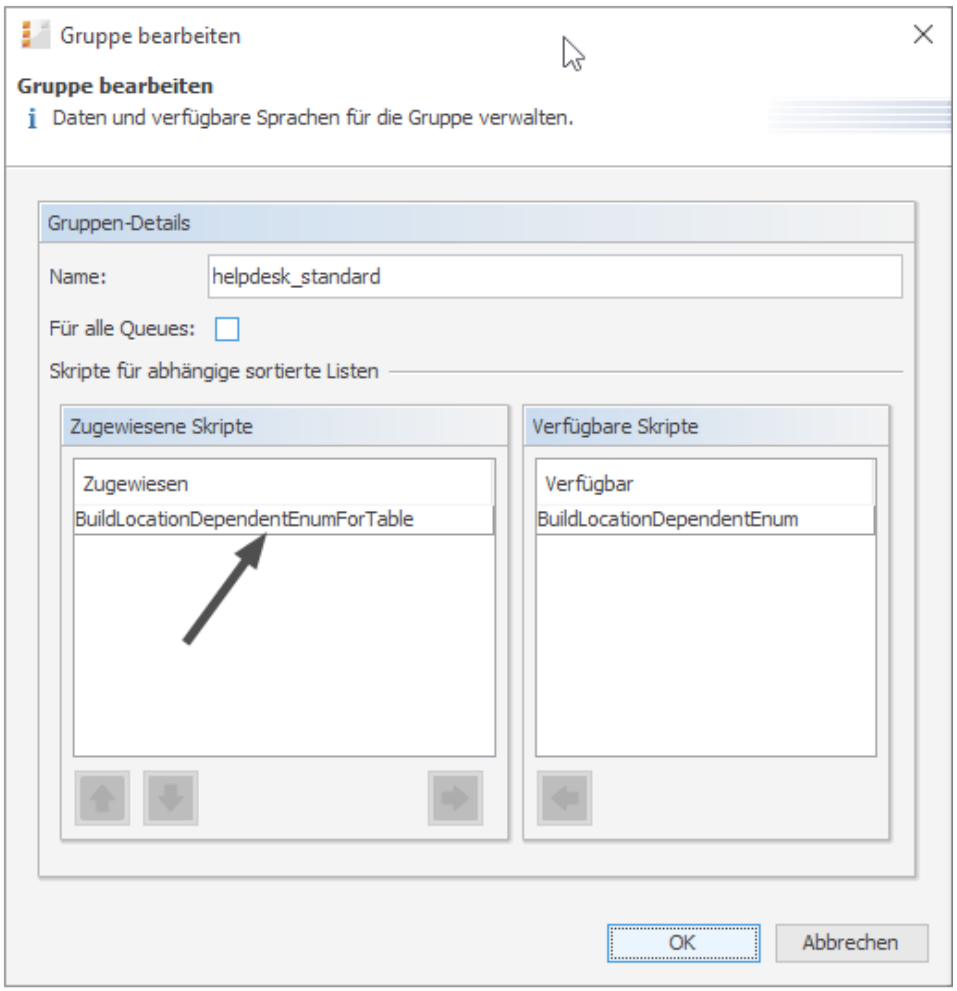


Abbildung 459: *ConSol CM Admin Tool - Tickets, Ticketfelder: Zuweisen eines Skripts für abhängige sortierte Listen zu einer Ticketfeldgruppe*

Im Web Client sieht der Bearbeiter nur die Unterliste des Wertes, den er auf der ersten Listenebene ausgewählt hat.

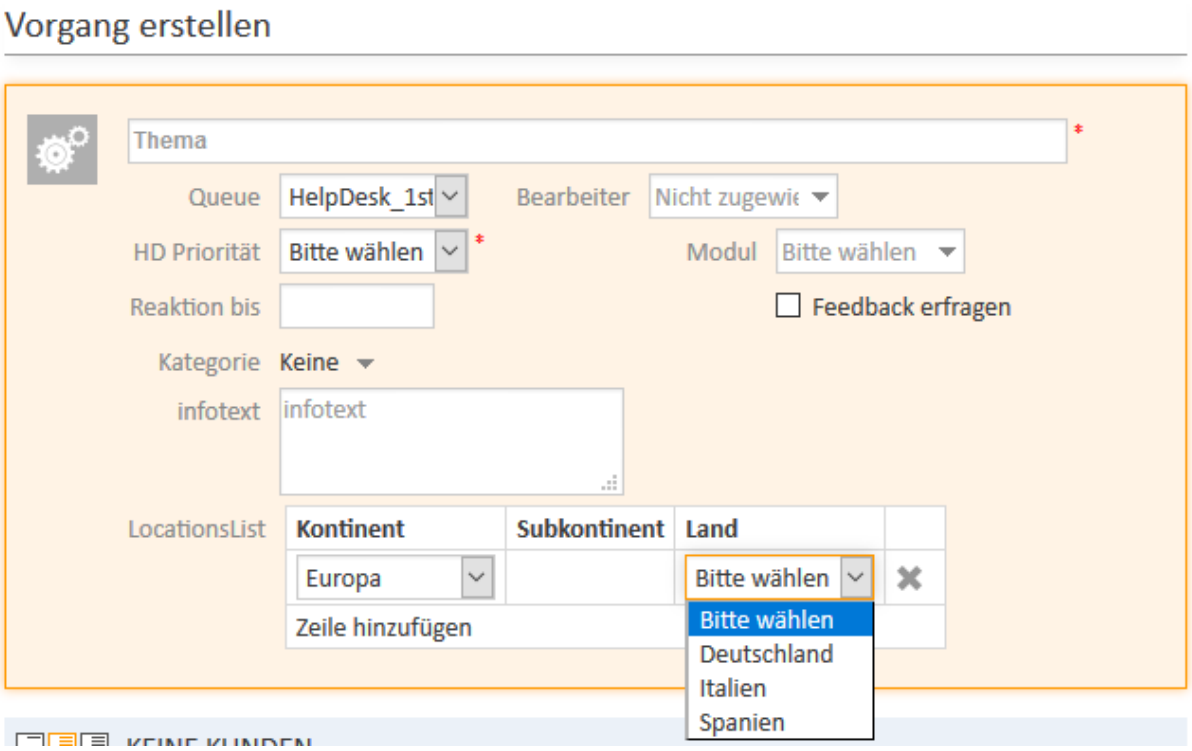


Abbildung 460: *ConSol CM Web Client - Unterliste des Kontinents Europa*

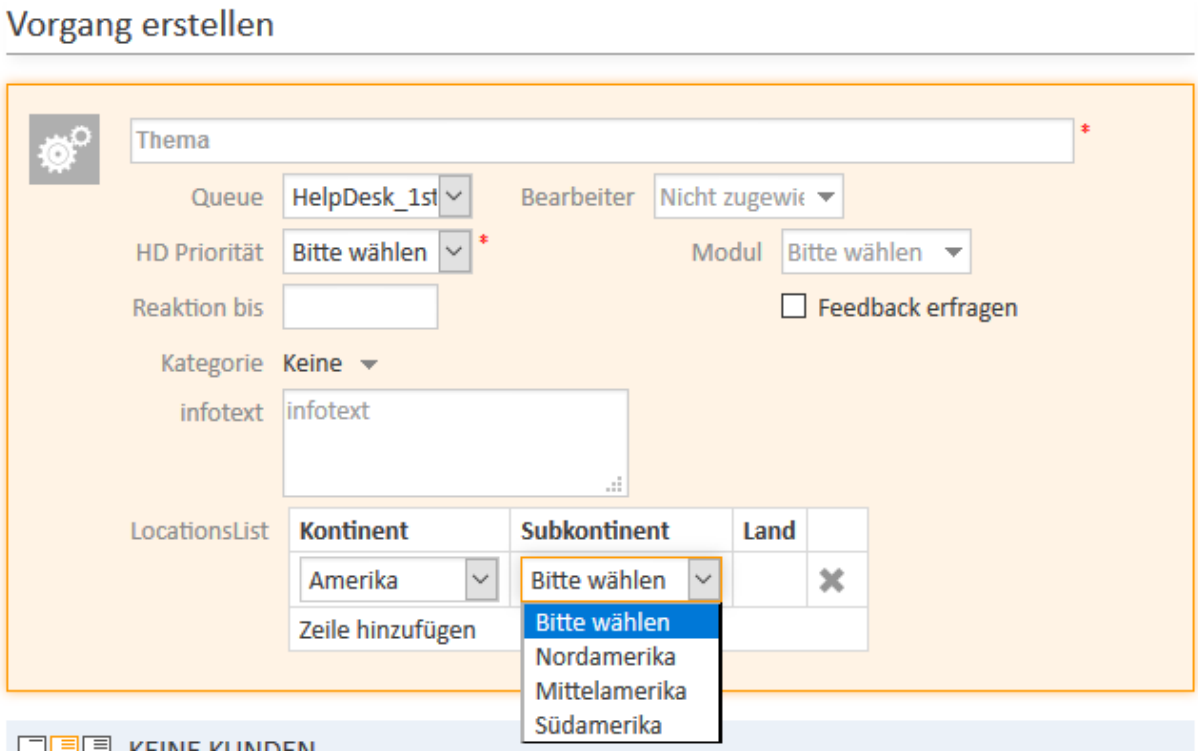


Abbildung 461: *ConSol CM Web Client - Unterliste des Kontinents Amerika*

F.8.1.6 Skripte des Typs *E-Mail*

Skripte dieses Typs haben verschiedene Funktionen. Einige der Skripte sind Teil der Standardsystemkonfiguration und müssen an die kundenspezifische Systemkonfiguration angepasst werden. Sie können auch eigene Skripte hinzufügen.

WICHTIGE INFORMATION

Seit ConSol CM-Version 6.9.4 gibt es zwei Modi für eingehende E-Mails:

- **Mule/ESB** - Dieser Modus ist seit Version 6 in ConSol CM verfügbar. Er ist in CM-Versionen 6.11 und höher veraltet. Deshalb wird dieser Modus nicht mehr in den ConSol CM-Handbüchern behandelt. Wenn Sie ein älteres CM-System mit Mule/ESB betreiben, finden Sie die nötigen Informationen in den jeweiligen Handbüchern. Alle ConSol CM 6.11-Systeme müssen im NIMH-Modus betrieben werden. Informationen über den Wechsel von Mule/ESB zu NIMH finden Sie in den älteren Administratorhandbüchern.
- **NIMH** - *New Incoming Mail Handler*, seit Version 6.9.4 verfügbar. Dies ist der einzige Modus für eingehende E-Mails in ConSol CM-Versionen 6.11 und höher.

Der aktive E-Mail-Modus wird in der System-Property cmas-core-server, nimh.enabled gesetzt:

- **true** - NIMH-Modus
- **false** - Mule/ESB-Modus

Das Senden von E-Mails, d. h. die Konfiguration des SMTP-Servers, wird vom Modus für eingehende E-Mails nicht beeinflusst.

E-Mail-Skripte für die Verarbeitung von eingehenden E-Mails im NIMH-Modus

Wenn ConSol CM eine E-Mail erhält, wird diese von mehreren Skripten verarbeitet, siehe folgende Abbildung.

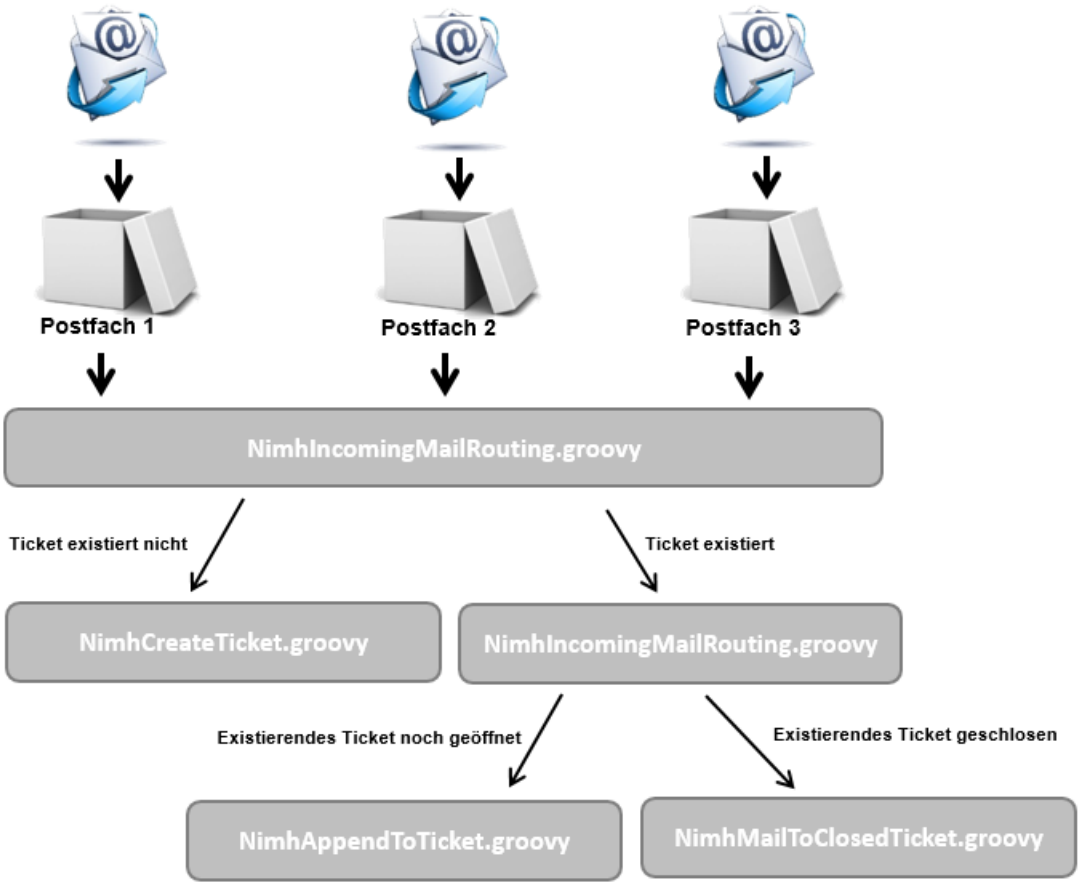


Abbildung 462: *ConSol CM Admin Tool - E-Mail-Skripte (NIMH)*

- **NimhIncomingMailRouting.groovy**
  Standardskript. Dies ist das erste Skript, das ausgeführt wird, wenn eine neue E-Mail eingeht. Hier wird entschieden, ob ein neues Ticket erstellt werden soll, ob die E-Mail zu einem vorhandenen Ticket gehört (dann wird `NimhAppendToTicket.groovy` ausgeführt), oder ob die E-Mail zu einem geschlossenen Ticket gehört (dann wird `NimhMailToClosedTicket.groovy` ausgeführt). Dieses Skript muss nicht geändert werden, um es an eine kundenspezifische Umgebung anzupassen.
- **NimhCreateTicket.groovy**
  Standardskript, das für die Erstellung eines Tickets zuständig ist, wenn eine E-Mail in einem der für ConSol CM konfigurierten Postfächer eingeht (siehe Abschnitt E-Mail-Konfiguration mit dem Admin Tool). Wenn der E-Mail-Betreff nicht zum regulären Ausdruck passt, mit dem die E-Mail an ein vorhandenes Ticket angehängt werden kann, wird dieses Skript ausgeführt. Hier werden, unabhängig vom Postfach, alle E-Mails verarbeitet, die ConSol CM erhalten hat (und die nicht einem vorhandenen Ticket zugewiesen wurden). In diesem Skript muss eine Standard-Queue für eingehende E-Mails definiert werden. Zusätzlich können weitere Werte von Ticketfeldern definiert werden (wie z. B. die Standardpriorität der per E-Mail erstellten Tickets). Es wird eine Standardfirma definiert. Wenn die Standardfirma (in einem neuen System) noch nicht vorhanden ist, wird sie automatisch erstellt. Im Skript können Entscheidungen getroffen werden,

z. B. kann anhand der To-Adresse oder anderer Parameter entschieden werden, in welcher Queue das neue Ticket erstellt werden soll. Dieses Skript muss normalerweise stark angepasst werden. Bitten Sie dafür einen ConSol CM-Consultant um Unterstützung.

- **NimhAppendToTicket.groovy**
  Standardskript, mit dem eine E-Mail an ein vorhandenes Ticket angehängt wird. Die Zuweisung der E-Mail zum Ticket wird normalerweise über einen Vergleich des E-Mail-Betreffs mit dem regulären Ausdruck vorgenommen. Eine detaillierte Beschreibung dieses Kontextes finden Sie im Abschnitt E-Mail-Konfiguration mit dem Admin Tool. Normalerweise muss dieses Skript nicht verändert werden.
- **NimhMailToClosedTicket.groovy**
  Standardskript, das für die Verarbeitung von E-Mails, die zu geschlossenen Tickets gehören, zuständig ist. Das Standardsystemverhalten ist, ein neues Ticket für den Kunden (Absender der E-Mail) zu erstellen und eine Referenz zum alten/geschlossenen Ticket zu setzen. Normalerweise muss dieses Skript nicht verändert werden.

E-Mail-Skripte für ausgehende E-Mails

Für jede Queue kann ein *E-Mail-Skript* konfiguriert werden. Im Abschnitt Queue-Verwaltung finden Sie eine Erklärung über die Konfiguration dieses Skripts. Eine E-Mail, die (automatisch durch den Workflow oder manuell durch einen Bearbeiter) aus einem Ticket in dieser Queue geschrieben wird, durchläuft dieses Skript, bevor sie das ConSol CM-System verlässt. In diesem *E-Mail-Skript* können Sie also queue-spezifische Einstellungen für die ausgehende E-Mail ändern oder setzen. Ein häufiger Anwendungsfall ist das Einstellen einer queue-spezifischen Reply-To-Adresse, um teamspezifische Reply-To-Adressen verwenden zu können.

Ein Beispiel für ein ausgehendes *E-Mail-Skript* ist folgendes Skript, das zur ConSol CM-Standardapplikation gehört:

- **ChangeOutgoingMail.groovy**
  Standardskript, das nicht verwendet wird, aber als Template für andere ausgehende *E-Mail-Skripte* dient. Sie können mit diesem Template queue-spezifische *E-Mail-Skripte* konfigurieren.

Innerhalb des ausgehenden *E-Mail-Skripts* ist das Java-Objekt `mailEntry` implizit als Objekt `mail` verfügbar. Sie müssen alle erforderlichen Attribute für die ausgehende E-Mail mit den Methoden `mail.setAttribute()` oder `mail.setAttributes()` setzen.

```
def queueReplyAddress = "serviceteam@mycompany.com"
// you might also use system properties for the queue-specific e-mail addresses and
 fetch an
// address using the configurationService!

mail.setAttribute('Reply-to', queueReplyAddress)
```

Code-Beispiel 85: *E-Mail-Skript*

Häufige E-Mail-Attribute sind:

- Bcc
- From

- Reply-to
- To
- Cc
- Subject

In RFC 5322 finden Sie eine sehr detaillierte Beschreibung des E-Mail-Formats.



Beachten Sie das folgende technische Verhalten von ConSol CM bei der Konfiguration von Reply-To-Adressen und passen Sie Ihr System entsprechend an!

Der technische Hintergrund:

Es gibt vier mögliche Reply-To-Adressen, mit denen Sie zu tun haben:

1. Die **Reply-To-Adresse, die in der System-Property `mail.reply.to` festgelegt ist.** Wenn sie gesetzt ist, wird sie im Ticket-E-Mail-Editor im Web Client angezeigt. Ob es tatsächlich die Reply-To-Adresse ist, die in einer E-Mail angewendet wird, hängt von der Konfiguration des queue-spezifischen Skripts für ausgehende E-Mails ab. Siehe nächster Punkt. Wenn das Attribut der Seitenanpassung *showReplyTo* für den Typ *mailTemplate* auf „false“ gesetzt ist, wird im Ticket-E-Mail-Editor keine Reply-To-Adresse angezeigt. Wenn die Property `mail.reply.to` gesetzt ist, wird diese Adresse trotzdem verwendet, außer es wird eine andere Adresse in einem Skript für ausgehende E-Mails gesetzt, siehe nächster Punkt.
2. Die **Reply-To-Adresse, die in einem queue-spezifischen Skript für ausgehende E-Mails gesetzt wird**. Da das Skript für ausgehende E-Mails die letzte Instanz ist, die eine ausgehende E-Mail verarbeitet, ist die in diesem Skript gesetzte E-Mail-Adresse immer die angewendete Reply-To-Adresse. Falls die Property `mail.reply.to` gesetzt ist, wird diese mail-reply.to-address nicht verwendet (aber trotzdem im Ticket-E-Mail-Editor angezeigt, was verwirrend sein kann. Im nächsten Abschnitt ist erklärt, was dies für die Systemkonfiguration bedeutet).
3. Die **E-Mail-Adresse, die in der System-Property `mail.from` gesetzt ist.** Wenn diese gesetzt ist und weder `mail.reply.to` noch eine queue-spezifische Reply-To-Adresse gesetzt ist, setzen die meisten E-Mail-Clients die From-Adresse als Reply-To-Adresse.
4. Die **E-Mail-Adresse des aktuellen Bearbeiters** (der Bearbeiter, der im Web Client angemeldet ist). Diese persönliche E-Mail-Adresse wird als Reply-To-Adresse für E-Mails aus dem Web Client verwendet, wenn weder die Property `mail.reply.to` gesetzt ist, noch ein queue-spezifisches Skript für ausgehende E-Mails konfiguriert ist, noch die Property `mail.from` gesetzt ist.

Im Web Client wird im Ticketprotokoll immer die Reply-To-Adresse angezeigt, die für eine ausgehende E-Mail tatsächlich verwendet wurde. So wird immer die angewendete E-Mail-Adresse angezeigt, auch wenn die im Ticket-E-Mail-Editor angezeigte Adresse (die Property

`mail.reply.to`) und die tatsächlich verwendete Reply-To-Adresse (die Reply-To-Adresse im queue-spezifischen Skript für ausgehende E-Mails) unterschiedlich sind. Die tatsächliche E-Mail-Adresse wäre in diesem Fall die Adresse aus dem Skript.

Unsere Empfehlung:

Es sollte immer eine Reply-To-Adresse für das System gesetzt werden! Sie können entscheiden, ob Sie

- mit der Reply-To-Adresse arbeiten, die in einem queue-spezifischen Skript für ausgehende E-Mails gesetzt wird

oder

- die System-Property `mail.reply.to` verwenden.

Da die E-Mail-Kommunikation immer über ConSol CM stattfinden soll und nicht über persönliche E-Mail-Adressen, sollte eine der beiden oben beschriebenen Systemeinstellungen verwendet werden, um zu verhindern, dass ConSol CM persönliche E-Mail-Adressen als Reply-To verwendet. Letzteres würde bedeuten, dass die E-Mails der Kunden an das persönliche E-Mail-Postfach des Bearbeiters gesendet werden statt an CM.

Auswirkungen auf die Systemkonfiguration:

1. Der einfachste Weg, um eine Reply-To-Adresse zu setzen, ist die Verwendung der System-Property `mail.reply.to`. Sie wird im Ticket-E-Mail-Editor angezeigt und ist die tatsächlich verwendete Reply-To-Adresse.
2. Wenn queue-spezifische Reply-To-Adressen erforderlich sind, empfehlen wir, ein Skript für ausgehende E-Mails zu schreiben, in dem die Queue-Namen den entsprechenden Reply-To-Adressen zugeordnet werden. Dieses Skript kann dann für Bcc, Cc oder andere Adressen ergänzt werden.
   Sie können die Property `mail.reply.to` und die queue-spezifischen Reply-To-Adressen auch kombinieren: Für alle Queues ohne ein spezielles Skript für ausgehende E-Mails wird die Adresse aus `mail.reply.to` verwendet; für alle Queues mit einem queue-spezifischen Skript für ausgehende E-Mails, das eine Reply-To-Adresse enthält, wird diese verwendet.

Auswirkungen auf Workflow-Skripte, die E-Mails versenden:

(Eine detaillierte Erklärung dazu finden Sie im *ConSol CM Process Designer-Handbuch*!)

- Verwenden Sie das Objekt und die Methode `configurationService.getValue("cmweb-server-adapter","mail.reply.to")`, um den Wert der System-Property abzurufen und in der ausgehenden E-Mail als Reply-To-Adresse zu setzen.
- Verwenden Sie das Objekt `Mail`, wenn das queue-spezifische Skript verwendet werden soll, z. B. `mail.useDefaultScript()`. Damit wird die Property `mail.reply.to` überschrieben!

> ⚠ Wenn weder die System-Property noch das queue-spezifische Skript für ausgehende E-Mails verwendet wird, d. h. wenn die Reply-To-Adresse nicht gesetzt ist, wird vom E-Mail-Client normalerweise die From-Adresse als Reply-To-Adresse verwendet.

F.8.1.7 Skripte des Typs *Duplizieren*

Im Web Client kann ein Ticket über die Option *Duplizieren* im Ticketmenü dupliziert werden. Sie können dies mit oder ohne Skript machen.

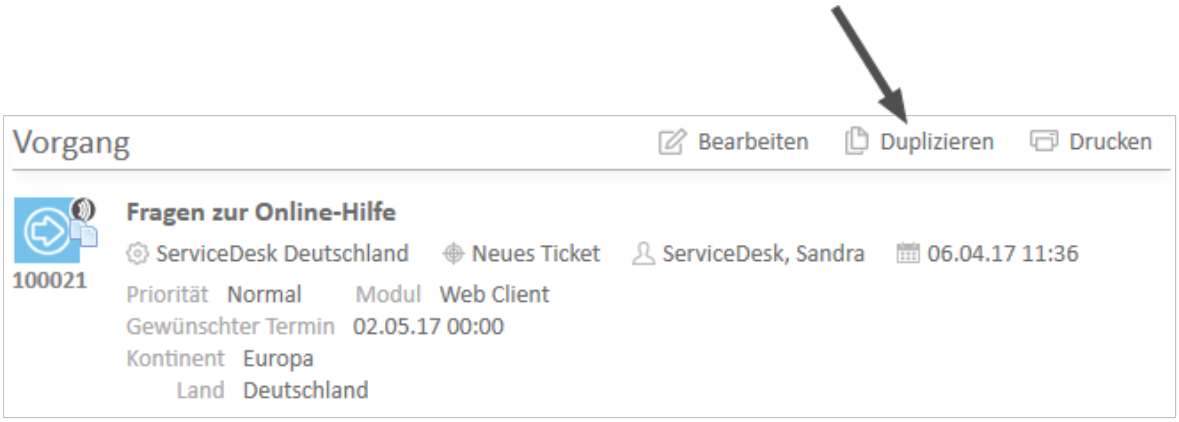


Abbildung 463: *ConSol CM Web Client - Option Duplizieren im Ticketmenü*

Die folgenden Abbildungen zeigen die Duplizierung eines Tickets ohne Duplizieren-Skript.

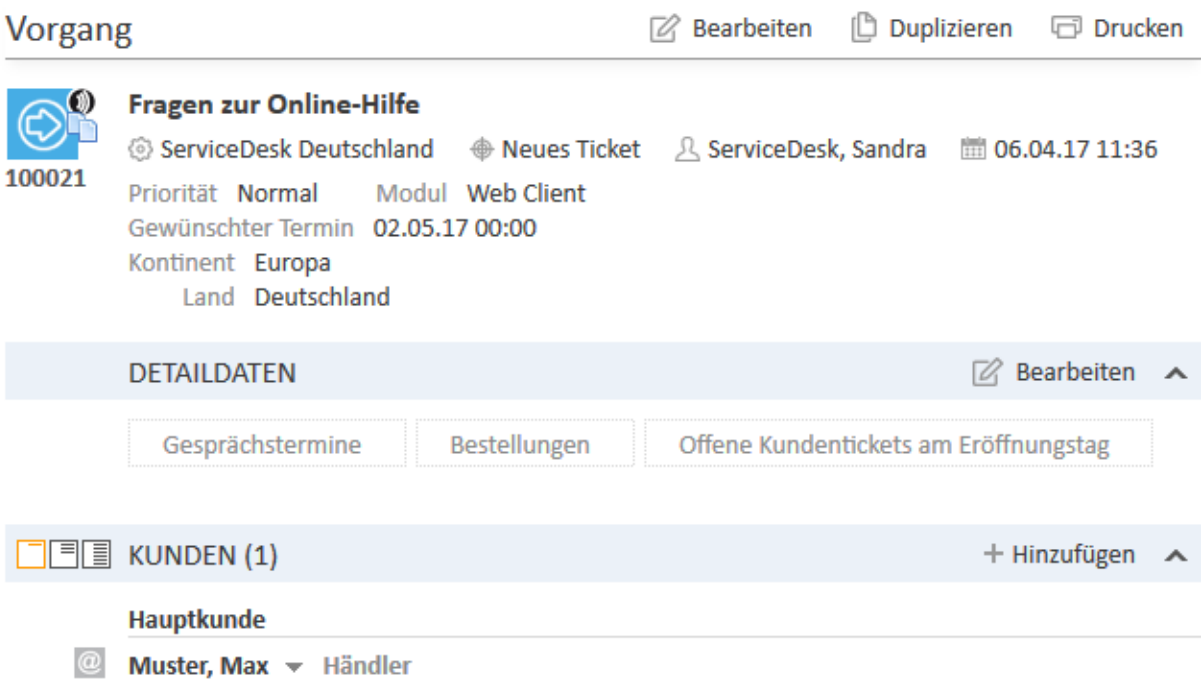


Abbildung 464: *ConSol CM Web Client - Ursprüngliches Ticket*

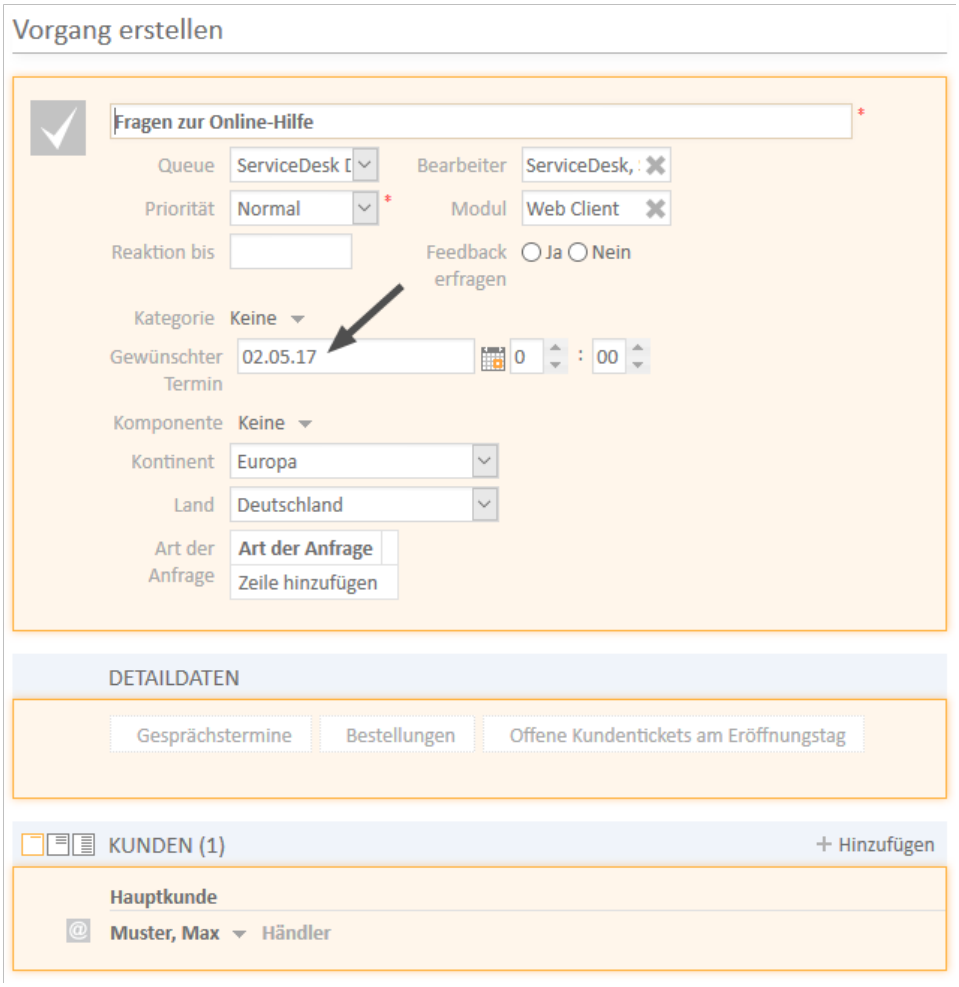


Abbildung 465: *ConSol CM Web Client - Duplizieren des Tickets (ohne Duplizieren-Skript)*

Wenn ein Ticket dupliziert wird, werden die folgenden Daten aus dem ursprünglichen Ticket übertragen (vergleichen Sie die beiden obigen Abbildungen):

- das Thema des Tickets
- die Queue
- der Bearbeiter (sofern gesetzt)
- die Werte aller Ticketfelder (Ticketdatenbereich und Bereich für Detaildaten)
- der Hauptkunde
- Zusatzkunden

Wenn ein Ticket dupliziert wird, werden die folgenden Daten **nicht** aus dem ursprünglichen Ticket übertragen (vergleichen Sie die beiden obigen Abbildungen):

- der gesamte Tickettext:
  - Kommentare
  - E-Mails
  - Attachments
- das Ticketprotokoll
- zusätzliche Bearbeiter
- verknüpfte Tickets

Falls die Queue, in der sich das Ticket befindet, ein *Duplizieren-Skript* verwendet (siehe Abschnitt Queue-Verwaltung), wird dieses Skript ausgeführt, wenn der Bearbeiter auf *Duplizieren* klickt. Das Skript kann ähnlich wie ein *Standardwerte-Skript* eingesetzt werden (siehe folgender Abschnitt): Sie können Werte im zu erstellenden Ticket vorbelegen. Während der Duplizierung, d. h. vor der Erzeugung des Tickets, werden die Ticketfelder im Web Client mit diesen Werten gefüllt. Der Bearbeiter kann die Werte bei Bedarf ändern.

Denken Sie daran, dass Sie bei einem Duplizieren-Skript nicht im Workflow-Kontext arbeiten! Das heißt, das Objekt `workflowApi` (Implementierung von `WorkflowContextService`) ist nicht verfügbar!

Im folgenden Beispiel wird das *Duplizieren-Skript* zum Zurücksetzen des Datenfeldes *Gewünschter Termin* verwendet, um zu vermeiden, dass (duplizierte) *ServiceDesk-Tickets* ein falsches Service-Datum enthalten.

```
ticket.set("serviceDesk_fields.desiredDeadline", null)
```

Code-Beispiel 86: *Duplizieren-Skript zum Zurücksetzen des Ticketfelds für den gewünschten Termin*

Wenn das Skript der Queue zugewiesen ist (in diesem Beispiel *ServiceDesk*), ist das Feld für den gewünschten Termin leer, wenn das duplizierte Ticket dem Bearbeiter angezeigt wird (siehe folgende Abbildung).

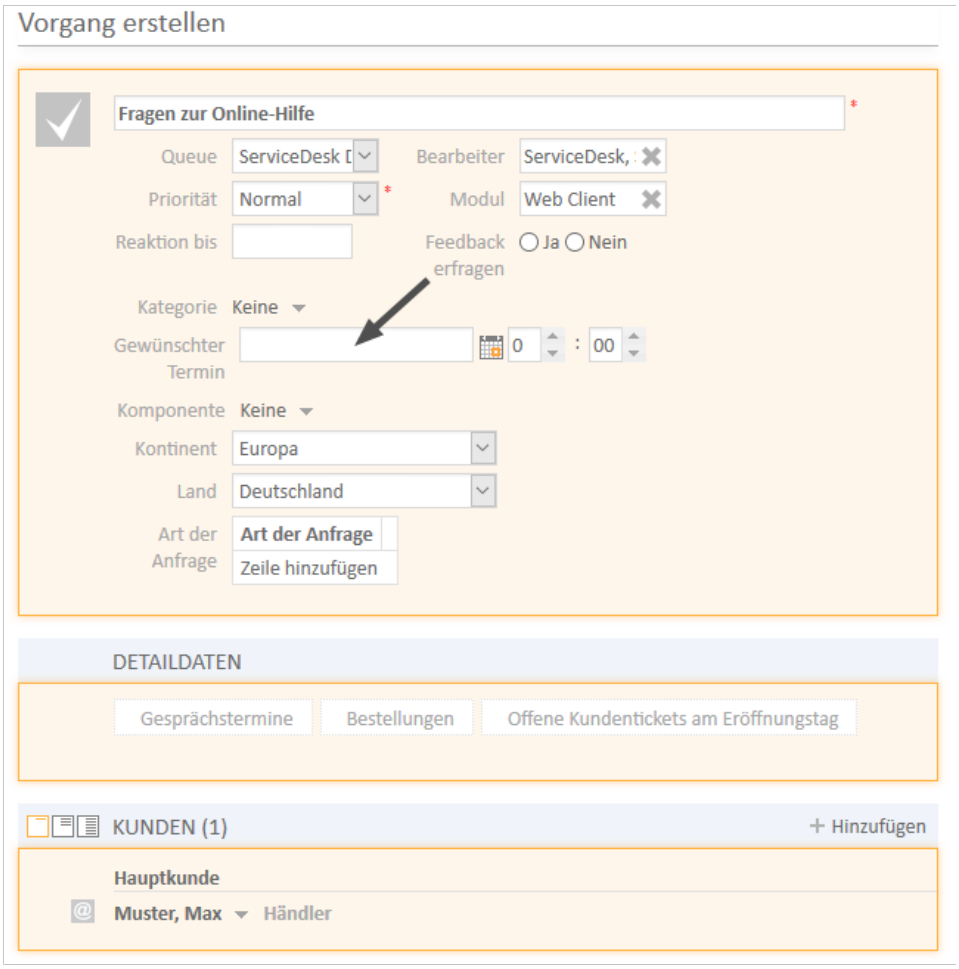


Abbildung 466: *ConSol CM Web Client - Dupliziertes Ticket (mit der Queue zugewiesenem Duplizieren-Skript)*

F.8.1.8 Skripte des Typs Feld-Visualisierung

Einleitung

Skripte des Typs *Feld-Visualisierung* ermöglichen die Anpassung von mehreren GUI-Aspekten im Web Client und CM/Track V2. Mögliche Anwendungsfälle sind:

- Organisieren von Datenfeldern in mehreren Gruppen mit Überschriften
- Ändern der Hintergrundfarbe von bestimmten Datenfeldern zum Hervorheben von wichtigen Informationen
- Hinzufügen von Bildern als Feldinhalt

- Hinzufügen von Karten als Feldinhalt
- Hinzufügen von Highcharts-Widgets als Feldinhalt

Diese Funktion kann für Datenfelder in Ticket-, Kunden- und Ressourcenobjekten im Web Client und für Datenfelder in Ticketobjekten in CM/Track V2 verwendet werden.

Konfigurieren eines Bildfeldes mit skriptbasierter Visualisierung

Im folgenden Beispiel wird die Anzeige eines Bildes auf der Kontaktseite des Web Clients implementiert. Der Rand um das Bild und die Formatierung der Beschriftung werden in einer angepassten CSS-Datei definiert.

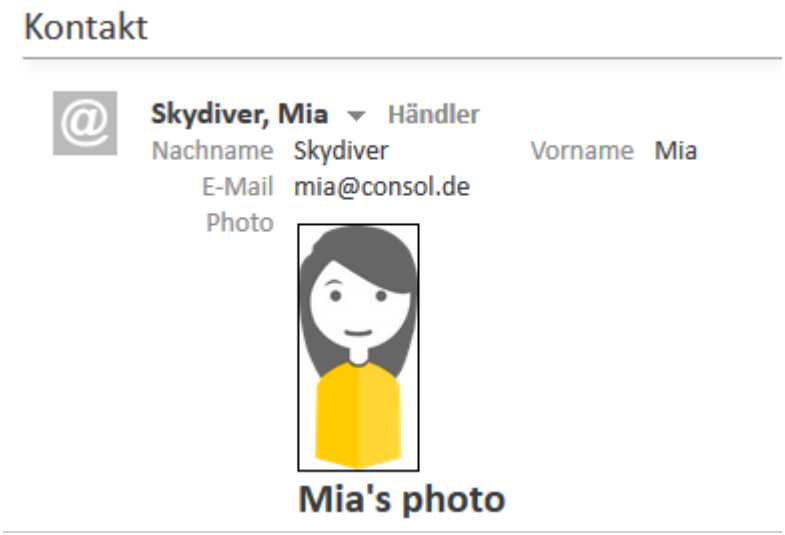


Abbildung 467: *ConSol CM Web Client - Kundenfeld mit Bild*

Führen Sie folgende Schritte aus, um solch ein Bild zu konfigurieren.

Schritt 1: Ressourcen bereitstellen

Zuerst müssen Sie die erforderlichen Ressourcen bereitstellen. Je nach Art der Visualisierung können das Bilder, Stylesheets, JavaScript-Dateien oder andere Dateien sein. Die Ressourcen werden im Dateisystem gespeichert. Alternativ können sie auch über URLs abgerufen werden.

Um die Ressourcen im Dateisystem zu speichern, erstellen Sie einen neuen Ordner `resources` im ConSol CM-Datenverzeichnis und speichern Sie die benötigten Ressourcen in diesem Ordner. In diesem Beispiel enthält der Ressourcenordner Unterordner für die beiden Arten von Unit (Kontakt oder Firma). In jedem Unterordner gibt es wiederum Unterordner, die den Namen der ID der jeweiligen Unit haben. Im ID-Unterordner gibt es ein Bild mit dem Namen `person.jpg`. Zusätzlich gibt es die Datei `person.css` im Ordner `resources`. Diese Datei enthält die Layout-Informationen für die Darstellung von Bildrand und Beschriftung.

Im aktuellen Beispiel arbeiten wir mit der folgenden Verzeichnisstruktur (in der Abbildung sind die Bilder von zwei Kontakten verfügbar):

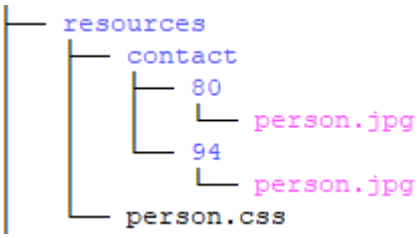


Abbildung 468: *Beispiel für Verzeichnisstruktur des Ressourcenverzeichnisses für die skriptbasierte Feldvisualisierung*

Schritt 2: Skript schreiben

Jetzt müssen Sie ein Skript des Typs *Feld-Visualisierung* erstellen, das die Logik der Visualisierung implementiert. Das Skript enthält zwei Methoden `resources()` und `render()`. Die Methode `resources()` ruft die Ressourcen aus dem Dateisystem oder einer URL ab und die Methode `render()` stellt die Logik für die Darstellung des Feldwertes zur Verfügung.

Die Methoden haben mehrere Parameter:

- **pContext**: Objekt, aus dem das Skript aufgerufen wird
- **pFieldKey**: Schlüssel des Feldes, das das Skript aufruft
- **pFieldValue**: Wert des Feldes
- **pClient**: Art des Clients, kann „web“, „track“ oder „rest“ sein

Mit diesen Parametern können Sie zum Beispiel das Verhalten an den Feldwert anpassen oder ein unterschiedliches Verhalten für den Web Client und CM/Track V2 konfigurieren.

Das folgende Skript implementiert die Anzeige eines Bildes, wie in der obigen Abbildung gezeigt.

```
import com.consol.cmas.common.model.customfield.meta.FieldKey;
import
 com.consol.cmas.common.model.customfield.visualization.FieldVisualizationContext;
// ######### renderCustomerImages.groovy ##############
/**
* It returns simple html document which will be used to render custom field in view
 mode.
*
* All additional resources like css, js, or image files need to be enumerated in
 the resources() method.
* Js or css file will be automatically registered on html page.
* To use other files simply use their names, they will be replaced with correct
 html url.
*
* @param pFieldKey - the field key
* @param pFieldValue - the field value
* @param pClient - the client type for which render method is called(e.g.: track,
 web)
```

```
* @param pContext - it provides id of an object from which given field comes from
*/
def render(FieldKey pFieldKey, Object pFieldValue, String pClient,
 FieldVisualizationContext pContext) {
   def path = "${pContext.getUnit().getDefinition().getType().name().toLowerCase
    ()}/${pContext.getUnit().getId()}/person.jpg";
   return """
   <div class='photoBox'>
   <img src='${path}' class='person'>
   <h2>${pFieldValue}<h2>
   </div>
   """ as String
}
/**
* It returns list of resources which are required to visualize this custom field.
* Any files from {data.directory}/resources or external CDN server resource can be
 used here.
*
* All js or css files will be automatically registered on html page.
* To use other files simply use their names in render() method, they will be
 replaced with correct html url.
*
* data.directory - is the server shared directory configured in CM property
*
* @param pFieldKey - the field key
* @param pFieldValue - the field value
* @param pClient - the client type for which render method is called(e.g.: track,
 web)
* @param pContext - it provide id of object from which given field comes from
*/
def resources(FieldKey pFieldKey, Object pFieldValue, String pClient,
 FieldVisualizationContext pContext) {
   def path = "${pContext.getUnit().getDefinition().getType().name().toLowerCase
    ()}/${pContext.getUnit().getId()}/person.jpg";
   List<String> resources = [
      "person.css",
      path
   ] as String[];
   log.info resources;
   return resources;
}
```

Code-Beispiel 87: *Feldvisualisierungsskript zum Anzeigen eines Bildes auf der Kundenseite*

Schritt 3: Datenfeld konfigurieren

Um die Konfiguration abzuschließen, erstellen Sie ein Datenfeld (Ticket-, Kunden- oder Ressourcenfeld), z. B. des Typs „String“, und geben Sie den Namen des Skripts als Wert der Annotation `common, visualization` ein. Positionieren Sie das Feld auf der GUI des Web Clients über die Pfeile nach oben/unten bzw. die Annotation `position`. Entscheiden Sie, ob das Bild auch dann

angezeigt werden soll, wenn das Feld leer ist. Wenn dies gewünscht ist, weisen Sie dem Datenfeld die Annotation `common, visualize-when-empty` zu. Andernfalls wird das Bild nur angezeigt, wenn das Feld Inhalt hat.

> ⚠ Die Annotation `common, visualize-when-empty` wird nur für die skriptbasierte Feldvisualisierung im Web Client verwendet, nicht in CM/Track V2.

Konfigurieren einer kontextabhängigen Hervorhebung des Inhalts eines Datenfeldes

Im folgenden Beispiel wird ein Ticketfeld, das die Anzahl der offenen Tickets des Hauptkunden enthält, hervorgehoben. Wenn die Anzahl kleiner als 10 ist, wird das Feld mit einem grünen Hintergrund angezeigt, wenn die Anzahl größer ist, ist der Hintergrund rot.

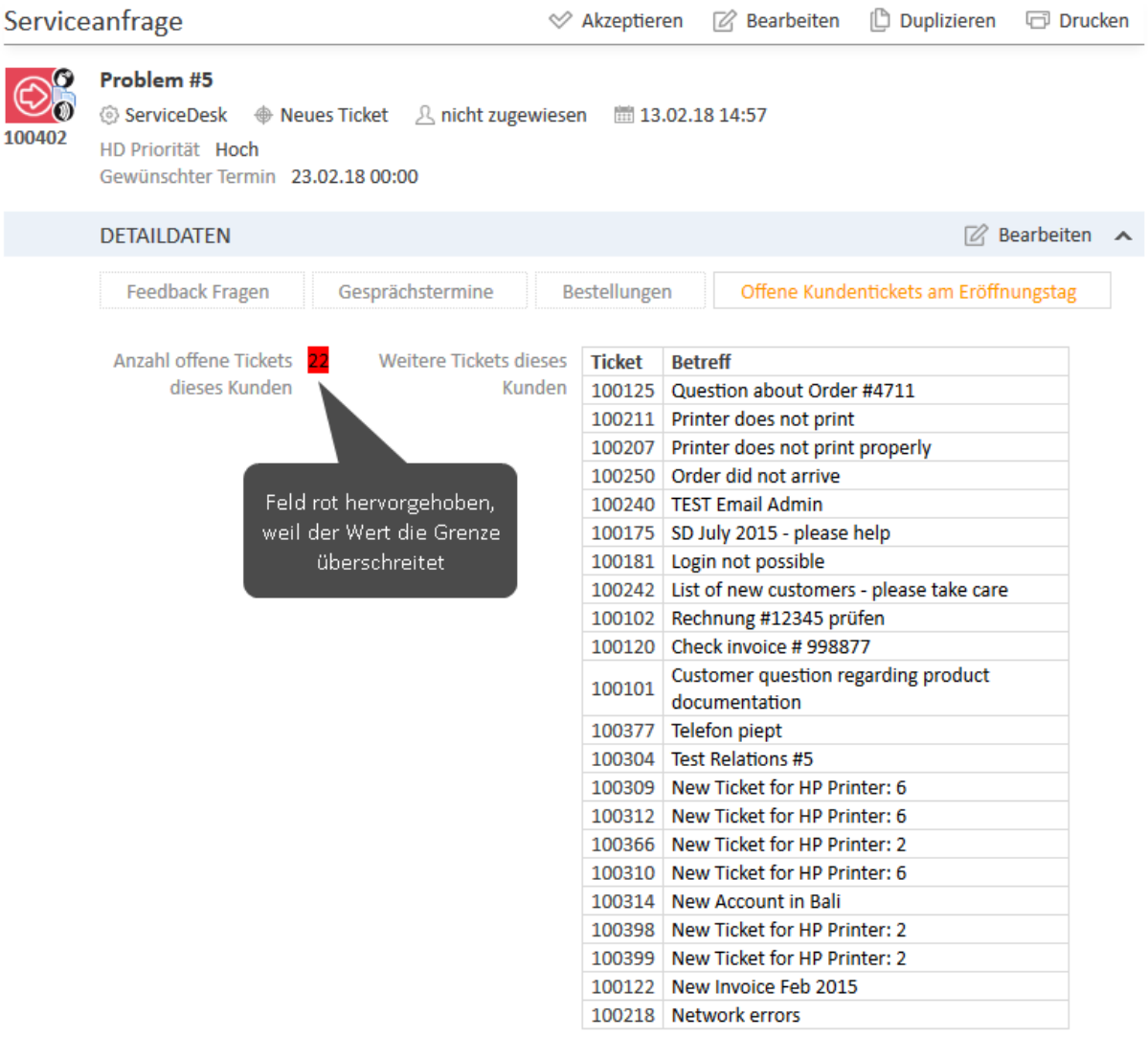


Abbildung 469: *ConSol CM Web Client - Feld mit Zahl hervorgehoben, weil der Wert die Grenze überschreitet*

Schritt 1: Ressourcen bereitstellen

Es ist lediglich eine .css-Datei erforderlich. In diesem Beispiel heißt sie `highlight.css` und ist sehr einfach aufgebaut:

```
.red {
   background-color:red;
}
.green {
   background-color:green;
}
.default {
   background-color:inherit;
}
```

Code-Beispiel 88: *Beispiel für Datei hightlight.css*

Schritt 2: Skript schreiben

```
import com.consol.cmas.common.model.customfield.meta.FieldKey;
import
 com.consol.cmas.common.model.customfield.visualization.FieldVisualizationContext;
// ################ highlightCriticalTicketNumbers.groovy ##############
/**
* It returns simple html document which will be used to render custom field in view
 mode.
*
* All additional resources like css, js, or image files need to be enumerated in
 the resources() method.
* Js or css file will be automatically registered on html page.
* To use other files simply use their names, they will be replaced with correct
 html url.
*
* @param pFieldKey - the field key
* @param pFieldValue - the field value
* @param pClient - the client type for which render method is called(e.g.: track,
 web)
* @param pContext - it provides id of an object from which given field comes from
*/

def render(FieldKey pFieldKey, Object pFieldValue, String pClient,
 FieldVisualizationContext pContext) {
   def highlight = "default";
   if(pClient.equalsIgnoreCase("track")) {
      return null
   } else {
      if (pFieldValue < 10) {
         highlight = "green";
      } else {
         highlight = "red";
      }
```

```
      return """
      <div>
         <span class="${highlight}">${pFieldValue}</span>
      </div>
      """ as String
   }
}
/**
* It returns list of resources which are required to visualize this custom field.
* Any files from {data.directory}/resources or external CDN server resource can be
 used here.
*
* All js or css files will be automatically registered on html page.
* To use other files simply use their names in render() method, they will be
 replaced with correct html url.
*
* data.directory - is the server shared directory configured in CM property
*
* @param pFieldKey - the field key
* @param pFieldValue - the field value
* @param pClient - the client type for which render method is called(e.g.: track,
 web)
* @param pContext - it provide id of object from which given field comes from
*/
def resources(FieldKey pFieldKey, Object pFieldValue, String pClient,
 FieldVisualizationContext pContext) {
   List<String> resources = [
      "highlight.css"
   ] as String[];
   return resources;
}
```

Code-Beispiel 89: *Admin-Tool-Skript des Typs Feld-Visualisierung, zur Hervorhebung von bestimmten Anzeigewerten in einem Ticketfeld verwendet*

Schritt 3: Datenfeld konfigurieren

Um die Konfiguration abzuschließen, erstellen Sie ein Ticketfeld des Typs „number“ und geben Sie den Namen des Skripts als Wert der Annotation `common`, `visualization` ein. Positionieren Sie das Feld auf der GUI des Web Clients über die Pfeile nach oben/unten bzw. die Annotation `position`.

Konfigurieren der Anzeige von Grafiken basierend auf der Highcharts-Bibliothek

In Datenfeldern können Grafiken, wie statistische Daten oder kleine Berichte, angezeigt werden. Die Anzeige basiert auf der Bibliothek ***Highcharts***. Als Administrator müssen Sie lediglich das Datenfeld (Typ *visualization*) erstellen und ein Skript des Typs *Feld-Visualisierung* schreiben, in dem die Bibliothek ***Highcharts*** zum Anzeigen der im Skript abgerufenen Werte verwendet wird.

Im folgenden Beispiel zeigt ein Kundenfeld, das sich im Gruppenbereich der Kundenseite befindet, Kundenstatistiken an. Im Beispielskript werden feste Werte verwendet. Im tatsächlichen Einsatz würden Sie mit Werten arbeiten, die in Echtzeit aus tatsächlichen Daten abgerufen werden.

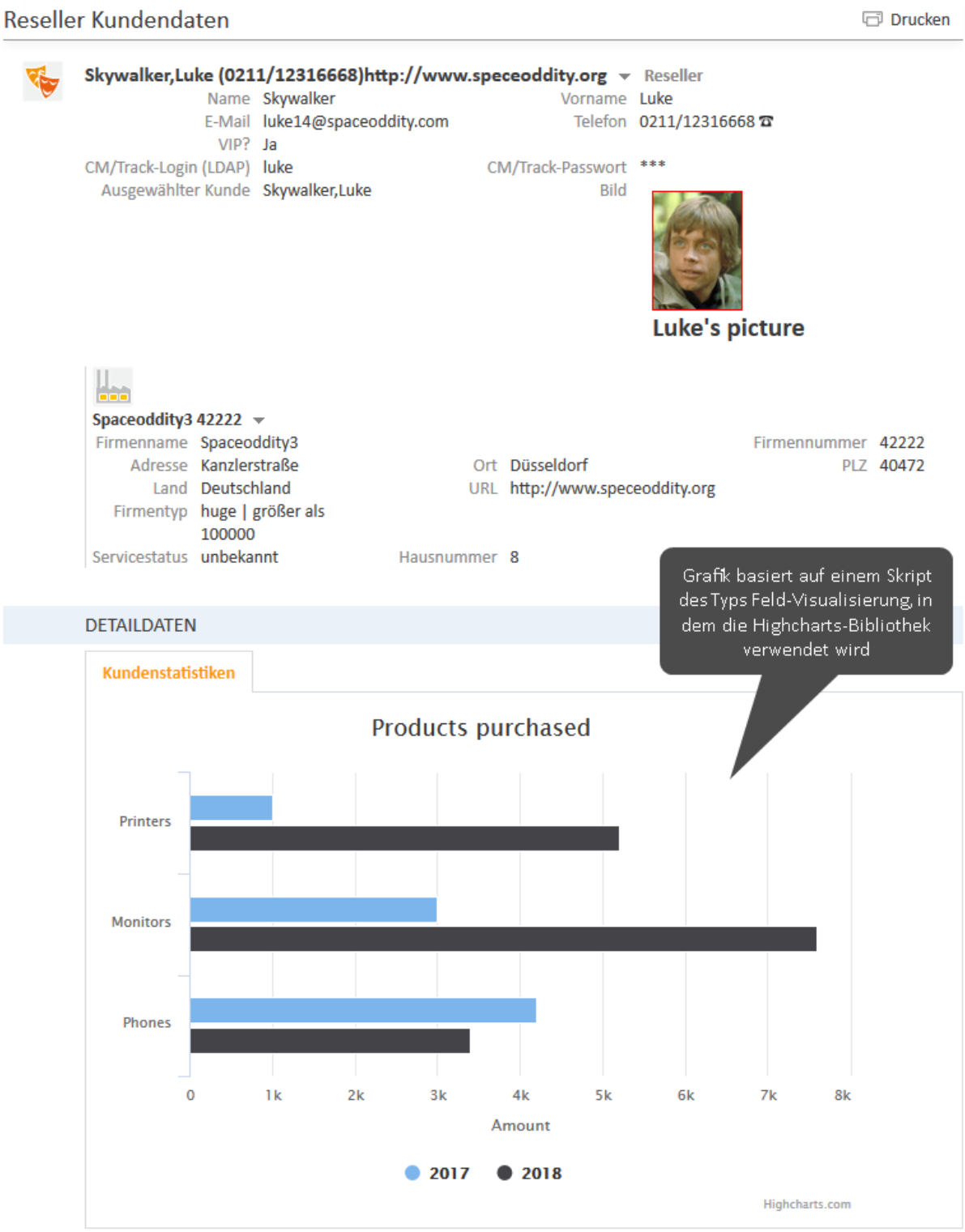


Abbildung 470: *Mit visualization annotiertes Kundenfeld im Gruppenbereich der Kundendaten*

Schritt 1: Skript schreiben

```
import com.consol.cmas.common.model.customfield.meta.FieldKey;
import
 com.consol.cmas.common.model.customfield.visualization.FieldVisualizationContext;
/**
* It returns simple html document which will be used to render custom field in view
 mode.
*
* All additional resources like css, js, or image files need to be enumerated in
 the resources() method.
* Js or css file will be automatically registered on html page.
* To use other files simply use their names, they will be replaced with correct
 html url.
*
* @param pFieldKey - the field key
* @param pFieldValue - the field value
* @param pClient - the client type for which render method is called(e.g.: track,
 web)
* @param pContext - it provides id of an object from which given field comes from
*/
def render(FieldKey pFieldKey, Object pFieldValue, String pClient,
 FieldVisualizationContext pContext) {
  return """
  <div id="container" style="width:100%; height:400px;"></div>
  <script>
  \$(function () {
     var myChart = Highcharts.chart('container', {
        chart: {
           type: 'bar'
        },
        title: {
           text: 'Products purchased'
        },
        xAxis: {
           categories: ['Printers', 'Monitors', 'Phones']
        },
        yAxis: {
           title: {
              text: 'Amount'
           }
        },
        series: [{
           name: '2017',
           data: [1000, 3000, 4200]
        }, {
           name: '2018',
           data: [5200, 7600, 3400]
        }]
     });
  });
</script>
""" as String
}
```

```
/**
* It returns list of resources which are required to visualize this custom field.
* Any files from {data.directory}/resources or external CDN server resource can be
 used here.
*
* All js or css files will be automatically registered on html page.
* To use other files simply use their names in render() method, they will be
 replaced with correct html url.
*
* data.directory - is the server shared directory configured in CM property
*
* @param pFieldKey - the field key
* @param pFieldValue - the field value
* @param pClient - the client type for which render method is called(e.g.: track,
 web)
* @param pContext - it provide id of object from which given field comes from
*/
def resources(FieldKey pFieldKey, Object pFieldValue, String pClient,
 FieldVisualizationContext pContext) {
   List<String> resources = [
      "https://code.highcharts.com/highcharts.js"
   ] as String[];
   return resources;
}
```

Code-Beispiel 90: *Admin-Tool-Skript des Typs Feld-Visualisierung für die Verwendung von statistischen Daten basierend auf der Highcharts-Bibliothek*

Schritt 2: Datenfeld konfigurieren

Erstellen Sie ein Kundenfeld wie im nachfolgenden Beispiel gezeigt. Geben Sie den Namen des entsprechenden Skripts des Typs *Feld-Visualisierung* an. Weisen Sie dem Datenfeld die Annotation `common, visualize-when-empty` zu. Andernfalls wird das Chart nur angezeigt, wenn das Feld Inhalt hat.

⚠ Die Annotation `common, visualize-when-empty` wird nur für die skriptbasierte Feld-visualisierung im Web Client verwendet, nicht in CM/Track V2.

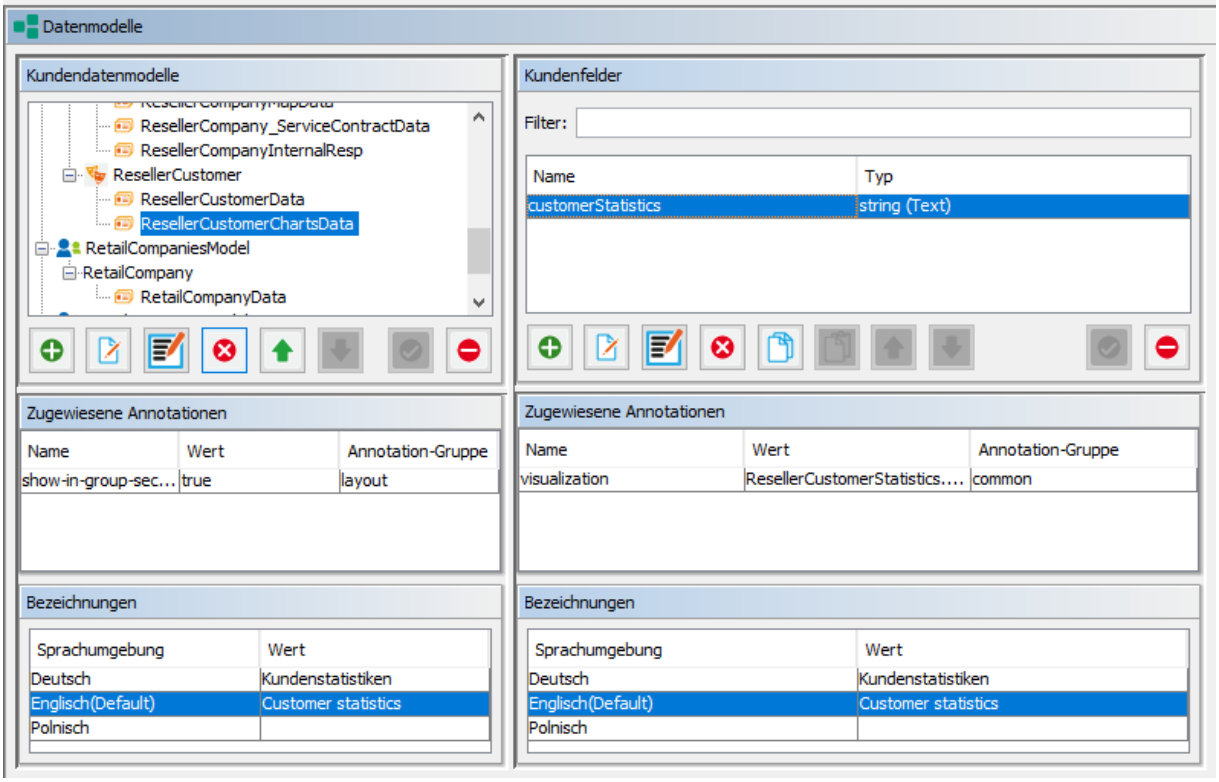


Abbildung 471: *ConSol CM Admin Tool - Kunden, Datenmodelle: Kundenfeld für die Feldvisualisierung*

Konfigurieren einer Ortsangabe auf einer Karte

Sie können Karten, z. B. von OpenStreetMap©, integrieren, um Orte auf einer interaktiven Karte anzuzeigen. Dies kann sehr hilfreich sein, um die Lage des Kundenbetriebs oder den Standort von Maschinen (wenn zum Beispiel Maschinen in Ihrem CM/Resource Pool verwaltet werden) anzuzeigen.

Im folgenden Beispiel wird ein Firmenfeld vom Typ *String* als Ausgangspunkt für die Anzeige einer interaktiven Karte verwendet. Der Standort des Kunden muss in den entsprechenden Firmenfeldern gespeichert sein: *Adresse*, *Hausnummer*, *PLZ* und *Ort*.

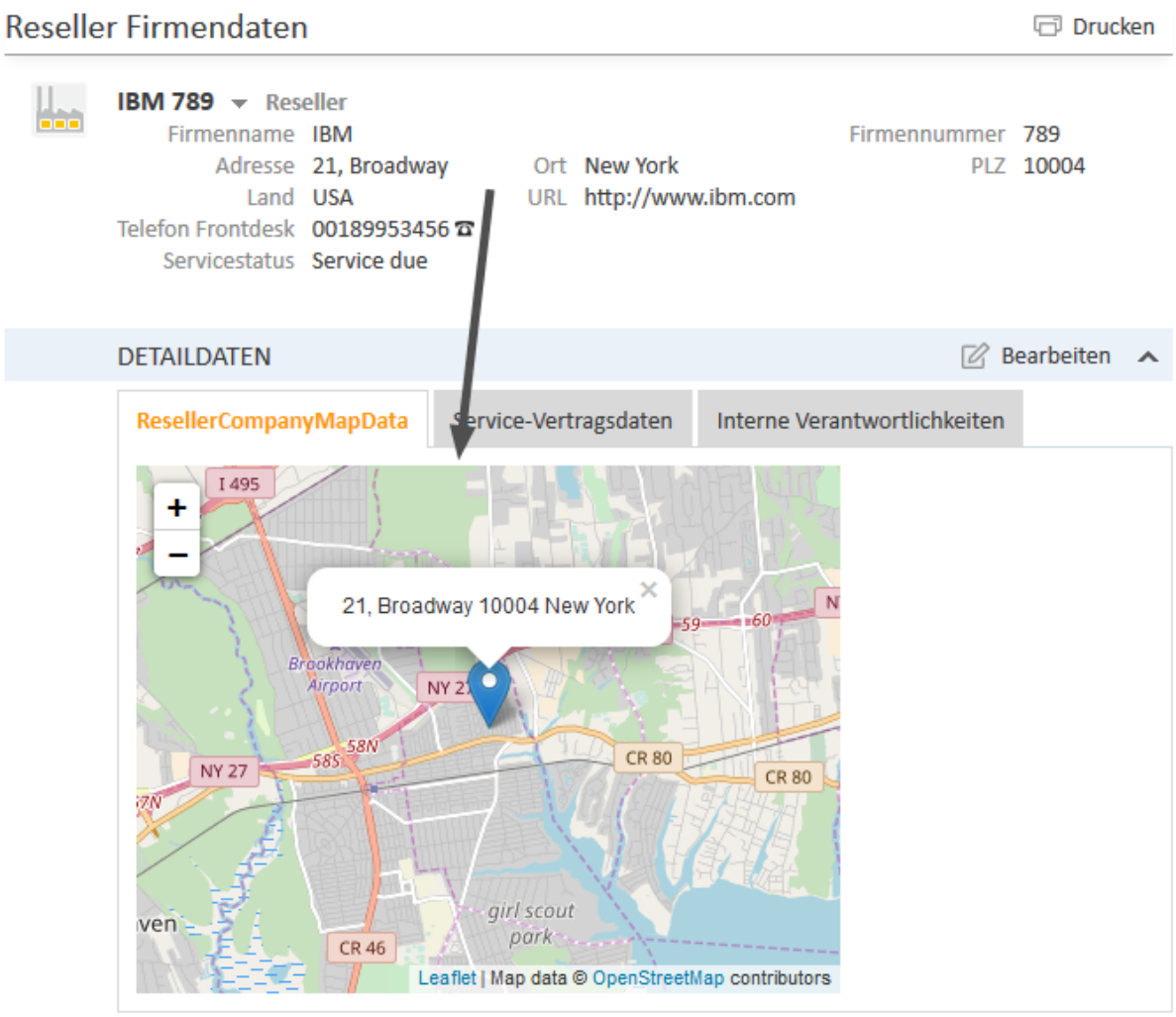


Abbildung 472: *ConSol CM Web Client - Interaktive Karte (OpenStreetMap), auf der die Lage eines Kundenstandorts angezeigt wird*

Schritt 1: Ressourcen bereitstellen

Es sind folgende Dateien erforderlich. Wenn Sie bei der Implementierung Unterstützung benötigen, fragen Sie bitte Ihren ConSol CM-Consultant.

```
└── resources
    ├── maps
    │   ├── images
    │   │   ├── marker-icon.png
    │   │   └── marker-shadow.png
    │   ├── leaflet.css
    │   └── leaflet.js
```

Abbildung 473: *Verzeichnisstruktur und erforderliche Dateien für die Kartenimplementierung als skriptbasierte Feldvisualisierung*

Schritt 2: Skript schreiben

```
import com.consol.cmas.common.model.customfield.meta.FieldKey;
import
 com.consol.cmas.common.model.customfield.visualization.FieldVisualizationContext;

log.info ' Maps visualization script started ...'

// Maps configuration
mapsDivHeight = 300
mapsDivWidth = 400
initialZoom = 12 //a value between 1 and 20


//String msg shown when we do not have enough address data to construct the map,
 customize at own will
notEnoughAddressDataMsg = messageProviderService.getMessage
 ("web.map.not.enough.address.data")


/**
* Fill this method with your system's logic how the address string is collected.
 You can collect and construct it from
* single fields or maybe your visualization field already contains the proper
 string. The implementation here is
* suited for showroom where we use this script for several unit definition models.
 Adapt as needed.
*
* Expected output: address string in the form Franziskanerstraße 38, 81669 München
 or null if no address data is given
*/
String buildAddress(FieldKey pFieldKey, Object pFieldValue,
 FieldVisualizationContext pContext) {
   def unit = pContext.getUnit() //this script is suited for units only, however
    ticket and resource are possible as well
   if(!unit) return null
   def unitDefName = unit.getDefinitionName()
   def street = unit.get("ResellerCompanyData.address")
   def nr = unit.get("ResellerCompanyData.house_nr")
   def zip = unit.get("ResellerCompanyData.zip")
   def city = unit.get("ResellerCompanyData.city")

   //check if we have enough data to construct an address, we need at least city
   if(!city) return null

   return "${street?street:''} ${nr?nr:''} ${zip?zip:''} $city"
}


/**
* Returns rendered map, should work ootb when you adapted buildAddress() properly
*
* @param pFieldKey - the field key
* @param pFieldValue - the field value
```

```
* @param pClient - the client type for which render method is called(e.g.: track,
 web)
* @param pContext - it provides id of an object from which given field comes from
*/
def render(FieldKey pFieldKey, Object pFieldValue, String pClient,
 FieldVisualizationContext pContext) {
  def address = buildAddress(pFieldKey, pFieldValue, pContext)
  if(!address) return "$notEnoughAddressDataMsg" as String
  return """
  <div id="mapid" style="height: ${mapsDivHeight}px; width:
   ${mapsDivWidth}px"></div>
  <script>
  var marker = L.icon({
     iconUrl: 'maps/images/marker-icon.png',
     shadowUrl: 'maps/images/marker-shadow.png',
     iconSize: [25, 41], // size of the icon
     shadowSize: [41, 41], // size of the shadow
     iconAnchor: [12, 41], // point of the icon which will correspond to marker's
      location
     popupAnchor: [1, -34] // point from which the popup should open relative to
      the iconAnchor
  });
  function geoCode(address, callback) {
     var xmlHttp = new XMLHttpRequest();
     xmlHttp.onreadystatechange = function() {
        if (xmlHttp.readyState == 4 && xmlHttp.status == 200) callback
         (xmlHttp.responseText);
     }
     xmlHttp.open("GET", "https://nominatim.openstreetmap.org/search?q=" + address
      + "&format=json", true);
     xmlHttp.send(null);
  }
  geoCode("${address}", function(geocodeStr) {
     var geocodeJson = JSON.parse(geocodeStr)[0];
     var lat = geocodeJson.lat;
     var lon = geocodeJson.lon;
     var map = L.map(mapid);
     var osmUrl="https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png";
     var osmAttrib='Map data © <a
      href=\"http://openstreetmap.org\">OpenStreetMap</a> contributors';
     var osm = new L.TileLayer(osmUrl, {attribution: osmAttrib});
     map.addLayer(osm);
     map.setView(new L.LatLng(lat, lon), ${initialZoom});
     L.marker([lat, lon], {icon: marker}).addTo(map).bindPopup
      ("${address}").openPopup();
  });
  </script>
  """ as String
}


/**
* It returns list of resources which are required to visualize this custom field.
```

```
* Any files from {data.directory}/resources or external CDN server resource can be
 used here.
*
* All js or css files will be automatically registered on html page.
* To use other files simply use their names in render() method, they will be
 replaced with correct html url.
*
* data.directory - is the server shared directory configured in CM property
*
* @param pFieldKey - the field key
* @param pFieldValue - the field value
* @param pClient - the client type for which render method is called(e.g.: track,
 web)
* @param pContext - it provide id of object from which given field comes from
*/
def resources(FieldKey pFieldKey, Object pFieldValue, String pClient,
 FieldVisualizationContext pContext) {
   List<String> resources = ["maps/leaflet.js", "maps/leaflet.css",
    "maps/images/marker-icon.png", "maps/images/marker-shadow.png"] as String[];
   return resources;
}
```

Code-Beispiel 91: *Admin-Tool-Skript des Typs Feld-Visualisierung, zur Angabe eines Ortes in OpenStreetMap verwendet*

Denken Sie daran, eine Bezeichnung (im obigen Beispiel in der Variable `notEnoughAddressDataMsg` verwendet) zu erstellen, in der die Meldung codiert wird, die angezeigt wird, um den Bearbeiter darüber zu informieren, dass die Karte nicht angezeigt werden kann, wenn Adressdaten fehlen. Im Abschnitt Bezeichnungen finden Sie weitere Informationen über die Arbeit mit Bezeichnungen.

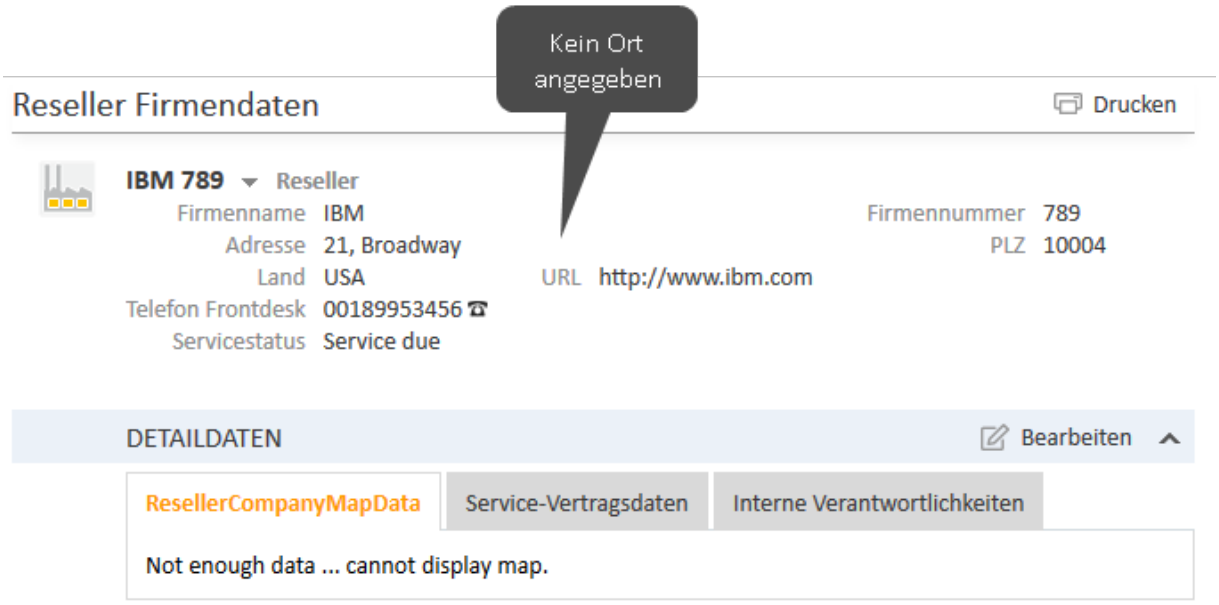


Abbildung 474: *ConSol CM Web Client - GUI-Meldung (basierend auf Bezeichnung), die angezeigt wird, wenn Adressdaten fehlen*

Datenfeld konfigurieren

Erstellen Sie ein Firmenfeld des Typs *String* und annotieren Sie es mit *visualization*. Geben Sie dabei den Namen des Skripts als Wert der Annotation an. Das Skript muss den Typ *Feld-Visualisierung* haben. Weisen Sie dem Datenfeld die Annotation `common, visualize-when-empty` zu. Andernfalls wird die Karte nur angezeigt, wenn das Feld Inhalt hat.

> ⚠ Die Annotation `common, visualize-when-empty` wird nur für die skriptbasierte Feldvisualisierung im Web Client verwendet, nicht in CM/Track V2.

F.8.1.9 Skripte des Typs *Standardwerte*

Manchmal ist es erforderlich, dass ein Datenfeld eines Tickets bei der Erstellung des Tickets einen bestimmten Wert hat, d. h. wenn der Bearbeiter auf *Neues Ticket* klickt und das entsprechende Formular im Web Client geöffnet wird, sollen ein oder mehrere Werte voreingestellt sein. Dies erspart dem Bearbeiter den Aufwand, den Wert jedes Mal setzen zu müssen, z. B. wenn die Standardpriorität *normal* ist, kann dieser Wert voreingestellt werden. Sollte *hoch* oder *niedrig* erforderlich sein, kann der Bearbeiter den Wert entsprechend ändern.

Dieses Verhalten von ConSol CM kann durch ein oder mehrere *Standardwerte-Skripte* erreicht werden. Die Standardwerte können für jede Queue einzeln definiert werden. Pro Queue kann genau ein *Standardwerte-Skript* zugewiesen werden. Siehe folgendes Beispiel.

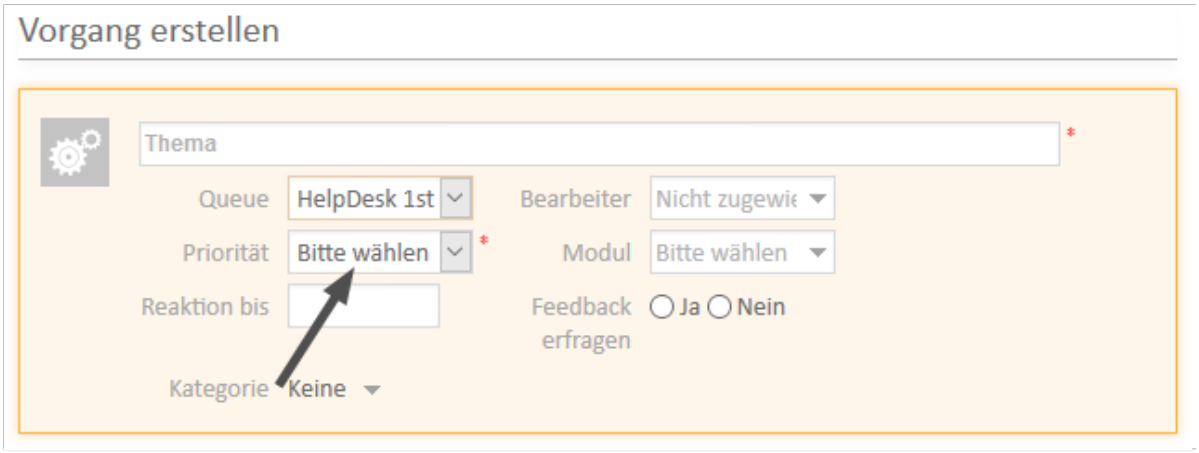


Abbildung 475: *ConSol CM Web Client - Neues Ticket ohne Standardwerte*

Ohne *Standardwerte-Skript* wird kein Wert für die Priorität gesetzt, wenn ein Bearbeiter ein neues Ticket im Web Client erstellt.

Um einen Standardwert zu definieren, muss ein Skript des Typs *Standardwerte* erstellt werden. Zuerst suchen wir in der Verwaltung der Ticketfelder das entsprechende Ticketfeld (in welcher Ticketfeldgruppe es sich befindet und welchen Namen es hat). Details dazu finden Sie im Abschnitt Verwaltung von Ticketfeldern (Einrichten des Ticketdatenmodells).

> Denken Sie daran, dass Sie bei einem Standardwerte-Skript nicht im Workflow-Kontext arbeiten! Das heißt, das Objekt `workflowApi` (Implementierung von `WorkflowContextService`) ist nicht verfügbar!

Die folgende Abbildung zeigt das Ticketfeld *priority*. Sie gelangen auf diese Seite, indem Sie das Navigationselement *Ticketfelder* in der Navigationsgruppe *Tickets* öffnen.

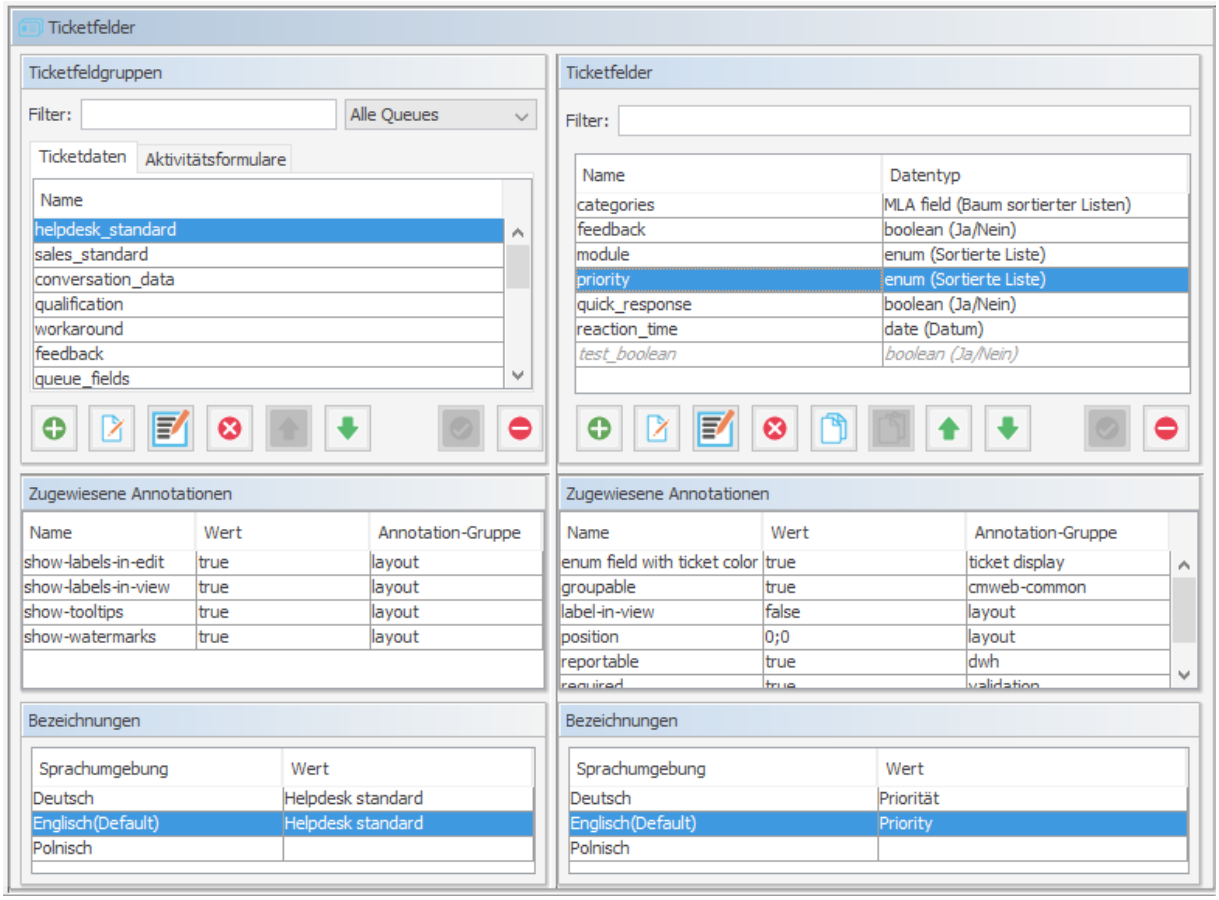


Abbildung 476: *ConSol CM Admin Tool - Tickets, Ticketfelder: priority*

Da es sich bei *priority* um ein ENUM-Feld (d. h. eine Sortierte Liste) handelt, müssen Sie die möglichen (technischen) Werte prüfen, die in der Liste auftreten können, und den gewünschten Standardwert in der Verwaltung der Sortierten Listen (Navigationsgruppe *Listen*, Navigationselement *Sortierte Listen*) suchen.

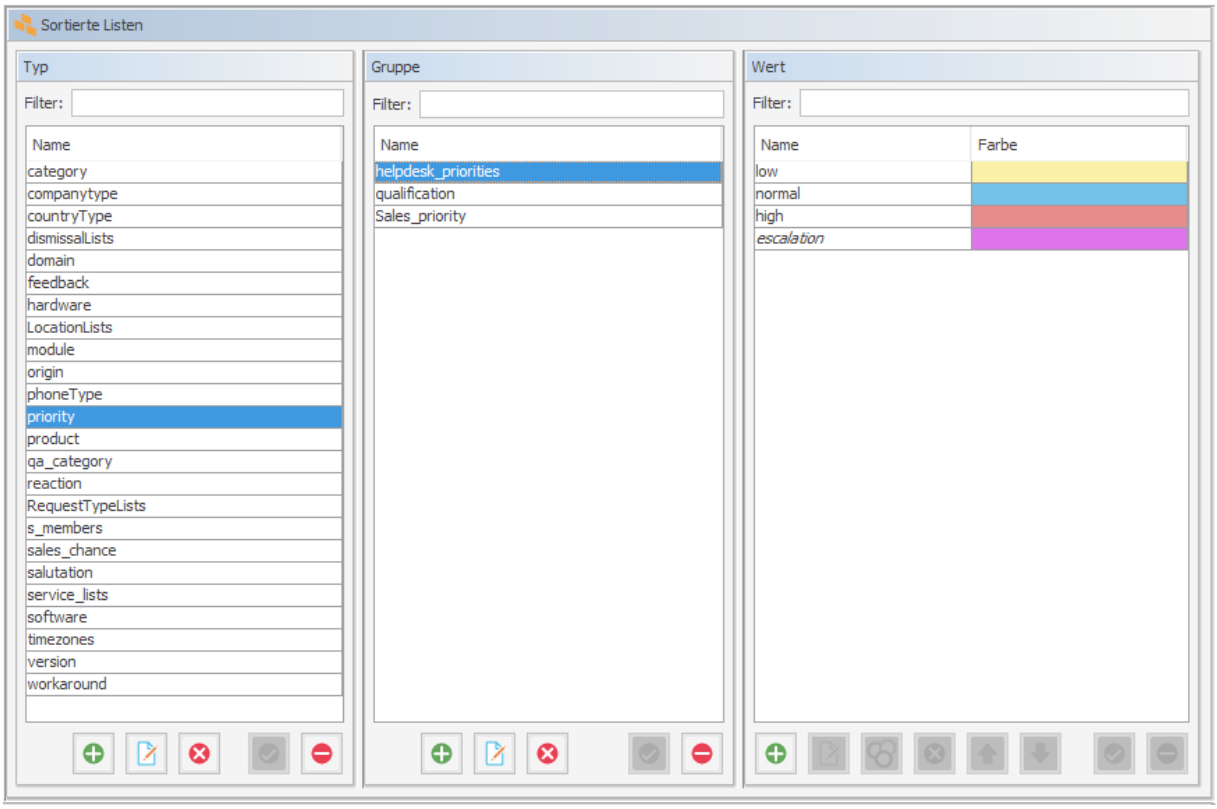


Abbildung 477: *ConSol CM Admin Tool - Listen, Sortierte Listen: Werte für die Priorität*

Die Gruppe, das Feld und der korrekte Wert können dann im neuen Skript (Navigationselement *Skripte und Templates*, Navigationsgruppe *System*) in der entsprechenden Groovy-Methode verwendet werden.

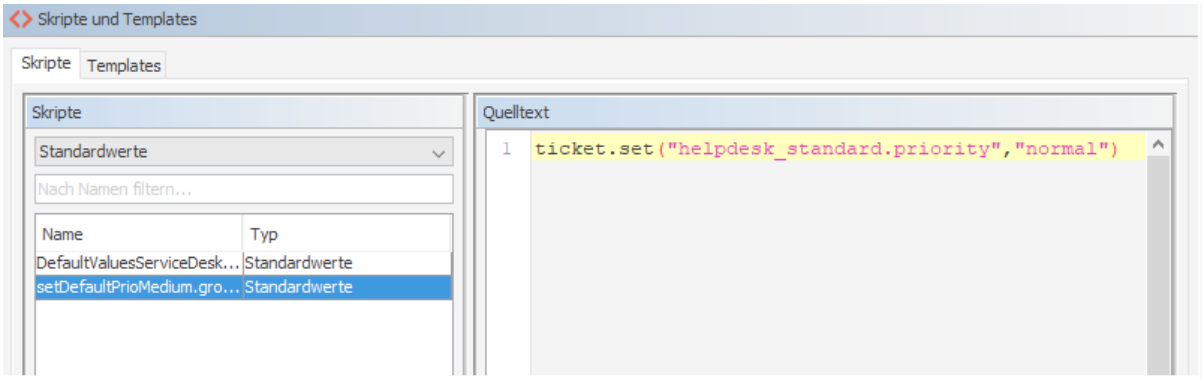


Abbildung 478: *ConSol CM Admin Tool - System, Skripte und Templates: Verwendung von Gruppe, Feld und Wert im Skript*

In der Queue-Verwaltung (Navigationsgruppe *Globale Konfiguration*, Navigationselement *Queues*) müssen wir das Skript der Queue zuweisen, in der der Standardwert verwendet werden soll.

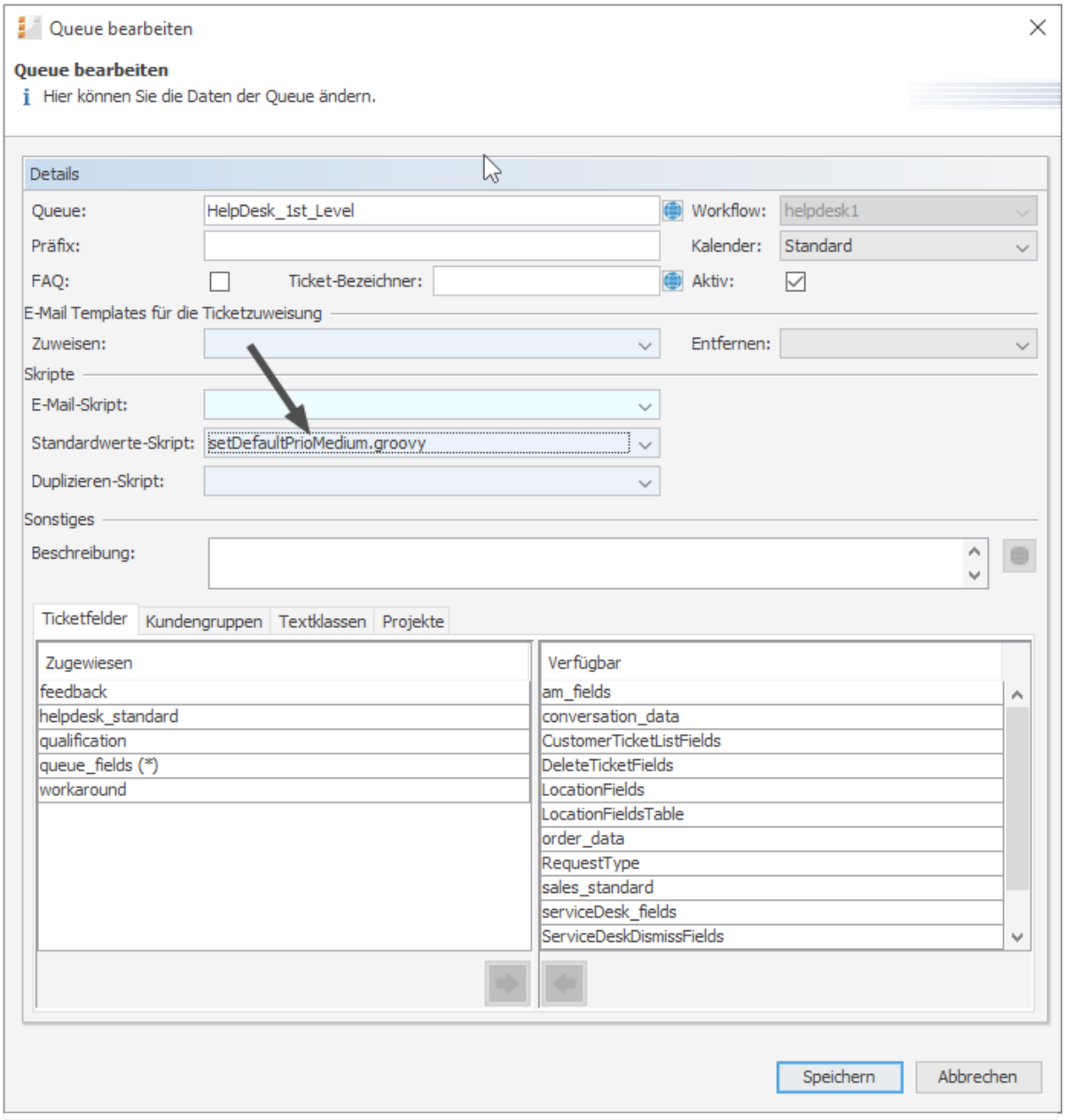


Abbildung 479: *ConSol CM Admin Tool - Globale Konfiguration, Queues: Zuweisen eines Standardwerte-Skripts zur Queue*

Wenn der Bearbeiter die Ticketerstellung jetzt im Web Client beginnt, wird der Standardwert *normal* für das Feld *Priorität* gesetzt.

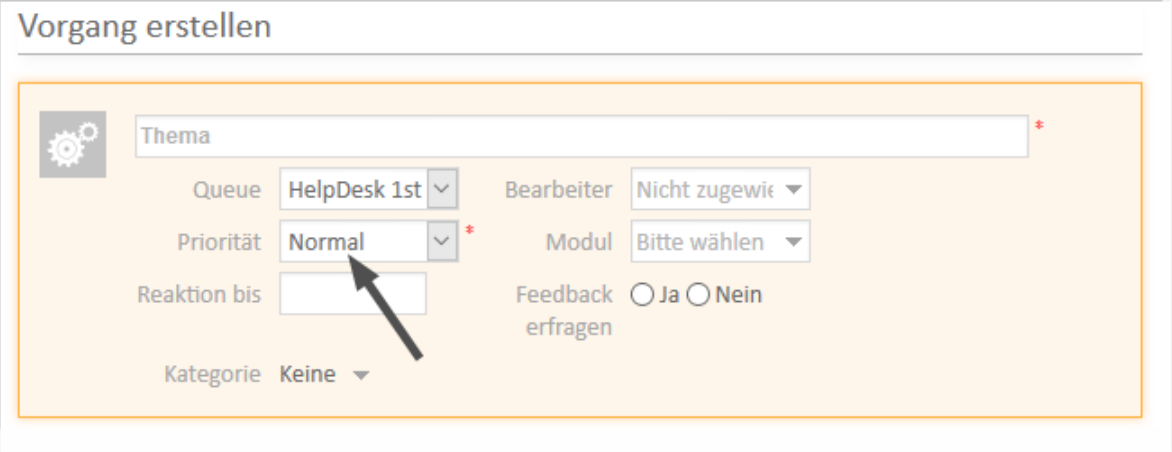


Abbildung 480: *ConSol CM Web Client - Neues Ticket mit Standardwert*

Denken Sie daran, dass es für jede Queue nur ein *Standardwerte-Skript* geben kann. Wenn Sie mehrere Standardwerte definieren möchten, müssen Sie dies in **einem** Skript tun. In diesem Fall können Sie auch den Namen des Skripts anpassen.

Wenn der gleiche Standardwert in unterschiedlichen Queues gesetzt werden soll und es noch andere Standardwerte gibt, muss dies ebenfalls in einem eigenen Skript pro Queue programmiert werden.

Überschreibmodus bei Standardwerte-Skripten

Standardmäßig werden die Felder eines Tickets, die mit dem *Standardwerte-Skript* vorausgefüllt werden, nicht überschrieben, d. h. wenn das Ticket in eine andere Queue verschoben wird, der ihr eigenes *Standardwerte-Skript* zugewiesen ist, versucht das zweite Skript Felder zu setzen, die bereits vom ersten Skript gefüllt wurden. Da dies nicht möglich ist, bleiben die Standardwerte des ersten Skripts erhalten.

Wenn ein *Standardwerte-Skript* vorhandene Werte überschreiben soll, muss der Überschreibe-Modus aktiviert werden. Um diesen Modus zu aktivieren, fügen Sie am Anfang des *Standardwerte-Skripts* folgenden Code ein:

```
import com.consol.cmas.common.model.ticket.TicketPrototypeContext
TicketPrototypeContext.enableOverwriteMode()
```

F.8.1.10 Skripte des Typs *Workflow*

Skripte dieses Typs sind im Admin Tool gespeichert, da sie in vielen Workflow-Skripten verwendet werden, d. h. der Code des Admin-Tool-Skripts wird mehr als einmal in einem oder mehreren Workflows benötigt. Es ist einfacher, weniger fehleranfällig und zeitsparender, die Skripte an einem zentralen Ort (im Admin Tool) zu speichern und sie in den Workflows nur zu referenzieren, anstatt den gleichen Code an unterschiedlichen Stellen in jedem Workflow, in dem er verwendet wird, zu editieren. Außer-

dem kann das Admin-Tool-Skript während der Workflow-Entwicklung einfach geändert werden und die Änderung ist sofort wirksam, wohingegen ein veränderter Workflow zuerst installiert werden muss.

Eine detaillierte Einführung in die Workflow-Programmierung finden Sie im *ConSol CM Process Designer Handbuch*. An dieser Stelle finden Sie ein kurzes Beispiel.

Der Code in einer Workflow-Aktivität referenziert lediglich das Skript, z. B.:

```
scriptExecutionService.execute(scriptProviderService.createDatabaseProvider
 ("DisplayCustomerData.groovy"))
```

Das entsprechende Skript wird im Admin Tool im Navigationselement *Skripte und Templates* der Navigationsgruppe *System* gespeichert:

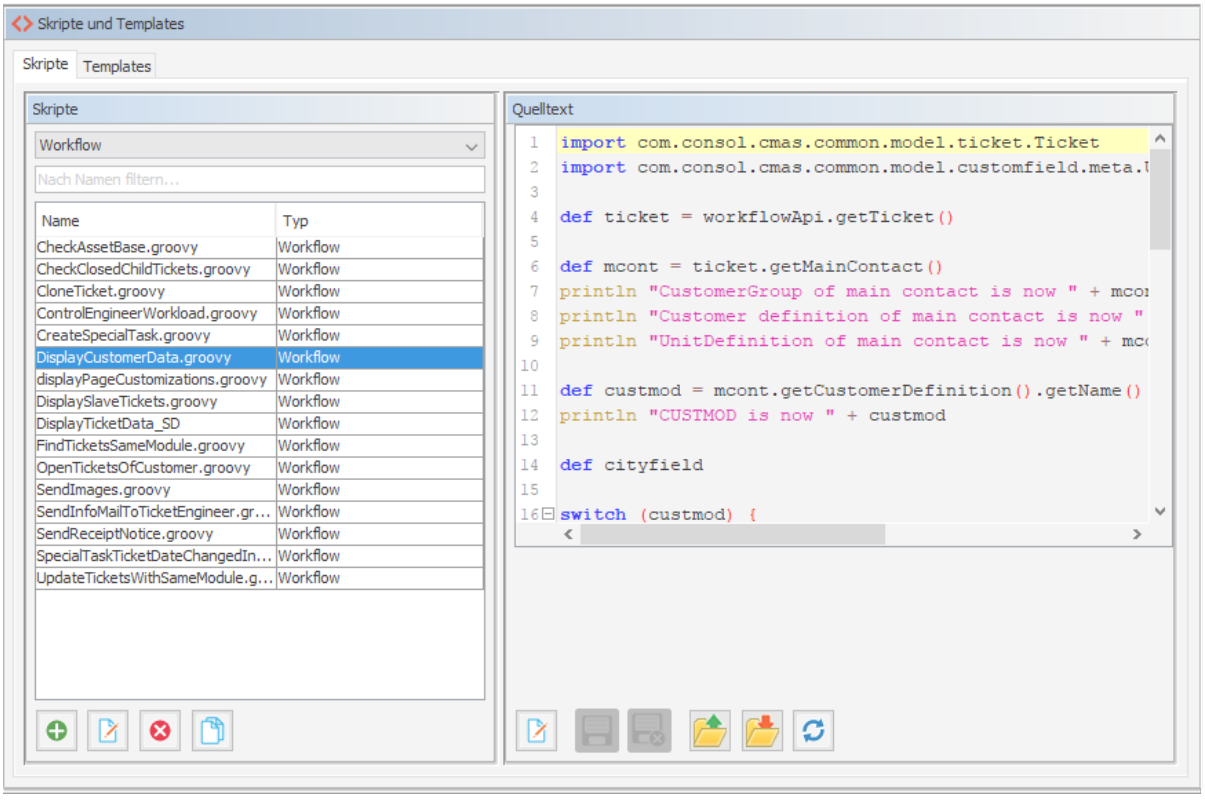


Abbildung 481: *ConSol CM Admin Tool - System, Skripte und Templates: Workflow-Skript*

Es ist auch möglich, Parameter (*key-value pairs*) an das Admin-Tool-Skript zu übergeben. Dies ist detailliert im *ConSol CM Process Designer Handbuch* beschrieben.

F.8.1.11 PostActivityExecutionScript

In bestimmten Anwendungsfällen ist es möglicherweise erforderlich, ein Skript auszuführen, nachdem ein Ticket eine Workflow-Aktivität durchlaufen hat. Diese Art von Skript kann zum Beispiel verwendet

werden, um nach der Ausführung einer Workflow-Aktivität ein anderes Ticket im Web Client anzuzeigen. Aus der Sicht des Bearbeiters „springt“ der Web Client zum nächsten Ticket. Dies kann, je nach Anwendungsfall, ein Child-Ticket oder das nächste Ticket in der Liste sein.

Das Systemverhalten wird in einem Admin-Tool-Skript definiert, dem *PostActivityExecutionScript* (manchmal auch *Standardskript für Workflow-Aktivität* genannt). Der Name des Skripts muss in der System-Property cmweb-server-adapter, postActivityExecutionScriptName gesetzt werden, siehe System-Properties.

Dieses Skript wird nach jeder **manuellen** Workflow-Aktivität ausgeführt. Dies bedeutet, dass Sie alle Kontrollmechanismen und die gesamte Intelligenz in das Skript einfügen müssen:

- Nach welcher Aktivität soll das Skript etwas machen?
  (Nach allen anderen Aktivitäten passiert nichts.)
- Was soll passieren?

Ab Version 6.10.2 kann ConSol CM mit einem PostActivityExecutionScript einen der vier unterschiedlichen Seitentypen (im Ansichtsmodus) öffnen:

- eine Ticketseite
- eine Firmendetailseite
- eine Kontaktdetailseite
- eine Ressourcenseite

Das Skript muss nur das entsprechende Objekt zurückgeben. Der folgende Code zeigt ein Beispiel für jeden der vier Typen.

```
switch(activity.name){
   case 'defaultScope/Goto_ticket':
      return ticketService.getByName("SUP-11")
   case 'defaultScope/Goto_contact':
      return unitService.getById(123)
   case 'defaultScope/Goto_company':
      return unitService.getById(456)
   case 'defaultScope/Goto_resource':
      return resourceService.getById(890)
}
```

Beispiel: Zum nächsten Ticket in einer Liste springen.

cmweb-server-adapter	pagemapLockDurationInSeconds	60
cmweb-server-adapter	postActivityExecutionScriptName	postActivityExecutionHandler
cmweb-server-adapter	queuesExcludedFromGS	

Abbildung 482: *ConSol CM Admin Tool - Property zur Definition von postActivityExecutionScriptName*

Das PostActivityExecutionScript ermöglicht es auch, zu einer Unit-Seite (d. h. einer Firmenseite oder einen Kontaktseite) oder einer Ressourcenseite zu springen, indem die Unit bzw. die Ressource im Skript zurückgegeben wird.

Siehe folgendes Beispielskript:

```
switch(activity.name){
   case 'defaultScope/Goto_the_ticket':
      return ticketService.getByName("SUP-11")
   case 'defaultScope/Goto_the_contact':
      return unitService.getById(123)
   case 'defaultScope/Goto_the_company':
      return unitService.getById(456)
   case 'defaultScope/Goto_the_resource':
      return resourceService.getById(890)
}
```

Code-Beispiel 92: *PostActivityExecutionScript*

Ein anderes **Beispiel**, das das Verhalten des Web Clients zeigt:

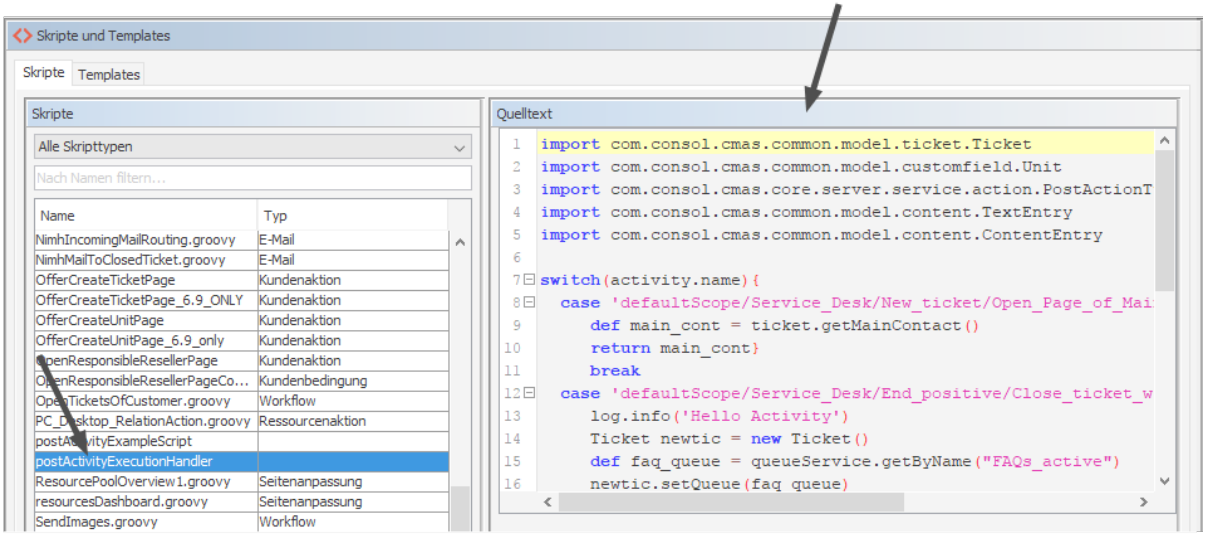


Abbildung 483: *ConSol CM Admin Tool - System, Skripte und Templates: PostActivityExecutionHandler-Skript*

```
switch(activity.name){
   case 'defaultScope/Service_Desk/New_ticket/Open_Page_of_Main_Contact':
      def main_cont = ticket.getMainContact()
         return main_cont
   // ( ... )
}
```

Code-Beispiel 93: *PostActivityExecutionHandler*

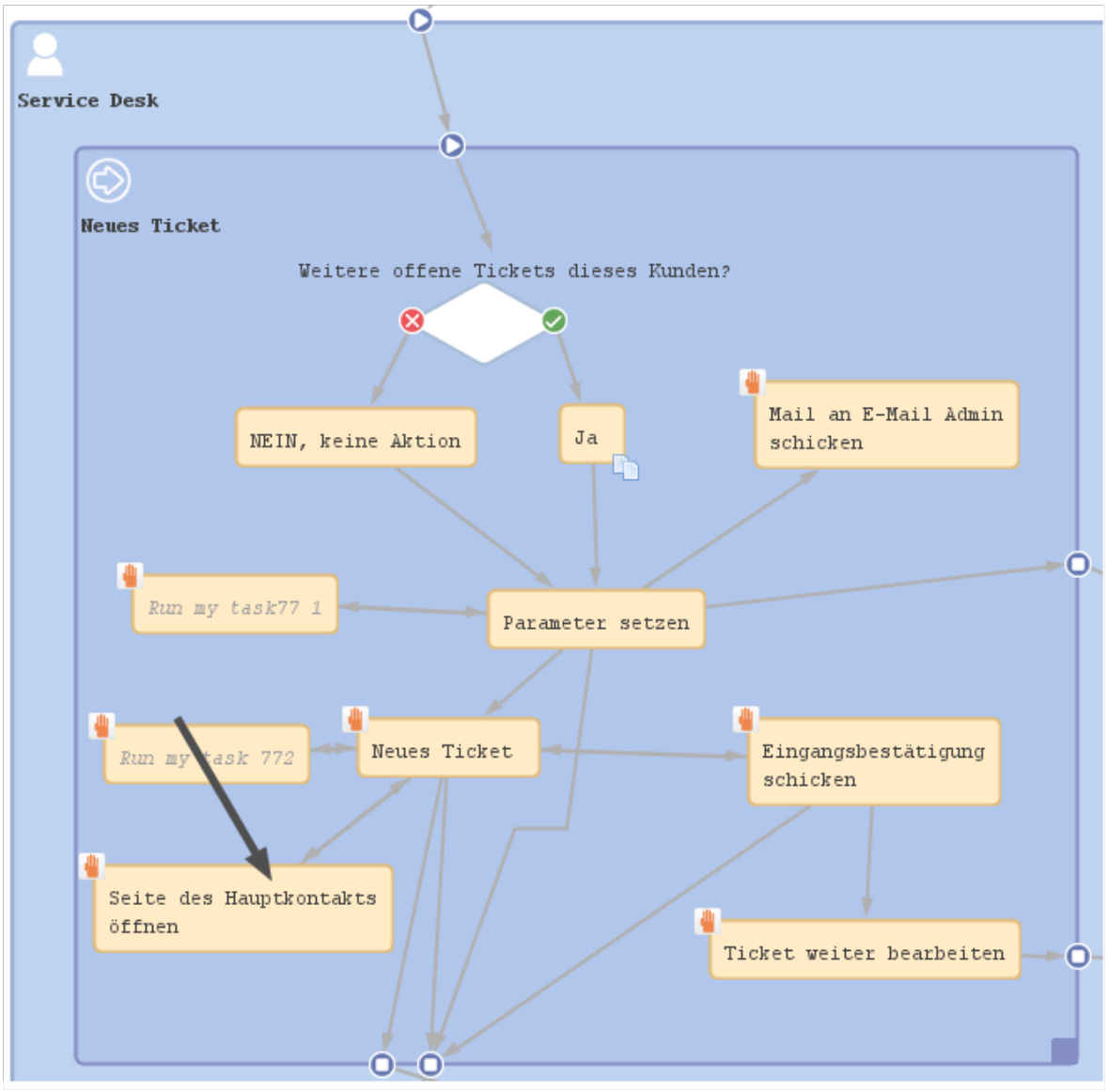


Abbildung 484: *ConSol CM Process Designer - Aktivität, die mit dem PostActivityExecutionScript gesteuert wird*

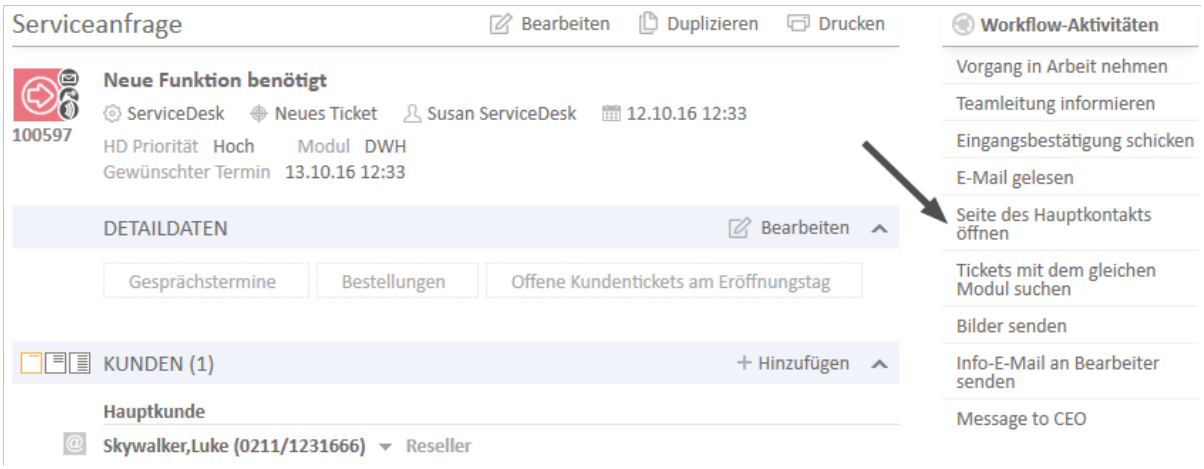


Abbildung 485: *ConSol CM Web Client - Ticket mit Workflow-Aktivität, mit der der Web Client zur Kontaktseite springt*

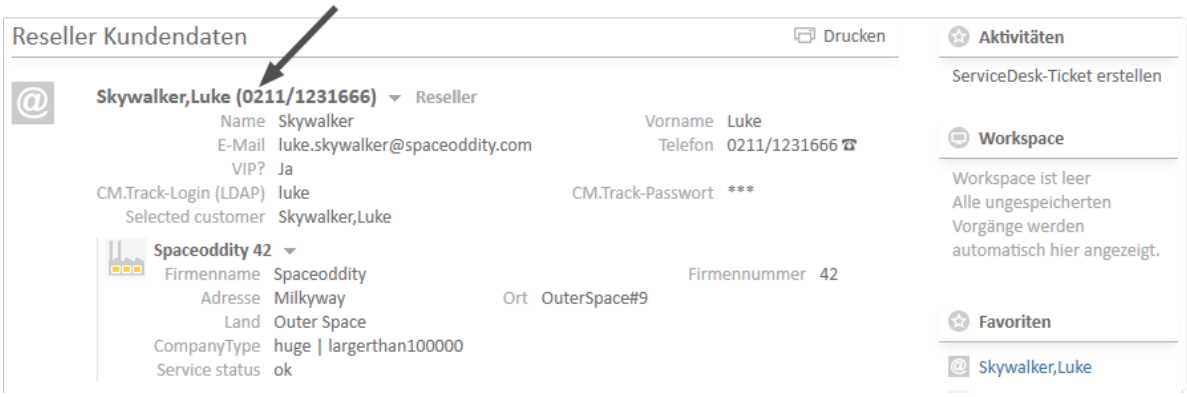


Abbildung 486: *ConSol CM Web Client - Kontaktseite, die direkt aus dem Ticket geöffnet wurde*

F.8.2 Admin-Tool-Templates

F.8.2.1 Einführung in die Templates im Admin Tool

In ConSol CM werden mehrere Arten von Templates und Vorlagen verwendet:

- **E-Mail-Vorlagen** werden an folgenden Stellen gespeichert:
  - im *Textvorlagen-Manager*, siehe Abschnitt Der ConSol CM Textvorlagen-Manager.
  - im Abschnitt *Templates* des Admin Tools. Details zu diesem Thema sind im vorliegenden Kapitel erklärt.
- **Dokumentvorlagen** werden im
  - Dokumentvorlagen-Manager (Teil von CM/Doc) gespeichert, siehe Abschnitt CM/Doc
- **Templates für die Datendarstellung** werden unter
  - *Skripte und Templates* im Admin Tool gespeichert. Sie werden verwendet zur:
    - Definition von Kunden-Templates, siehe Abschnitt Templates für Kundendaten
    - Definition von Ressourcen-Templates, siehe Abschnitt CM/Resource Pool - Templates für Ressourcendaten.
- **Allgemeine Templates** sind unter
  - *Skripte und Templates* im Admin Tool gespeichert. Sie werden im vorliegenden Kapitel erklärt.

In diesem Kapitel werden die allgemeinen Templates erklärt, die im Admin Tool unter *Skripte und Templates* gespeichert sind.

Admin-Tool-Templates sind in FreeMarker-Notation geschrieben (siehe FreeMarker-Website) und sollten nur von erfahrenen ConSol CM-Consultants und Administratoren editiert werden. Eine ConSol CM-Standardinstallation enthält bereits System-Templates und einige Beispiel-Templates, an denen Sie sich als Administrator bei der Definition neuer Templates für Ihren speziellen Anwendungsfall orientieren können.

F.8.2.2 Der Template-Editor des Admin Tools

Um mit Templates zu arbeiten, öffnen Sie das Navigationselement *Skripte und Templates* in der Navigationsgruppe *System* und wechseln Sie zum Tab *Templates*.

Die Template-Liste enthält alle Templates mit folgenden Angaben:

- **Name**
  Pflichtfeld. Ein Template wird über seinen Namen in anderen Objekten referenziert.
- **Gruppe**
  Optional. Gruppen helfen Ihnen dabei, die Templates in der Template-Liste vorübergehend zu sortieren. Diese Einstellung hat keine technischen Auswirkungen und geht beim Neuladen der Daten im Admin Tool verloren.

Um ein Template im Editorbereich zu öffnen, markieren Sie es in der Liste und öffnen Sie es, indem Sie auf den Button *Bearbeiten* klicken. Jedes Template muss einen Namen haben, die Gruppe ist optional.

Wenn Ihr System mit mehreren Sprachen arbeitet, können Sie für jede Sprache ein Template definieren. Verwenden Sie dazu das Drop-down-Menü *Sprache* über dem Editorbereich. Im Web Client wird das Template für die Sprache angezeigt, die der Benutzer im Browser eingestellt hat. Wenn für die Sprache kein Template vorhanden ist, wird das Template der Standardsprache verwendet. Jedes Template muss in der Standardsprache definiert sein.

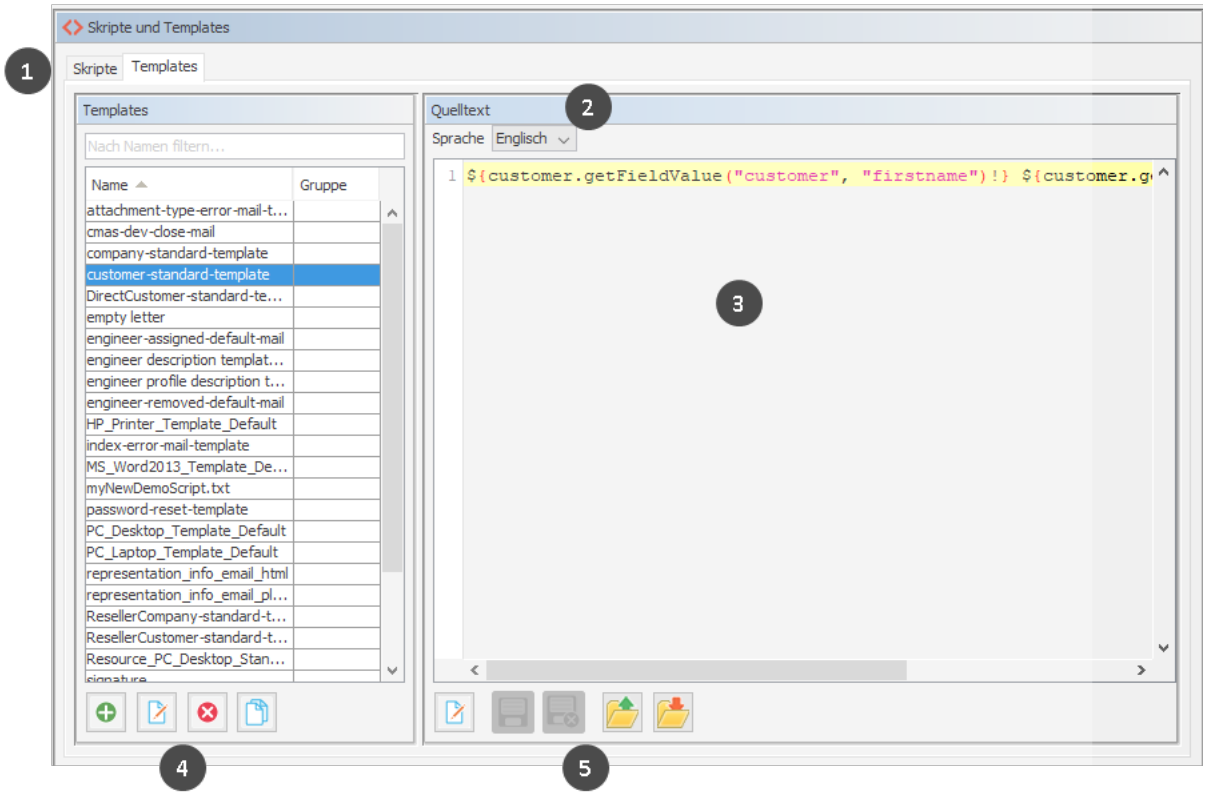


Abbildung 487: *ConSol CM Admin Tool - System, Skripte und Templates: Template-Editor*

Die Seite Templates hat folgende Elemente:

- Tabs zum Wechseln zwischen Skripten und Templates (1)
- Sprachauswahl (2)
- Editorbereich (3). Zeigt das in der Liste ausgewählte Template an.
- Allgemeine Buttons zum Hinzufügen, Löschen oder Kopieren eines Templates und zum Editieren von Template-Name und -Gruppe (4)
- Buttons für das offene Template (5)
    - Ausgewähltes Template editieren
    - Template speichern
    - Abbrechen (schließen ohne die Änderungen zu speichern)
    - Textdatei (als Template) aus dem Dateisystem hochladen
    - Template als Datei im Dateisystem speichern

F.8.2.3 Arbeiten mit Admin-Tool-Templates

Die Admin-Tool-Templates bilden einen Template-Pool. Jedes Template kann über seinen Namen in unterschiedlichen Modulen des Systems referenziert werden. In den folgenden Abschnitten werden alle Module, in denen Templates verwendet werden können, erklärt. In einem Template werden Kundenfelder, Ressourcenfelder und Ticketfelder über den Gruppennamen und den Feldnamen referenziert, z. B. wird der Firmenname in der Kundenfeldgruppe *ResellerCompanyData* wie folgt referenziert.

```
${ResellerCompany.getFieldValue("ResellerCompanyData","company_name")!}
```

Eine detaillierte Beschreibung der Arbeit mit Ticketfeldern finden Sie im Abschnitt Verwaltung von Ticketfeldern (Einrichten des Ticketdatenmodells). Kundenfelder sind im Abschnitt Einrichten des Kundendatenmodells erklärt. Ressourcenfelder sind im Abschnitt CM/Resource Pool - Einrichten des Ressourcenmodells beschrieben.

⚠ Verwenden Sie in Template-Statements keine Zeilenumbrüche!

System-Templates

Eine Standardinstallation von ConSol CM enthält mehrere System-Templates. Sie werden in Standardsituationen, wie z. B. Fehlermeldungen an einen Administrator, verwendet. Die folgende Liste zeigt eine Übersicht über die System-Templates:

- **attachment-type-error-mail-template**
  Eine auf diesem Template basierende E-Mail wird an den E-Mail-Administrator (die E-Mail-Adresse, die in der System-Property cmas-nimh-extension, mail.process.error angegeben ist) gesendet, wenn der Attachment-Typ einer eingehenden oder ausgehenden E-Mail nicht unterstützt wird und die E-Mail deshalb nicht verarbeitet werden kann.
- **cmas-dev-close-mail**
  Nicht verwendet. Mit ConSol CM-Version 6.10.1 aus der Standardinstallation entfernt.
- **engineer description template name**
  Template für die Darstellung der Bezeichnung des Bearbeiters, z. B. Ticketbesitzer. Der Template-Name „engineer description template name“ ist der Standardwert. In der System-Property cmas-core-server, engineer.description.template.name können Sie einen anderen Namen für das Template definieren, das für die Darstellung von Bearbeiternamen verwendet wird.
- **engineer profile description template name**
  Template für die Darstellung der Bezeichnung des Bearbeiters im Kopfbereich der Seite neben dem Logout-Button.
- **index-error-mail-template**
  Nicht verwendet. Mit ConSol CM-Version 6.10.1 aus der Standardinstallation entfernt.
- **password-reset-template**
  Template für den Text der E-Mail, die gesendet wird, wenn ein Benutzer (auf der Login-Seite) die Zurücksetzung seines Passworts anfordert.

- **representation_info_email_html**
  Alle E-Mails, die von CM an den vertretenen Bearbeiter gesendet werden, werden auch an den Vertreter gesendet (siehe *Allgemeine Berechtigungen: Vertretungs-Berechtigungen* im Abschnitt Rollenverwaltung). Das Template wird zur Konfiguration des Textes, der zur weitergeleiteten E-Mail hinzugefügt wird, verwendet.
- **representation_info_email_plain_text**
  Gleiche Funktion wie `representation_info_email_html` nur für E-Mails im reinen Textformat.
- **representation-create-email**
  Das Template für die E-Mail, die gesendet wird, wenn eine neue Vertretungskonfiguration eingerichtet wurde, d. h. wenn ein Bearbeiter einen Kollegen als Vertreter ausgewählt hat, weil er zum Beispiel im Urlaub ist. Die E-Mail wird beiden gesendet, dem neuen vertretenden Bearbeiter und dem Bearbeiter, der vertreten wird. Alle bearbeiterspezifischen Properties sind als Variablen im Template verfügbar, für den vertretenden Bearbeiter zum Beispiel:
  - Nachname: ${representing.lastname}
  - Vorname: ${representing.firstname}
  - Login: ${representing.name}
  - E-Mail: ${representing.email}
  - Fax: ${representing.fax}
  - Mobilnummer: ${representing.mobile}
  - Telefonnummer: ${representing.phone}

  Die Variablen für den Bearbeiter, der vertreten wird (represented), werden nach dem gleichen Muster gebildet, z. B. Nachname: ${represented.name}.
- **representation-delete-email**
  Das Template für die E-Mail, die gesendet wird, wenn eine neue Vertretungskonfiguration beendet wurde, d. h. wenn ein Bearbeiter eine Vertretungskonfiguration für den Kollegen, der ihn vertritt, gelöscht hat, weil er zum Beispiel aus dem Urlaub zurück ist. Die E-Mail wird beiden gesendet, dem alten vertretenden Bearbeiter und dem Bearbeiter, der vertreten wurde. Die verfügbaren Variablen finden Sie im vorherigen Listenpunkt.

Templates für die Darstellung der Kundendaten im Web Client

Die Darstellung von **Kundendaten** (z. B. Name, Telefonnummer und Raumnummer oder nur Name und Vorname) an unterschiedlichen Stellen des Web Clients kann mit Templates festgelegt werden. Die Definition muss für jedes Kundenobjekt einzeln vorgenommen werden (d. h. für jede Firmendefinition und für jede Kontaktdefinition), sodass für jede Kundengruppe ein spezielles Template verwendet werden kann. Die Konfiguration der Templates für Kundenobjekte ist im Abschnitt Templates für Kundendaten erklärt.

Das gleiche Prinzip gilt auch für die Darstellung von **Ressourcendaten** (wenn CM/Resource Pool in Ihrem ConSol CM-System aktiv ist). Eine detaillierte Beschreibung finden Sie im Abschnitt CM/Resource Pool - Templates für Ressourcendaten.

Im folgenden Beispiel sollen die Kundendaten im Kundenobjekt *ResellerCustomer* im Standard-Template als Vorname und Name dargestellt werden.

Die folgende Abbildung zeigt das Template in der Template-Verwaltung. Sie gelangen auf diese Seite, indem Sie das Navigationselement *Skripte und Templates* der Navigationsgruppe *System* öffnen, auf den Tab *Templates* klicken und das gewünschte Template in der Liste auswählen.

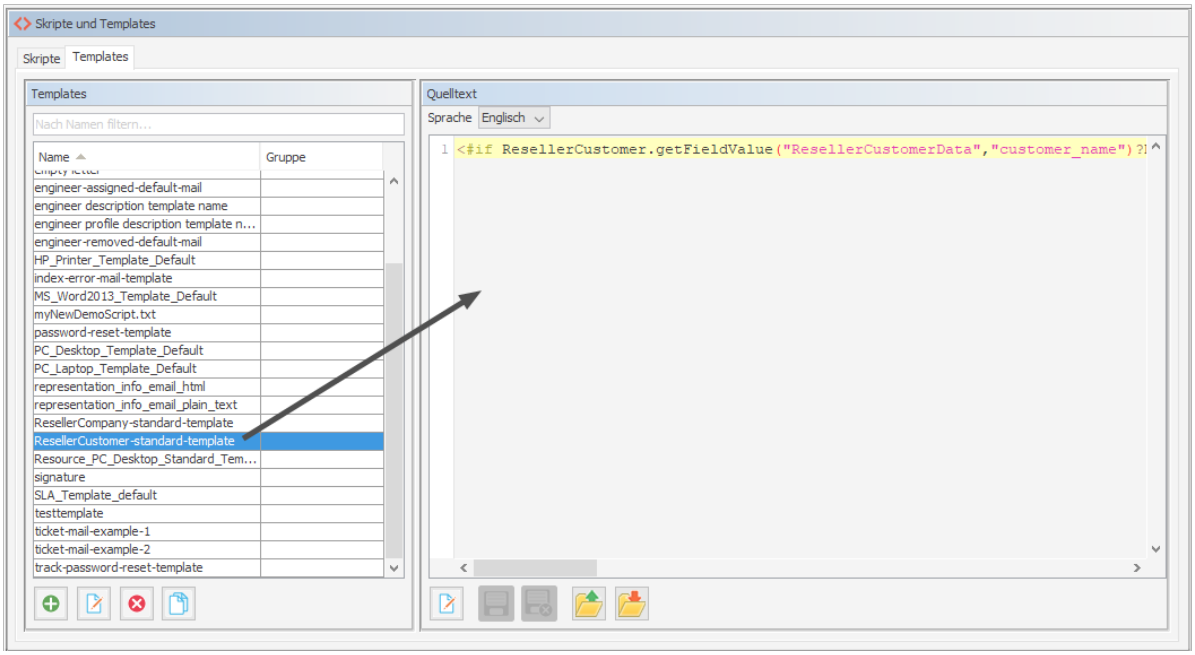


Abbildung 488: *ConSol CM Admin Tool - System, Skripte und Templates: Beispiel für ein Template für die Darstellung von Kundendaten*

```
<#if ResellerCustomer.getFieldValue("ResellerCustomerData","customer_name")?has_
 content && ResellerCustomer.getFieldValue("ResellerCustomerData","forename")?has_
 content>${ResellerCustomer.getFieldValue("ResellerCustomerData","customer_
 name")!},${ResellerCustomer.getFieldValue
 ("ResellerCustomerData","forename")!}<#else> ${ResellerCustomer.getFieldValue
 ("ResellerCustomerData","customer_name")!}</#if>Search for contacts of a certain
 company
```

Code-Beispiel 94: *Template für die Darstellung von Kundendaten*

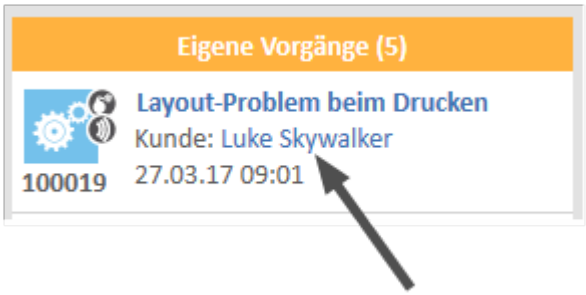


Abbildung 489: *ConSol CM Web Client - Beispiel für ein Template für die Darstellung von Kundendaten*

E-Mail-Templates für die Ticketzuweisung

In der Queue-Verwaltung (siehe Abschnitt Queue-Verwaltung) können E-Mail-Templates für die Ticketzuweisung ausgewählt werden. Es gibt Templates für zwei Anwendungsfälle: **Zuweisen** und **Entfernen**.

- Das Template *Zuweisen* wird als Text-Template für die automatische E-Mail verwendet, die das System an den (neuen) Bearbeiter schickt, wenn ihm ein Ticket zugewiesen wird.
- Das Template *Entfernen* wird als Text-Template für die automatische E-Mail verwendet, die das System an den (alten) Bearbeiter schickt, wenn ihm ein Ticket entzogen wird.

Sie müssen die Templates vorher im Abschnitt *Templates* schreiben und abspeichern. Dann sind sie im Drop-down-Menü im Abschnitt *E-Mail-Templates für die Ticketzuweisung* der Queue-Verwaltung verfügbar (siehe Abschnitt Queue-Verwaltung). Sie können einen beliebigen Namen definieren. Wir empfehlen Ihnen, einen Namen zu vergeben, der die Verwendung des Templates beschreibt, wie im folgenden Beispiel gezeigt.

Die folgende Abbildung zeigt das Template in der Template-Verwaltung. Sie gelangen auf diese Seite, indem Sie das Navigationselement *Skripte und Templates* der Navigationsgruppe *System* öffnen, auf den Tab *Templates* klicken und das gewünschte Template in der Liste auswählen.

Als Standardadresse `From`wird die E-Mail-Adresse des CM-Administrators (CM-System-Property admin.email) verwendet. Sie können die From-Adresse auch explizit im Template setzen (siehe folgendes Beispiel).

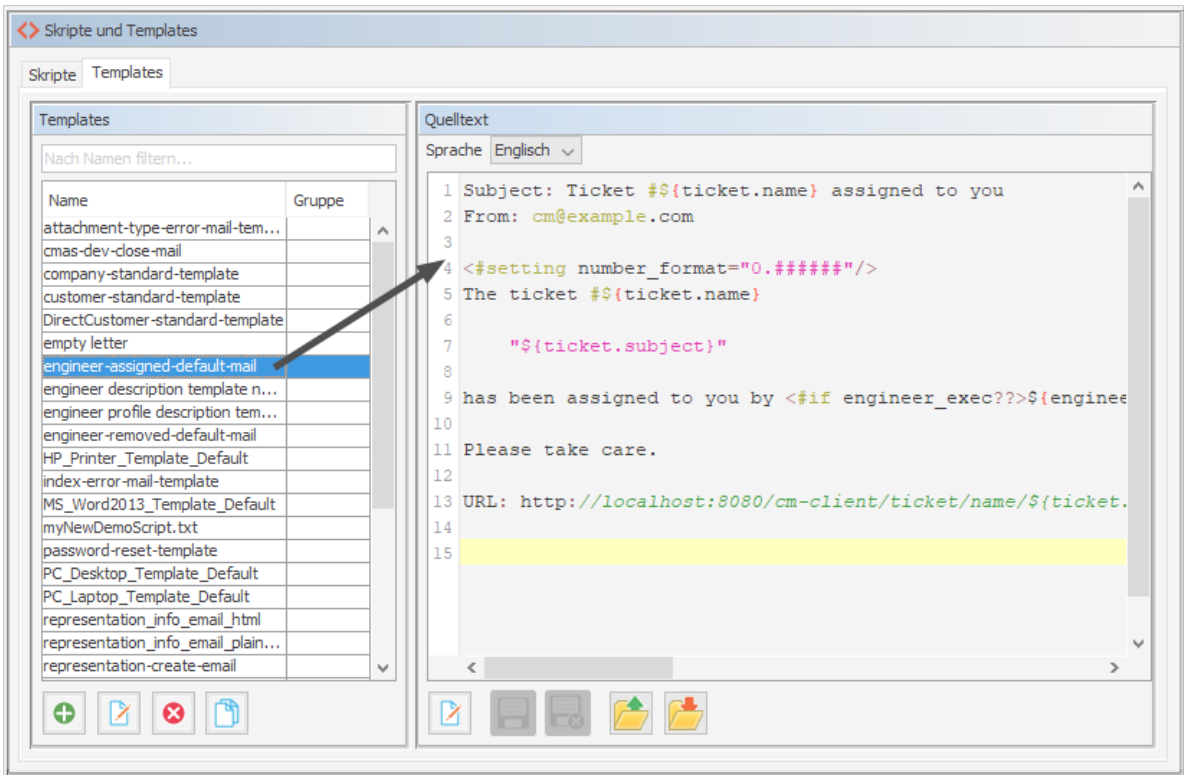


Abbildung 490: *ConSol CM Admin Tool - System, Skripte und Templates: Beispiel für ein E-Mail-Template für die Ticketzuweisung zu einem Bearbeiter*

```
Subject: Ticket #${ticket.name} assigned to you
From: cm@example.com

<#setting number_format="0.######"/>
The ticket #${ticket.name}

   "${ticket.subject}"

has been assigned to you by <#if engineer_exec??>${engineer_exec.name}<#else>the
 workflow</#if> <#if engineer_old??>(former engineer: ${engineer_old.name})<#else>
 (no former engineer)</#if>

Please take care.

URL: http://localhost:8080/cm-client/ticket/name/${ticket.name}
```

Code-Beispiel 95: *Text des Beispiel-Templates engineer-assigned-default-mail*

Die folgende Abbildung zeigt die Queue-Details. Sie gelangen auf diese Seite, indem Sie das Navigationselement *Queues* in der Navigationsgruppe *Globale Konfiguration* öffnen und die gewünschte Queue in der Liste auswählen.

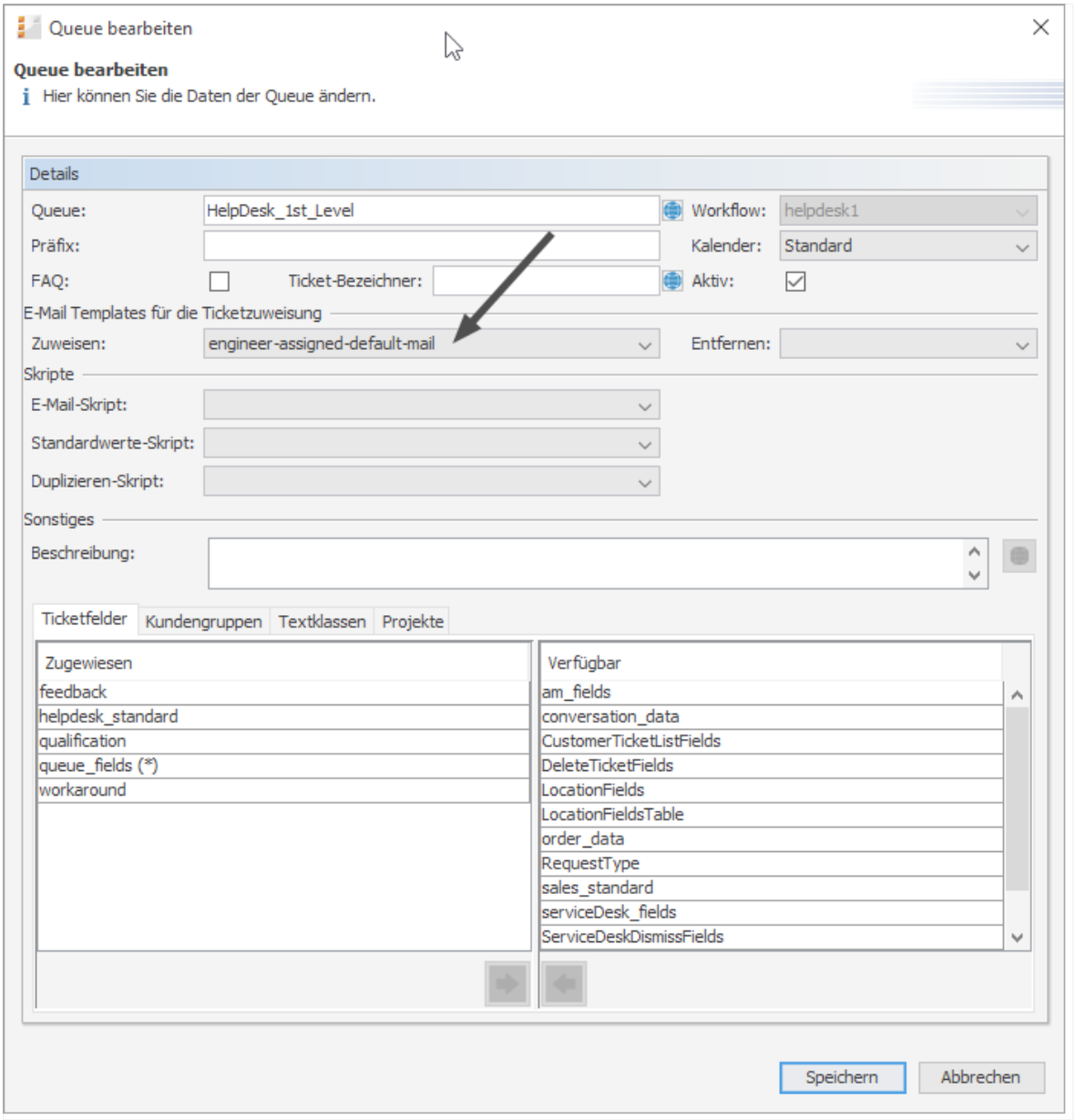


Abbildung 491: *ConSol CM Admin Tool - Globale Konfiguration, Queues: Queue bearbeiten, um E-Mail-Template für die Ticketzuweisung anzugeben*

Templates zum Zurücksetzen des Passworts

Wenn Benutzer ihre Passwörter vergessen haben, können sie über die ConSol CM-Standardfunktion ein neues Passwort anfordern. Der Benutzer kann ein Bearbeiter sein, der mit dem Web Client arbeitet, oder ein Kunde, der sich bei CM/Track anmeldet. Für beide Fälle kann der Administrator ConSol CM so konfigurieren, dass eine E-Mail an den Benutzer gesendet wird, der ein neues Passwort

benötigt. Diese E-Mail basiert auf einem Template, das im Abschnitt *Templates* des Admin Tools gespeichert ist. Die E-Mail enthält einen Hyperlink auf eine Seite, auf der das Passwort zurückgesetzt werden kann.

Zurücksetzen des Passworts für Bearbeiter im Web Client

Wenn ein Bearbeiter sein Web-Client-Passwort vergessen hat, kann er ein neues Passwort anfordern, indem er auf der Login-Seite auf den Link *Passwort vergessen?* klickt. Der Bearbeiter erhält eine E-Mail mit einem Link zu einer URL, über die er ein neues Passwort setzen kann.

> ⚠ Beachten Sie, dass dieser Vorgang nur funktioniert, wenn der Bearbeiter ein gültiges E-Mail-Konto hat und der entsprechende Wert in den Bearbeiterdaten im Feld *E-Mail* eingetragen ist!

Die E-Mail, die an den Bearbeiter gesendet wird, basiert auf dem Template `password-reset-template`. Das Template muss diesen Namen haben, damit das Zurücksetzen des Passworts funktioniert. Das Template kann zum Beispiel wie folgt aussehen:

```
Subject: Password reset procedure

<#setting number_format="0.######"/>
To reset your password please click the following link:

http://localhost:8888/cm-client/passwordChange?resetCode=${resetCode}

This link expires at ${expirationDate?string("yyyy.MM.dd HH:mm:ss")}.
```

Code-Beispiel 96: *Template zum Zurücksetzen des Passworts eines Bearbeiters*

Die From-Adresse der E-Mail, die an den Bearbeiter gesendet wird, wird in der CM-System-Property cmas-core-security, password.reset.mail.from definiert.

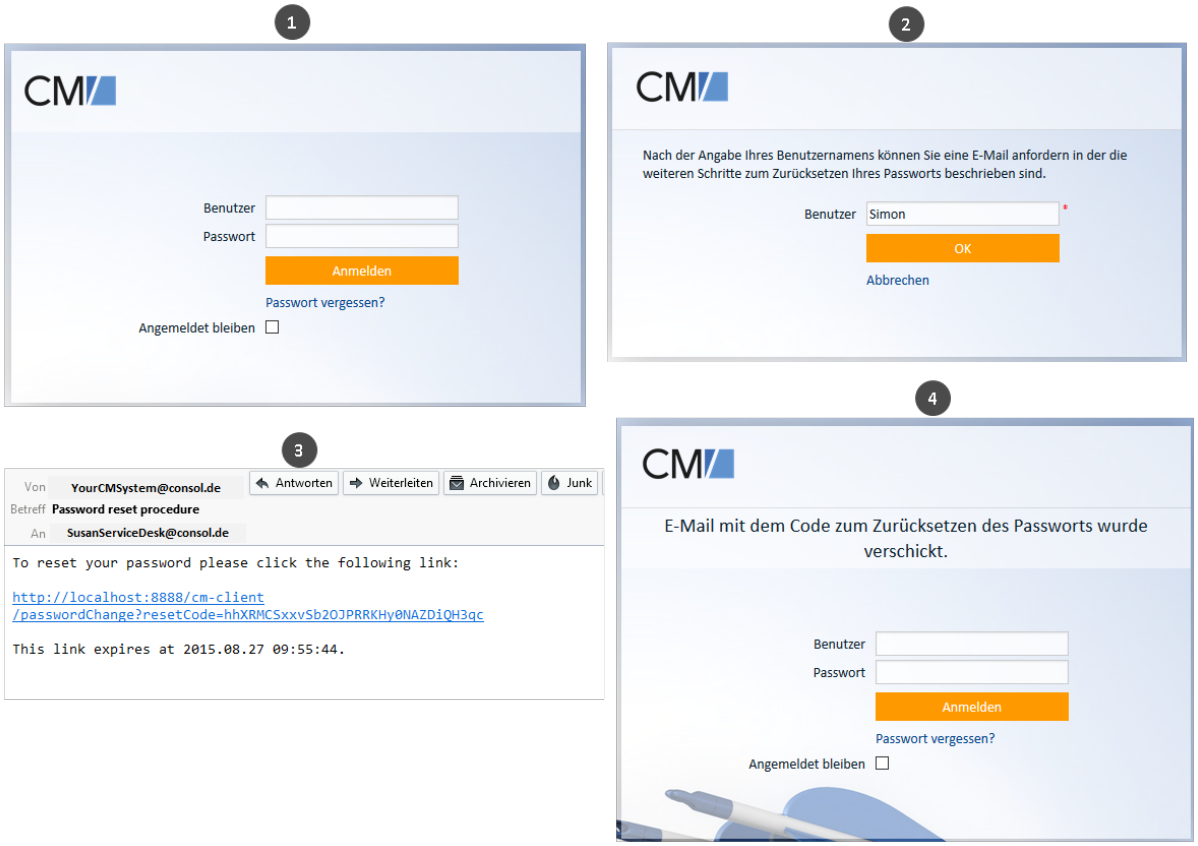


Abbildung 492: *ConSol CM Web Client - Zurücksetzen des Passworts durch einen Bearbeiter*

Beachten Sie, dass das Zurücksetzen des Passworts im Web Client nur möglich ist, wenn der Standardmodus verwendet wird. Bei LDAP- oder Kerberos-Authentifizierung ist dies nicht möglich. Eine Beschreibung aller möglichen Authentifizierungsmethoden finden Sie im Abschnitt Authentifizierungsmethoden für Bearbeiter im Web Client.

Template zum Zurücksetzen des Passworts durch Kunden in CM/Track V1

Wenn ein Kunde sein CM/Track-Passwort vergessen hat, kann er über den Link *Passwort vergessen?* auf der Login-Seite ein neues Passwort anfordern. Der Kunde erhält eine E-Mail mit einem Link zu einer URL, über die er ein neues Passwort setzen kann.

Beachten Sie, dass dieser Vorgang nur funktioniert, wenn der Kunde ein gültiges E-Mail-Konto hat und der entsprechende Wert im entsprechenden Kundenfeld als E-Mail-Adresse für den Kunden eingetragen ist!

Die E-Mail, die an den Kunden gesendet wird, basiert auf dem Template `track-password-reset-template`. Das Template muss diesen Namen haben, damit das Zurücksetzen des Passworts funktioniert. Das Template muss manuell erstellt/hinzugefügt werden, da es standardmäßig nicht im System verfügbar ist.

Die Formatierung des Templates sollte dem folgenden Beispiel entsprechen (Sie können einen beliebigen Text hinzufügen, der an die Kunden gesendet werden soll, achten Sie aber auf die Variablen `$resetCode` und die URL!):

```
Your Password Reset Link:
http://mycm6system/cm-track/#track=set_new_password/resetCode-${resetCode}(Reset
 Code: ${resetCode})
Valid 24 hours only, please visit before expiry!
```

Code-Beispiel 97: *Zurücksetzen des Passworts eines Kunden in CM/Track*

Als From-Adresse für die E-Mail an den Kunden wird die E-Mail-Adresse des Administrators verwendet.

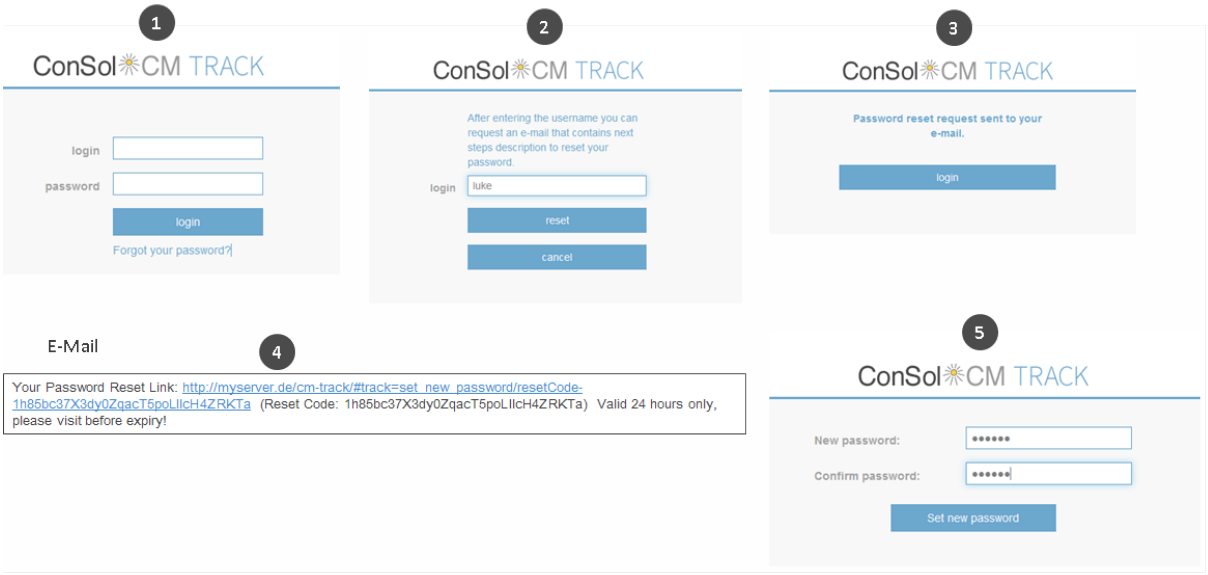


Abbildung 493: *CM/Track - Zurücksetzen des Passworts durch einen Kunden*

Beachten Sie, dass das Zurücksetzen des Passworts in CM/Track V1 nur möglich ist, wenn der Datenbankmodus verwendet wird. Bei LDAP-Authentifizierung ist dies nicht möglich. Eine Beschreibung aller verfügbaren Authentifizierungsmethoden finden Sie im Abschnitt Authentifizierungsmethoden für Kunden in CM/Track.

Template zum Zurücksetzen des Passworts durch Kunden in CM/Track V2

Wenn ein Kunde sein CM/Track-Passwort vergessen hat, kann er über den Link *Passwort vergessen?* auf der Login-Seite ein neues Passwort anfordern. Der Kunde erhält eine E-Mail mit einem Link zu einer URL, über die er ein neues Passwort setzen kann.

⚠ Beachten Sie, dass dieser Vorgang nur funktioniert, wenn der Kunde ein gültiges E-Mail-Konto hat und der entsprechende Wert im entsprechenden Kundenfeld als E-Mail-Adresse für den Kunden eingetragen ist!

Die E-Mail, die an den Kunden gesendet wird, basiert auf dem Template `track-password-reset-template`. Das Template muss diesen Namen haben, damit das Zurücksetzen des Passworts funktioniert. Das Template muss manuell erstellt/hinzugefügt werden, da es standardmäßig nicht im System verfügbar ist.

Die Formatierung des Templates sollte dem folgenden Beispiel entsprechen (Sie können einen beliebigen Text hinzufügen, der an die Kunden gesendet werden soll, achten Sie aber auf die Variablen `$resetCode` und die URL!):

```
Subject: Your Password Reset Link
Your Password Reset Link:
<#setting number_format="0.######"/>
To reset your password please click the following link:
http://myserver:myport/track/#/password-reset/resetCode-${resetCode}
This link expires at ${expirationDate?string("yyyy.MM.dd HH:mm:ss")}.
```

Abbildung 494: *Admin-Tool-Template zum Zurücksetzen des Kundenpassworts in CM/Track V2 (Ersetzen Sie myserver und myport mit Ihren Systemparametern.)*

ⓘ Beachten Sie, dass die Variable `expirationDate` eine Systemvariable ist, die standardmäßig auf ein Datum 24 Stunden nach dem Zeitpunkt der Anfrage zum Zurücksetzen des Passworts gesetzt ist. Sie können die Variable über die CM-System-Property cmas-core-security, resetCode.expiriationPeriod (Integer, Millisekunden) ändern. Die System-Property ist standardmäßig nicht vorhanden und muss bei Bedarf erstellt werden.

Als From-Adresse für die E-Mail an den Kunden wird die E-Mail-Adresse des Administrators verwendet.

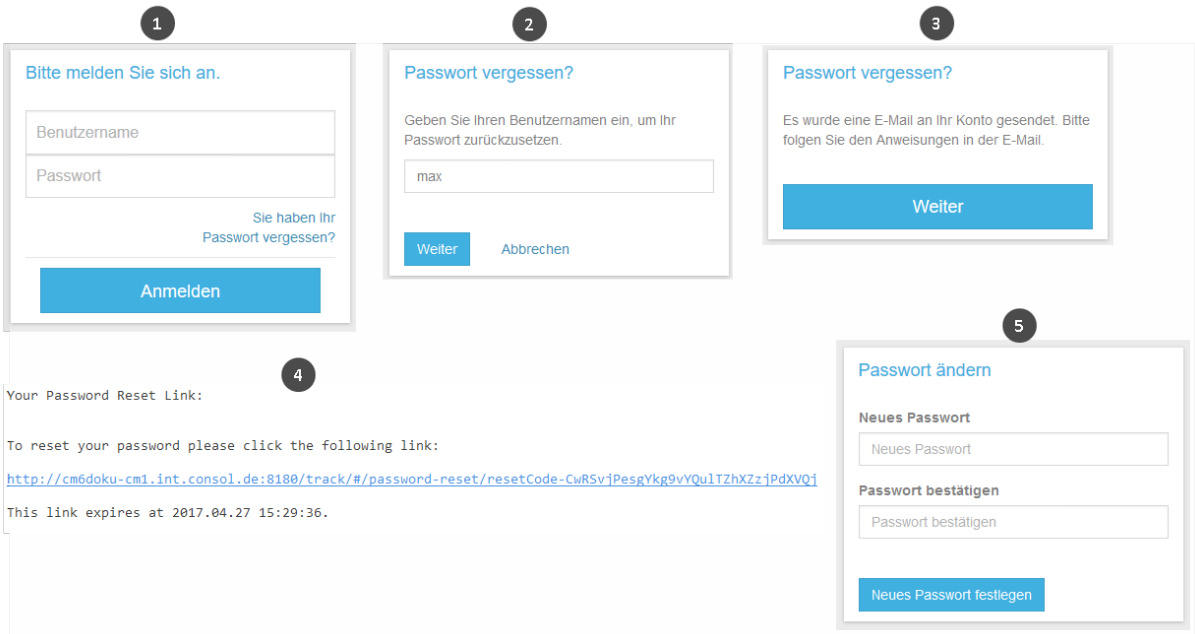


Abbildung 495: *CM/Track - Zurücksetzen des Passworts durch einen Kunden*

Beachten Sie, dass das Zurücksetzen des Passworts in CM/Track V2 nur möglich ist, wenn der Datenbankmodus verwendet wird. Bei LDAP-Authentifizierung ist dies nicht möglich. Eine Beschreibung aller verfügbaren Authentifizierungsmethoden finden Sie im Abschnitt Authentifizierungsmethoden für Kunden in CM/Track.

Benutzerdefinierte Templates

Als ConSol CM-Administrator oder Workflow-Entwickler können Sie alle erforderlichen Templates definieren und in den *Skripten und Templates* speichern. Wenn Sie das Template in automatischen E-Mails verwenden, die von einer Workflow-Aktivität gesendet werden, können Sie das Template mit der Methode `renderTemplate()` der Workflow-API referenzieren. Die meisten E-Mail-Templates sollten allerdings im *Textvorlagen-Manager* verwaltet werden (siehe Abschnitt Der ConSol CM Textvorlagen-Manager). Es gibt nur wenige Anwendungsfälle, in denen es erforderlich ist, E-Mail-Templates oder Teile von E-Mail-Templates im Navigationselement *Skripte und Templates* des Admin Tools zu speichern.

F.9 Deployment (Import/Export)

In diesem Kapitel werden folgende Themen behandelt:

F.9.1 Einleitung 770

F.9.2 Szenarios 771

F.9.3 Deployment (Import/Export) mit dem Admin Tool 772

F.9.1 Einleitung

ConSol CM bietet die Möglichkeit, die Systemkonfiguration mit oder ohne Laufzeitdaten in eine Datei zu exportieren, und die Datei in einem anderen ConSol CM-System zu importieren. Die Datei, mit der die Daten übertragen werden, heißt *Szenario*. Sie enthält verschiedene Daten. Die Details sind in den folgenden Abschnitten beschrieben.

Normalerweise wird ein Szenario mit Konfigurationsdaten benutzt, um Daten von einem Testsystem auf ein Staging- oder Produktivsystem zu übertragen.



Da ConSol CM für alle Objekte mit Übertragungsschlüsseln (*transfer keys*) arbeitet, wird dringend empfohlen, für die Übertragung von Testumgebungen auf Staging- oder Produktivumgebungen ein Szenario zu verwenden und die gleiche Funktionalität **nicht** parallel auf beiden Systemen zu implementieren. Ansonsten würden Objekte dupliziert, wenn zu einem späteren Zeitpunkt eine Übertragung (Export/Import) durchgeführt wird! Lesen Sie die detaillierten Beschreibungen dazu in den folgenden Abschnitten.

F.9.2 Szenarios

Ein Szenario ist eine Datei in einem proprietären ConSol CM-Format (ähnlich wie `zip`, `jar` und `tar`), die die Daten der ConSol CM-Installation enthält. Sie kann aus einem ConSol CM-System exportiert werden und auf demselben oder einem anderen System wieder importiert werden. Dies kann sehr hilfreich sein, z. B. wenn ein Testszenario auf einem Testsystem gebaut wird, und später auf einen Produktionsserver übertragen werden soll.

Der Administrator kann beim Erstellen der Exportdatei (siehe detaillierte Erklärung in den Abschnitten unten) entscheiden, welche Daten darin enthalten sein sollen.

Ein Szenario enthält **immer**:

- alle kundenspezifischen System-Properties,
  d. h. System-Properties, deren Modulname mit *custom-* beginnt, eine detaillierte Erklärung dazu finden Sie im Abschnitt CM-System-Properties!

Ein Szenario **kann** abhängig von der Auswahl, die der Administrator trifft, Folgendes enthalten (siehe Abbildung unten):

- Laufzeitdaten
- Konfigurationsdaten

Ein Szenario enthält **nie**:

- allgemeine (nicht kundenspezifische) System-Properties (z. B. Mail-Server, LDAP-Verzeichnis usw.)

F.9.3 Deployment (Import/Export) mit dem Admin Tool

Im Admin Tool werden die Deployments im Navigationselement *Import/Export* der Navigationsgruppe *System* verwaltet. In diesem Navigationselement können Sie Szenarios (d. h. die gesamte Konfiguration oder einen Teil der Konfiguration) in einem anwendungsspezifischen Format importieren oder exportieren. Normalerweise tun Sie dies, um Daten zwischen unterschiedlichen ConSol CM-Installationen zu übertragen. Ein typisches Beispiel ist die Übertragung der Konfiguration von einem Testsystem auf ein Produktivsystem.

Alle Export-/Importaktionen sind in der Datei `transfer.log` protokolliert. Eine detaillierte Beschreibung dieser Datei finden Sie im Abschnitt Logging und Log-Dateien.

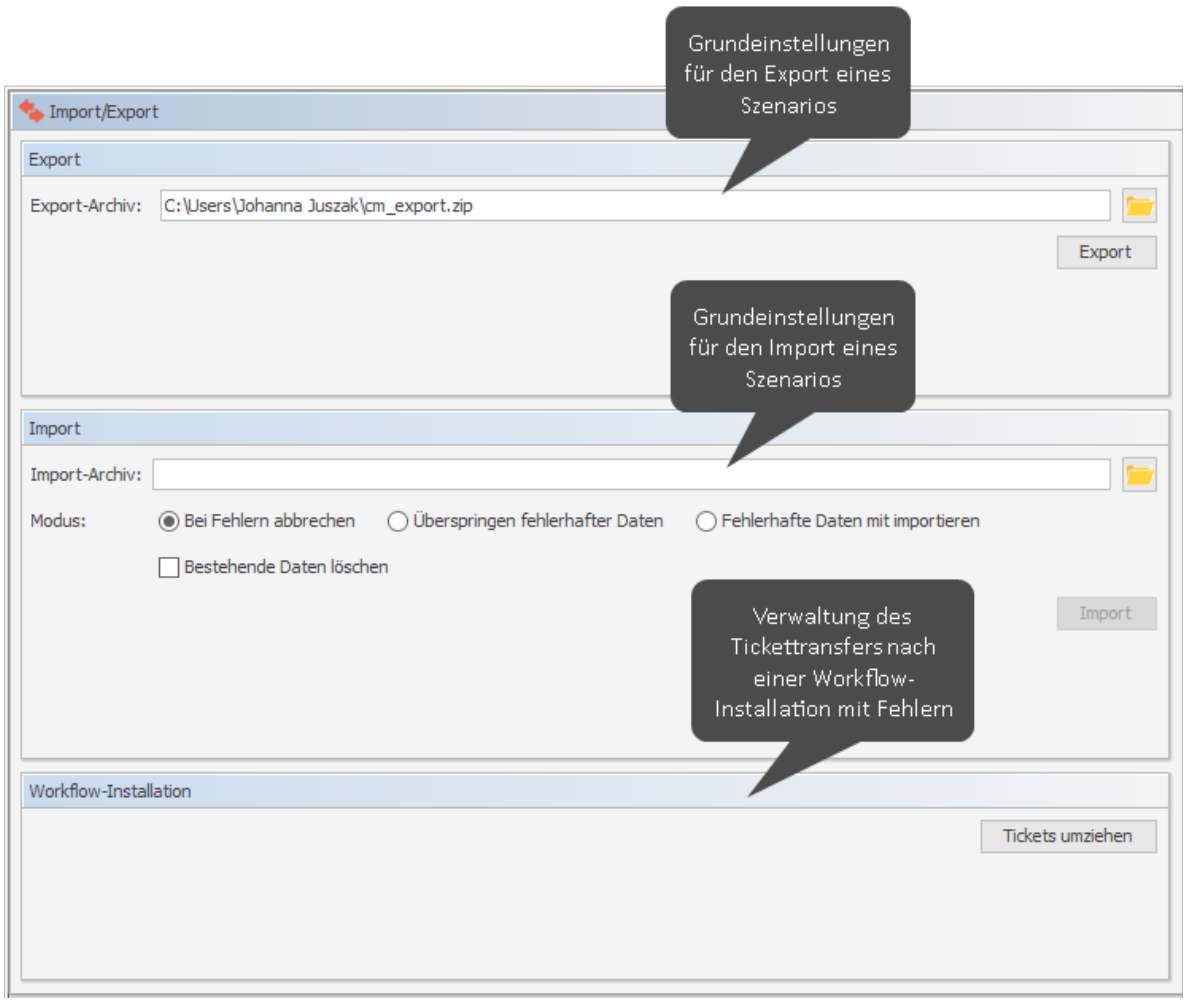


Abbildung 496: *ConSol CM Admin Tool - System, Import/Export*



Die Änderung, das Überschreiben bzw. das Löschen vorhandener Daten bei einem Import von externen Daten kann nicht rückgängig gemacht werden. Obwohl der Benutzer entscheidende Punkte des Deployments extra bestätigen muss, können irrtümliche Handlungen nicht verhindert werden. Benutzen Sie diese Funktion **nur**, wenn Sie genau wissen, was Sie tun. Bitten Sie das ConSol CM-Supportteam oder einen ConSol CM-Consultant um Hilfe, wenn Sie sich unsicher sind.

F.9.3.1 Export

- **Export-Archiv:**
  Pfad zum Exportarchiv. Wenn Sie das Navigationselement *Import/Export* öffnen, ist als Standardpfad für den Export (`<Startverzeichnis des Admin-Benutzers>/cm_export.zip`) gesetzt. Verwenden Sie diesen Wert oder geben Sie den Pfad und Namen der Datei ein, die Sie erstellen möchten. Alternativ können Sie auf den Button *Ordner* klicken, um den Dateibrowser zu öffnen.

Klicken Sie danach auf *Export*, um den Datenexport zu beginnen.

Sie müssen die Daten auswählen, die in der Exportdatei (dem Szenario) enthalten sein sollen:

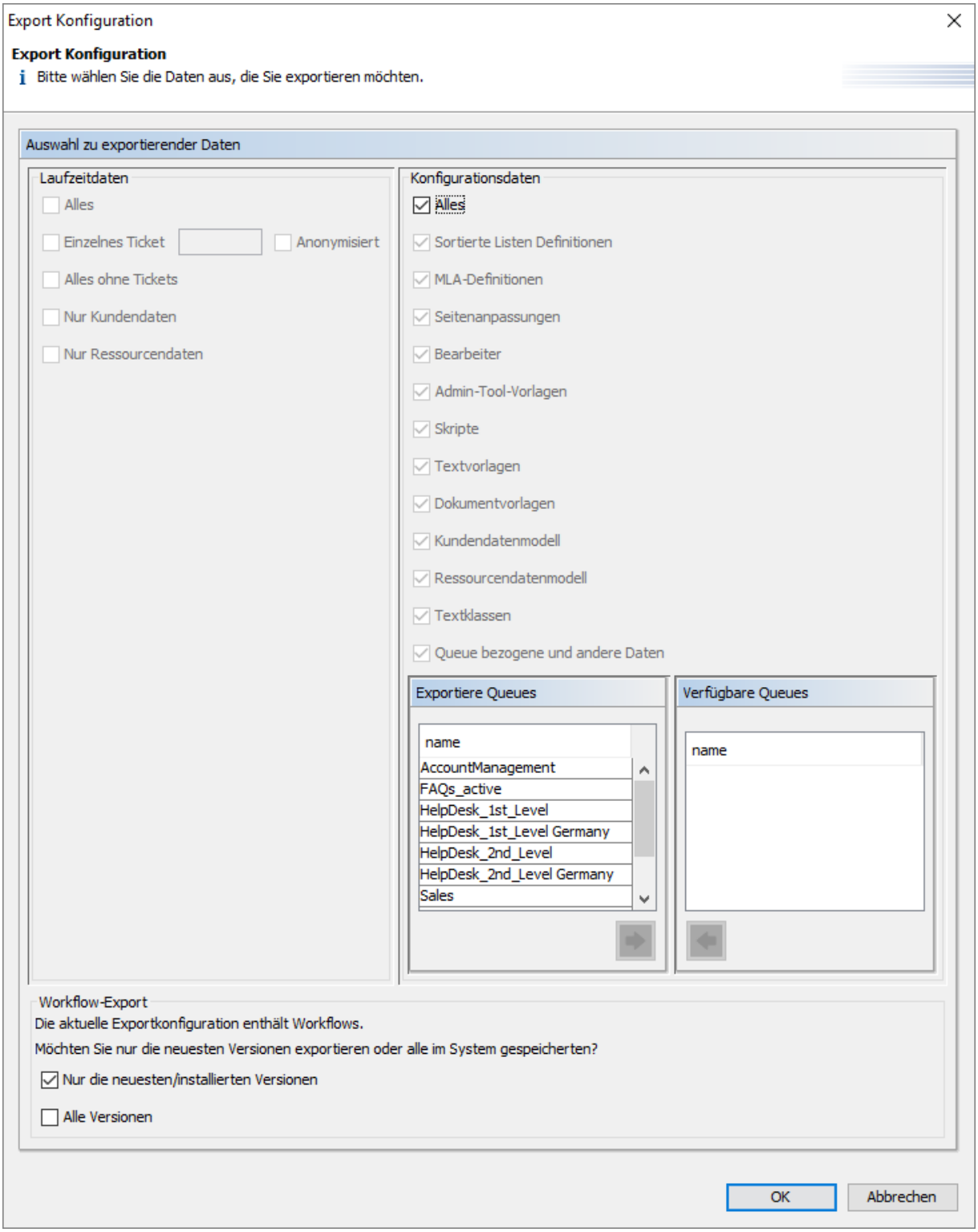


Abbildung 497: *ConSol CM Admin Tool - System, Import/Export: Exportkonfiguration*

- **Laufzeitdaten**
  Dies bezieht sich auf die Daten, die aus dem laufenden Betrieb stammen, z. B. Tickets, Ressourcendaten und Kundendaten.
  - **Alles**
    Ticketdaten und Kundendaten werden komplett exportiert **und** die gesamte Konfiguration wird exportiert. Wenn Sie die Checkbox *Alles* markieren, werden alle anderen Checkboxen automatisch auch markiert. Wenn CM/Resource Pool aktiv ist, werden die Ressourcendaten ebenfalls exportiert.
  - **Einzelnes Ticket**
    Wenn die Checkbox *Einzelnes Ticket* markiert ist, müssen Sie den Ticketnamen in das Eingabefeld eingeben. (Der Ticketname wird im Web Client unter dem Ticket-Icon angezeigt.)
    Markieren Sie die Checkbox *Anonymisiert*, um die Exportdaten zu anonymisieren (Hash). Auf diese Weise werden keine echten Kunden-, Bearbeiter- oder Ressourceninformationen in das Importsystem übertragen. Die folgenden Daten werden anonymisiert:
    - Inhalt von Ticketfeldern
    - Ticket-Attachments
    - Ticketprotokoll
    - Inhalt von Kundenfeldern
    - Kundenprotokoll

    > Die Option *Einzelnes Ticket* ist nicht für die Verwendung im normalen Systembetrieb gedacht, sondern ist eine Funktion zum Erleichtern der Fehleranalyse. Mit diesem Export wird eine Datei im Format .zip/.jar erzeugt, die die Ticketdaten in XML-Dateien enthält. Alle für das Ticket erforderlichen Daten (d. h. alle verknüpften Objekte) werden mit dem Ticket exportiert. Beachten Sie dies, wenn Sie das Ticket in ein anderes System importieren - die Daten könnten dort geändert werden!

  - **Alles ohne Tickets**
    Die komplette Installation mit Ausnahme der Tickets wird exportiert, d. h. die Kundendaten **und** die gesamte Konfiguration. Wenn Sie die Checkbox *Alles ohne Tickets* markieren, werden alle anderen Checkboxen außer *Alles* automatisch auch markiert. Wenn CM/Resource Pool aktiv ist, werden die Ressourcendaten ebenfalls exportiert.
  - **Nur Kundendaten**
    Es werden nur die Kundendaten (d. h. das Kundendatenmodell und die eigentlichen Kundendaten) exportiert. Nichts anderes. (Die Checkbox *Kundendatenmodell* wird automatisch markiert.)

- **Nur Ressourcendaten**
  Es werden nur die Ressourcendaten (d. h. das Ressourcendatenmodell und die eigentlichen Ressourcendaten) exportiert. Nichts anderes. (Die Checkbox *Ressourcenmodell* wird automatisch markiert.)

- **Konfigurationsdaten**
  Dies bezieht sich nur auf die Konfiguration im Admin Tool. Es werden keine Laufzeitdaten exportiert.

  - **Alles**
    Die gesamte Konfiguration wird exportiert. Wenn Sie die Checkbox *Alles* auswählen, werden alle anderen Checkboxen unter *Konfigurationsdaten* automatisch auch markiert.
  - **Sortierte Listen Definitionen**
    Es werden die Definitionen und Lokalisierungen von Sortierten Listen (Enums) exportiert. Dies sind die Daten, die im Navigationselement *Sortierte Listen* (Navigationsgruppe *Listen*) definiert werden. Details über Sortierte Listen finden Sie im Abschnitt Verwaltung von Sortierten Listen
  - **MLA-Definitionen**
    Es werden die Definitionen und Lokalisierungen von MLAs (Multi Level Attributes) exportiert. Dies sind die Daten, die im Navigationselement *MLAs* (Navigationsgruppe *Listen*) definiert werden. Details über MLAs finden Sie im Abschnitt MLA-Verwaltung.
  - **Seitenanpassungen**
    Es werden die speziell für das System definierten Seitenanpassungen (d. h. nicht die Standardwerte) exportiert. Dies sind die Werte, die in den Datenbanktabellen `cmas_web_customization` und `cmas_web_customization_values` gespeichert sind. Eine Detailübersicht über dieses Thema finden Sie im Abschnitt Seitenanpassung.
  - **Bearbeiter**
    Es werden nur die Bearbeiter mit ihren Daten exportiert. Dies umfasst auch die Rollen, die den Bearbeitern zugewiesen sind.
  - **Admin-Tool-Vorlagen**
    Es werden nur die Admin-Tool-Templates (siehe Abschnitt Admin-Tool-Templates) exportiert.
  - **Skripte**
    Es werden nur die Admin-Tool-Skripte (siehe Abschnitt Admin-Tool-Skripte) exportiert.
  - **Textvorlagen**
    Es werden nur die im Textvorlagen-Manager (siehe Abschnitt Der ConSol CM Textvorlagen-Manager) gespeicherten Vorlagen exportiert.
  - **Dokumentvorlagen**
    Es werden nur die Microsoft Word- oder OpenOffice-Vorlagen exportiert. Diese Einstellung ist nur relevant, wenn CM/Doc aktiv ist (siehe Abschnitt CM/Doc).
  - **Kundendatenmodell**
    Es werden nur die Kundenfelder, mit denen das Kundendatenmodell definiert ist, exportiert. Kundendaten aus dem laufenden Betrieb gehören nicht dazu.

- **Ressourcendatenmodell**
  Es werden nur die Ressourcenfelder, mit denen das Ressourcendatenmodell definiert ist, exportiert. Ressourcendaten aus dem laufenden Betrieb gehören nicht dazu.
- **Textklassen**
  - Wenn diese Checkbox aktiviert ist, werden alle im CM-System verfügbaren Textklassen exportiert.
  - Wenn diese Checkbox nicht aktiviert ist, werden nur die Textklassen exportiert, die einer Queue zugewiesen sind, die im Export enthalten ist. Daraus folgt, dass Textklassen, die keiner Queue zugewiesen sind, in diesem Fall **nicht** exportiert werden.
- **Queue-bezogene und andere Daten**
  Es werden nur die Queue-Konfiguration und die allgemeinen Konfigurationseinstellungen exportiert (Workflows, Queues, Ticketfelder, Werte in Sortierten Listen, MLAs, Rollen, Sichten, System-Properties ...). Kurz gesagt: alles, was nicht einem der obigen Punkte zugeordnet ist.

- **Workflow-Export**
  Dieser Bereich wird aktiviert, sobald Sie im Bereich *Konfigurationsdaten* die Option *Alle* oder *Queue bezogene und andere Daten* ausgewählt haben. Die Auswahl im Bereich *Workflow-Export* kann die Größe des exportierten Szenarios stark beeinflussen. Unter Umständen ist es hilfreich, einige Workflows zu löschen, bevor Sie ein Szenario mit Workflows exportieren.
  - **Nur die neuesten/installierten Versionen**
    Es werden nur die installierten Versionen der Workflows exportiert, d. h. die Versionen, die aktuell installiert und im System aktiv sind. Alle älteren Versionen werden nicht exportiert.
  - **Alle Versionen**
    Es werden alle Versionen aller Workflows exportiert, sowohl die aktuell installierten als auch die älteren Versionen. Beachten Sie, dass das Markieren dieser Checkbox je nach Anzahl der im System gespeicherten Workflows dazu führen kann, dass der Export sehr lange dauert und ein sehr großes Szenario exportiert wird.

Wenn Sie die gesamte Konfiguration exportieren möchten, wählen Sie im Abschnitt *Konfigurationsdaten* die Option *Alles*. Der Export/Import von Teilmengen (z. B. nur den Templates) wird normalerweise eingesetzt, wenn die ausgewählten Daten (z. B. von einer Testumgebung) auf ein anderes System (z. B. eine Live-Umgebung) übertragen werden sollen.

F.9.3.2 Import

Allgemein

Beachten Sie, dass während des Imports der DWH-Modus „LIVE“ ausgeschaltet werden muss!

Das Grundprinzip des Imports eines ConSol CM-Szenarios ist:

1. Wenn die Checkbox *Bestehende Daten löschen* **nicht ausgewählt** ist, werden die Szenarios wie hier beschrieben **zusammengeführt**:
    - Es werden nur Daten **hinzugefügt**. Es wird nichts gelöscht.
    - Wenn das importierte Szenario **das gleiche Feld / den gleichen Parameter** enthält wie das ursprüngliche Szenario, **überschreibt** der Wert des importierten Szenarios den Wert des ursprünglichen Szenarios.
      Beispiel: Das Feld *priority* hat im importierten Szenario die Annotation `position` = „0;2“. Das ursprüngliche Szenario enthält für das Feld *priority* den Wert `position` = „2;2“, d. h. in dem Szenario, das aus dem Import resultiert, hat die Annotation `position` den Wert „0;2“.
    - Wenn das importierte Szenario **mehr Parameter** enthält als das ursprüngliche Szenario, werden die Parameter zum ursprünglichen Szenario **hinzugefügt**.
      Beispiel: Im importierten Szenario gibt es eine Annotation `visibility` = „none“ für das Feld *PersonID*. Im ursprünglichen Szenario ist das Feld *PersonID* vorhanden, es hat aber keine Annotation, d. h. im aus dem Import resultierenden Szenario hat das Feld *PersonID* die Annotation `visibility` = „none“ und ist daher unsichtbar.
    - Wenn das importierte Szenario **weniger Daten/Parameter** enthält als das ursprüngliche Szenario, bleiben die ursprünglichen Daten im resultierenden Szenario erhalten. **Es wird nichts gelöscht**.
      Beispiel: Wenn das Feld *PersonID* im importierten Szenario die Annotation `visibility` = „none“ nicht mehr hat, das ursprüngliche Szenario diese aber noch enthält, bleibt die Annotation erhalten, d. h. im resultierenden Szenario ist das Feld *PersonID* immer noch unsichtbar.
    - Bei **Skripten und Templates** wird die **letzte Version** (entsprechend dem Zeitstempel) verwendet, unabhängig davon, aus welchem Szenario sie stammt.
    - **Objekte** werden über einen **internen Schlüssel** (*Übertragungsschlüssel, transfer key*) identifiziert.
      Wenn ein importiertes Szenario ein Objekt enthält, das den gleichen Namen aber einen anderen Übertragungsschlüssel hat, handelt es sich technisch gesehen um zwei unterschiedliche Objekte und das neue Objekt aus dem Importszenario wird zum ursprünglichen Szenario hinzugefügt (z. B. wenn es in beiden Szenarios einen Benutzer *Hr. Müller* gibt, enthält das resultierende Szenario nach dem Import einen Benutzer *Hr. Müller* und einen Benutzer *Hr. Müller (1)*.
      Um ein weiteres Importszenario aus der gleichen Quelle (Testsystem) übertragen zu können, können Sie den ursprünglichen Benutzer *Hr. Müller* löschen und seine Tickets *Hr. Müller (1)* zuweisen. Dieser Vorgang wird vom ConSol CM Web Client unterstützt. Benennen Sie dann *Hr. Müller (1)* in *Hr. Müller* um. Jetzt hat der Benutzer *Hr. Müller* den Übertragungsschlüssel des Importszenarios und beim nächsten Import tritt das Problem nicht mehr auf.
      Der allgemeine Anwendungsfall ist: Der Übertragungsschlüssel wird vom ConSol CM-System erzeugt und ermöglicht den Import und die Aktualisierung von Konfigurationsdaten.

- **CM-System-Properties**, deren Modulname mit *custom-* beginnt, werden in einer Installation während des Imports hinzugefügt (d. h. erstellt), wenn sie vorher noch nicht vorhanden waren. Falls eine kundenspezifische Property im importierenden System bereits vorhanden ist, wird der Wert aktualisiert, d. h. der importierte Wert wird gesetzt. Dieses Systemverhalten ist in CM-Versionen 6.11.0.5 und höher implementiert. In CM-Versionen 6.11.0.4 und niedriger wurde der Wert nicht überschrieben.

2. Wenn die Checkbox *Bestehende Daten löschen* **markiert** ist, wird das gesamte System gelöscht, d. h. es werden **alle** vorhandenen Daten gelöscht. Alle Daten bedeutet:
   - Konfigurationsdaten
   - Laufzeitdaten

   ⚠ Das heißt, wenn *Bestehende Daten löschen* ausgewählt ist, ist es nicht möglich, irgendetwas aus dem ursprünglichen Szenario zu behalten. **Es wird alles gelöscht! Nur** die System-Properties werden **nicht** gelöscht.

3. Import von Laufzeitdaten:
   - Der Export und Import von Laufzeitdaten ist für die Verwendung auf Test- oder Entwicklungssystemen gedacht. Für den Export und Import von größeren Datenmengen müssen ETL-Tools verwendet werden. Der Szenario-Export und -Import darf **nicht** verwendet werden.
   - Wenn ein Objekt bereits in der Zieldatenbank existiert (über den Transferschlüssel identifiziert), wird das Objekt in keiner Weise durch den Import verändert, d. h. es werden keine Felder des Objekts hinzugefügt oder gelöscht und bestehende Felder werden nicht geändert. Auf diese Weise sind vorhandene Objekte (Tickets, Kunden, Ressourcen) davor geschützt, versehentlich durch Testdaten verändert zu werden.

Für einen Importvorgang müssen folgende Parameter gesetzt sein:

- **Import-Archiv:**
  Geben Sie den Pfad und den Dateinamen der Datei an, aus der die Daten importiert werden sollen. Alternativ können Sie auf den Button *Ordner* klicken, um den Dateibrowser zu öffnen.
- **Modus:**
  Hier können Sie auswählen, wie der Import auf Fehler reagieren soll:
  - **Bei Fehlern abbrechen**
    Dieser Modus wird für Produktionssysteme empfohlen.
  - **Überspringen fehlerhafter Daten**
    Dieser Modus wird für Importe in Testsysteme empfohlen. Er kann auch für Produktionssysteme sinnvoll sein, da ein unerwarteter Fehler zu einem beschädigten System führen kann, der Import aber trotz eines aufgetretenen Fehlers fortgesetzt wird. Das Problem kann wahrscheinlich schnell gelöst werden und ein neuer Import würde länger dauern.
    Beispiel: Ein referenziertes Objekt wird nicht gefunden, z. B. beim Import einer Sicht, die eine nicht gefundene Queue referenziert.

- **Fehlerhafte Daten mit importieren**
  Wählen Sie diesen Modus nur, wenn Sie ein System mit fehlerhaften Daten duplizieren möchten, z. B. auf einem Entwicklungsserver oder wenn das Supportteam einen Fehler analysiert.

Klicken Sie danach auf *Import*, um den Datenimport zu starten.

> Achtung:
> Laufzeitdaten werden nur dann in die Zielfelder importiert, wenn diese Felder leer sind. Wenn ein Feld einen Eintrag enthält, wird dieser Eintrag beim Import des Szenarios nicht überschrieben. Verwenden Sie ETL-Operationen (extract - transform - load) für den Export/Import von Laufzeit-/Betriebsdaten. Dies ist kein Anwendungsfall für einen Export/Import eines ConSol CM-Szenarios.

Konfigurieren des Systemverhaltens beim Import eines Szenarios

Der Import der Konfigurationsdaten in einem Szenario wird in mehrere Datenbanktransaktionen aufgeteilt. Dadurch wird der Speicherverbrauch auf dem Datenbankserver verringert. Sollten allerdings während des Imports schwerwiegende Fehler auftreten, kann dies zu einem inkonsistenten Systemstatus führen.

Mit einer ConSol CM-System-Property kann das System so konfiguriert werden, dass alle Konfigurationselemente eines Szenario-Imports in einer großen Datenbanktransaktion gekapselt werden. Wenn diese Option aktiviert ist und während des Imports eines Konfigurationselements ein schwerwiegender Fehler auftritt, führt dies nicht mehr zu einem inkonsistenten Systemstatus. Stattdessen wird die gesamte Transaktion zurückgerollt und das System im Hinblick auf die Konfiguration wieder in den Zustand vor dem Import zurückgeführt. Die Kapselung der Transaktion schließt weder lokalisierte Strings noch Laufzeitdaten ein, die unabhängig von dieser Einstellung verarbeitet werden.

Verwenden Sie die ConSol CM-System-Property cmas-core-server, config.import.global.transaction.enabled.

> Achtung:
> Das Aktivieren der Transaktionskapselung kann zu längeren Ausführungszeiten beim Import und höherem Speicherbedarf auf dem Datenbankserver führen.

F.9.3.3 Transformation von Benutzernamen- und Passwortfeldern bei einem Import nach CM 6.11

In ConSol CM-Version 6.11 hat sich das Verhalten von Kundenfeldern (früher Datenobjektgruppenfeldern) geändert:

- Mit `username` annotiertes Kundenfeld
  Es wird jetzt überprüft, dass der Benutzername eindeutig ist.
- Mit `password` annotiertes Kundenfeld
  Die Passwörter werden nur als verschlüsselter String gespeichert.

Wenn Sie also ein Szenario aus einem ConSol CM-System mit einer Version unter 6.11 in ein System mit der Version 6.11 (oder höher) importieren, werden automatisch zwei Transformationen durchgeführt. Die Checkbox *Bestehende Daten löschen* des Importvorgangs hat keine Auswirkungen auf dieses Verhalten.

- **username:**
  Es wird das mit `username` annotierte Feld verwendet. Wenn in den Kundendaten mehrere identische Benutzernamen gefunden werden, werden Zahlen an die später gefundenen Feldwerte angehängt, z. B. *Huber_1* für den zweiten ursprünglichen Benutzernamen *Huber* und so weiter. Der Dialog nach dem Import zeigt eine Nachricht bezüglich der Änderungen der Benutzernamen an.
  Beachten Sie, dass in CM-Versionen 6.11 und höher konfiguriert werden kann, dass bei den Benutzernamen zwischen Groß- und Kleinschreibung unterschieden wird. Dies können Sie in der CM-System-Property cmas-core-security, policy.track.username.case.sensitive einstellen.
- **password:**
  Es wird das mit `password` annotierte Kundenfeld verwendet. Die Passwörter werden mit dem neuen Mechanismus als verschlüsselte Strings gespeichert.

F.9.3.4 Workflow-Installation (nur für die Fehlerbehebung nach der Installation)

Normalerweise werden alle Vorgänge im Zusammenhang mit der Workflow-Entwicklung und -Installation mit dem ***Process Designer*** durchgeführt. Wenn bei der Workflow-Installation ein Fehler aufgetreten ist, können Sie die Tickets, die nicht übertragen werden konnten, mit einer der folgenden Optionen in den neuen Workflow übertragen.

Wählen Sie zuerst die Queue(s) und danach den Übertragungsmodus:

- **Position im Prozess behalten**
  Das Ticket versucht, seine Position im Prozess beizubehalten:
  - Wenn die Aktivität und der Bereich nicht geändert wurden, ändert sich die Position des Tickets nicht.
  - Wenn die Aktivität nicht mehr vorhanden ist, kehrt das Ticket an die letzte konsistente Position im Prozess zurück.
- **Prozess neu starten**
  Das Ticket kehrt zum START-Knoten des Prozesses/Workflows zurück.

Eine detaillierte Beschreibung der Workflow-Installation finden Sie im *ConSol CM Process Designer Handbuch*.

F.10 Lizenzverwaltung

In diesem Kapitel werden folgende Themen behandelt:

F.10.1 Allgemeine Informationen über Lizenzen in ConSol CM 782

F.10.2 Abschnitte einer Lizenzdatei 783

F.10.3 Verwaltung der ConSol CM-Lizenz mit dem Admin Tool 784

F.10.4 Experteninformationen über den Zugriff auf den Inhalt der CM-Lizenzen 786

F.10.1 Allgemeine Informationen über Lizenzen in ConSol CM

Eine ConSol CM-Lizenzdatei ist eine Textdatei, die Einträge für mehrere Module enthält. Für jedes Modul ist die Anzahl der gültigen Lizenzen angegeben. Der folgende Ausschnitt aus einer Lizenzdatei zeigt den Abschnitt CONCURRENT_USERS des ConSol CM Web Clients. Es wurden zehn Lizenzen erworben.

```
[CONCURRENT_USERS]
contractParty = Demo-Licence ConSol
products = WEB_CLIENT,REST
version = 6.10
expirationDate = 31.12.2016
licenses = 10
signature = XXX
```

ConSol CM arbeitet mit der Anzahl der gleichzeitigen Benutzer (**concurrent users**, auch Floating-Lizenzen genannt), d. h. es wird erfasst, wie viele Benutzer gleichzeitig angemeldet sind. Die Benutzernamen werden nicht überprüft. Dies bedeutet, dass die Anzahl der Bearbeiter, die im Admin Tool verwaltet werden (siehe Abschnitt Bearbeiterverwaltung), nicht mit der Anzahl der Web-Client-Lizenzen übereinstimmen muss.

Eine Lizenz wird verbraucht, wenn sich der Benutzer anmeldet. Die Lizenz wird an den Server zurückgegeben, wenn die Session des Benutzers endet, d. h. wenn sich der Benutzer abmeldet oder wenn die Session vom Server automatisch beendet wird, weil das Timeout erreicht ist (siehe System-Property cmas-core-server, server.session.timeout).

F.10.2 Abschnitte einer Lizenzdatei

Eine ConSol CM-Lizenzdatei kann folgende Abschnitte enthalten. Alle Lizenzen sind Lizenzen für die **gleichzeitige** Nutzung, siehe obige Erklärung.

- [ADMINTOOL_USERS]
  Die Anzahl der Benutzer, die sich im CM Admin Tool anmelden können.
- [CONCURRENT_USERS]
  Die Anzahl der CM-Bearbeiter, die sich im Web Client anmelden können.
- [PROCESS_DESIGNER]
  Die Anzahl der Benutzer, die sich im CM Process Designer anmelden können.
- [TRACK]
  Die Anzahl der Kunden, die sich im Portal CM/Track anmelden können.
- [TRACK_USERS]
  Die Anzahl der Benutzerprofile für das Portal CM/Track. Dies ist die Anzahl der Bearbeiter, die in der Bearbeiterverwaltung als *Track* gekennzeichnet sind.
- [REST_USERS]
  Die Anzahl der Benutzer, die auf die REST API zugreifen können. Die Anzahl der TRACK_USERS ist darin nicht enthalten (CM/Track benutzt die REST API ebenfalls.)
  Auch CM/Phone braucht REST-Lizenzen: eine Lizenz pro Client (PC/Laptop), auf dem CM/Phone installiert und aktiv ist.

Da mehrere (oder sogar eine große Anzahl) an Kunden dasselbe Benutzerprofil in CM/Track verwenden können, kann die Anzahl der Lizenzen für [TRACK] und [TRACK_USERS] deutliche Unterschiede aufweisen. Es kann zum Beispiel sein, dass es zwei Benutzerprofile gibt und sich 1000 Kunden gleichzeitig bei CM/Track anmelden können. Dafür sind dann 2 Lizenzen [TRACK_USERS] und 1000 Lizenzen [TRACK] erforderlich. Eine detaillierte Erklärung dazu finden Sie in CM/Track V1: Systemzugang für CM/Track-Benutzer (Kunden) oder CM/Track V2: Systemzugang für CM/Track-Benutzer (Kunden).

F.10.3 Verwaltung der ConSol CM-Lizenz mit dem Admin Tool

Sie müssen in der Navigationsgruppe *System*, Navigationselement *Lizenz* eine gültige Lizenz für Ihr ConSol CM-System importieren. Sie erhalten eine Lizenz für das Test- und/oder Produktivsystem, wenn Sie die entsprechenden Softwareverträge mit ConSol unterschrieben haben. Wenn die Lizenz abgelaufen ist, aber bald ablaufen wird, können Sie eine neue Lizenzdatei importieren. Das Admin Tool startet natürlich immer, auch dann wenn die Lizenz abgelaufen ist. Die Anmeldung im Web Client, CM/Track und Process Designer ist bei abgelaufener Lizenz nicht möglich.

Fragen Sie Ihren Consultant nach näheren Informationen. Die Lizenz ist eine Datei im reinen Textformat. Sie kann während des ConSol CM-Betriebs geändert werden. Ein Herunterfahren des Systems ist nicht erforderlich.

⚠ Es gibt keinen Button *Zurück*, mit dem Sie Ihre Änderungen rückgängig machen können, wenn Sie im Feld *Lizenz* Text ändern oder löschen. Wenn Sie versehentlich Teile der Lizenz geändert haben, schließen Sie das Admin Tool **ohne** auf *Speichern* zu klicken. Damit werden alle Änderungen an der Lizenzdatei verworfen. Wenn Sie das Admin Tool danach neu starten, befindet sich die Lizenz wieder in dem Zustand, den sie vor Ihren Änderungen hatte.

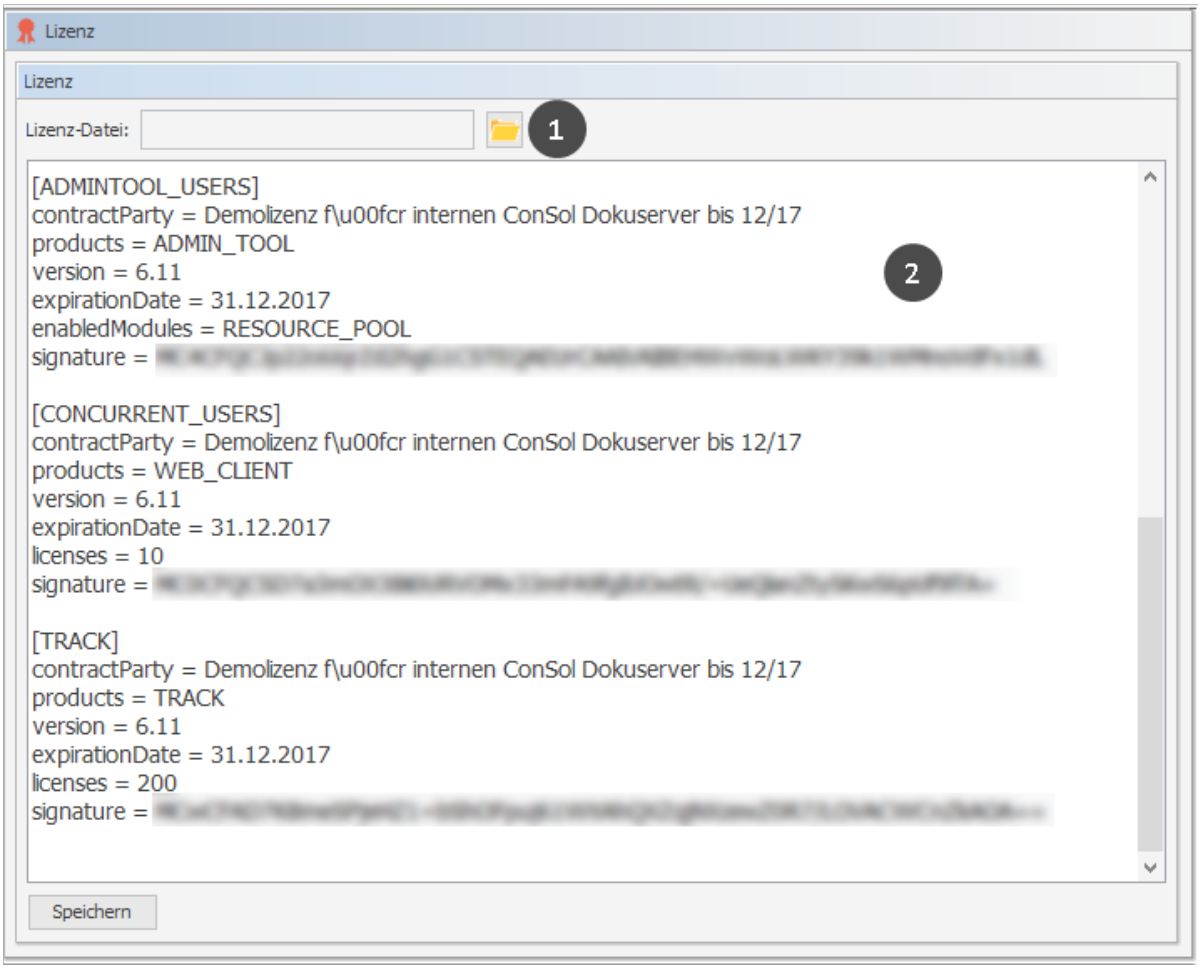


Abbildung 498: *ConSol CM Admin Tool - System, Lizenz*

Importieren Sie die ConSol CM-Lizenzdatei auf einem der beiden folgenden Wege:

- Laden Sie die Lizenz mithilfe des Dateibrowsers neben dem Feld *Lizenz-Datei*. Klicken Sie auf *Speichern*. (1)
- Fügen Sie den kompletten Text der Lizenzdatei durch Kopieren und Einfügen ein. Falls eine alte Lizenz vorhanden ist, ersetzen Sie einfach den gesamten Text. Klicken Sie auf *Speichern*. (2)

Es sollte eine Meldung angezeigt werden, dass die Lizenz erfolgreich ins System importiert wurde. Sie ist sofort aktiv, es ist keine weitere Aktion erforderlich.

F.10.4 Experteninformationen über den Zugriff auf den Inhalt der CM-Lizenzen

Der Inhalt der CM-Lizenzen kann über die MBean `licenseDeployer` abgefragt werden. Diese MBean bietet drei Methoden:

- getRemainingDays
- deployLicense
- getLicenseInfo

Wenden Sie sich an Ihren CM-Consultant, wenn Sie an dieser Funktion interessiert sind und Unterstützung beim Implementieren eines Systems, das den Zugriff über MBean verwendet, haben möchten, z. B. um ein Monitoring für Ihr CM-System aufzusetzen.

Die folgende Abbildung zeigt ein Beispiel mit JConsole.

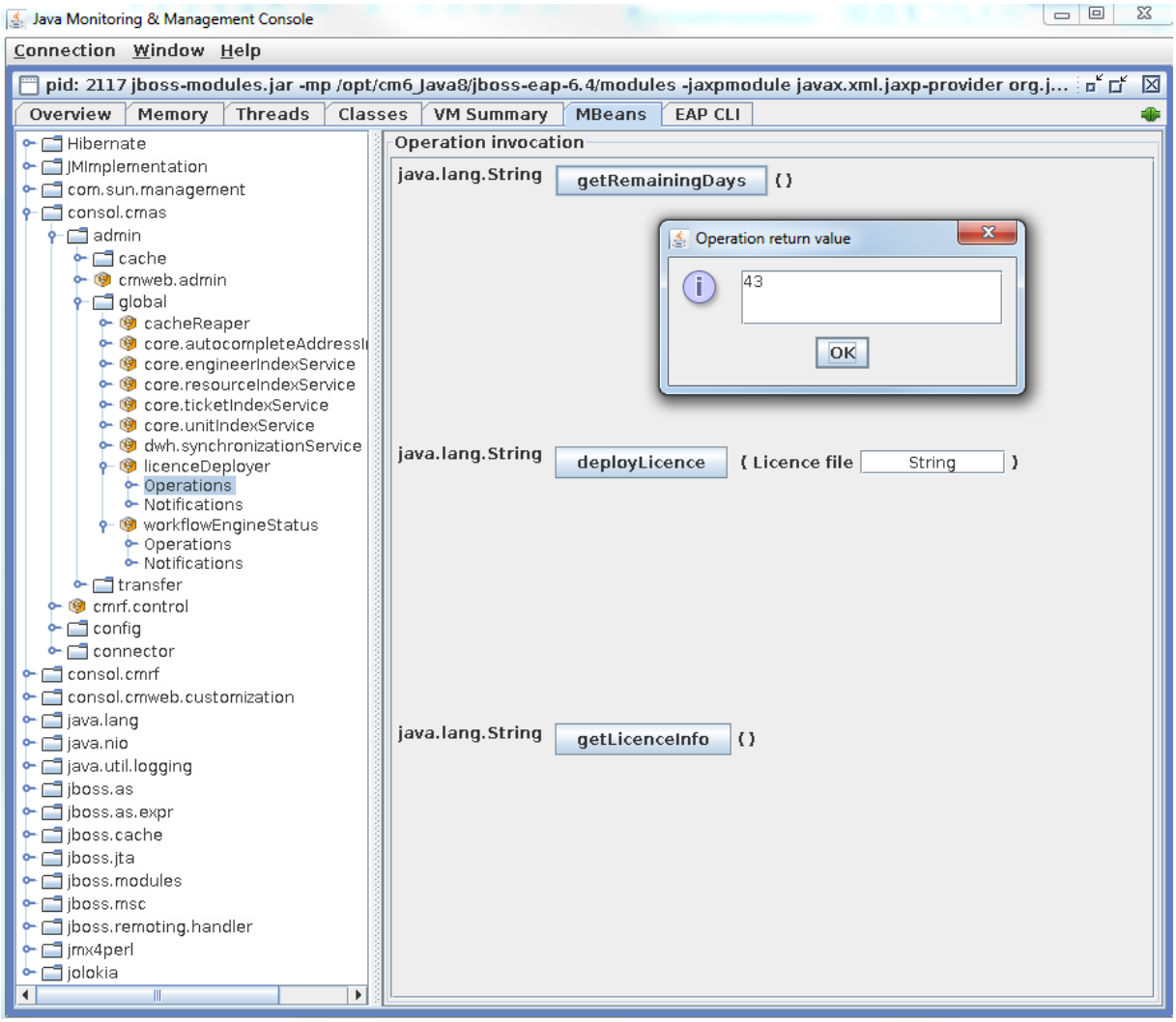


Abbildung 499: *Abrufen der verbleibenden Tage einer CM-Lizenz mit JConsole*

Wenn Sie eine CM-Lizenz mit einem beschränkten Gültigkeitszeitraum gekauft haben, empfehlen wir es, ein Lizenz-Monitoring aufzusetzen, das eine bestimmte Zeit vor Ablauf der Lizenzen eine Benachrichtigung sendet.

F.11 System-Properties

In diesem Kapitel werden folgende Themen behandelt:

F.11.1 Einleitung 788

F.11.2 Übersicht über die System-Properties 789

F.11.3 Setzen der System-Properties 792

F.11.4 Programmieren mit System-Properties 793

F.11.1 Einleitung

Im Navigationselement *System-Properties* der Navigationsgruppe *System* können Sie Systemeinstellungen, die sogenannten *System-Properties*, für die ConSol CM-Applikation hinzufügen, ändern oder löschen. System-Properties werden zum Beispiel verwendet, um die Werte für die Anzahl der Sekunden vor dem Timeout der Session, die E-Mail-Adressen der Administratoren oder die Größe der Suchseite zu speichern.



ARBEITEN SIE NUR IN DIESEM NAVIGATIONSELEMENT, WENN SIE GENAU WISSEN, WAS SIE TUN!

Auf dieser Seite haben Sie Zugriff auf grundlegende Systemeinstellungen, die *System-Properties*.

Ändern, editieren oder löschen Sie nur dann Werte von System-Properties, wenn Sie genau wissen, welche Auswirkungen dies hat.

F.11.2 Übersicht über die System-Properties

Eine System-Property hat folgende Parameter:

- **Modul**
  Pflichtangabe. Gibt an, in welchem ConSol CM-Modul die System-Property verwendet wird.
- **Schlüssel** (Name)
  Pflichtangabe. Der Name der System-Property. Mit diesem Namen wird die System-Property im System referenziert.
- **Typ**
  Pflichtangabe. Datentyp des Werts der System-Property.
  - **Text**
    Ein normales String-Feld.
  - **Passwort**
    Ein Feld, das ein Passwort enthält und daher nicht als Klartext angezeigt wird.
  - **E-Mail**
    Ein Feld, das eine E-Mail-Adresse enthält, d. h. das Format muss dem Standardformat einer E-Mail-Adresse entsprechen (<Adresse>@<Domain>).
  - **Ja/Nein**
    Ein Boolean-Feld (true/false)
  - **Ganzzahl**
    Eine Feld für Ganzzahlen (ohne Komma). Wird zum Beispiel für Zeitintervalle oder die Anzahl der Neustarts verwendet.
- **Wert**
  Der Wert der System-Property. Muss entsprechend dem angegebenen Datentyp gesetzt werden.
- **Beschreibung**
  Optional. Ein Beschreibungstext für die System-Property.
  Wenn Sie kundenspezifische System-Properties hinzufügen, die Ganzzahlen für Zeitintervalle enthalten, empfehlen wir Ihnen dringend, in der Beschreibung die Einheit festzuhalten, da es einen großen Unterschied macht, ob eine Eskalation z. B. nach fünf Minuten, fünf Stunden oder fünf Tagen erfolgt!
- **Neustart erforderlich**
  Boolean-Feld (Checkbox), mit dem festgelegt wird, ob die Änderung an einer System-Property erst nach dem Neustart des Systems wirksam wird.
- **Optional**
  Boolean-Feld (Checkbox), mit dem festgelegt wird, ob die System-Property vorhanden sein muss oder optional ist. Es wird folgendes Verhalten konfiguriert:
  - optional = „true“:
    - Der Wert der System-Property darf NULL sein.
    - Die System-Property kann gelöscht werden, z. B. mit dem Admin Tool.

  - optional = „false“:
    - Der Wert der System-Property darf nicht NULL sein.
    - Die Property kann nicht gelöscht werden, d. h. im Feld *Wert* muss ein Wert stehen.
- **REST zugänglich**
  Boolean-Feld. Ab CM-Version 6.11 kann der Wert einer System-Property über die REST-API abgerufen werden, wenn diese Checkbox auf „true“ gesetzt ist. Detaillierte Informationen über die REST-API und die benötigten Befehle finden Sie in der *ConSol CM REST API Dokumentation*.

  ⚠ Überlegen Sie es sich zweimal, bevor Sie den Wert auf „true“ setzen. Über REST reicht ein normales Systemkonto aus, um Daten abzurufen.

Die System-Properties werden in einer Tabelle verwaltet, die Spalten für drei der Parameter einer System-Property hat:

- **Modul**
  Name des Moduls
- **Schlüssel**
  Der Name der System-Property.
- **Wert**
  Der Wert der System-Property.

Sie können die Tabelle nach einer Spalte sortieren, indem Sie auf die Kopfzeile der Spalte klicken. Mit einem zweiten Klick können Sie die Sortierreihenfolge umdrehen.

Das folgende Beispiel zeigt das Popup-Menü, das zum Setzen einer System-Property verwendet wird, in diesem Fall `cmas-core-server, server.session.timeout`. Sie gelangen auf diese Seite, indem Sie in der Navigationsgruppe *System* das Navigationselement *System-Properties* öffnen und die Property im Bearbeitungsmodus auswählen.

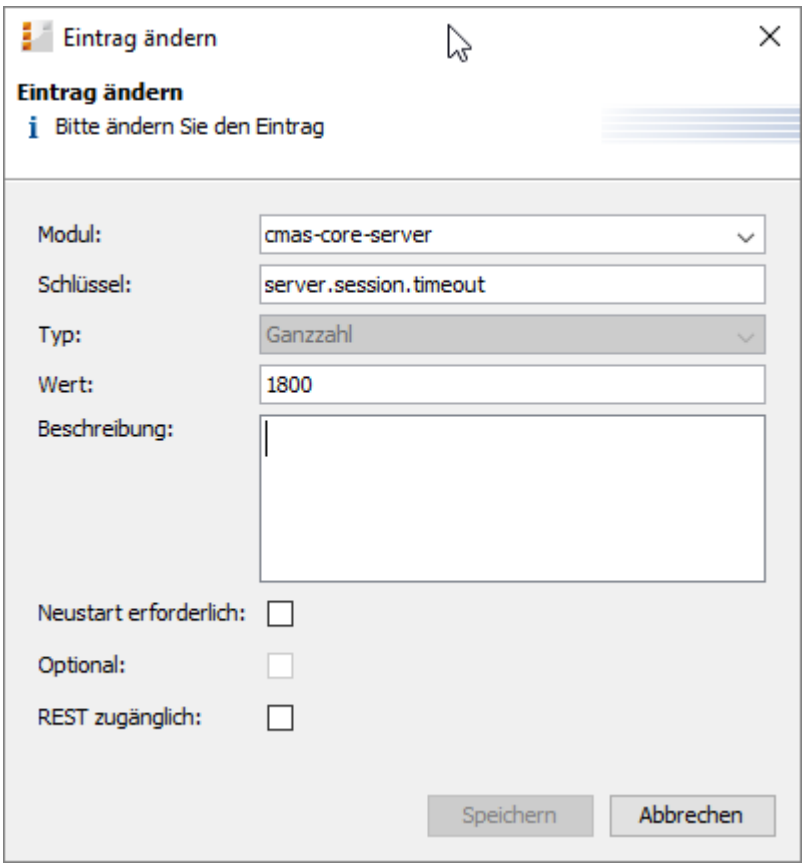


Abbildung 500: *ConSol CM Admin Tool: System, System-Properties: Editieren einer System-Property*

F.11.3 Setzen der System-Properties

Die Werte für die System-Properties werden auf unterschiedliche Art und Weise gesetzt. System-Properties können ...

- vom ConSol CM-System automatisch mit Standardwerten gefüllt werden, z. B. `cmweb-server-adapter, ticketListRefreshIntervalInSeconds` wird auf 180 Sekunden gesetzt. Sie können die Werte wenn nötig ändern.
- vom System vorbereitet werden, d. h. die System-Properties sind in der Liste vorhanden, aber nicht mit den erforderlichen Werten gefüllt. Die System-Properties, die sich auf LDAP beziehen, werden zum Beispiel nur (manuell durch einen Administrator) gefüllt, wenn die LDAP-Authentifizierung konfiguriert wird.
- durch Konfigurationen im Admin Tool gefüllt werden, die Sie auf der grafischen Benutzeroberfläche des Admin Tools vornehmen. Wenn Sie zum Beispiel die Namen der Postfächer in der *E-Mail-Konfiguration* eingeben, werden die Werte in den entsprechenden NIMH-System-Properties gesetzt, z. B. `cmas-nimh, mailbox.0.connection.host`. In diesen Fällen sollten Sie die System-Property nicht direkt editieren, sondern immer die grafische Konfiguration nutzen!
- in einer Standardinstallation gar nicht vorhanden sein. In diesem Fall müssen sie manuell hinzugefügt werden, z. B. die Aktivierung des TEF (Task Execution Framework) durch Hinzufügen der System-Property `cmas-app-admin-tool, start.groovy.task.enabled` und Setzen des Wertes auf „true“.
- als kundenspezifische System-Properties manuell zu einem System hinzugefügt werden. Wenn der Modulname mit `custom-` beginnt, werden diese System-Properties in einem Szenario mit exportiert (siehe Abschnitt Deployment (Import/Export)). Sie können Ihre eigenen System-Properties definieren und hinzufügen, um zum Beispiel Eskalationszeiten zu verwalten. Auf diese Weise können Sie die Werte in der Verwaltung der System-Properties speichern und ein Administrator ohne Programmierkenntnisse kann diese ändern. Im Skriptcode, in dem die System-Properties referenziert sind, wird nur der Name der Property verwendet. Dies ist im folgenden Abschnitt erklärt.

F.11.4 Programmieren mit System-Properties

Um eine System-Property in einem Groovy-Skript zu verwenden, d. h. den Wert der Property abzurufen, verwenden Sie folgende Klasse und Methode:

- `configurationService.getValue(String pModule, String pProperty).`

Zum Abrufen einer speziellen Eskalationszeit können Sie z. B. folgenden Code verwenden:

```
def mytime = configurationService.getValue("custom-
 servicedesk","escalation.time.medium2")
```

Damit wird der Wert „10“ abgerufen.

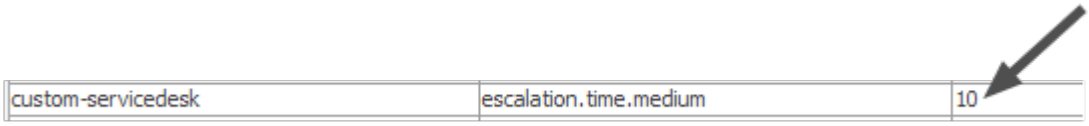


Abbildung 501: *ConSol CM Admin Tool - System, System-Properties: Kundenspezifische System-Property*

F.12 Arbeiten mit Textvorlagen

Textvorlagen sind vordefinierte Texte, die ein Bearbeiter öffnen und direkt verwenden oder ändern kann. Textvorlagen können für E-Mails oder Ticketkommentare verwendet werden, wo Text, Kopf- und Fußzeile angegeben werden können. Ein anderes Beispiel sind Dokumente, die mit Microsoft Word oder OpenOffice bearbeitet werden.

In beiden Fällen enthalten die Vorlagen nicht nur Text sondern auch bestimmte Datenfelder, die mit Daten aus dem Ticket gefüllt werden, wie zum Beispiel dem Kundennamen oder dem Ticketthema.

ConSol CM hat zwei Module, die Textvorlagen bereitstellen:

- Der Textvorlagen-Manager zum Bearbeiten und Verwalten von Vorlagen für E-Mails und Ticketkommentare (siehe Abschnitt Der ConSol CM Textvorlagen-Manager)

und

- CM/Doc mit dem Dokumentvorlagen-Manager zum Bearbeiten und Verwalten von Microsoft Word- und OpenOffice-Vorlagen (siehe Abschnitt CM/Doc).

F.12.1 Der ConSol CM Textvorlagen-Manager

F.12.1.1 Einführung in die Arbeit mit E-Mail- und Tickettext-Vorlagen

Mit dem ***Textvorlagen-Manager*** können zwei Arten von Vorlagen definiert werden:

- **E-Mail-Vorlagen** für E-Mails, die aus dem ConSol CM-System geschrieben werden:
  - manuelle E-Mails (von einem Bearbeiter mit dem Ticket-E-Mail-Editor geschrieben)
  - automatische E-Mails (z. B. von einem Workflow-Skript gesendet, wenn eine bestimmte Workflow-Aktivität ausgeführt wird)
- **Textvorlagen** für Tickettexte (Kommentare)
  - beim Erstellen eines Tickets
  - beim Bearbeiten eines Tickets

E-Mail-Vorlagen

Wozu dienen E-Mail-Vorlagen?

Bei einem System, das mit E-Mails arbeitet, müssen mehrere Kriterien beachtet werden. Wenn all diese Anforderungen erfüllt sind, sind E-Mail-Vorlagen ein hilfreiches Werkzeug für die tägliche Arbeit.

- Die E-Mails müssen ein klar definiertes Layout haben, normalerweise entsprechend dem Corporate Design des Unternehmens.
- Die Texte müssen den Unternehmensrichtlinien für Briefe bzw. Texte entsprechen.
- Häufig verwendete Texte müssen in Vorlagen zur Verfügung gestellt werden, um Zeit zu sparen und Tippfehler und andere Fehler bei der Texteingabe zu vermeiden.
- Kunden-, system- und bearbeiterspezifische Daten müssen in den Text integriert sein.
- Für die Verwaltung der Vorlagen sollte ein Administrator bzw. ein Power User zuständig sein. Es sollte keine Systemkonfiguration durch die Firma erforderlich sein.

ConSol CM liefert den nötigen Funktionsumfang, der all diese Kriterien berücksichtigt.

E-Mails in ConSol CM

E-Mails werden für Kernfunktionen in ConSol CM eingesetzt. Details zu diesen Funktionen sind im Abschnitt E-Mail-Funktionen in ConSol CM beschrieben. Daher finden Sie an dieser Stelle nur einen kurzen Überblick.

ConSol CM kann E-Mails empfangen und senden. Das Senden von E-Mails kann verschiedenen Zwecken dienen:

- **Ein Bearbeiter schreibt direkt aus dem Ticket eine E-Mail und verwendet dazu den Ticket-E-Mail-Editor.**
  Dies kann eine E-Mail an den Kunden, an einen Kollegen oder an eine beliebige andere Person mit einer gültigen E-Mail-Adresse sein. Häufig gibt es Standardtexte, die jeden Tag für mehrere Empfänger verwendet werden. Um nicht immer wieder den gleichen Text eingeben zu müssen, stellt ConSol CM E-Mail-Vorlagen bereit. Das sind Textvorlagen, in die Parameter wie der Kundenname, die Ticketnummer oder der Name und die Telefonnummer des Bearbeiters integriert

werden können. Wenn die Vorlage verwendet wird, füllt das System die Parameter automatisch mit gültigen Daten aus dem aktuellen Ticket. Der Bearbeiter kann zusätzlichen Text eingeben oder den Text nach Bedarf ändern, da die E-Mail-Vorlagen nicht statisch sondern dynamisch sind.
E-Mails, die manuell gesendet werden, verwenden entweder gar keine Vorlage oder eine Vorlage aus dem Textvorlagen-Manager. Vorlagen aus dem Bereich *Skripte und Templates* des Admin Tools sind hier nicht verfügbar.

- **Das System sendet eine E-Mail automatisch.**
  Dies kann eine interne E-Mail sein, zum Beispiel einer Erinnerung an den Bearbeiter, dass ein Ticket einen Eskalationsstatus erhalten hat, oder eine E-Mail an den Vorgesetzten, wenn ein Ticket eine Genehmigung braucht, bevor es den Prozess weiter durchlaufen kann. Es kann sich auch um eine externe E-Mail an einen Kunden handeln, zum Beispiel eine Empfangsbestätigung oder eine Benachrichtigung, dass ein Ticket gelöst wurde. Die E-Mail wird automatisch auf Basis der entsprechenden E-Mail-Vorlage erzeugt. Dies kann eine E-Mail-Vorlage aus dem Textvorlagen-Manager oder aus dem Bereich *Skripte und Templates* des Admin Tools sein.

Tickettext-Vorlagen

Wozu dienen Tickettext-Vorlagen?

Die Verwendung von Tickettext-Vorlagen, d. h. vordefinierten Textsegmenten, auf die Sie bei der Erstellung oder Bearbeitung eines Tickets zugreifen können, dient mehreren Zwecken

- Sie als Ticketbearbeiter sparen eine Menge Zeit, weil Sie den gleichen Text nicht immer wieder eingeben müssen.
- Sie laufen nicht Gefahr, wichtige Punkte zu vergessen (z. B. bei Fragen für eine Vorqualifizierung, wenn Sie mit dem Kunden telefonieren).
- Sie müssen sich keine Sorgen über Tippfehler machen.
- Sie müssen Ticket- bzw. Kundendaten nicht nachschlagen, weil die Daten automatisch in den Text integriert werden.

Tickettext-Vorlagen in ConSol CM

Tickettext-Vorlagen werden auf ähnliche Weise definiert wie E-Mail-Vorlagen. Es wird lediglich der Parameter *Verwendung in* anders gesetzt, wenn die Vorlage im Textvorlagen-Manager erstellt wird.

> ⚠ Technisch gesehen gibt es keinen Unterschied zwischen E-Mail- und Tickettext-Vorlagen! Für jede Vorlage können Sie als Vorlagenmanager im Textvorlagen-Manager entscheiden, ob die Vorlage als Tickettext-Vorlage, als E-Mail-Vorlage oder für beide Fälle verwendet werden soll.

E-Mail- und Tickettext-Vorlagen in ConSol CM

Komponenten von E-Mail- und Tickettext-Vorlagen

In E-Mail- und Tickettext-Vorlagen in ConSol CM können Sie Freitext und alle Daten, die für einen Kunden, Bearbeiter bzw. das Ticket verfügbar sind, verwenden. Im Abschnitt Die Bibliothek der Templatefelder sind alle verfügbaren Komponenten erklärt.

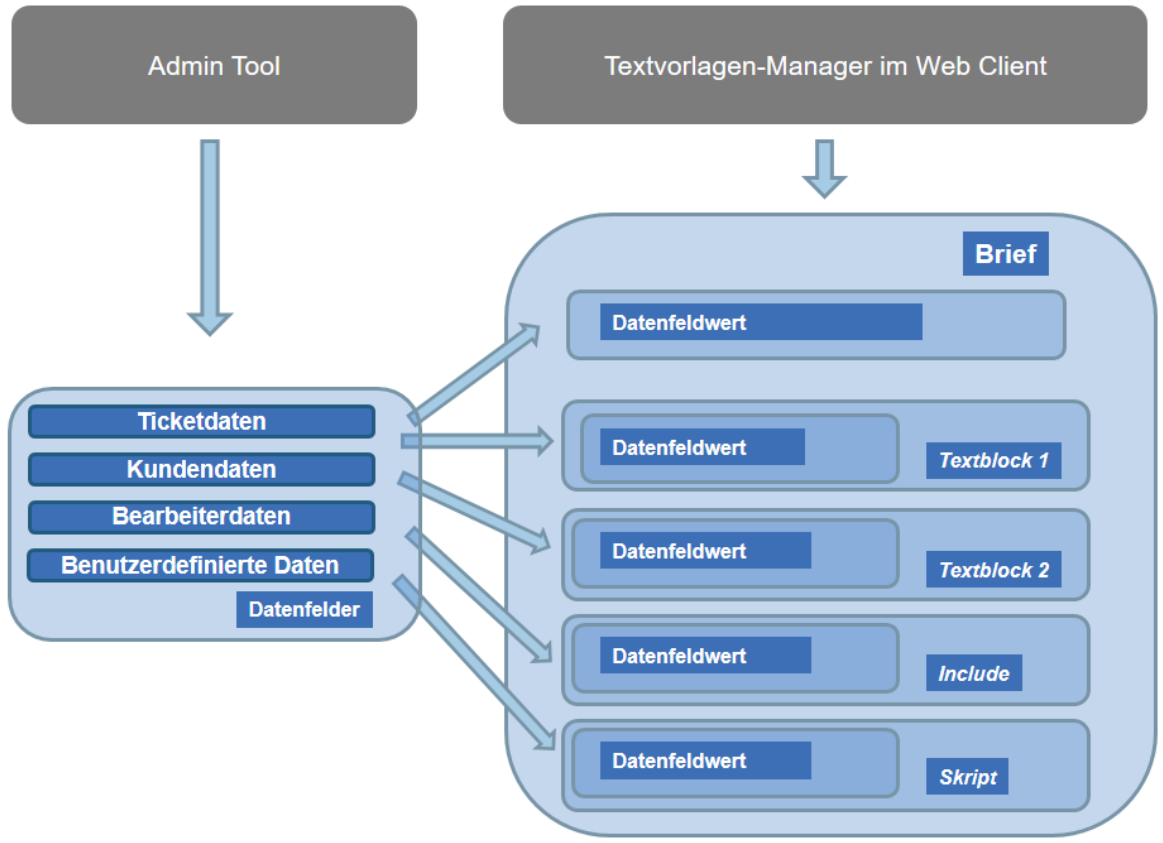


Abbildung 502: *ConSol CM - Verfügbare Komponenten / Daten für E-Mail-Vorlagen*

Eine detaillierte Beschreibung der Verwendung von Funktionen zur Ticketbearbeitung und des Ticket-E-Mail-Editors finden Sie im *ConSol CM Benutzerhandbuch* in den Abschnitten *Erstellen eines neuen Tickets*, *Bearbeiten von Tickets* und *Versenden von E-Mails*.

Speichern und Verwalten von E-Mail- und Tickettext-Vorlagen

E-Mail-Vorlagen

E-Mail-Vorlagen werden an zwei unterschiedlichen Stellen in ConSol CM gespeichert und verwaltet:

1. Im Textvorlagen-Manager
2. Im Navigationselement *Skripte und Templates* des Admin Tools (wird nicht hier sondern im entsprechenden Abschnitt dieses Handbuchs behandelt, siehe Admin-Tool-Templates)

Tickettext-Vorlagen

Tickettext-Vorlagen werden an zwei unterschiedlichen Stellen in ConSol CM gespeichert und verwaltet:

1. Im Textvorlagen-Manager (hier werden Tickettext-Vorlagen verwaltet)
2. In ***CM/Doc*** (hier werden Microsoft Word- und OpenOffice-Dokumente für CM/Doc gespeichert, siehe Abschnitt CM/Doc)

F.12.1.2 Einführung in den Textvorlagen-Manager

Der ConSol CM Textvorlagen-Manager ist ein auf dem Web Client basierendes Tool für die Erstellung und Verwaltung von E-Mail- und Tickettext-Vorlagen. Siehe Abschnitt Arbeiten mit dem Textvorlagen-Manager.

Die folgende Abbildung zeigt den Textvorlagen-Manager. Sie gelangen auf diese Seite, indem Sie im Web Client auf den Link *Textvorlagen* klicken.

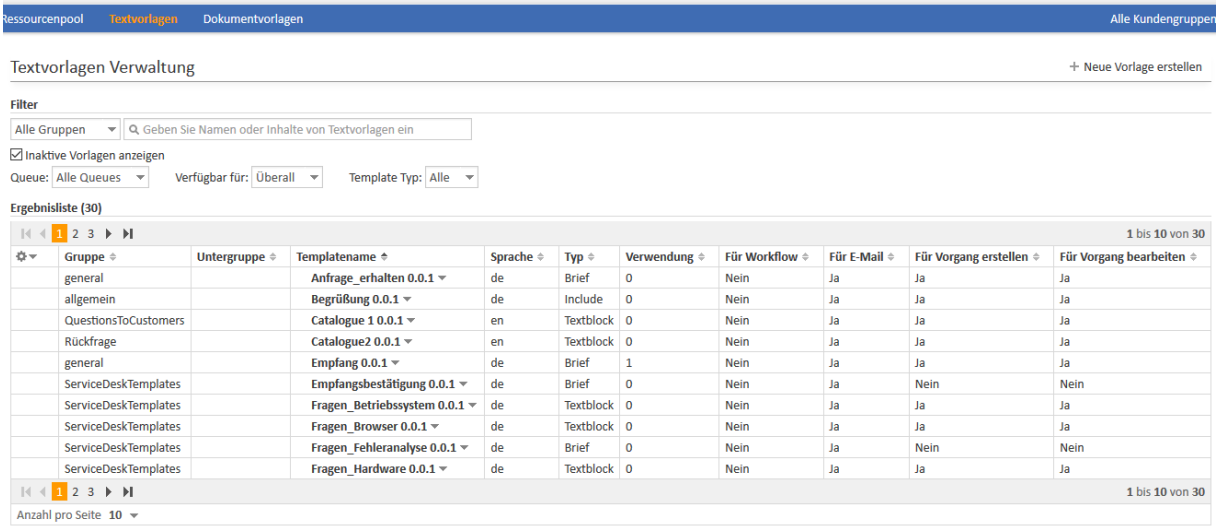


Abbildung 503: *ConSol CM Web Client - Textvorlagen-Manager*

Jeder Benutzer, dem eine Rolle mit der Berechtigung *Template speichern* zugewiesen ist, kann auf das Element *Textvorlagen* im Hauptmenü zugreifen (und den Textvorlagen-Manager öffnen).

Die folgende Abbildung zeigt die Seite der Rollenverwaltung. Sie gelangen auf diese Seite, indem Sie das Navigationselement *Rollen* der Navigationsgruppe *Zugriff und Rollen* öffnen und eine Rolle auswählen.

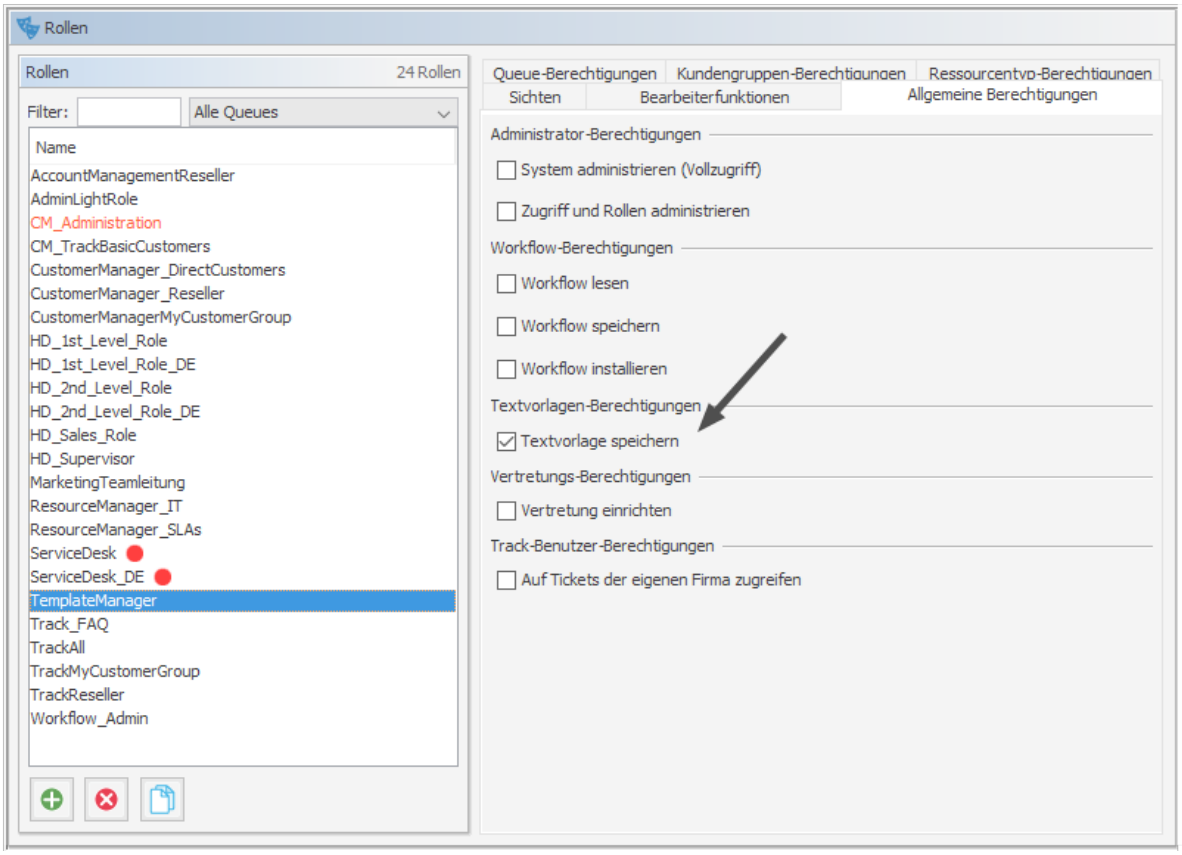


Abbildung 504: *ConSol CM Admin Tool - Zugriff und Rollen, Rollen: Berechtigungen für die Rolle TemplateManager*

Wir empfehlen Ihnen, eine Rolle (z. B. *TemplateManager*) zu erstellen, die nur die Berechtigung *Template speichern* hat, und an die keine Queue-Berechtigungen oder anderen Berechtigungen vergeben werden. Jedem Benutzer, der Zugang zum Textvorlagen-Manager soll, kann diese Rolle zugewiesen werden. Auf diese Weise werden normale Berechtigungen (z. B. Queue- und Kundengruppen-Berechtigungen) und Berechtigungen für den Textvorlagen-Manager nicht vermischt und Sie können die Berechtigung für den Textvorlagen-Manager schnell vergeben und entziehen.

Wenn die Berechtigung erteilt wurde, hat der Benutzer Zugriff auf das Menüelement *Textvorlagen*.

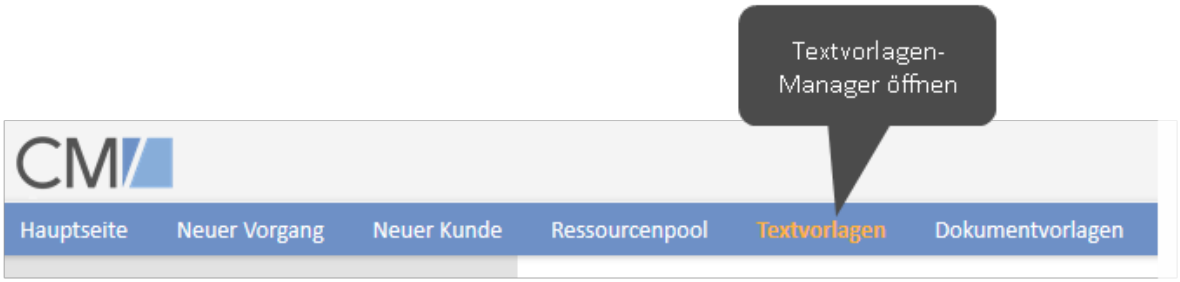


Abbildung 505: *ConSol CM Web Client - Hauptmenü mit Zugriff auf den Textvorlagen-Manager*

F.12.1.3 Arbeiten mit dem Textvorlagen-Manager

Die Textvorlagen-Verwaltung

Wenn Sie den Textvorlagen-Manager öffnen, wird die Seite *Textvorlagen Verwaltung* angezeigt:

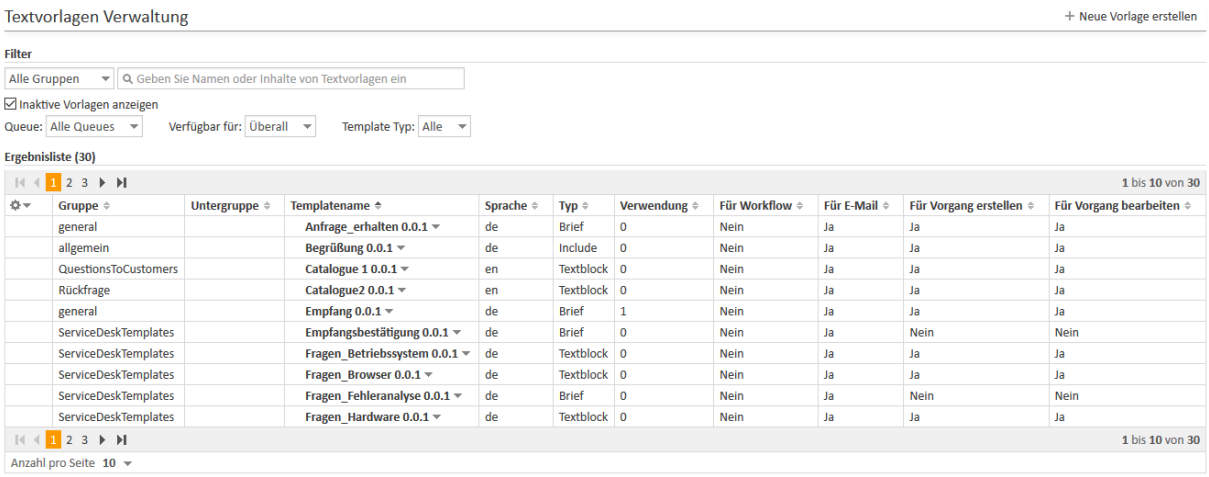


Abbildung 506: *ConSol CM Web Client - Textvorlagen-Manager*

Es wird eine Liste aller vorhandenen Vorlagen angezeigt. Wenn eine Vorlage Datenfelder (Ticketfelder oder Kundenfelder) verwendet, die nicht mehr verfügbar sind, wird die Zeile der entsprechenden Vorlage in Rot angezeigt. Vorlagen, die ein Attachment haben, sind mit dem Attachment-Icon (Büroklammer) gekennzeichnet. Inaktive Vorlagen werden in grau angezeigt.

Filter

Am Anfang werden alle Vorlagen in der *Ergebnisliste* angezeigt. Sie können die angezeigten Listeneinträge mit den Filtern im oberen Abschnitt der Seite filtern:

- **Gruppen/Untergruppen**
  Wählen Sie eine Gruppe oder Untergruppe in diesem Drop-down-Menü aus (eine Mehrfachauswahl ist nicht möglich). In der Liste der Vorlagen (*Ergebnisliste*) werden nur die Vorlagen angezeigt, die zu dieser Gruppe/Untergruppe gehören.

- **Textfilter**
  Geben Sie einen Vorlagennamen oder den Teil eines Vorlagennamens oder einen Teil des Vorlageninhalts ein. In der *Ergebnisliste* werden nur die Vorlagen angezeigt, die die angegebenen Kriterien erfüllen.
- **Inaktive Vorlagen anzeigen**
  Markieren Sie diese Checkbox, damit alle Vorlagen angezeigt werden, sowohl aktive Vorlagen als auch inaktive Vorlagen. Wenn diese Checkbox nicht markiert ist, werden nur die aktiven Vorlagen angezeigt.
- **Queue**
  Wählen Sie eine Queue aus der Liste (eine Mehrfachauswahl ist hier nicht möglich). In der *Ergebnisliste* werden nur die für die ausgewählte Queue verfügbaren Vorlagen angezeigt.
- **Verfügbar für**
  Wählen Sie einen der Werte aus der Liste (eine Mehrfachauswahl ist hier nicht möglich). In der *Ergebnisliste* werden nur die Vorlagen angezeigt, die das angegebenen Kriterium erfüllen. Mögliche Werte sind:
  - **Überall** (Standardwert)
    Es werden alle Vorlagen angezeigt.
  - **E-Mail**
    In der *Ergebnisliste* werden nur Vorlagen angezeigt, die für E-Mails verfügbar sind.
  - **Ticket erzeugen**
    In der *Ergebnisliste* werden nur Vorlagen angezeigt, die als Textvorlagen bei der Ticketerstellung verfügbar sind.
  - **Ticket bearbeiten**
    In der *Ergebnisliste* werden nur Vorlagen angezeigt, die als Textvorlagen bei der Ticketbearbeitung verfügbar sind.
  - **Workflow**
    In der *Ergebnisliste* werden nur Vorlagen angezeigt, die als Workflow-Vorlagen gekennzeichnet sind.

  In den folgenden Abschnitten ist beschrieben, welche Auswirkungen die Definition der Verfügbarkeit einer Vorlage hat.

  Die Verfügbarkeit der einzelnen Vorlagen ist auch in der entsprechenden Spalte der *Ergebnisliste* vermerkt (*Für E-Mail*, *Für Ticket erstellen*, *Für Ticket bearbeiten*, *Für Workflow*), siehe Erklärung unten.
- **Template Typ**
  Wählen Sie einen der Werte aus der Liste (eine Mehrfachauswahl ist hier nicht möglich). In der *Ergebnisliste* werden nur die Vorlagen angezeigt, die das angegebenen Kriterium erfüllen. Mögliche Werte sind:
  - **Alle** (Standard)
  - **Brief**

- **Include**
- **Textblock**
- **Skript**

Alle Typen sind detailliert in den folgenden Abschnitten beschrieben.

Liste der Vorlagen / Ergebnisliste

Die Liste hat folgende Spalten. Sie können die Tabelle nach einer Spalte sortieren, indem Sie auf die Kopfzeile der Spalte klicken. Mit einem zweiten Klick können Sie die Sortierreihenfolge umkehren.

- **Gruppe**
  Pflichtfeld. Die Gruppe einer Vorlage hat keine technischen oder funktionalen Auswirkungen innerhalb des Textvorlagen-Managers. Sie dient lediglich zum Sortieren der Liste im Textvorlagen-Manager und im Web Client. Wenn in Ihrem CM-System eine große Anzahl an Vorlagen definiert ist, ist es daher sinnvoll Gruppen zu nutzen, um den Bearbeitern die Arbeit mit den Vorlagen im Ticket-E-Mail-Editor und Ticket-Kommentar-Editor zu erleichtern.
- **Untergruppe**
  Optional. Die Untergruppe einer Vorlage hat keine technischen oder funktionalen Auswirkungen innerhalb des Textvorlagen-Managers. Sie dient lediglich zum Sortieren der Liste im Textvorlagen-Manager und im Web Client. Wenn in Ihrem CM-System eine große Anzahl an Vorlagen definiert ist, ist es daher sinnvoll zusätzlich zu den Gruppen auch Untergruppen zu nutzen, um den Bearbeitern die Arbeit mit den Vorlagen im Ticket-E-Mail-Editor und Ticket-Kommentar-Editor zu erleichtern.
- **Templatename**
  Der Name der Vorlage. Der Name wird in Workflows verwendet, um die benötigte Vorlage zu referenzieren. Außerdem wird er im Web Client (Ticket-E-Mail-Editor und Ticket-Kommentar-Editor) bei der Auswahl der Vorlage angezeigt.
- **Sprache**
  Die Sprache, die bei der Erstellung der Vorlage ausgewählt wurde (kann geändert werden). Der Webbrowser des Bearbeiters zeigt die Vorlage für die Sprache an, die der Bearbeiter im Browser eingestellt hat. Wenn Sie eine Vorlage in anderen Sprachen benötigen, achten Sie darauf, diese richtig einzustellen.
- **Typ**
  Es gibt fünf unterschiedliche Typen von Vorlagen, die in den folgenden Abschnitten detailliert erklärt werden:
  - **Brief**
    Dies ist die Grundform einer Vorlage. Vorlagen des Typs *Brief* werden im Ticket-E-Mail-Editor und Ticket-Kommentar-Editor angeboten und können als E-Mail-Vorlagen im Workflow verwendet werden. Alle anderen Vorlagentypen sind Teilkomponenten eines *Briefs*.
  - **Include**
    Dies ist eine Teilkomponente des *Briefs*, die in Briefen verwendet werden kann. Auf diese Weise können Sie den gleichen Text in mehreren Vorlagen verwenden. Ein typisches Beispiel dafür ist die Signatur des Unternehmens, die in allen anderen Vorlagen verwendet wird. Die Signatur sollte als *Include* definiert werden und dann in alle anderen Vorlagen

(des Typs *Brief*) integriert werden, in denen die Signatur benötigt wird. Der Vorlagenadministrator muss die Signatur also nur an einer Stelle verwalten und kann sicher sein, dass sie in allen anderen Vorlagen richtig verwendet wird.

- **Workflow Include**
  Dies ist das Gleiche wie ein *Include* wird aber in Workflows verwendet.
- **Textblock**
  Dies ist eine Teilkomponente eines *Briefs*. Sie kann beim Schreiben von E-Mails markiert werden oder nicht, je nachdem ob der Text angezeigt werden soll oder nicht. Ein gutes Beispiel ist eine Erstanalyse in einem Helpdesk-Team, bei der jedem Kunden die gleichen Fragen gestellt werden. Ein Textblock kann Fragen zur Hardware enthalten und ein anderer Fragen zur Software. Je nach Zweck der E-Mail kann der Bearbeiter einen der beiden Textblöcke verwenden.
- **Skript**
  Dieser Vorlagentyp steht nur Administratoren zur Verfügung (d. h. Benutzern, die sich im Web Client mit einem Administratorkonto anmelden). Damit können „intelligente“ Vorlagen erstellt werden, wie z. B. Vorlagen, die, abhängig vom Wert des Feldes *salutation*, für männliche Kunden *Sehr geehrter Herr* und für weibliche Kunden *Sehr geehrte Frau* setzen.

- **Verwendung**
  Gibt an, wie oft die Vorlage verwendet wird.
- **Für Workflow**
  Boolean. Eine Vorlage kann als *Workflow-Vorlage* markiert werden. Sie ist dann im Ticket-E-Mail-Editor oder Ticket-Kommentar-Editor nicht verfügbar, kann aber vom Workflow in automatischen E-Mails verwendet werden.
- **Für E-Mail**
  Boolean. Alle Vorlagen, die als *Verfügbar in E-Mail* markiert sind, haben hier den Eintrag *Ja*.
- **Für Ticket erzeugen**
  Boolean. Alle Vorlagen, die als *Verfügbar in Ticket erzeugen* markiert sind, haben hier den Eintrag *Ja*.
- **Für Ticket bearbeiten**
  Boolean. Alle Vorlagen, die als *Verfügbar in Ticket bearbeiten* markiert sind, haben hier den Eintrag *Ja*.

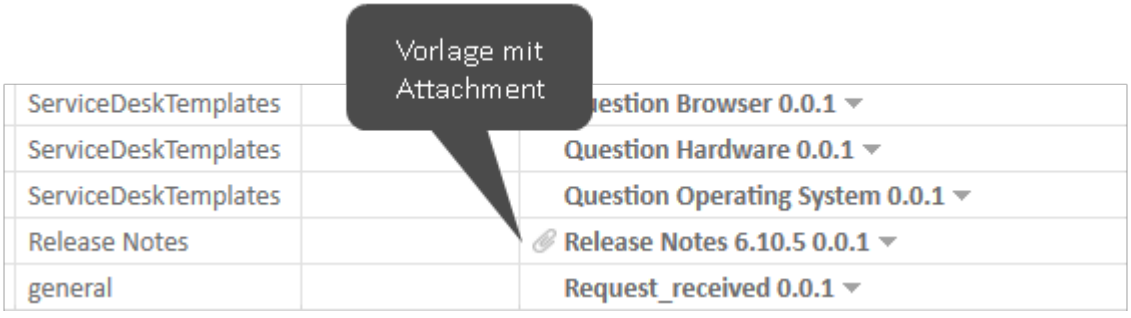


Abbildung 507: *ConSol CM Web Client - Textvorlagen-Manager: Vorlage mit Attachment*

Für jede Vorlage können Sie im Kontextmenü eine Aktion auswählen:

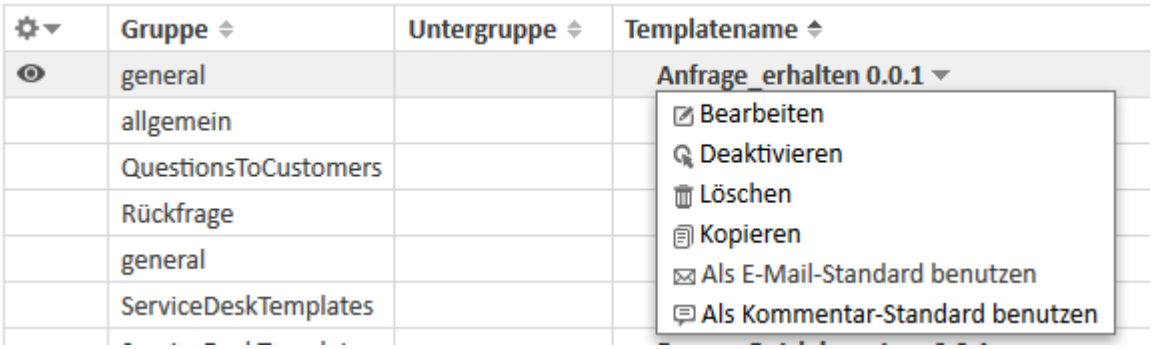


Abbildung 508: *ConSol CM Web Client - Textvorlagen-Manager: Kontextmenü einer Vorlage*

- **Bearbeiten**
  Bearbeiten Sie die Vorlage. Es sind die gleichen Funktionen verfügbar wie für die Erstellung einer neuen Vorlage.
- **Deaktivieren**
  (bzw. **Aktivieren** für deaktivierte Vorlagen)
  Nur aktivierte Vorlagen sind aktiv und im System verfügbar.
- **Löschen**
  Löschen Sie die Vorlage. Dies ist nicht möglich, wenn die Vorlage von einem Workflow verwendet wird oder wenn ein *Include* oder *Textblock* in anderen Vorlagen (*Briefen*) verwendet wird.
- **Kopieren**
  Erstellen Sie eine Kopie der Vorlage. In diesem Fall ist ein neuer Name erforderlich.
- **Als E-Mail-Standard benutzen**
  (oder **E-Mail-Standard entfernen** bei der aktuellen Standardvorlage)
  Es kann nur eine Vorlage als Standardvorlage für E-Mails verwendet werden. Diese wird automatisch in eine E-Mail eingefügt, die im Ticket-E-Mail-Editor geöffnet wird. Sie kann danach vom Bearbeiter gelöscht, oder in der E-Mail verwendet werden. Normalerweise ist als Standardvorlage eine Signatur oder eine Fußzeile definiert.
- **Als Kommentar-Standard benutzen**
  (oder **E-Mail-Standard entfernen** bei der aktuellen Standardvorlage)
  Es kann nur eine Vorlage als Standardvorlage für Kommentare verwendet werden. Diese wird automatisch in einen Kommentar eingefügt, der im Ticket-Kommentar-Editor geöffnet wird. Sie kann danach vom Bearbeiter gelöscht, oder für den Kommentar verwendet werden.


Eine Standardvorlage darf keine Textblöcke oder Variablen enthalten.

Die Vorlage wird in roter Schrift angezeigt, wenn sie Referenzen zu Feldern enthält, die dem System nicht bekannt sind. Bearbeiten Sie die Vorlage, um sicherzustellen, dass sich alle verwendeten Felder auf verfügbare Felder beziehen. Weitere Informationen über die Vorlagenfelder finden Sie im Abschnitt Die Bibliothek der Templatefelder.

Vorschaufunktion innerhalb der Ergebnisliste

Mit der Vorschaufunktion erhalten Sie einen schnellen Überblick über die Vorlagen, z. B. indem Sie in der Liste nach einem bestimmten Thema suchen und durch die Liste scrollen. Klicken Sie auf das Augensymbol in der ersten Spalte einer Zeile, um die Vorschau für die entsprechende Vorlage zu öffnen.

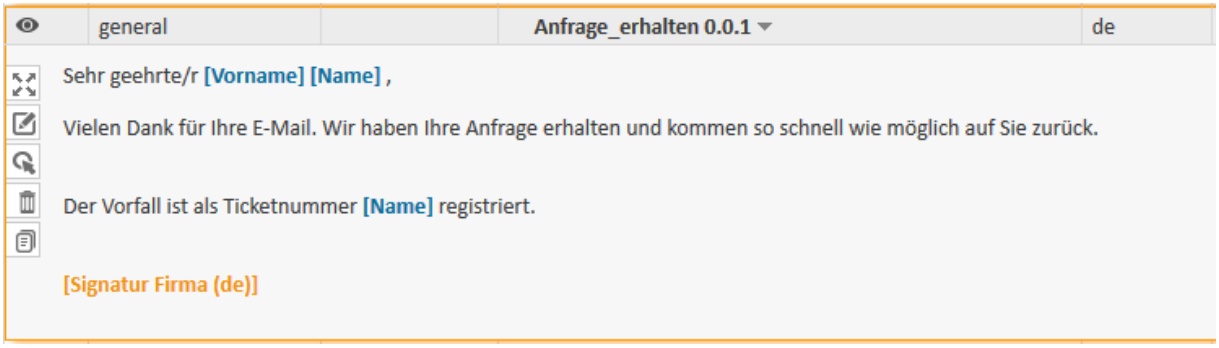


Abbildung 509: *ConSol CM Web Client - Textvorlagen-Manager: Vorschaufunktion*

Folgende Icons/Funktionen sind in der Vorschau verfügbar (fahren Sie mit der Maus über das Icon, um den Namen anzuzeigen):

- **Ansicht erweitern**
  Zeigt die gesamte Vorlage an.
- **Bearbeiten**
  Öffnet den Editiermodus. Details dazu finden Sie in den folgenden Abschnitten.
- **Deaktivieren**
  (bzw. **Aktivieren** für deaktivierte Vorlagen)
  Bietet die Möglichkeit, eine Vorlage schnell zu deaktivieren/aktivieren.
- **Löschen**
  Löscht die Vorlage. Dies ist nicht möglich, wenn die Vorlage von einem Workflow verwendet wird oder wenn ein *Include* oder *Textblock* in anderen Vorlagen (*Briefen*) verwendet wird.
- **Kopieren**
  Erstellt eine Kopie der Vorlage. In diesem Fall ist ein neuer Name erforderlich.
- **Attachment**
  Wenn die Vorlage ein oder mehrere PDF-Attachments enthält (in der obigen Abbildung nicht gezeigt), können Sie mit den beiden Buttons *Attachments* durch diese Attachments scrollen. Es wird eine Vorschau von jedem angehängten Dokument angezeigt.

Erstellen einer neuen Vorlage

Dies wird hier am Beispiel einer E-Mail-Vorlage gezeigt. Dasselbe Prinzip gilt auch für Tickettext-Vorlagen.

Erstellen eines neuen Briefs

Um eine neue Vorlage zu erstellen, klicken Sie im Textvorlagen-Manager auf dem Link *Neue Vorlage* erstellen. Auf der Seite *Neue Textvorlage erstellen* können Sie alle Parameter für die neue Vorlage eingeben. In unserem ersten Beispiel wird ein *Brief* erstellt, der als Empfangsbestätigung für den Kunden dient. Er kann automatisch vom Workflow gesendet werden oder im Web Client verwendet werden.

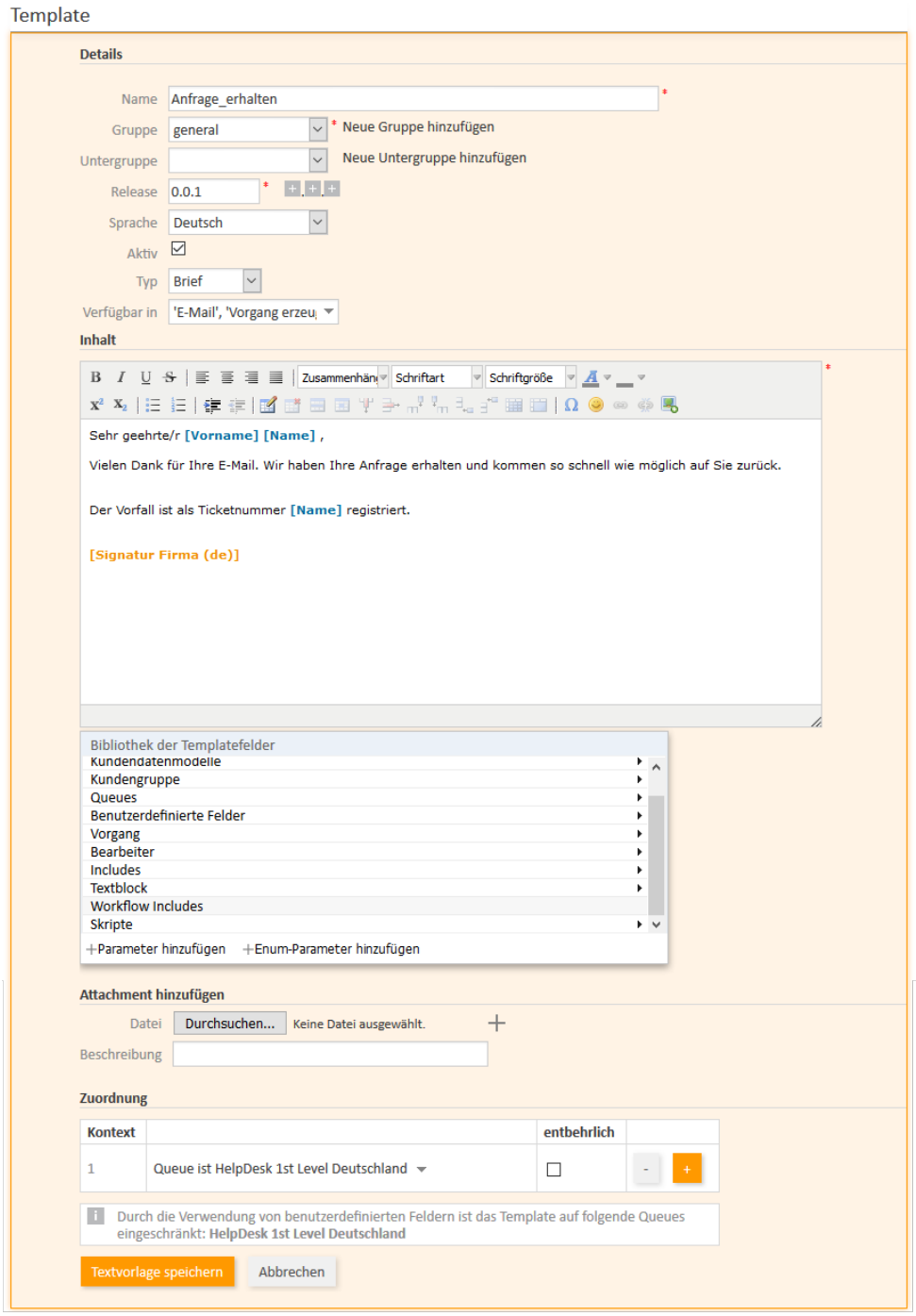


Abbildung 510: *ConSol CM Web Client - Neue Vorlage erstellen*

- **Name**
  Der Name der Vorlage. Dieser Name wird im Web Client im Ticket-E-Mail und Ticket-Kommentar-Editor angezeigt, und dient als technische Referenz, wenn eine Vorlage in einem Skript verwendet wird (Admin Tool oder Workflow).

- **Gruppe**
  Die Gruppe der Vorlage (siehe vorheriger Abschnitt). Sie können entweder eine vorhandene Gruppe verwenden oder eine neue Gruppe erstellen. Die Gruppe dient zum Sortieren der Vorlagen in der *Ergebnisliste* im Textvorlagen-Manager. Außerdem wird sie im Ticket-E-Mail-Editor und Ticket-Kommentar-Editor angezeigt, um den Bearbeitern die Arbeit zu erleichtern.
- **Untergruppe**
  Die Untergruppe der Vorlage (siehe vorheriger Abschnitt). Sie können entweder eine vorhandene Untergruppe verwenden oder eine neue Untergruppe erstellen. Die Untergruppe dient zum Sortieren der Vorlagen in der *Ergebnisliste* im Textvorlagen-Manager. Außerdem wird sie im Ticket-E-Mail-Editor und Ticket-Kommentar-Editor angezeigt, um den Bearbeitern die Arbeit zu erleichtern.
- **Release**
  Wenn Sie ein Versionierungssystem für die E-Mail-Vorlagen aufsetzen möchten, können Sie das Release, d. h. die Version hier eingeben. Sie können die Zahlen (drei Ziffern) eingeben, oder den Plus-Button zur Hilfe nehmen.
- **Sprache**
  Wählen Sie die Sprache der Vorlage aus. Dies kann wichtig sein, wenn Sie in einem internationalen Team arbeiten. ConSol CM kann in so vielen Sprachen verwendet werden, wie benötigt werden. Dies kann im Admin Tool und im Process Designer konfiguriert werden. Damit die E-Mails in der richtigen Sprache gesendet werden, muss hier die entsprechende Sprachumgebung gesetzt werden.
- **Aktiv**
  Wählen Sie, ob die Vorlage aktiv (= aktiviert) oder inaktiv (deaktiviert) sein soll. Dies kann später geändert werden. Das heißt, Sie können die Vorlage schreiben und bearbeiten und erst dann auf *aktiv* setzen, wenn sie fertig ist.
- **Typ**
  Wählen Sie den Typ (*Brief*, *Include*, *Textblock*, *Skript*) der Vorlage. Eine Erklärung dazu finden Sie im vorherigen Abschnitt.
- **Verfügbar in**
  - **Workflow**
    Wählen Sie diese Option, wenn die Vorlage in Workflows verfügbar sein soll (d. h. nicht im Ticket-E-Mail-Editor oder im Ticket-Kommentar-Editor).
  - **E-Mail**
    Wählen Sie diese Option, wenn die Vorlage in E-Mails verfügbar sein soll.
  - **Ticket erzeugen**
    Wählen Sie diese Option, wenn die Vorlage bei der Ticketerstellung verfügbar sein soll.
  - **Ticket bearbeiten**
    Wählen Sie diese Option, wenn die Vorlage bei der Ticketbearbeitung verfügbar sein soll.
- **Inhalt**
  Hier können Sie den Inhalt der Vorlage / des Briefs definieren. Sie können Freitext mit Komponenten der *Bibliothek der Templatefelder* (unter dem Inhalt, siehe Abschnitt Die Bibliothek

der Templatefelder) kombinieren. Schreiben Sie den Text und wählen Sie das gewünschte Element aus der Bibliothek durch einen Doppelklick aus.
Alle Funktionen des Rich-Text-Editors stehen für das Design des Vorlageninhalts zur Verfügung. Beachten Sie, dass die Verfügbarkeit von zusätzlichen Formatierungsfunktionen vom Seitenanpassungsattribut *cmRichTextEditor* abhängt. Eine detaillierte Beschreibung dieser Funktionen finden Sie im Abschnitt cmRichTextEditor des Kapitels Seitenanpassung.

> Beachten Sie, dass <script>-Tags automatisch vom System in Escape-Zeichen gesetzt werden. Auf diese Weise ist es nicht möglich, ausführbaren JavaScript-Code in Text einzufügen.

- **Attachment hinzufügen**
  Hier können Sie ein oder mehrere Attachments zur Vorlage hinzufügen. Wählen Sie mithilfe des Dateibrowsers eine Datei aus dem Dateisystem ihres Computers aus. Es sind alle gebräuchlichen Dateitypen möglich. Das Systemverhalten im Hinblick auf Attachments hängt von der definierten Verfügbarkeit ab.
  - **Attachment in E-Mail-Vorlagen**
    Ein Attachment, das in einer für E-Mails verfügbaren Vorlage definiert ist, wird als normales Ticket-Attachment an die Tickets angehängt. Zusätzlich ist es als E-Mail-Attachment vorausgewählt, wenn der Bearbeiter mit dem Ticket-E-Mail-Editor eine E-Mail schreibt und die entsprechende E-Mail-Vorlage auswählt.
  - **Attachment in Ticketkommentaren (*Ticket bearbeiten*, *Ticket erzeugen*)**
    Ein Attachment, das in einer Vorlage definiert ist, die bei der Erstellung und Bearbeitung eines Tickets verfügbar ist, wird wie ein normales Attachment an das Ticket angehängt. Danach kann es auch als E-Mail-Attachment verwendet werden, es ist im Ticket-E-Mail-Editor aber nicht vorausgewählt.
- **Zuordnung/Kontext**
  Hier können Sie einen bestimmten Kontext für die Vorlage definieren. Wenn hier keine Einstellungen vorgenommen werden, ist die Vorlage in allen Queues verfügbar und die Verfügbarkeit ist nicht von Parametern abhängig. Wenn einschränkende Queues und/oder Ticketfelder definiert sind, wird die Verfügbarkeit des Vorlage entsprechend beschränkt. Sie können
  - Ticketfelder für Queues
  - Queues

  definieren, in denen die Vorlage verfügbar sein soll, siehe Abschnitt Zuordnung von Vorlagen zu Queues oder bestimmten Parametern.

> Beachten Sie, dass die Verfügbarkeit von Vorlagen im Web Client auch von der Kundengruppe des Tickets abhängt. Wenn Parameter verwendet werden, die Kundenfelder eines bestimmten Kundendatenmodells referenzieren, ist die Vorlage nur verfügbar, wenn das Ticket einem Kunden aus einer Kundengruppe, die dieses Modell verwendet, zugewiesen ist.

Im Web Client, d. h. im Ticket-E-Mail-Editor hätte die Vorlage *Anfrage_erhalten* folgendes Layout:

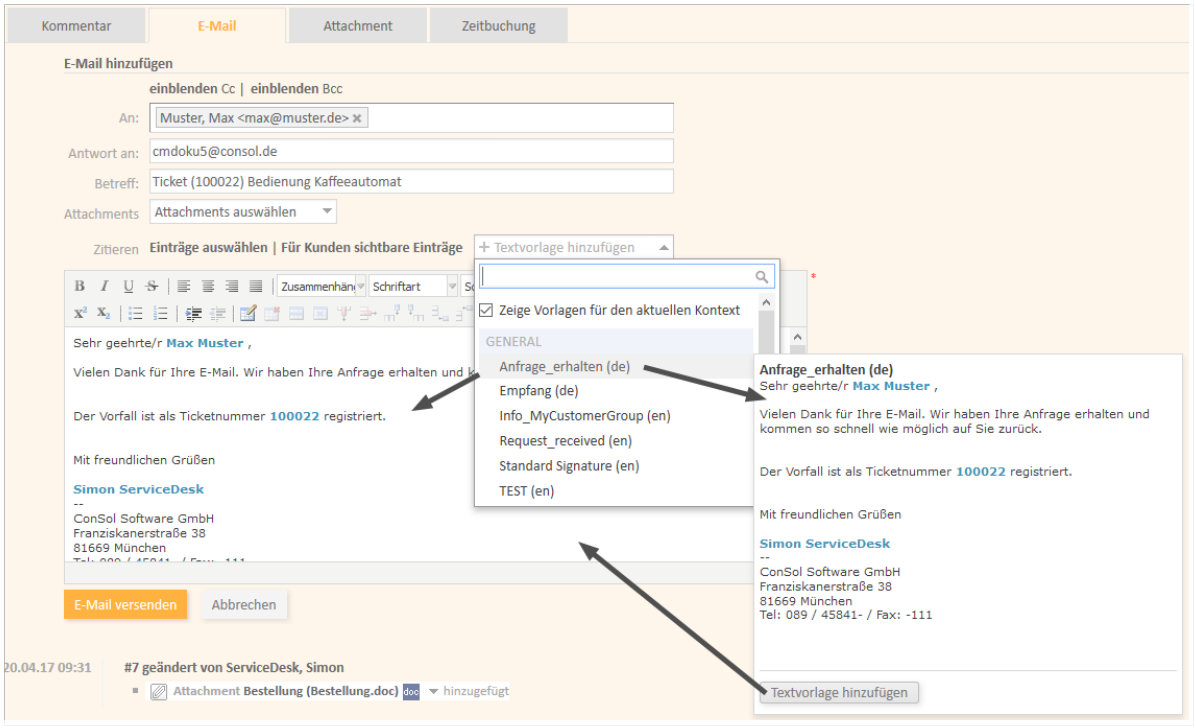


Abbildung 511: *ConSol CM Web Client - E-Mail-Vorlage im Ticket-E-Mail-Editor*

Erstellen einer neuen Vorlage aus einer gesendeten E-Mail

Sie können den Textvorlagen-Manager auch über das Kontextmenü einer gesendeten E-Mail im Protokollbereich eines Tickets öffnen. Der Textvorlagen-Manager wird im Modus *Neue Textvorlage erstellen* geöffnet, wobei die folgenden beiden Einstellungen vorausgewählt sind:

- Typ: *Brief*
- Verfügbar in: *E-Mail*

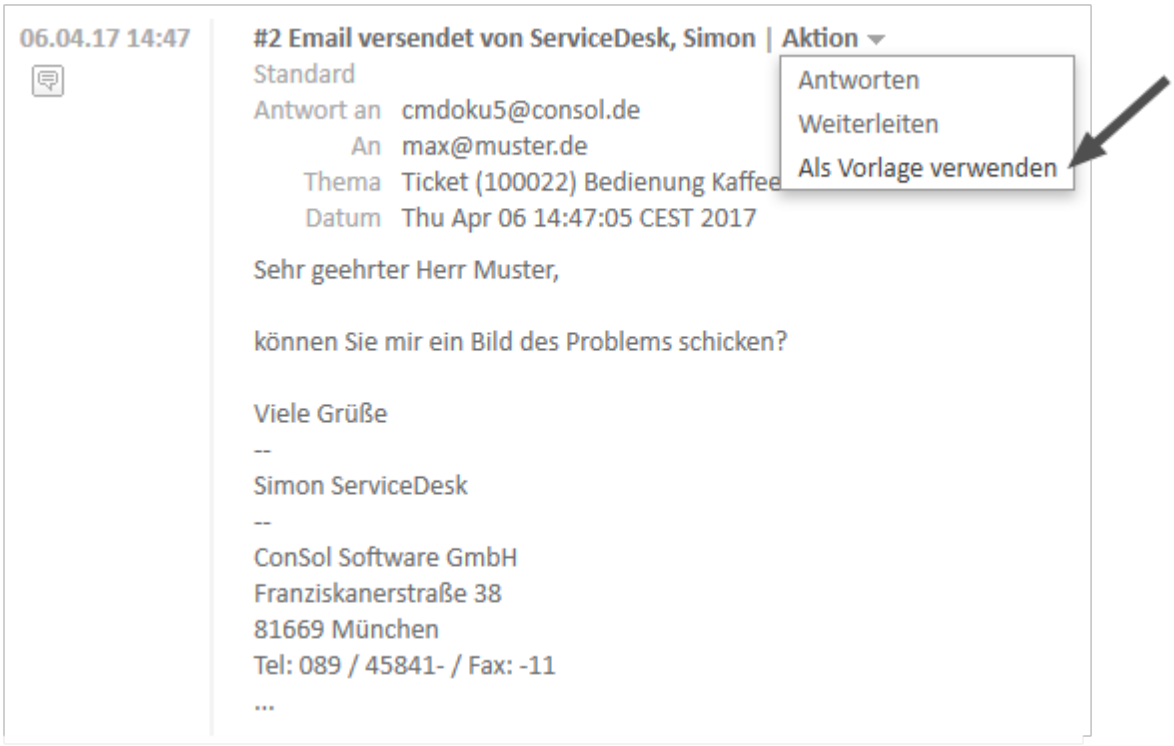


Abbildung 512: *ConSol CM Web Client - Erstellen einer neuen E-Mail-Vorlage aus einer gesendeten E-Mail*

Die Bibliothek der Templatefelder

Die Bibliothek der Templatefelder ist eine Sammlung aller Datenfelder, die im System verfügbar sind. Dies sind:

- **Standardfelder**
  Wie *Queue* oder *Bearbeiter* mit allen zugehörigen Daten wie *Queue-Name* oder *Vorname des Bearbeiters* oder *Nachname des Bearbeiters*.
- **Ticketfelder und/oder Kundenfelder**
  Diese wurden speziell für das System angelegt, z. B. *Kundenservicenummer*. Beachten Sie, dass in der Bibliothek der Templatefelder die lokalisierten Werte der Ticketfelder und Kundenfelder angezeigt werden.
- **Komponenten des Textvorlagen-Managers**
  Diese werden in anderen Komponenten verwendet, z. B. *Includes* oder *Workflow Includes*.
- **Skripte**
  Diese werden von einem Administrator definiert und helfen Ihnen dabei, Inhalt dynamisch bereitzustellen.

Die folgende Tabelle enthält Beispiele für Felder, die es in einem CM-System geben könnte. Die in der Bibliothek der Templatefelder angezeigten Namen sind die lokalisierten Namen der Ticketfelder bzw. Kundenfelder. Wenn keine Lokalisierung verfügbar ist, wird der (technische) Feldname angezeigt.

Wenn Sie Informationen über Ticketfelder lesen möchten, finden Sie diese in Kapitel Verwaltung von Ticketfeldern (Einrichten des Ticketdatenmodells). Für Kundenfelder (d. h. Kundendaten), siehe Abschnitt Einrichten des Kundendatenmodells.

Feldgruppe oder Hauptkomponente	Ticketfeld bzw. Kundenfeld (Beispiel)	Erklärung
Kundendatenmodelle		<Einstiegspunkt für alle kundenspezifischen Felder>
Kundendatenmodelle	Kundengruppen	<Einstiegspunkt für alle kundenspezifischen Felder der ausgewählten Kundengruppe>
<Kontakt oder Firma>	Anrede	
	Akademischer Titel	
	Vorname	
	Nachname	
	Telefonnummer	
	E-Mail	
	<weitere Felder abhängig von der FlexCDM-Definition>	
Kundengruppe	Name der Kundengruppe	
Queue	Name	Der Name der Queue, in der das Ticket momentan verarbeitet wird
Ticketfelder der Queue	Alle Ticketfelder der Ticketfeldgruppen, die der Queue zugewiesen sind	
Ticket	ID	Die interne Ticket-ID, nicht auf der GUI des Web Clients angezeigt
	Name	Der Ticketname, die *ID* im Web Client
	Thema	

Feldgruppe oder Hauptkomponente	Ticketfeld bzw. Kundenfeld (Beispiel)	Erklärung
	Bearbeiter	Der aktuelle Bearbeiter, dem das Ticket zugewiesen ist. Kann „NULL “(leer) sein, wenn kein Bearbeiter gesetzt ist.
	Eröffnungsdatum	Das Eröffnungsdatum des Tickets
Bearbeiter	Login	Das Login des Bearbeiters, der aktuell im System angemeldet ist
	Vorname	Vorname, Nachname, E-Mail-Adresse des Bearbeiters. Stellen Sie sicher, dass das entsprechende Feld in den Bearbeiterdaten ausgefüllt ist. Details zur Bearbeiterverwaltung finden Sie im Kapitel Bearbeiterverwaltung.
	Nachname	
	E-Mail	
Includes	<alle verfügbaren Includes>	
Textblöcke	<alle verfügbaren Textblöcke>	
Workflow Includes	<alle verfügbaren Workflow Includes>	
Skripte	<alle verfügbaren Skripte>	

⚠ Wenn kundenspezifische Felder in einer Vorlage verwendet werden, wird diese Vorlage nur angeboten, wenn das Ticket einem Hauptkunden der entsprechenden Kundengruppe zugewiesen ist!

Da die Kundenfelder in den einzelnen Kundendatenmodellen unterschiedlich sind, ist es möglicherweise erforderlich, für mehrere Kundengruppen ähnliche Vorlagen zu definieren.

Durch Klicken auf *Parameter hinzufügen* oder *Enum-Parameter* hinzufügen können Sie ein Feld definieren, das der Bearbeiter zur Laufzeit mit Daten füllen muss. Im folgenden Beispiel wird der zuständige Consultant als String-Parameter hinzugefügt und die Ticketpriorität wird als Enum-Parameter hinzugefügt. Eine Sortierte Liste (ENUM, siehe Abschnitt Verwaltung von Sortierten Listen) wird nur dann als Enum-Parameter angeboten, wenn die Option *Template aktiviert* für die Sortierte Liste gesetzt wurde.

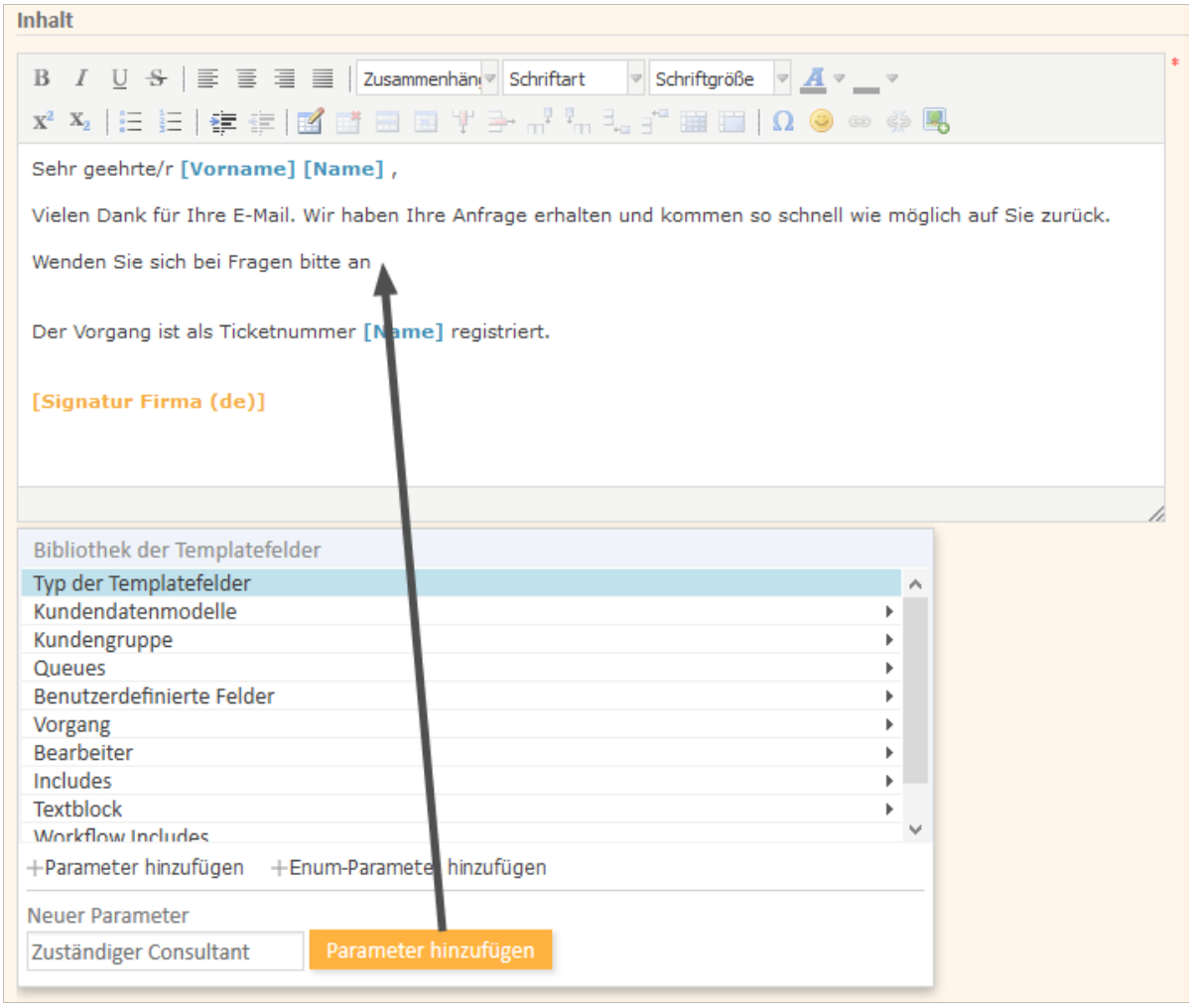


Abbildung 513: *ConSol CM Web Client - Textvorlagen-Manager: String-Parameter in Bibliothek der Templatefelder hinzufügen*

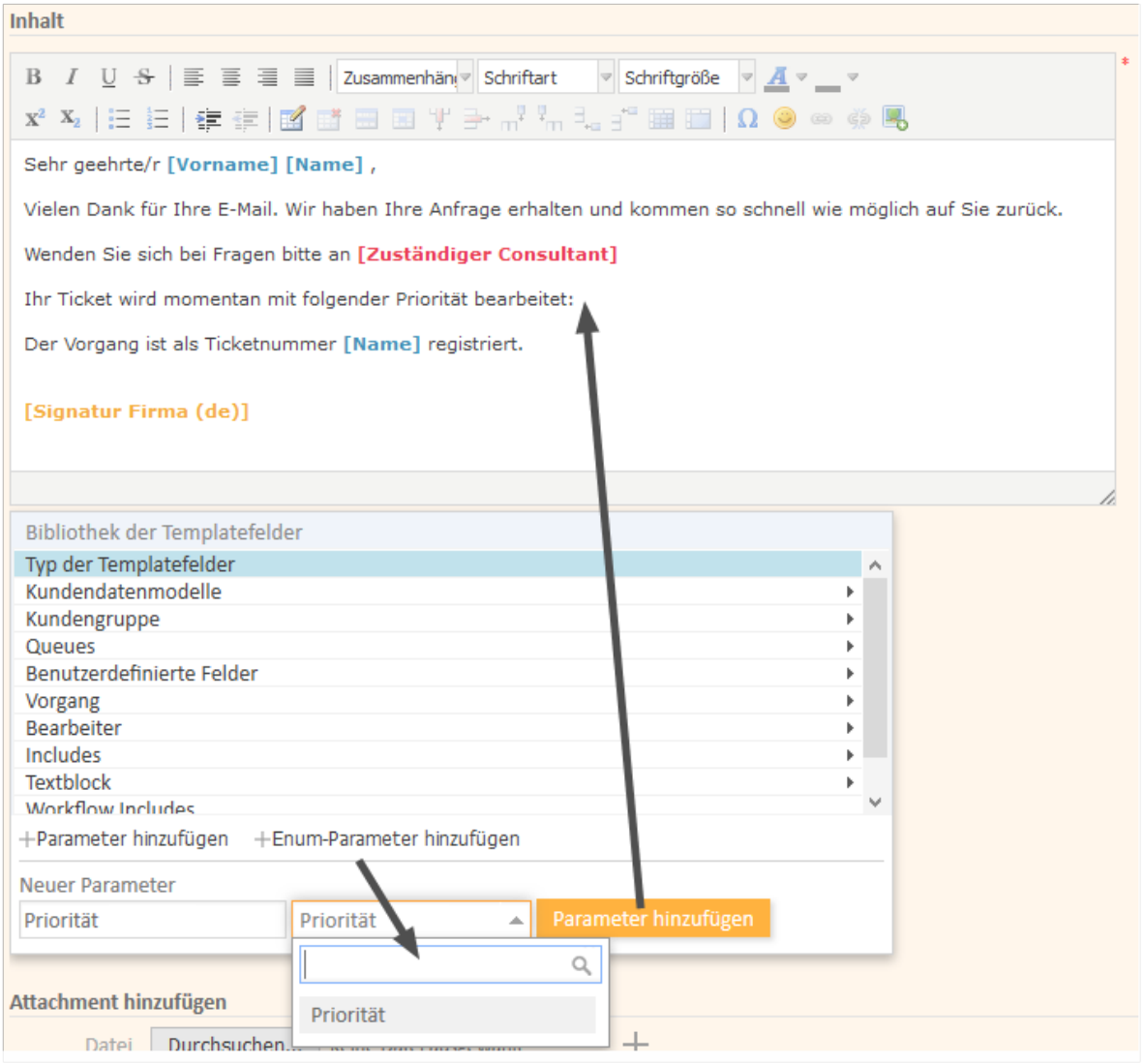


Abbildung 514: *ConSol CM Web Client - Textvorlagen-Manager: Enum-Parameter in Bibliothek der Templatefelder hinzufügen*

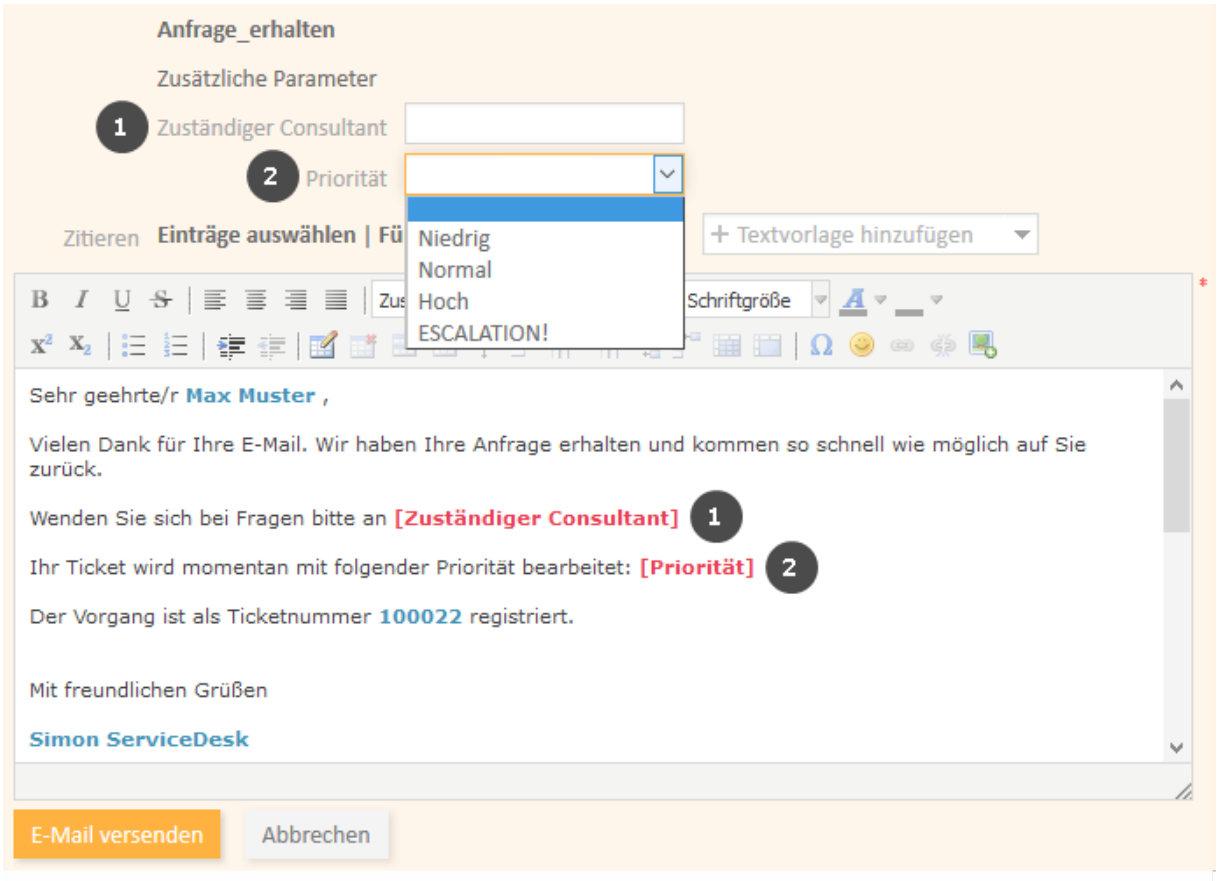


Abbildung 515: *ConSol CM Web Client - Ticket-E-Mail-Editor: Ausfüllen von Parametern, 1*

Im obigen Beispiel ist *Zuständiger Consultant* (1) ein String-Parameter, bei dem der Bearbeiter den gewünschten Wert in ein Textfeld eingeben kann. *Priorität* (2) ist ein Enum-Parameter, bei dem der Bearbeiter den gewünschten Wert aus einer Drop-down-Liste auswählen kann.

Wenn der Bearbeiter im Ticket den Ticket-E-Mail-Editor öffnet und Daten in die Felder eingibt (hier *Herrn Schmidt* als *Zuständiger Consultant* und *Niedrig* als *Priorität*), werden die (neuen) Daten automatisch in die entsprechenden Felder der Vorlage geschrieben.

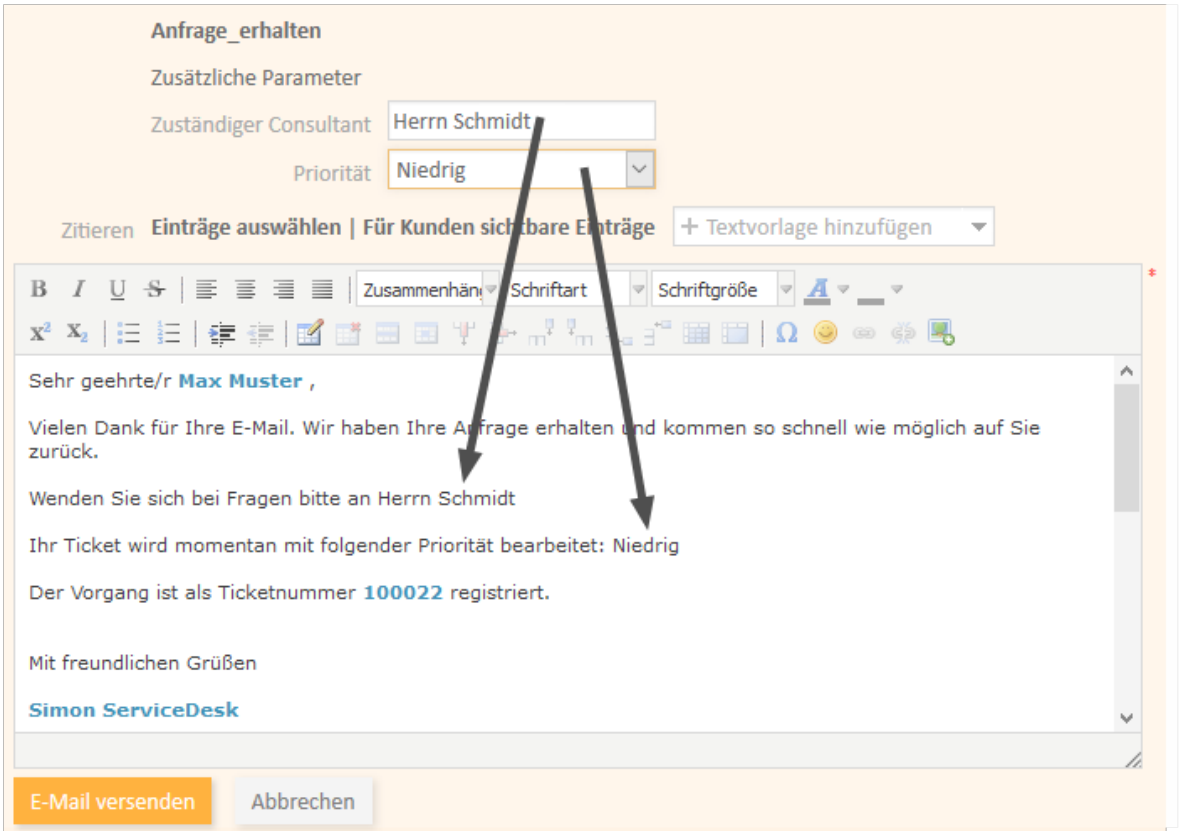


Abbildung 516: *ConSol CM Web Client - Ticket-E-Mail-Editor: Ausfüllen von Parametern, 2*

Erstellen eines neuen Include oder Workflow Include

Ein *Include* ist eine Vorlage, die vom Bearbeiter nicht direkt (im Ticket-E-Mail-Editor oder Ticket-Kommentar-Editor) ausgewählt werden kann, sondern als Komponente in andere E-Mail- oder Tickettext-Vorlagen, nämlich in *Briefe* integriert wird.

Ein Standardanwendungsfall für ein *Include* ist die Signatur. Deshalb zeigen wir Ihnen dieses Beispiel. Um eine Standardsignatur für eine Firma zu definieren, definieren Sie die Signatur als *Include* und integrieren Sie sie in den *Brief*, der die Standardsignatur der Firma darstellt.

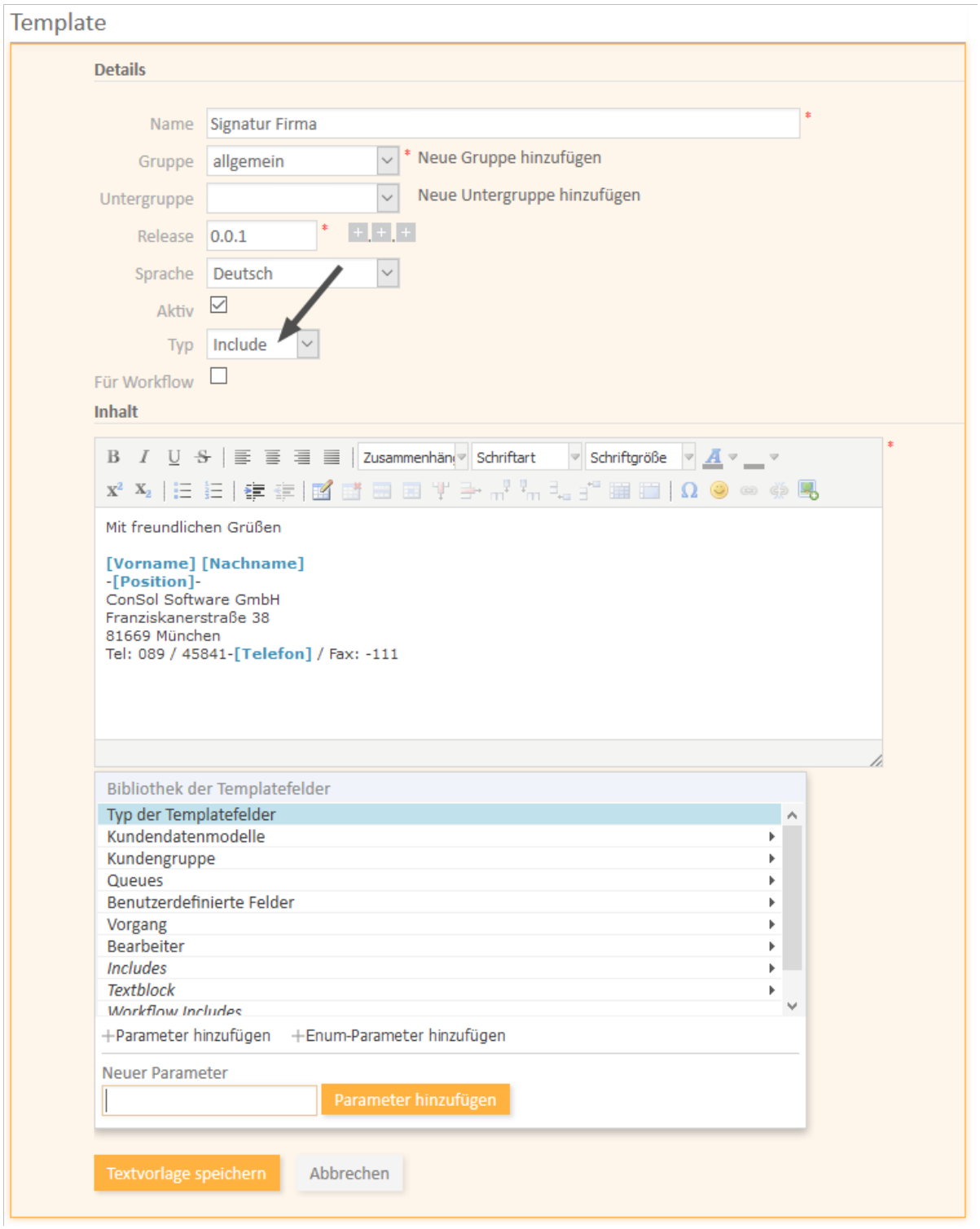


Abbildung 517: *ConSol CM Web Client - Textvorlagen-Manager: Include für Signatur*

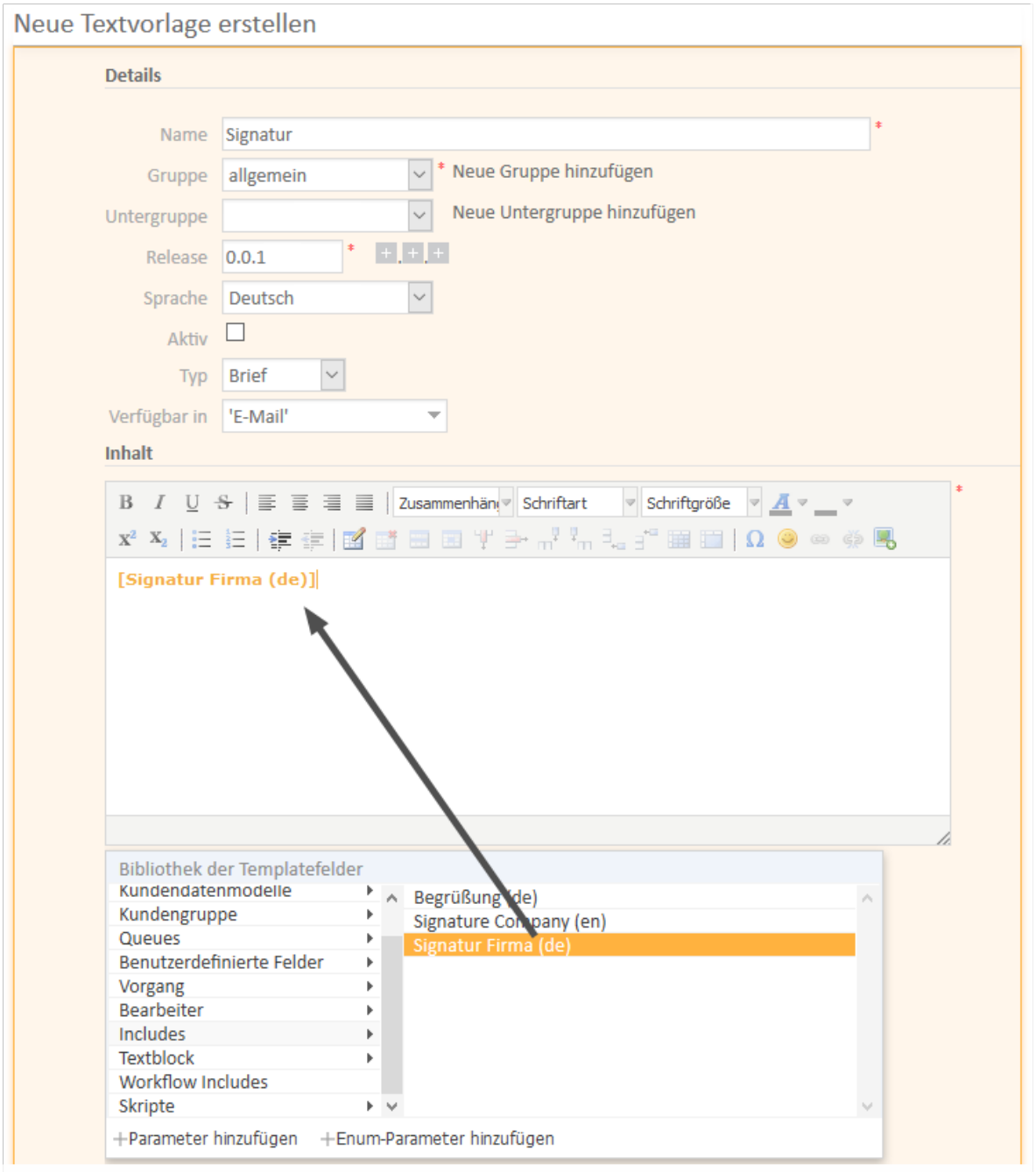


Abbildung 518: *ConSol CM Web Client - Textvorlagen-Manager: Brief für Standardsignatur*

Wenn Sie dem Beispiel folgen, können zwei Zwecke erfüllt werden:

- Die *Standard_Signatur* kann als E-Mail-Standard definiert werden. Auf diese Weise wird sie automatisch für jede neue E-Mail angezeigt. Der Bearbeiter kann die Vorlage natürlich ändern.
- Die *Signatur_Firma* kann in einer beliebigen anderen Vorlage verwendet werden, falls erforderlich (vergleichen Sie die Abbildung der neuen Vorlage).

Erstellen neuer Textblöcke

Ein *Textblock* ist eine Vorlage, die vom Bearbeiter nicht direkt (im Ticket-E-Mail-Editor oder Ticket-Kommentar-Editor) ausgewählt werden kann, sondern als Komponente in andere E-Mail- oder Tickettext-Vorlagen, hauptsächlich in *Briefe* integriert wird. Normalerweise werden in einem *Brief* mehrere *Textblöcke* angeboten, sodass der Bearbeiter auswählen kann, welche er verwenden möchte.

Das folgende Beispiel zeigt, wie Sie dem Kunden mithilfe von drei *Textblöcken* einige Fragen zur Erstanalyse stellen können.

Zuerst werden die *Textblöcke* erstellt:

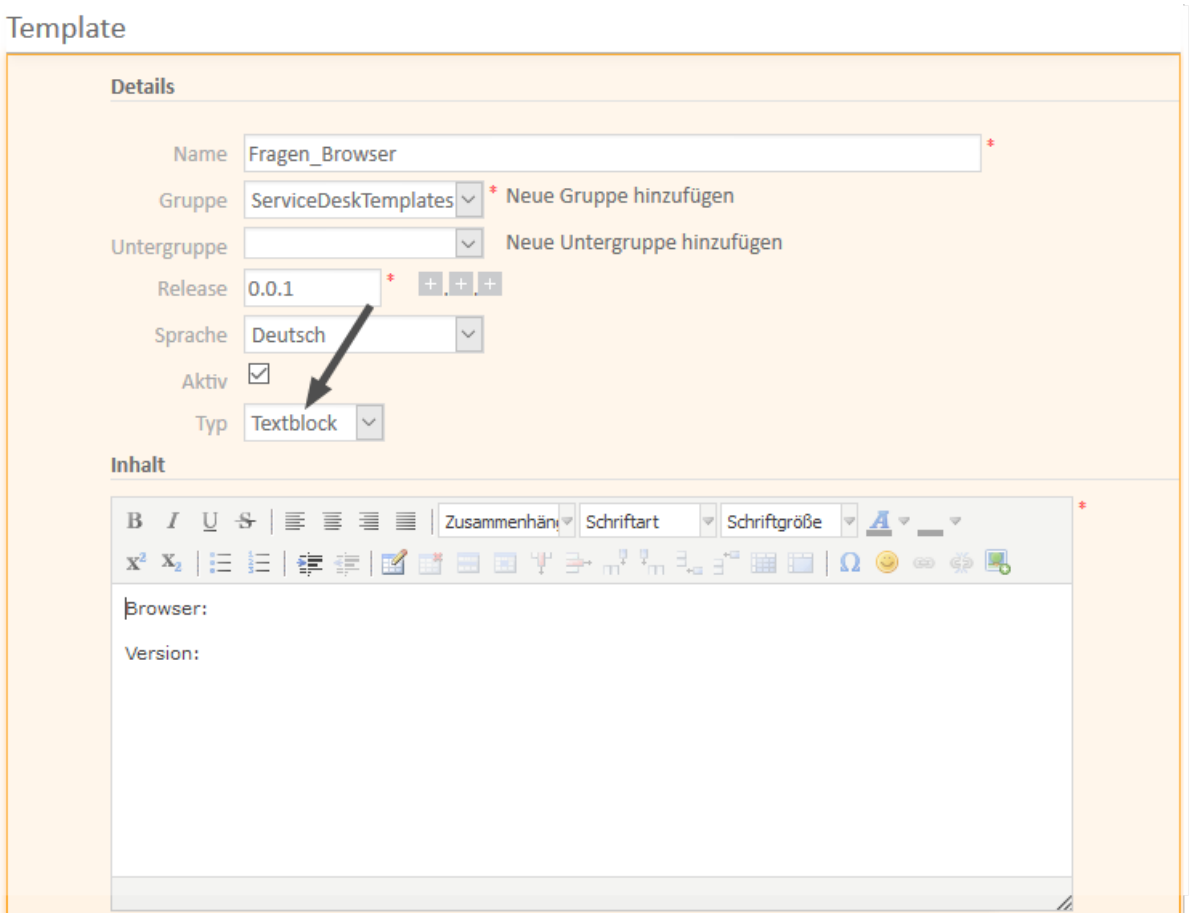


Abbildung 519: *ConSol CM Web Client - Textvorlagen-Manager: Erstellen des ersten Textblocks*

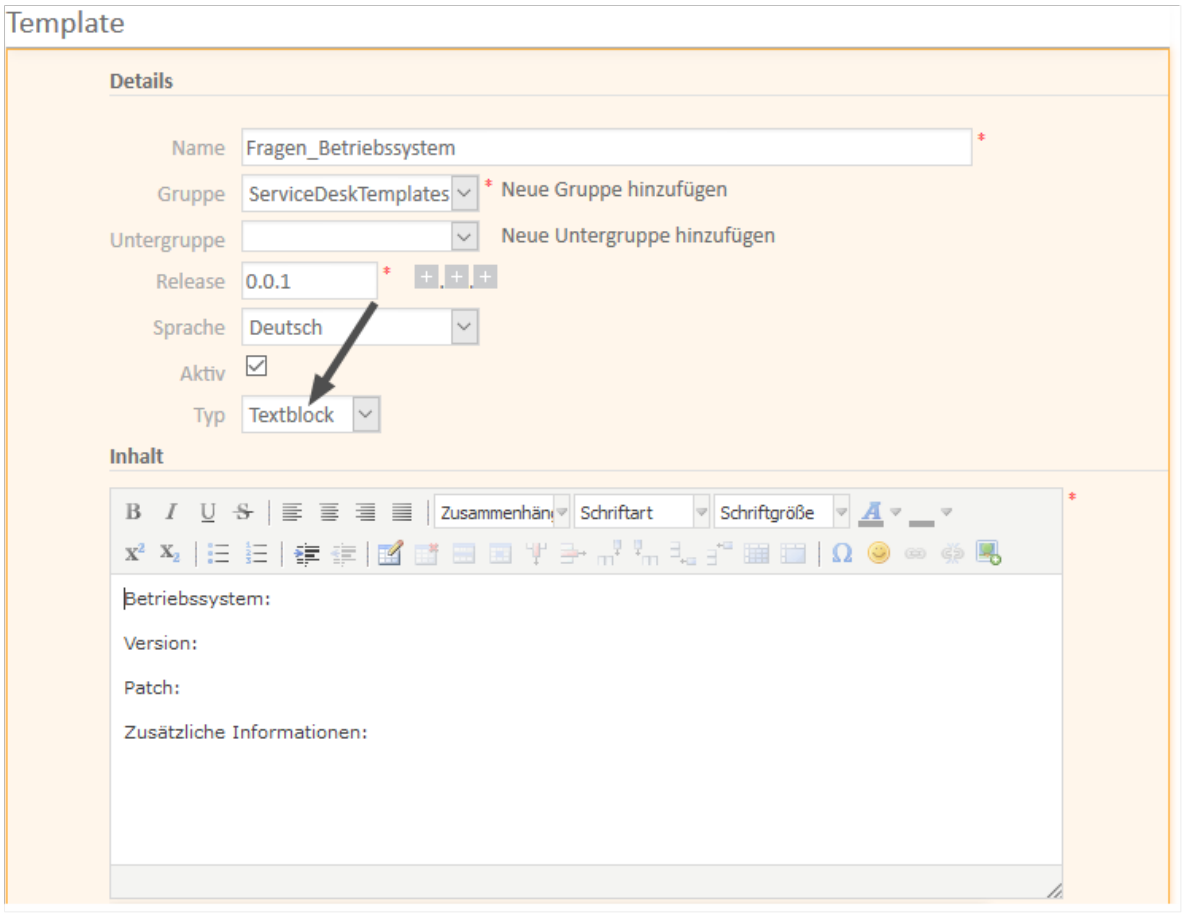


Abbildung 520: *ConSol CM Web Client - Textvorlagen-Manager: Erstellen des zweiten Textblocks*

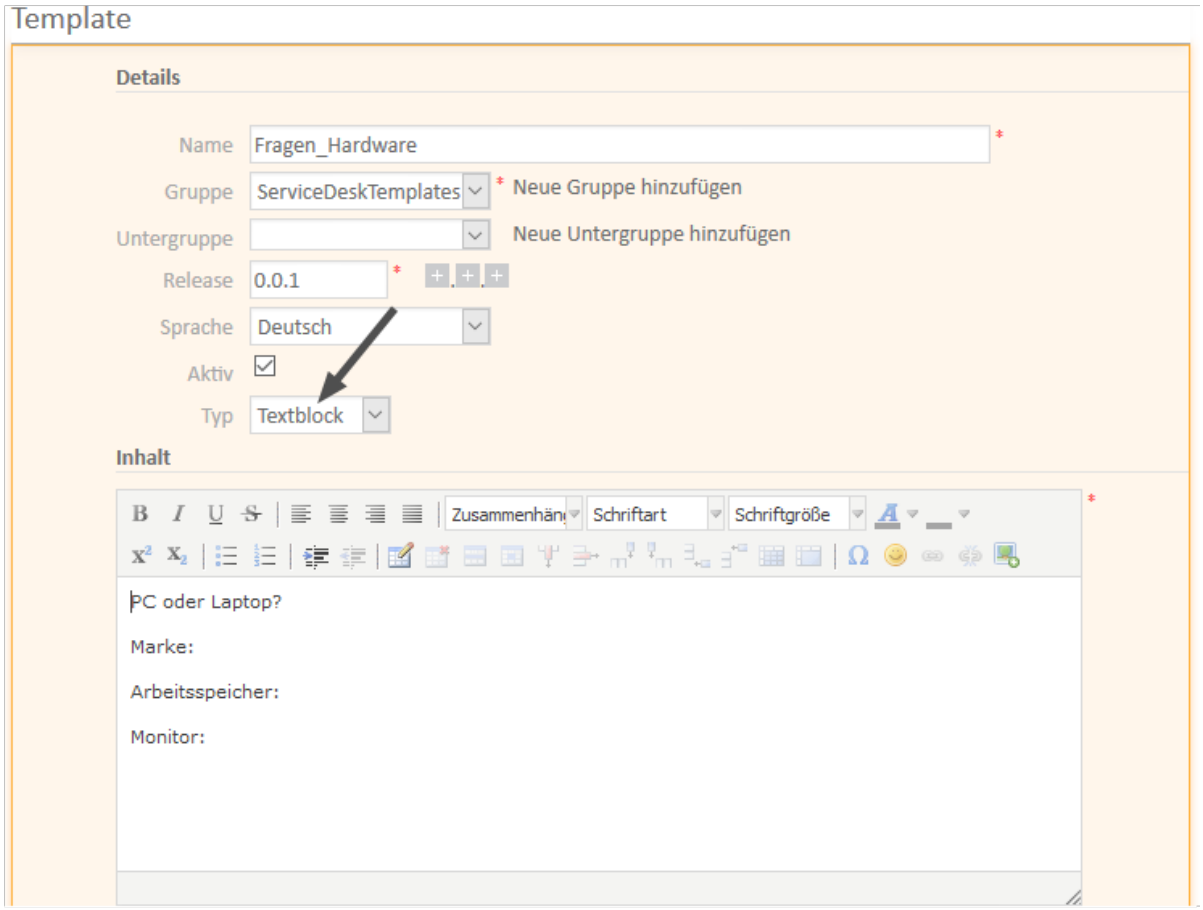


Abbildung 521: *ConSol CM Web Client - Textvorlagen-Manager: Erstellen des dritten Textblocks*

Danach wird der *Brief* erstellt, in dem die *Textblöcke* verwendet werden sollen:

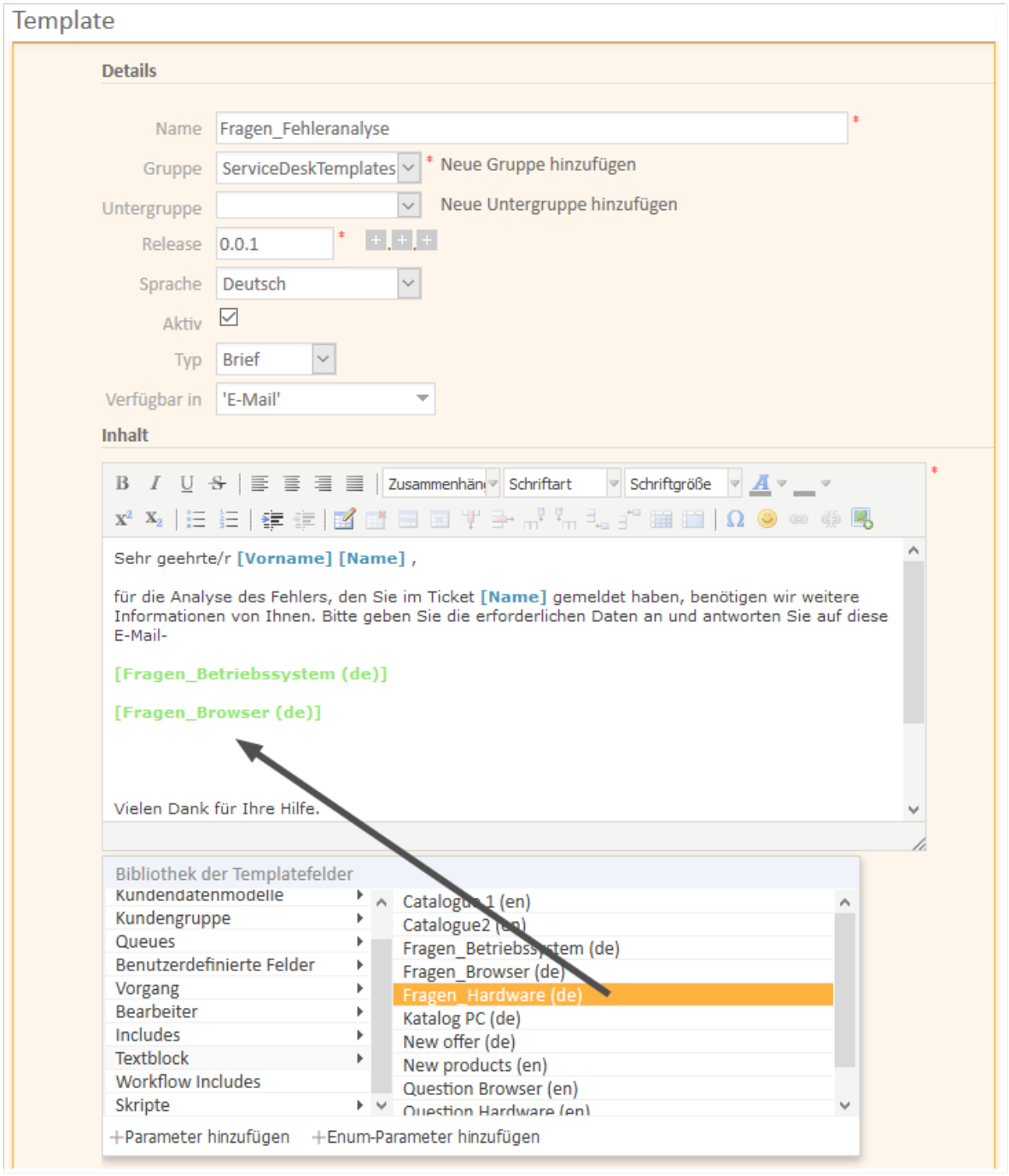


Abbildung 522: *ConSol CM Web Client - Textvorlagen-Manager: Erstellen des Briefs mit den Textblöcken*

Im Web Client kann der Bearbeiter entscheiden, welche *Textblöcke* verwendet werden sollen und welche deaktiviert werden sollen:

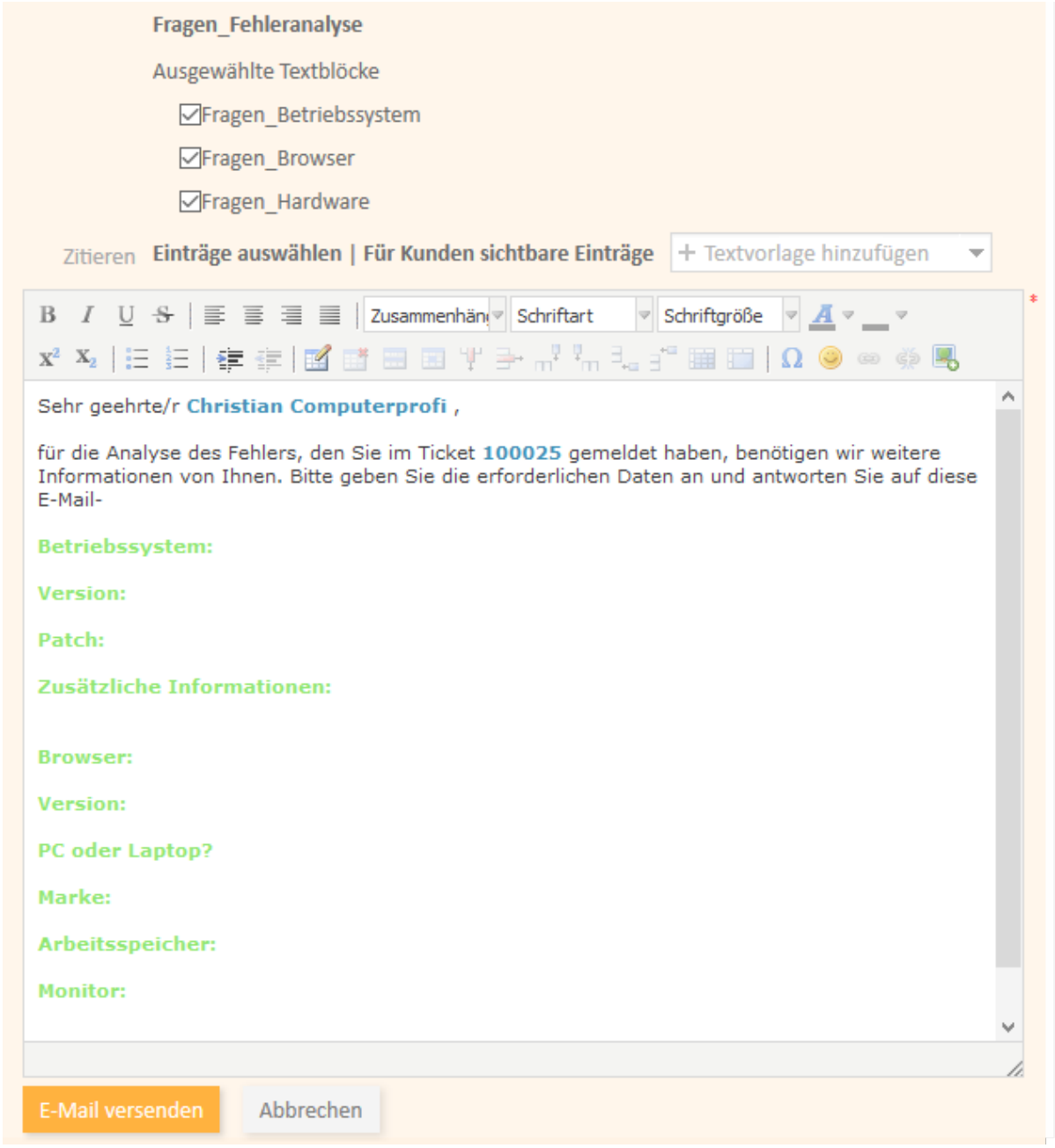


Abbildung 523: *ConSol CM Web Client - Ticket-E-Mail-Editor: Alle Textblöcke ausgewählt*

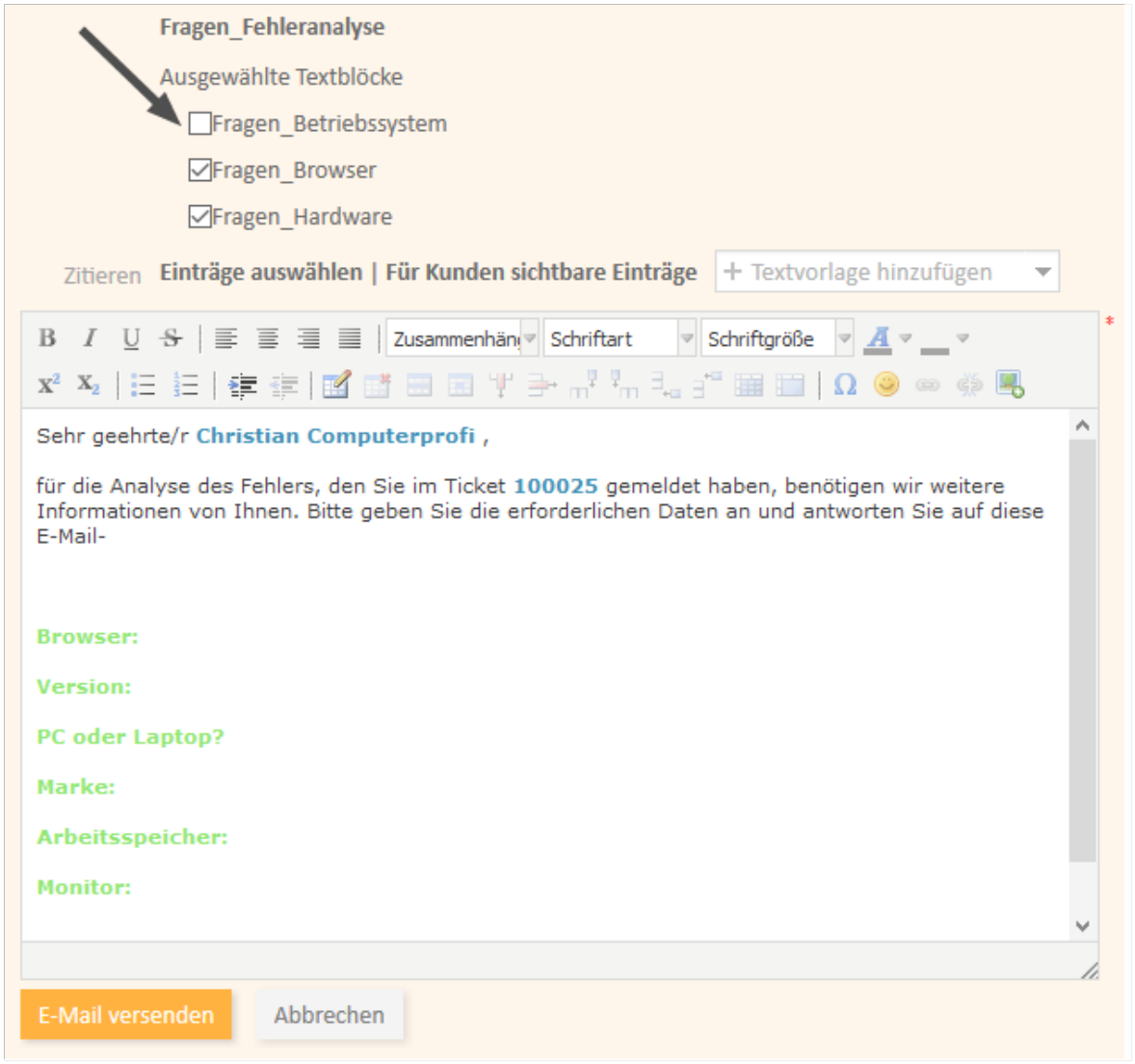


Abbildung 524: *ConSol CM Web Client - Ticket-E-Mail-Editor: Ein Textblock deaktiviert*

Erstellen und Verwenden eines Skripts

Manchmal muss ein System eine gewisse „Intelligenz“ bezüglich der in E-Mail-Vorlagen verwendeten Wörter und Ausdrücke haben, da diese nicht statisch sind, sondern dynamisch angepasst werden müssen. Ein Standardbeispiel ist die Verwendung von *Sehr geehrter Herr* für männliche Kunden (Anrede = „Herr“) und *Sehr geehrte Frau* für weibliche Kunden (Anrede = „Frau“).

Diese Anwendungsfälle können normalerweise durch die Verwendung von **Skripten** im Textvorlagen-Manager abgedeckt werden.

Diese Aufgabe kann nur von einem ConSol CM-Administrator durchgeführt werden. Wenn Sie sich als normaler Benutzer mit Berechtigungen zur Vorlagenverwaltung im Web Client anmelden, können Sie alle Vorlagentypen außer Skripten definieren. Um *Skript* als Vorlagentyp auswählen zu können, müssen Sie sich mit einem Administratorkonto anmelden.

Informationen über die zu verwendende Syntax finden Sie auf folgenden Websites:

- FreeMarker
- FreeMarker-Richtlinien

> Beachten Sie, dass in der Freemarker-Standardnotation spitze Klammern (<>) für Statements verwendet werden. Da diese in einer HTML-Umgebung nicht funktionieren, müssen Sie stattdessen eckige Klammern ([ ]) verwenden.

Die folgende Abbildung zeigt ein Beispiel für ein *Skript*:

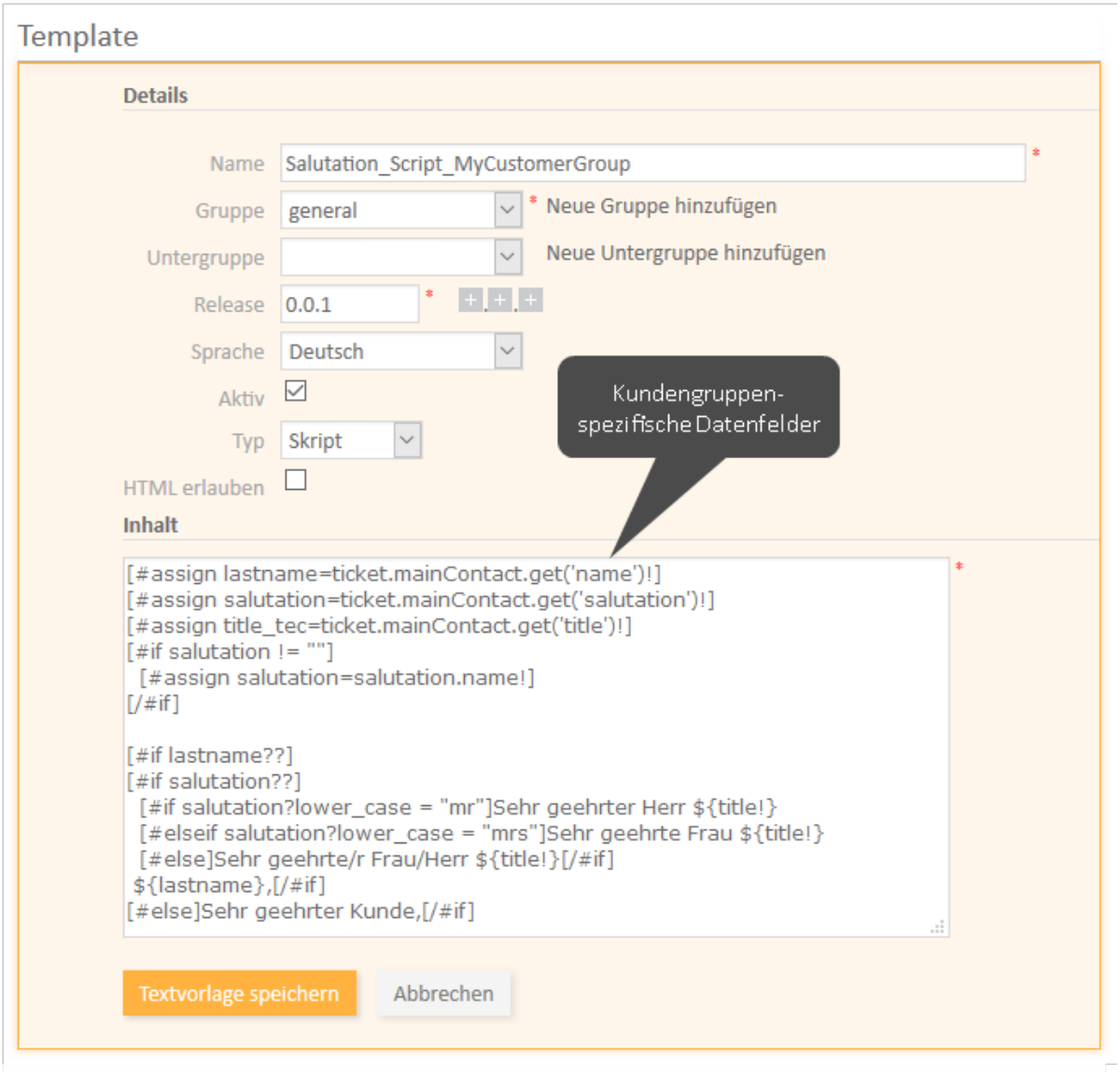


Abbildung 525: *ConSol CM Web Client - Textvorlagen-Manager: Beispielskript für Anrede*

Sie können im Skript auch HTML-Syntax verwenden, wie im folgenden Beispiel gezeigt.

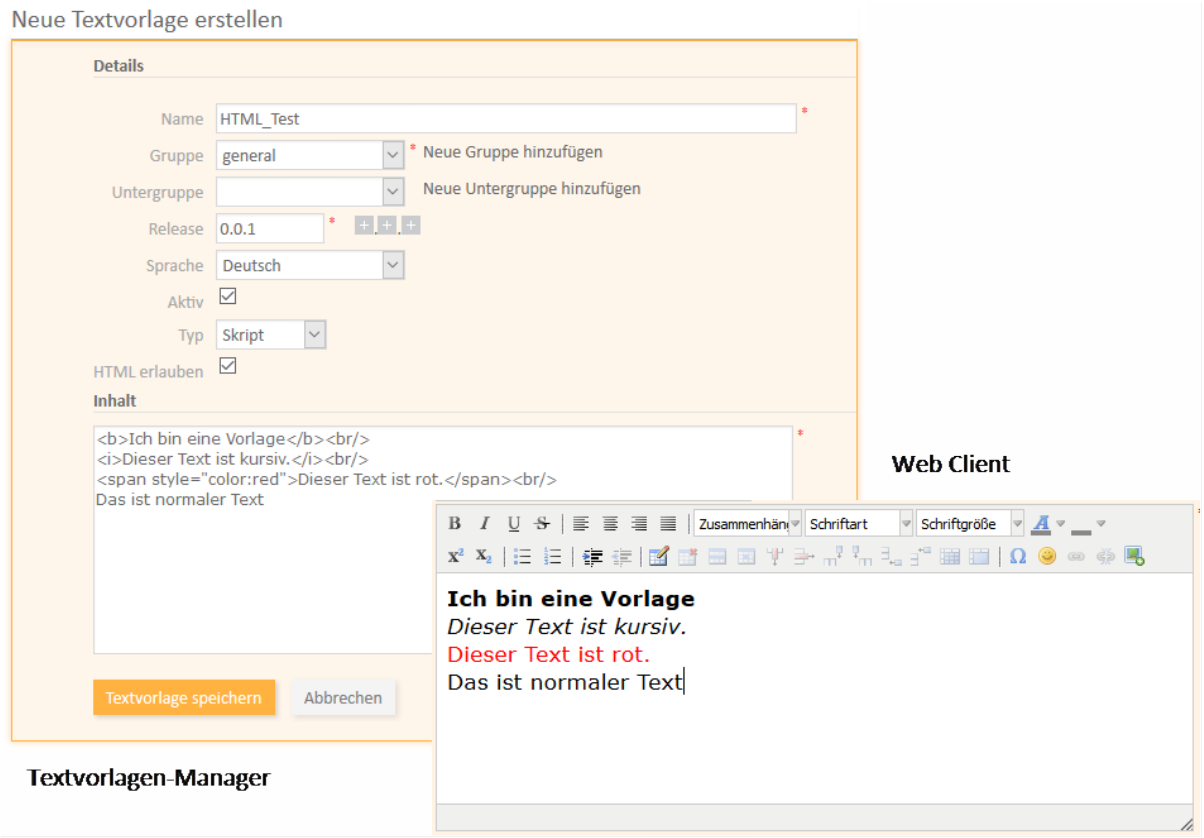


Abbildung 526: *ConSol CM Web Client: Editieren eines Skripts mit HTML-Syntax im Textvorlagen-Manager und Verwenden der Vorlage in einem Ticket*

Beachten Sie, dass ...

- es sich bei den Werten der Felder um die technischen Werte handelt (in diesem Beispiel ist der technische Wert des Felds *salutation mr / mrs*, der lokalisierte Wert für DE wäre *Herr / Frau*). Benutzen Sie immer die technischen Werte!
- es sich bei den Feldern um Ticketfelder und Kundenfelder handelt, die im Admin Tool verwaltet werden. Eine detaillierte Erklärung finden Sie in den Abschnitten Verwaltung von Ticketfeldern (Einrichten des Ticketdatenmodells) und Einrichten des Kundendatenmodells.

Das *Skript* wird dann in eine Vorlage des Typs *Brief* integriert:

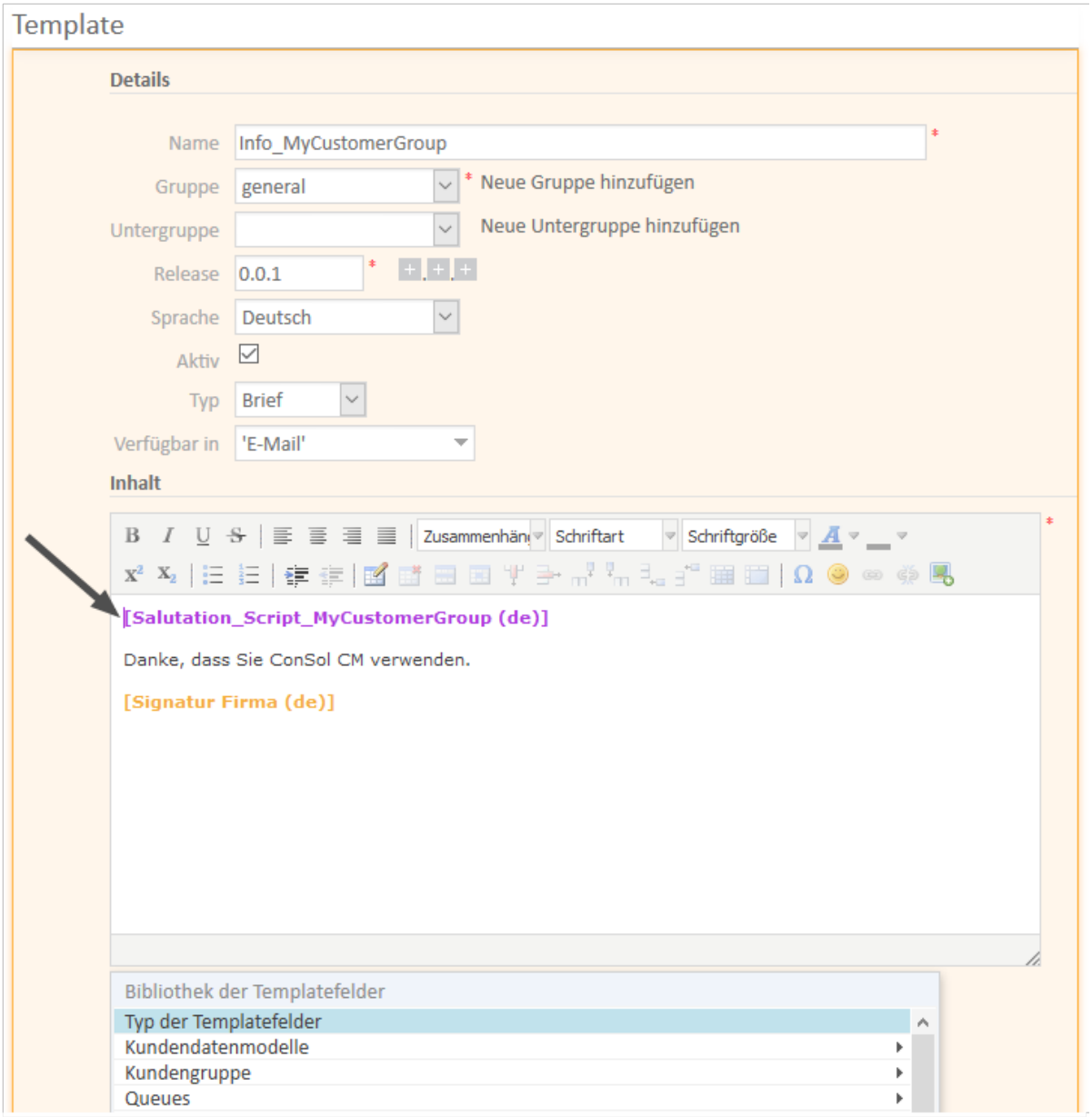


Abbildung 527: *ConSol CM Web Client - Textvorlagen-Manager: Skript in Brief einfügen*

Im Web Client sind die E-Mails dann entsprechend formatiert.

Beispiel 1 (für **Frau**):



Abbildung 528: *ConSol CM Web Client - Mit Skript formatierte E-Mail (Kundendaten, Beispiel 1)*

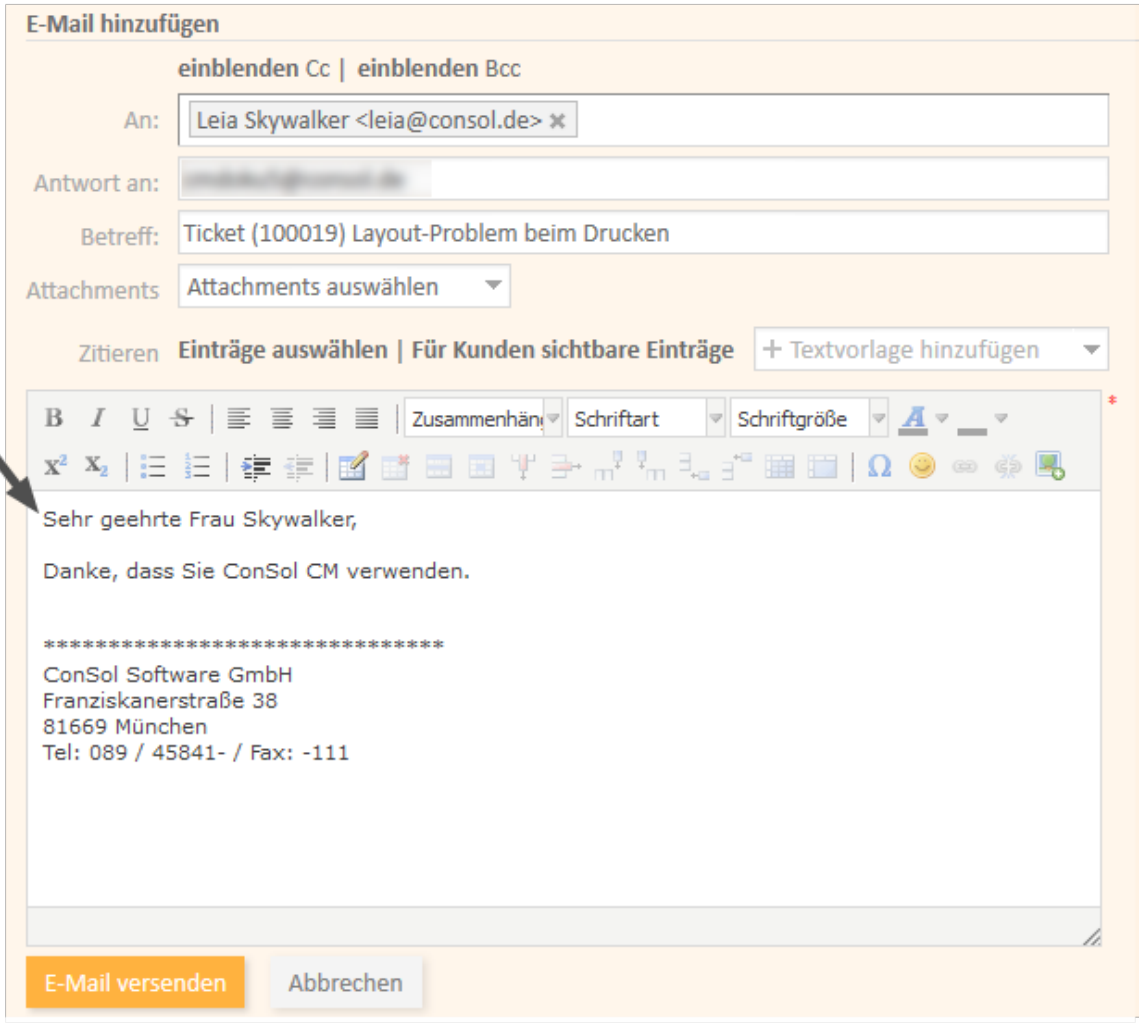


Abbildung 529: *ConSol CM Web Client - Mit Skript formatierte E-Mail (E-Mail, Beispiel 2)*

Beispiel 2 (für **Herr**):

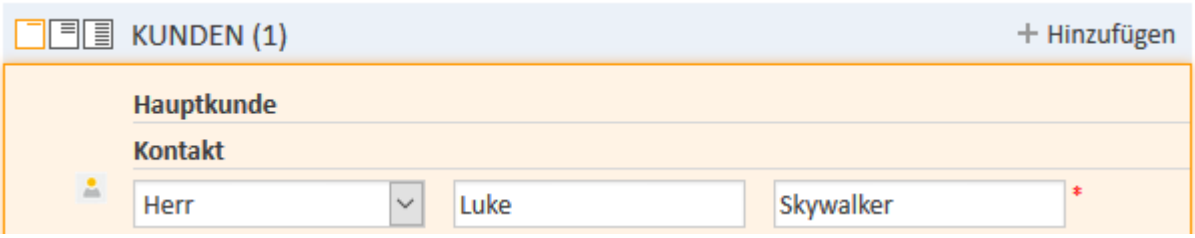


Abbildung 530: *ConSol CM Web Client - Mit Skript formatierte E-Mail (Kundendaten, Beispiel 1)*

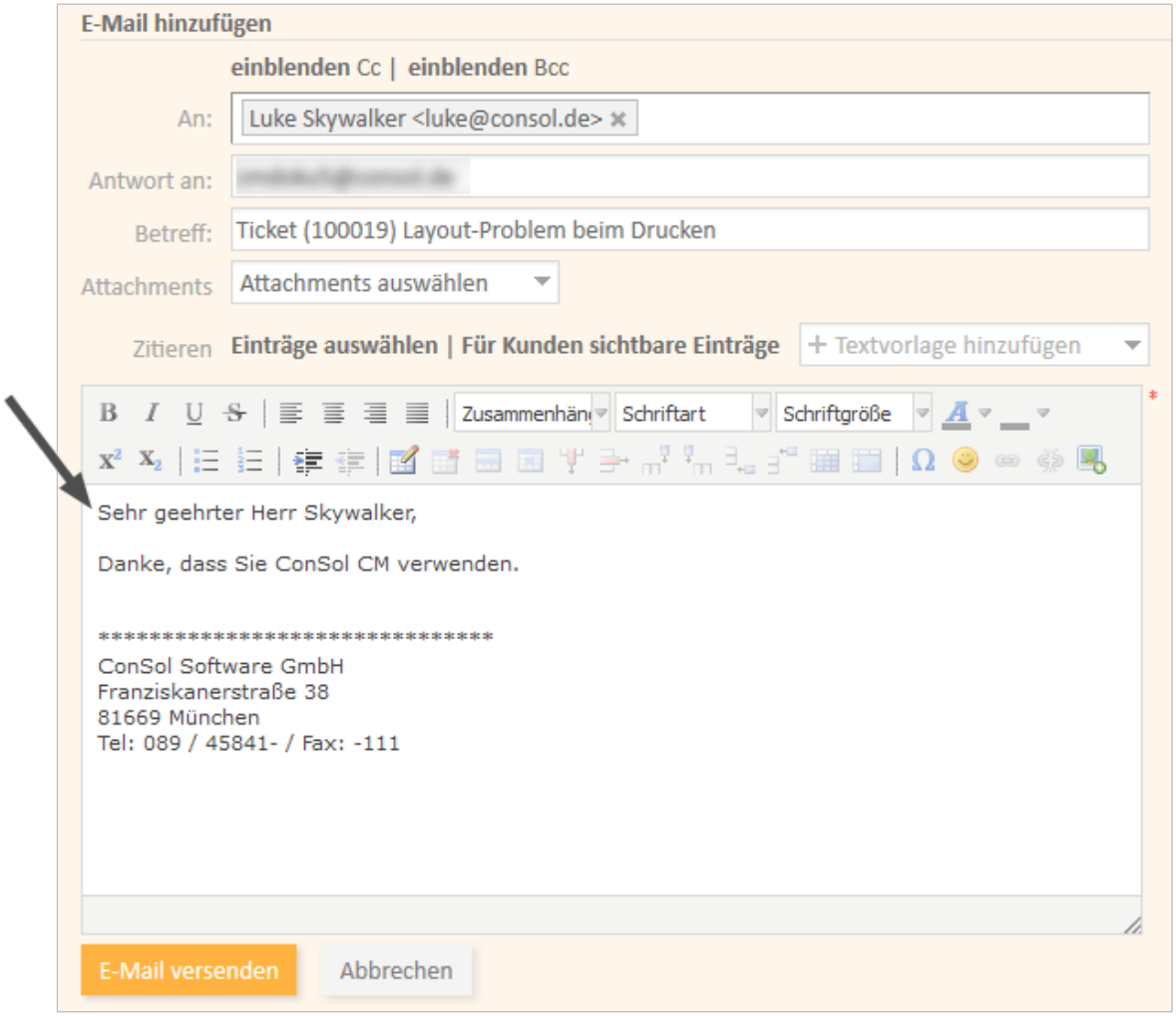


Abbildung 531: *ConSol CM Web Client - Mit Skript formatierte E-Mail (E-Mail, Beispiel 2)*

Verwenden eines Skripts zum Setzen von queue-spezifischen Signaturen

Wenn Ihr Unternehmen mehr als eine Standardsignatur verwendet, z. B. in E-Mails, können Sie dies mit einem Vorlagenskript umsetzen. Das folgende Beispiel zeigt, wie Sie für jede Queue eine spezielle Signatur verwenden können. Dies kann praktisch sein, wenn jede Abteilung mit einer eigenen Queue arbeitet. Durch die Definition von queue-spezifischen Signaturen in E-Mails können Sie also eine abteilungsspezifische Signatur bereitstellen.

Schritt 1:
Erstellen Sie eine Signatur für jede Abteilung. Dies sind reguläre Textvorlagen des Typs *Brief*. Zum Beispiel:

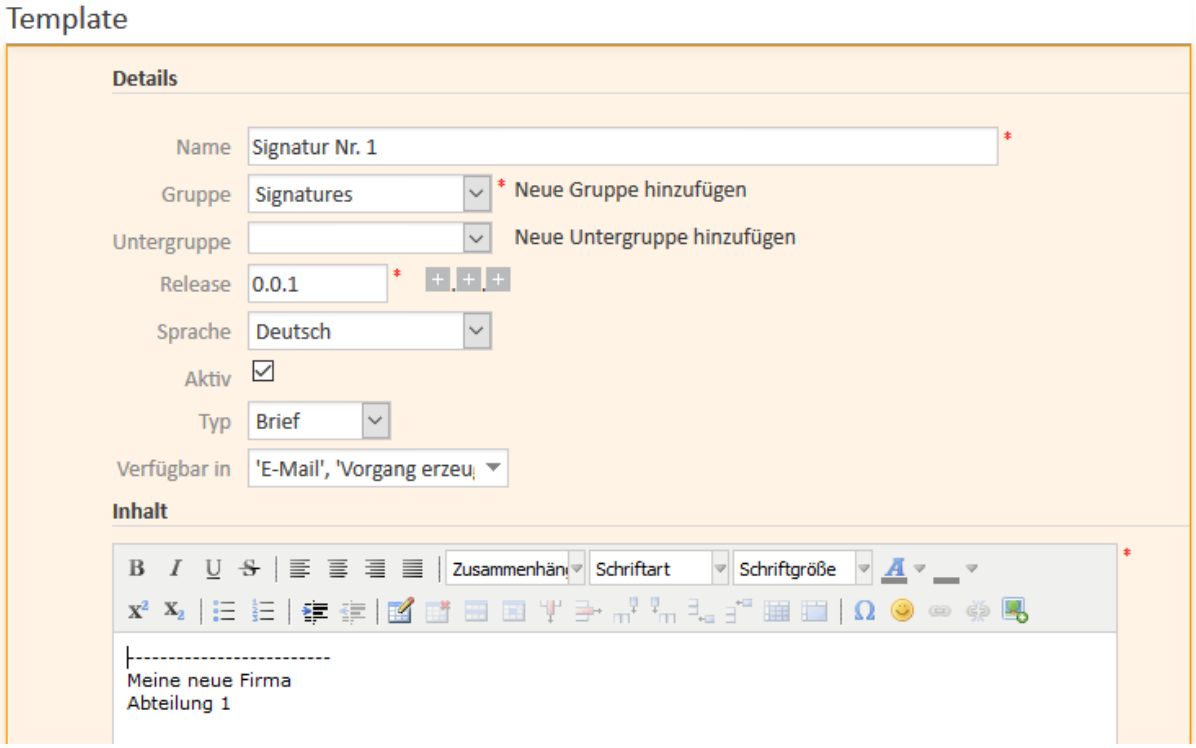


Abbildung 532: *ConSol CM Web Client - Einfache Signatur*

Schritt 2:
Erstellen Sie eine Vorlage des Typs *Skript*, mit der die Auswahl der queue-spezifischen Vorlage verwaltet wird, zum Beispiel:

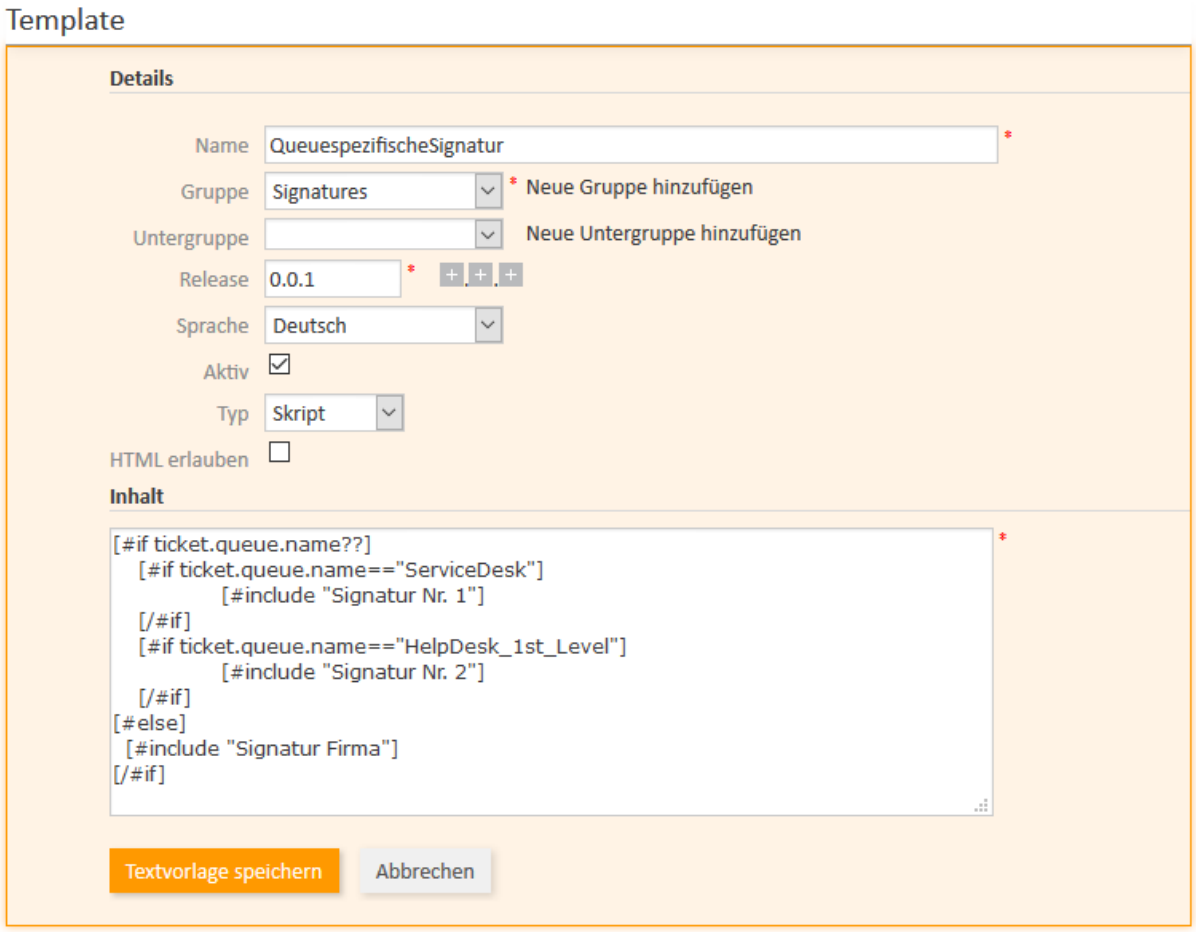


Abbildung 533: *ConSol CM Web Client - Vorlage des Typs Skript zur Verwaltung der Auswahl der queue-spezifischen Signatur*

Schritt 3:
Erstellen Sie eine einfache Vorlage des Typs *Brief*, die als Standard-E-Mail-Vorlage verwendet wird, und fügen Sie das Skript dort ein.

Das ist alles!

Der Bearbeiter sieht jetzt im Ticket-E-Mail-Editor eine unterschiedliche Signatur je nach Queue des Tickets. Siehe die folgenden beiden Abbildungen als Beispiel.

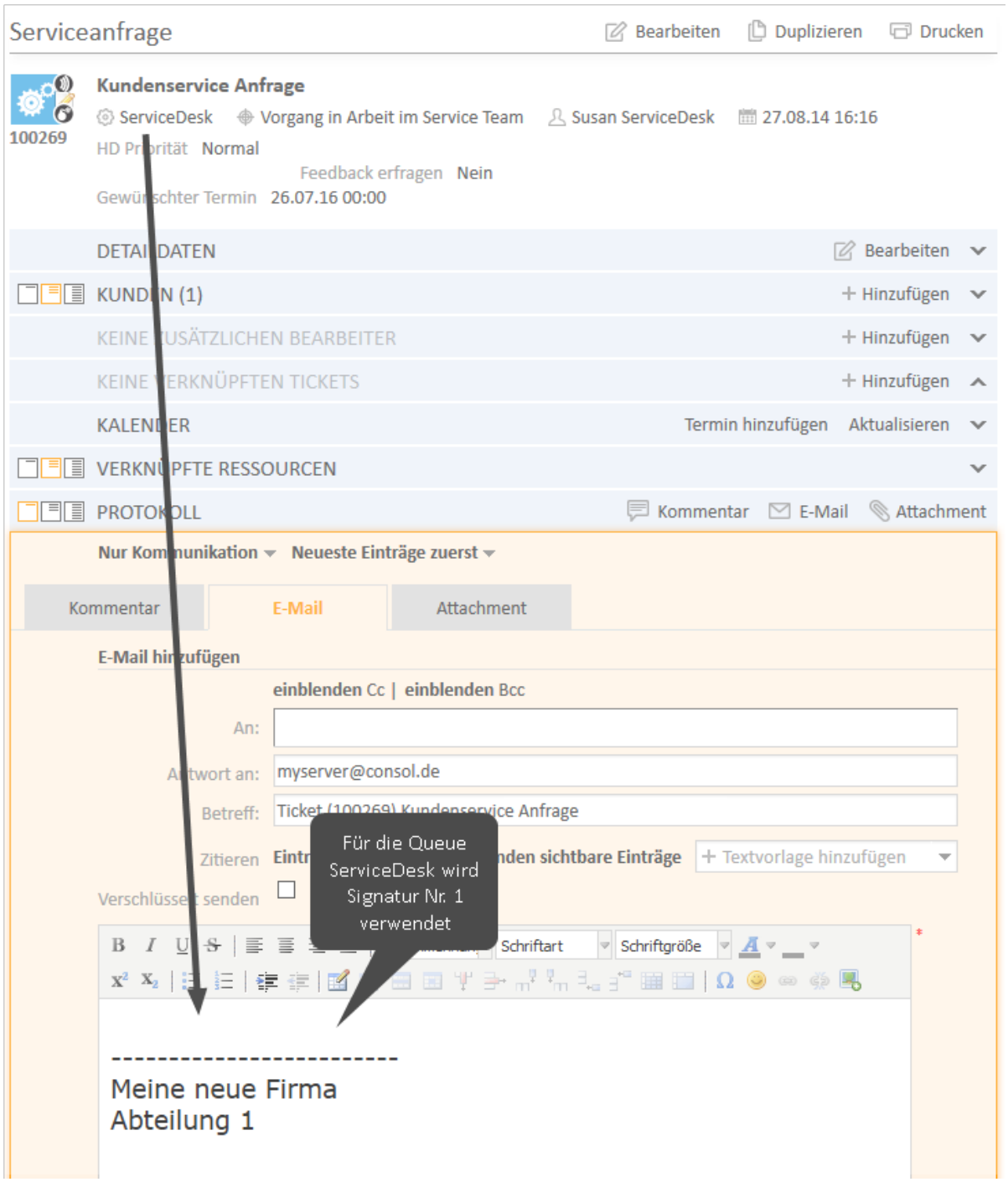


Abbildung 534: *ConSol CM Web Client - Queue-spezifische Signatur, verwaltet durch Vorlagenskript, 1*

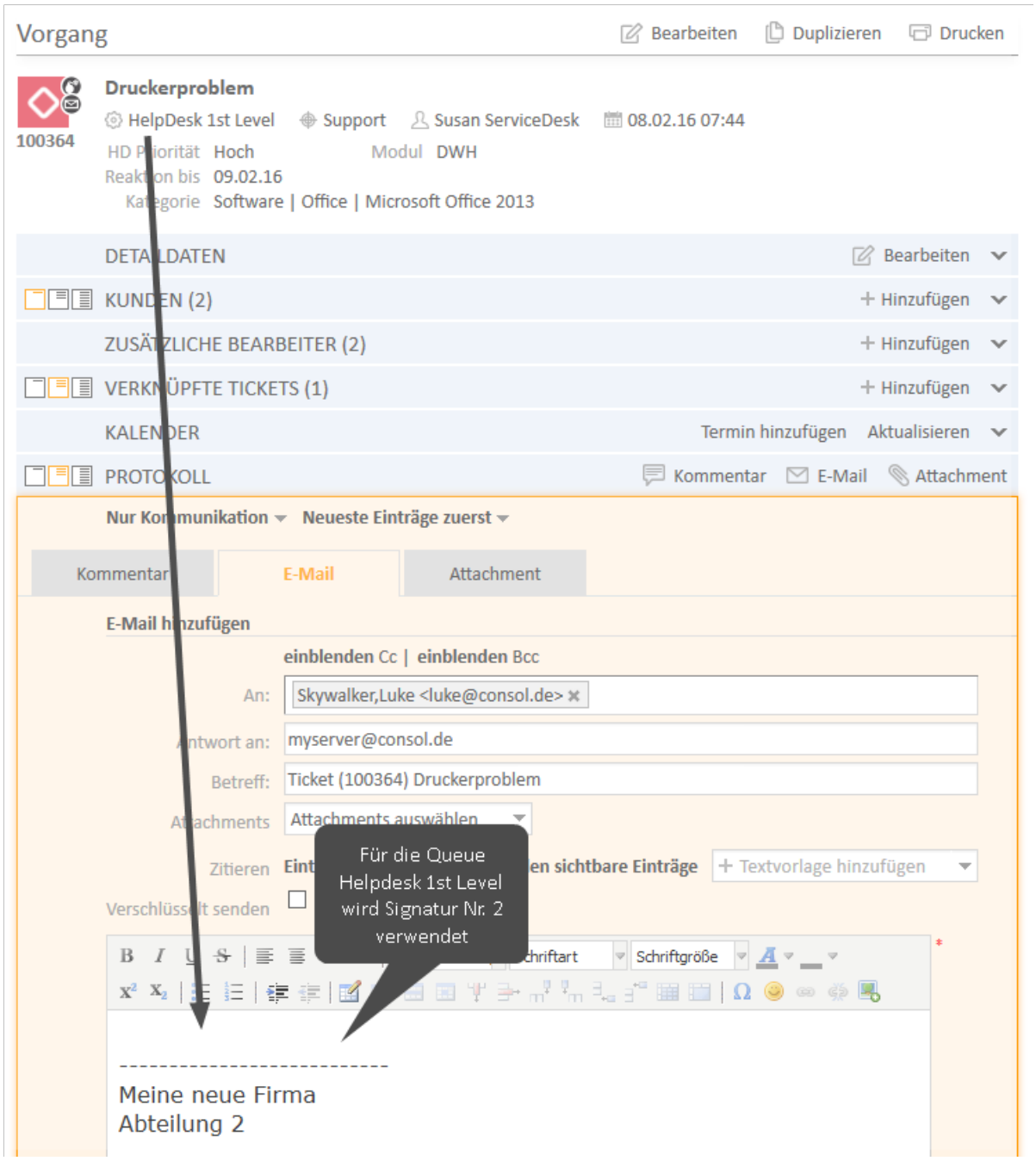


Abbildung 535: *ConSol CM Web Client - Queue-spezifische Signatur, verwaltet durch Vorlagenskript, 2*

Zuordnung von Vorlagen zu Queues oder bestimmten Parametern

Der Abschnitt *Zuordnung* ist der letzte Abschnitt der Seite *Neue Textvorlage* im Textvorlagen-Manager.

Für jede Vorlage können Sie entscheiden, ob sie ohne Einschränkungen überall angezeigt werden soll (d. h. in jeder Queue und ohne Berücksichtigung von Parametern) oder ob sie an bestimmte Kriterien gebunden (= eingeschränkt) sein soll. Das kann Folgendes sein:

- Queues
- queue-spezifische Parameter (z. B. Vorlage wird nur angezeigt, wenn das Ticket die Priorität *hoch* hat)

Sie können Queues und/oder Parameter auswählen, indem Sie einen *Kontext* auswählen, siehe Beispiele in den folgenden Abbildungen. Mit dem Plus-Button (+) können Sie weitere Zuordnungsparameter hinzufügen. Mit dem Minus-Button (-) können Sie vorhandene Parameter löschen.

Beispiel 1: Die Vorlage soll nur in der Queue *Helpdesk 1st Level Deutschland* angezeigt werden:

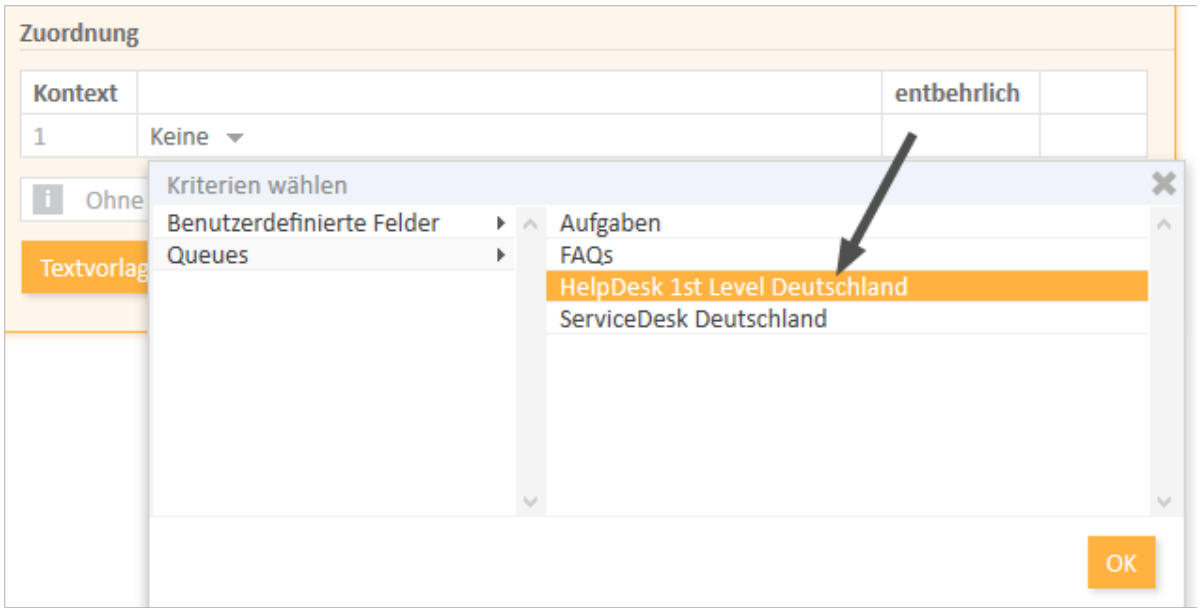


Abbildung 536: *ConSol CM Web Client - Vorlage für bestimmte Queue anzeigen*

Beispiel 2: Diese Vorlage soll nur angezeigt werden, wenn die Queue *Helpdesk 1st Level Deutschland* und die Priorität *Hoch* ist:

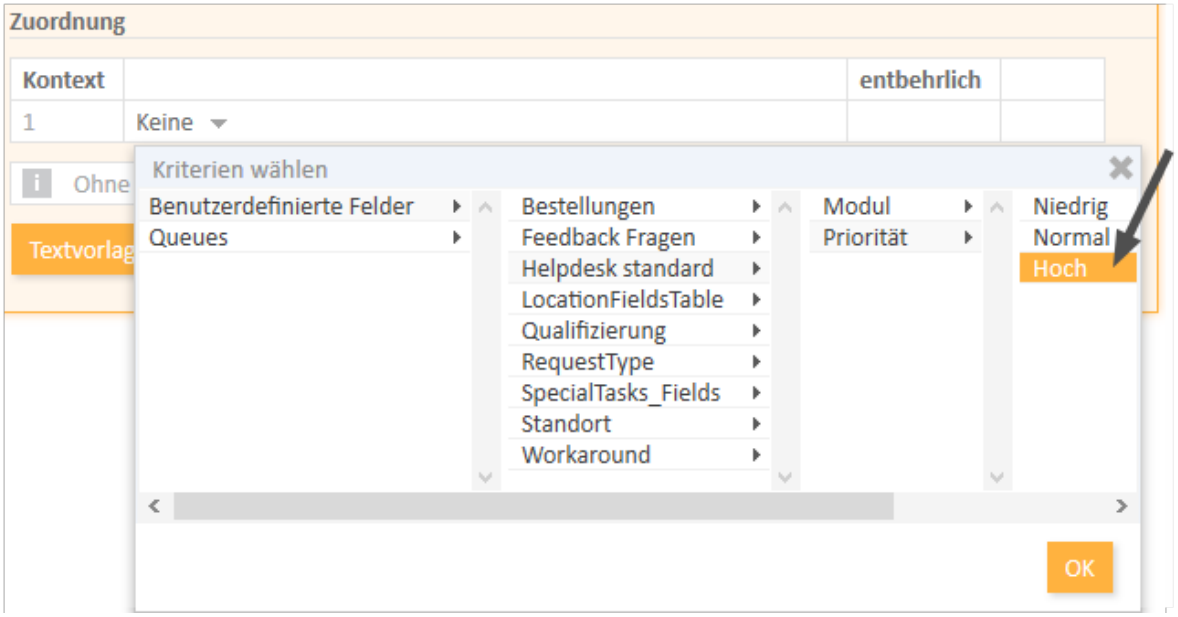


Abbildung 537: *ConSol CM Web Client - Vorlage für bestimmte Queue und Priorität anzeigen*

Beachten Sie bezüglich der Zuordnung einer Vorlage an spezifische Parameter Folgendes:

1. Als spezifische Parameter sind nur Ticketfelder des Typs ENUM verfügbar.
2. Ein oder mehrere spezifische Parameter (d. h. Ticketfelder des Typs ENUM) können ausgeschlossen werden. Das bedeutet, dass diese nicht mehr im Menü *Kriterien wählen* angezeigt werden. Wenn eine große Anzahl an Ticketfeldern bei jedem Öffnen des Textvorlagen-Editors geladen wird, kann die Performance beeinträchtigt werden. Das Laden der Seite zum Bearbeiten kann so ziemlich lange dauern. In solchen Fällen kann es helfen, Ticketfelder aus der Liste auszuschließen, solange klar ist, dass sie für den Entwurf der Vorlagen auf keinen Fall benötigt werden.
   Sie können spezielle Ticketfelder in der Seitenanpassung von markersLibrary (Typ) ausschließen.

Unentbehrliche und entbehrliche Zuordnung

Wenn die Vorlage, wie im obigen Beispiel gezeigt, nur in einer (oder mehreren) ausgewählten Queues angezeigt wird, ist die Vorlage diesen Queues bzw. den anderen ausgewählten (einschränkenden) Parametern *zugeordnet*. Es gibt zwei Arten von Zuordnung:

- **Unentbehrliche Zuordnung**
  Die Checkbox *entbehrlich* ist **nicht** markiert:
  Die Vorlage wird nur in den ausgewählten Queues und für die ausgewählten Parameter angezeigt (im Ticket-E-Mail-Editor oder Ticket-Kommentar-Editor angeboten). Der Bearbeiter, der

mit der Vorlage arbeitet, kann diese Konfiguration nicht ändern.

- **Entbehrliche Zuordnung**
  Die Checkbox *entbehrlich* ist markiert:
  Standardmäßig wird die Vorlage nur in den ausgewählten Queues und für die ausgewählten Parameter angezeigt (im Ticket-E-Mail-Editor oder Ticket-Kommentar-Editor angeboten). Der Bearbeiter kann die Anzeige aber ändern, indem er auf den Button *Mehr Templates* klickt. Dann werden alle Vorlagen mit einer entbehrlichen Zuordnung ebenfalls angezeigt.

F.12.1.4 Migrieren von Vorlagen von CM-Version 6.8 und niedriger auf CM-Version 6.9 und höher

Wenn ein ConSol CM-System von Version 6.8 und niedriger auf Version 6.9 und höher aktualisiert wird, müssen die Skripte im Textvorlagen-Manager manuell überprüft und geändert werden.

Für alle Parameter, die sich auf Tickets und Queues beziehen, ist keine Änderung erforderlich. Aufgrund des neuen Kundendatenmodells FlexCDM wurde die Syntax für Referenzen auf Kundenfelder allerdings angepasst.

F.12.1.5 Seitenanpassung für E-Mail-Vorlagenfunktionen

Im Abschnitt Seitenanpassung erfahren Sie, wie Sie Parameter für E-Mail-Vorlagen anpassen können.

F.12.2 CM/Doc

F.12.2.1 Einführung in CM/Doc

Auch in Unternehmen, in denen die meisten Prozesse über IT-Applikationen gesteuert werden, müssen häufig noch viele Dokumente ausgedruckt werden, oder werden aus unterschiedlichen Gründen in den Formaten `.doc` oder `.pdf` benötigt. Mögliche Beispiele sind:

- Rechnungen
- Verträge
- Dokumente über die Abnahme von IT-Systemen
- Bestellungen

ConSol CM bietet das Add-on CM/Doc, mit dem Dokumente direkt aus dem Geschäftsprozess gedruckt werden können. CM/Doc unterstützt Microsoft Word-Dokumente und (in CM-Versionen 6.10.1 und höher) auch OpenOffice-Dokumente.

Vorlagen garantieren, dass ...

- alle Dokumente eines Typs identisch sind (Text und Layout).
- alle Dokumente dem Corporate Design des Unternehmens entsprechen.
- die Bearbeiter nicht immer wieder den gleichen Text eingeben müssen.

Folgende Daten aus Tickets können automatisch in die Vorlage integriert werden:

- Ticketdaten (z. B. Betrag einer Rechnung, Servicelevel in einem Vertrag)
- Kundendaten (Name und Adresse des Hauptkunden (und der Firma, sofern der Hauptkunde ein Kontakt ist und zu einer Firma gehört), Zusatzkunden sind in CM/Doc **nicht** relevant!)
- Bearbeiterdaten (Name, Telefonnummer, E-Mail-Adresse des Bearbeiters, der den Vorfall bearbeitet)

Wenn CM/Doc in ConSol CM aktiv ist, kann der Bearbeiter die benötigte Microsoft Word-Vorlage (bzw. OpenOffice-Vorlage) im Ticket auswählen. Das Dokument wird automatisch in Microsoft Word (OpenOffice) geöffnet und die benötigten Datenfelder sind schon ausgefüllt. Der Bearbeiter kann das Dokument dann bearbeiten und speichern. Es wird automatisch (als normales Attachment) an das Ticket angehängt und kann von den Benutzern geöffnet werden, die **Lesezugriff** auf das Ticket haben und auf deren PCs die entsprechende Software (Microsoft Word, OpenOffice) installiert ist.

Mit besonderen Anpassungen durch das ConSol CM Consulting-Team kann das ConSol CM-System so erweitert werden, dass .docx-Dokumente in .pdf-Dateien konvertiert werden, damit sichergestellt ist, dass keine weiteren Änderungen am Dokument vorgenommen werden können.

F.12.2.2 Voraussetzungen für die Verwendung von CM/Doc

Der Client-PC oder -Laptop, auf dem CM/Doc verwendet werden soll, muss folgende Voraussetzungen erfüllen:

- JRE (Java Runtime Environment) muss für den Webbrowser installiert sein, da CM/Doc auf Java-Applets basiert. Die unterstützten Java-Versionen finden Sie in den aktuellen *Systemanforderungen*.

- Ein Webbrowser, der Java-Applets unterstützt, siehe aktuelle *Systemanforderungen*.
- Microsoft Word / OpenOffice muss installiert sein. Die unterstützten Microsoft Word- und OpenOffice-Versionen finden Sie in den aktuellen *Systemanforderungen*.

F.12.2.3 Verfügbarkeit von CM/Doc

CM/Doc ist in ConSol CM-Version 6.7 und höher verfügbar und gehört zu den Standardfunktionen der Applikation.

F.12.2.4 Konfigurieren des ConSol CM-Systems für CM/Doc

Wenn Sie CM/Doc in Ihrem ConSol CM-System aktivieren möchten, müssen Sie zuerst die System-Property cmweb-server-adapter, cmoffice.enabled auf „true“ setzen.

Sie gelangen auf diese Seite, indem Sie das Navigationselement *System-Properties* der Navigationsgruppe *System* öffnen und auf den entsprechenden Eintrag klicken.

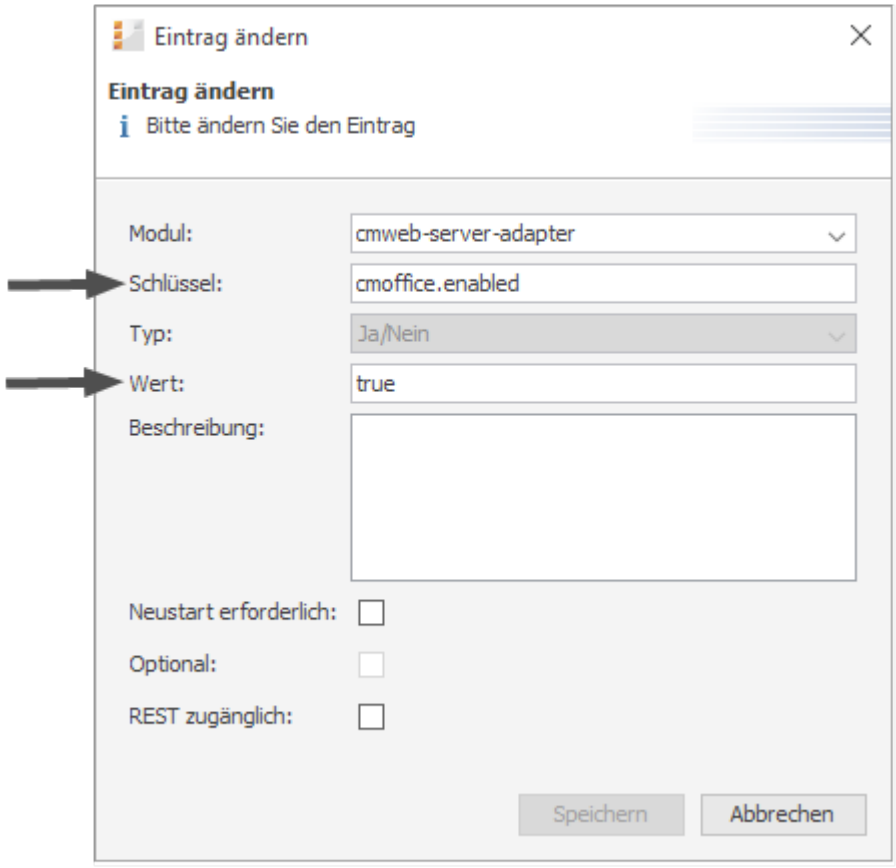


Abbildung 538: *ConSol CM Admin Tool - System, System-Properties: Konfiguration der System-Property für CM/Doc*

F.12.2.5 Erstellen einer Bearbeiterrolle mit Berechtigungen für den Dokumentvorlagen-Manager

Nur Bearbeiter mit der Berechtigung *Template speichern* (siehe folgende Abbildung) können den *Dokumentvorlagen-Manager* im Web Client starten. Im zweiten Schritt müssen Sie daher eine oder mehrere Rollen mit der entsprechenden Berechtigung erstellen. Eine detaillierte Beschreibung über das Einstellen der Rollenberechtigungen finden Sie im Abschnitt Rollenverwaltung.

Sie gelangen auf diese Seite, indem Sie das Navigationselement *Rollen* der Navigationsgruppe *Zugriff und Rollen* öffnen und eine Rolle auswählen.

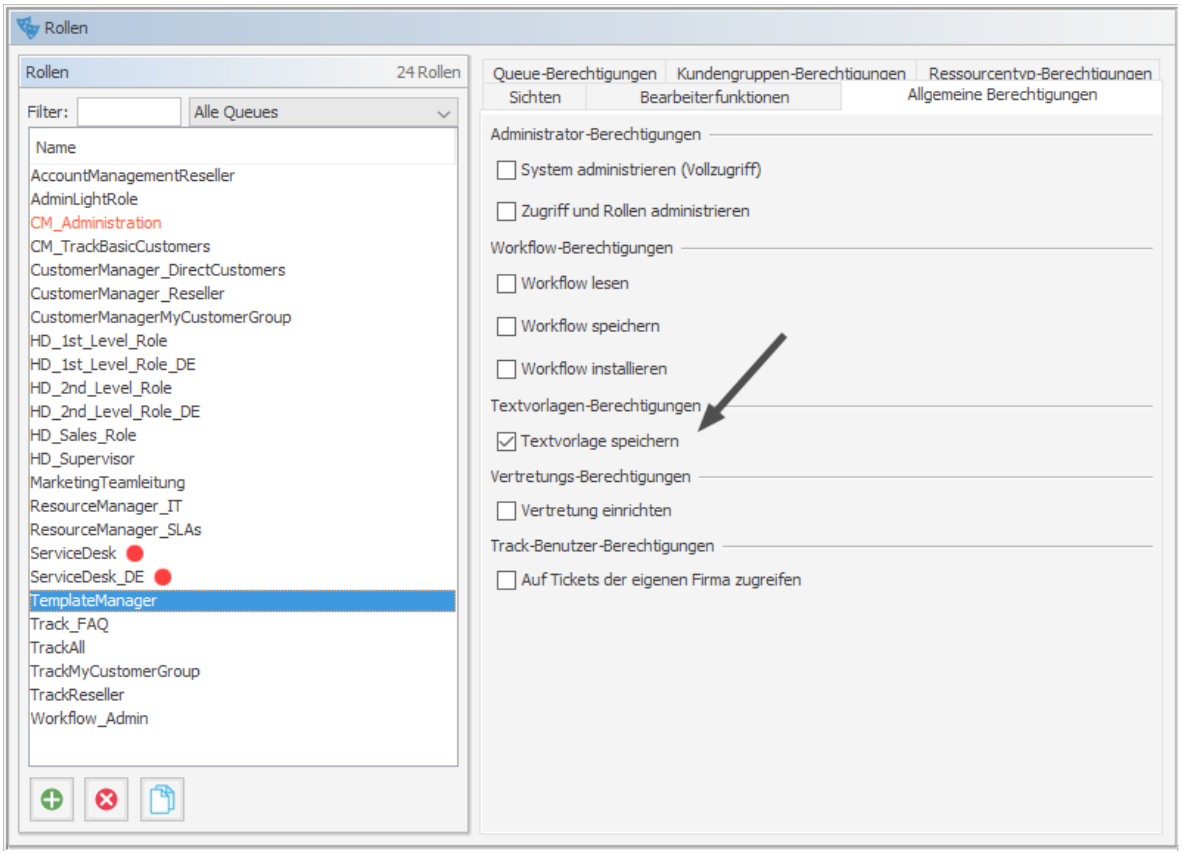


Abbildung 539: *ConSol CM Admin Tool - Zugriff und Rollen, Rollen: Erforderliche Vorlagenberechtigungen*

Wir empfehlen Ihnen, eine Rolle (z. B. *TemplateManager*) zu erstellen, die nur die Berechtigung *Template speichern* hat, und an die keine Queue-Berechtigungen oder anderen Berechtigungen vergeben werden. Jedem Benutzer, der Zugang zum Dokument-Vorlagenmanager soll, kann diese Rolle zugewiesen werden. Auf diese Weise werden normale Benutzerberechtigungen und Berechtigungen für Dokumentvorlagen nicht vermischt und Sie können die Berechtigung für Dokumentvorlagen schnell erteilen und entziehen.

Mit der Berechtigung *Template speichern* wird auch der Zugriff auf den Textvorlagen-Manager erteilt (für Textvorlagen, siehe Abschnitt Der ConSol CM Textvorlagen-Manager).

F.12.2.6 Erstellen und Bereitstellen von Microsoft Word-Vorlagen

Erstellen von Microsoft Word-Vorlagen

Im dritten Schritt müssen Sie die Microsoft Word-Vorlagen erstellen. Dies erfolgt in Microsoft Word. Erstellen Sie als Vorlage Dateien mit den Dateiendungen `.doc` oder `.docx`, **keine** Dateien mit der Dateiendung `.dot`!

Die spezielle Konfiguration für OpenOffice ist im Abschnitt Verwenden von CM/Doc mit OpenOffice erklärt. Die meisten Konfigurationsschritte sind allerdings für Microsoft Word- und OpenOffice-Vorlagen identisch oder sehr ähnlich.

Erstellen des Vorlagenobjekts im Vorlagen-Archiv

Im vierten Schritt müssen Sie die erforderlichen Datenfelder als *MergeFields* in der Microsoft Word-Vorlage anlegen, d. h. Sie erstellen eine CM/Doc-Vorlage aus einem normalen Microsoft Word-Dokument. Dies geschieht im ConSol CM Web Client.

Um Schritte drei und vier durchzuführen, melden Sie sich im Web Client an und klicken Sie im Hauptmenü auf *Dokumentvorlagen*, um den Dokumentvorlagen-Manager zu öffnen.

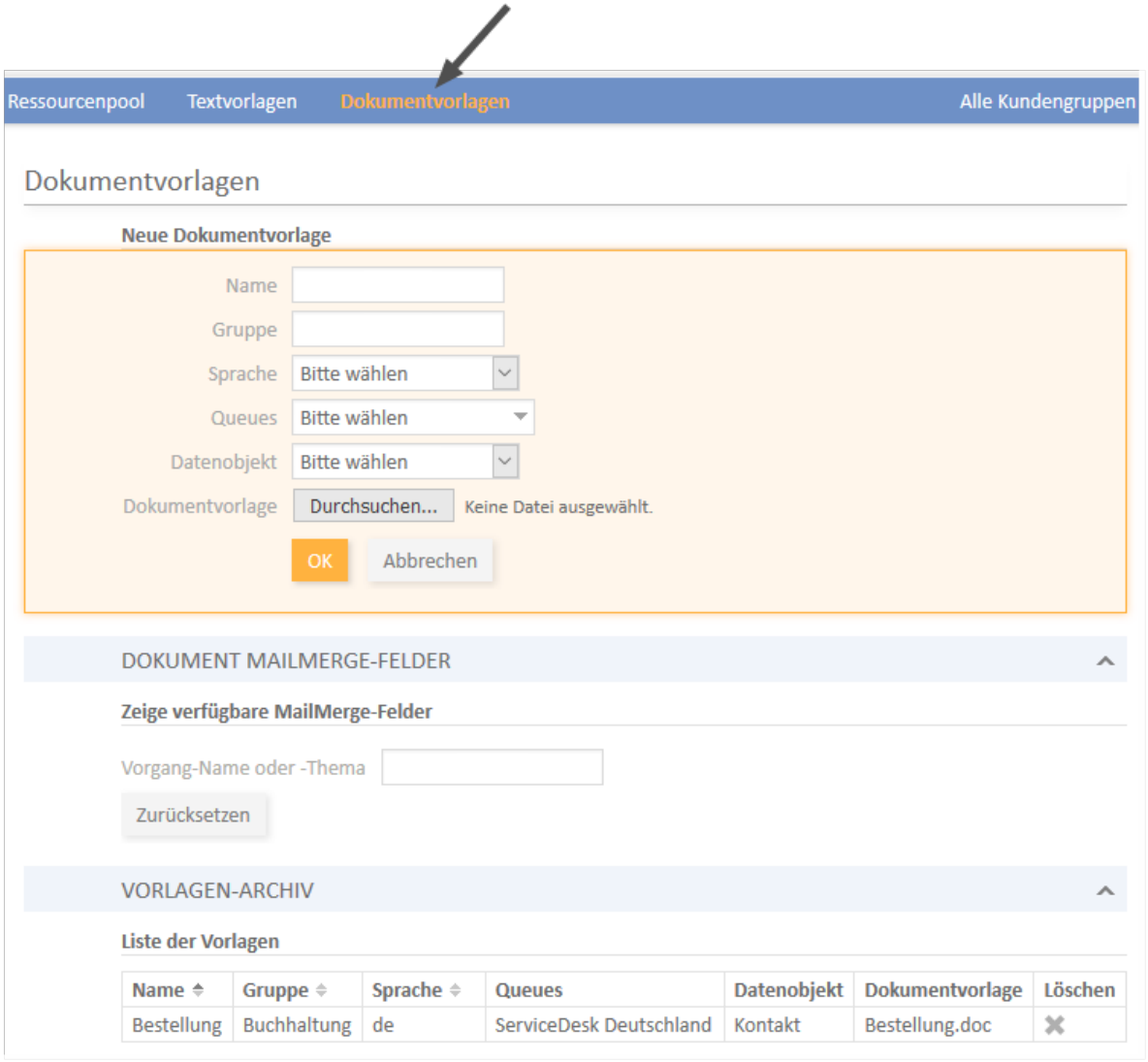


Abbildung 540: *ConSol CM Web Client - Dokumentvorlagen-Manager*

Der Dokumentvorlagen-Manager wird geöffnet.

Geben Sie für jede neue Vorlage folgende Daten ein und klicken Sie auf *OK*:

- **Name**
  Der Name der (neuen) Vorlage. Er wird den Bearbeitern im Web Client angezeigt.
- **Gruppe**
  Der Name der Vorlagengruppe. Dieser hat keine technischen Auswirkungen und dient lediglich dazu, die Vorlagen in der Liste im Dokumentvorlagen-Manager einfach sortieren zu können.
- **Sprache**
  Wählen Sie die benötigte Sprache aus. Es sind die im Admin Tool aktivierten Sprachen aufgeführt.

- **Queues**
  Wählen Sie die Queues aus, in denen die Vorlage zur Verfügung stehen soll.
- **Datenobjekt**
  Wählen Sie das Kundenobjekt aus, das in der Vorlage als Referenzobjekt für die Kundendaten verwendet werden soll. Es sind alle Kundenobjekte aufgeführt. Eine detaillierte Erklärung dazu, welches Kundenobjekt Sie auswählen müssen, finden Sie im Abschnitt Die Wahl des Kundenobjekts und ihre Konsequenzen.
- **Dokumentvorlage**
  Wählen Sie mithilfe des Dateibrowsers im Dateisystem die .doc- oder .docx-Datei aus, die als Vorlage dienen soll.

Die Wahl des Kundenobjekts und ihre Konsequenzen

In dem Formular, in dem Sie alle Felder für eine Microsoft Word-Vorlage ausfüllen, müssen Sie ein Kundenobjekt auswählen, d. h. ein Kundenobjekt (Kontakt- oder Firmenobjekt) aus Ihrem Kundendatenmodell. Wenn Sie mit dem ConSol CM-Kundendatenmodell nicht vertraut sind, lesen Sie bitte zuerst den Abschnitt Kundendatenmodell.

Im Drop-down-Menü *Datenobjekt* sind die Kundenobjekte im folgenden Format aufgeführt:

- [lokalisierter Name des Kundenobjekts (lokalisierter Name des Kundendatenmodells)]

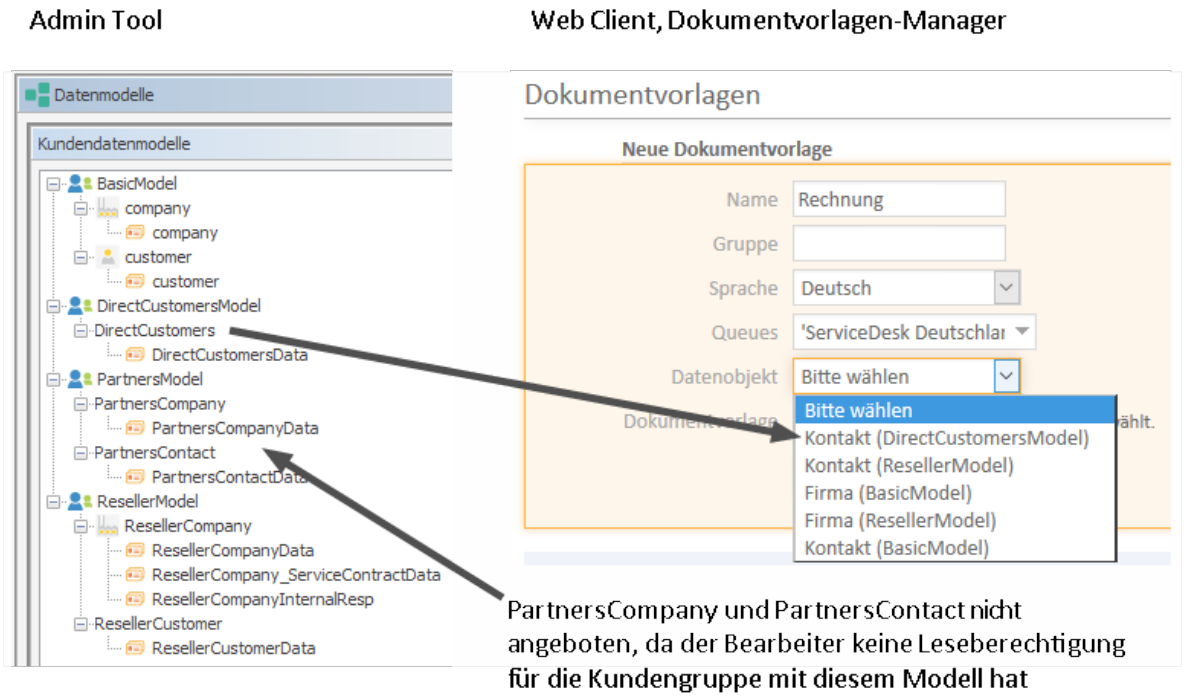
Damit ein Kundenobjekt im Drop-down-Menü aufgeführt wird, müssen folgende Bedingungen erfüllt sein:

1. Das Kundendatenmodell mit dem entsprechenden Kundenobjekt muss im Admin Tool erstellt worden sein.
2. Das Kundendatenmodell muss mindestens einer Kundengruppe zugewiesen sein.
3. Der Bearbeiter, der mit den Dokumentvorlagen arbeitet, muss mindestens Lesezugriff auf mindestens eine Kundengruppe mit dem entsprechenden Kundendatenmodell haben.

Sie müssen das Kundenobjekt auswählen, das der Hauptkunde der Tickets verwendet. Die Microsoft Word-Vorlage ist nur in Tickets verfügbar, deren Hauptkunde aus einer Kundengruppe mit diesem Kundendatenmodell stammt. Dies bedeutet, dass Sie zwei Microsoft Word-Vorlagen (eine für jedes Kundendatenmodell) in ConSol CM erstellen müssen, wenn Sie die gleiche Microsoft Word-Vorlage für zwei Kundengruppen mit zwei unterschiedlichen Kundendatenmodellen verwenden möchten. Natürlich können Sie dieselbe Microsoft Word-Vorlage für zwei oder mehr Kundengruppen verwenden, die dasselbe Kundendatenmodell haben.

Wenn Sie versuchen, ein Kundenobjekt einer Microsoft Word-Vorlage zuzuweisen, die mit der ausgewählten Queue nicht kompatibel ist (weil das ausgewählte Datenmodell der ausgewählten Queue nicht zugewiesen ist), wird eine Fehlermeldung angezeigt.

Beachten Sie, dass es nur sinnvoll ist, ein Objekt auf Firmenstufe auszuwählen, wenn im entsprechenden Kundendatenmodell der Parameter *Firma als Kunde* gesetzt ist.



> Bedenken Sie, dass die Microsoft Word-Vorlage theoretisch für alle Kundengruppen zur Verfügung steht, wenn Sie das Feld *Kundenobjekt* leer lassen. Es kann dann aber zu Laufzeitfehlern kommen, wenn die erforderlichen Felder beim Laden der Vorlage nicht gefunden werden. ConSol CM zeigt in diesem Fall eine Meldung an, dass Feldinhalt fehlt, und die Felder werden mit der Zeichenfolge "n/a" gefüllt. Wir empfehlen Ihnen, immer ein Kundenobjekt auszuwählen, um das System in gutem Zustand zu halten.

So steuern Sie, in welchen Tickets die Microsoft Word-Vorlage angeboten wird

Ob die Microsoft Word-Vorlage in einem bestimmten Ticket angeboten wird (siehe Abschnitt Verwendung von Microsoft Word-Vorlagen im Web Client unten) hängt von zwei Parametern ab:

- **Queue**
  Die Vorlage ist nur in Tickets der ausgewählten Queues aufgeführt
- **Kundendatenmodell**
  Die Vorlage ist nur in Tickets aufgeführt, bei denen der Hauptkunde auf dem ausgewählten Kundenobjekt basiert

Beispiel (Wir nehmen hier die Verwendung von Microsoft Word-Vorlagen im Web Client vorweg. Details dazu finden Sie im Abschnitt Verwendung von Microsoft Word-Vorlagen im Web Client):

- Zuerst wird die Microsoft Word-Vorlage für die Queue *ServiceDesk* und für das Kundenobjekt *Reseller Kundendaten* (erste Abbildung) definiert.

- Im Web Client **kann** die Vorlage in einem Ticket verwendet werden, das sich in der Queue *ServiceDesk* befindet **und** als Hauptkunden ein Kundenobjekt des Typs *Reseller Kundendaten* hat (zweite Abbildung).
- Im Web Client **kann** die Vorlage **nicht** in einem Ticket verwendet werden, das sich in der Queue *ServiceDesk* befindet und als Hauptkunden ein Kundenobjekt eines **anderen** Typs als *Reseller Kundendaten* hat (dritte Abbildung).
- Im Web Client kann die Vorlage nicht in Tickets verwendet werden, die sich **nicht** in der Queue *ServiceDesk* befinden (nicht abgebildet).

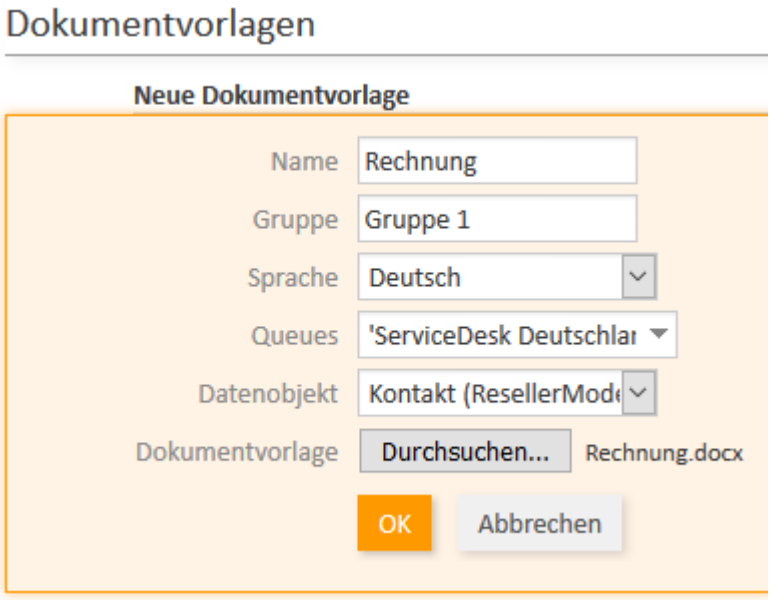


Abbildung 541: *ConSol CM Web Client - Definition einer Microsoft Word-Vorlage für eine Queue und ein bestimmtes Kundenobjekt*

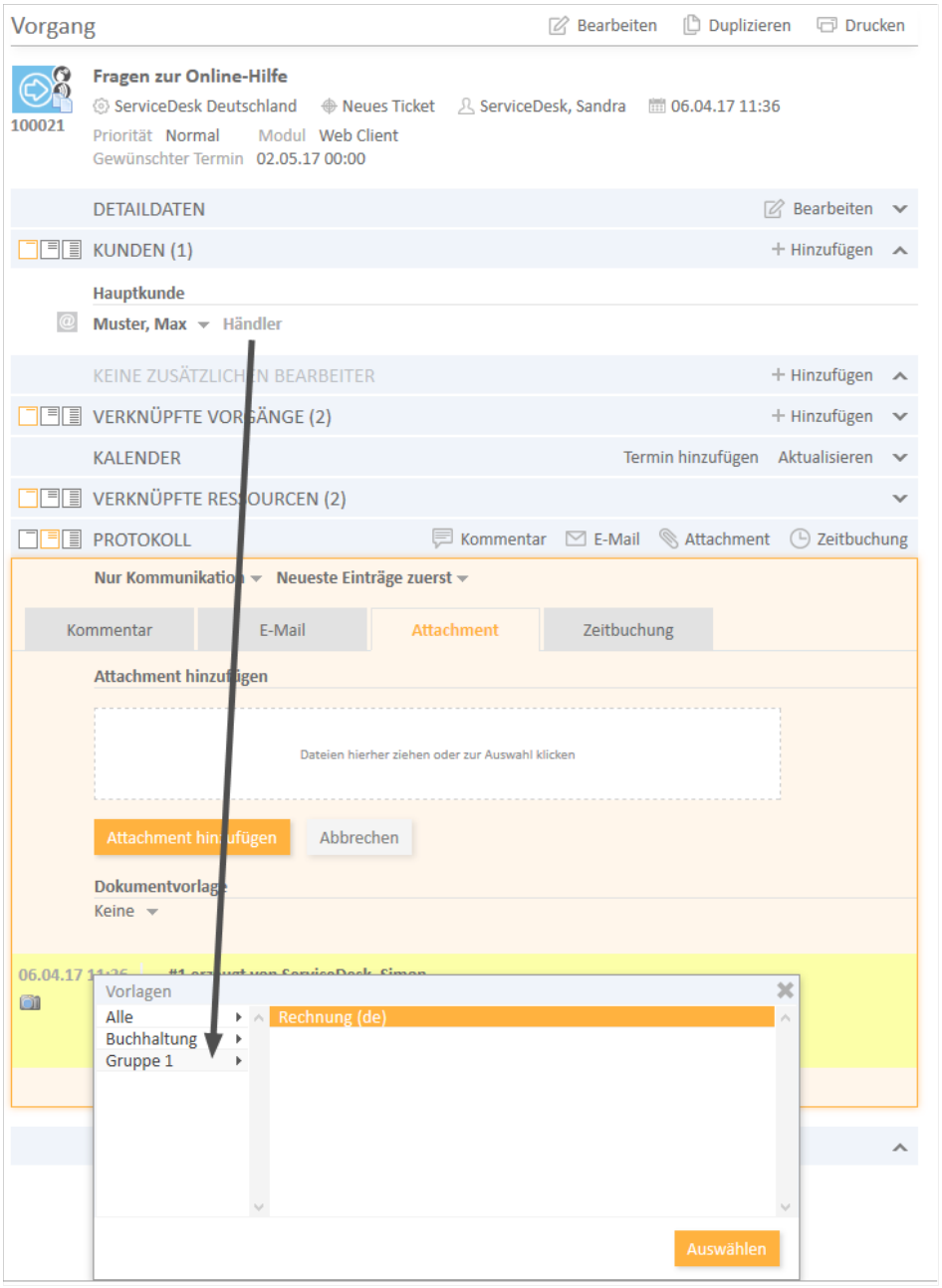


Abbildung 542: *ConSol CM Web Client - Microsoft Word-Vorlage wird aufgrund von Queue und Kundengruppe angeboten*

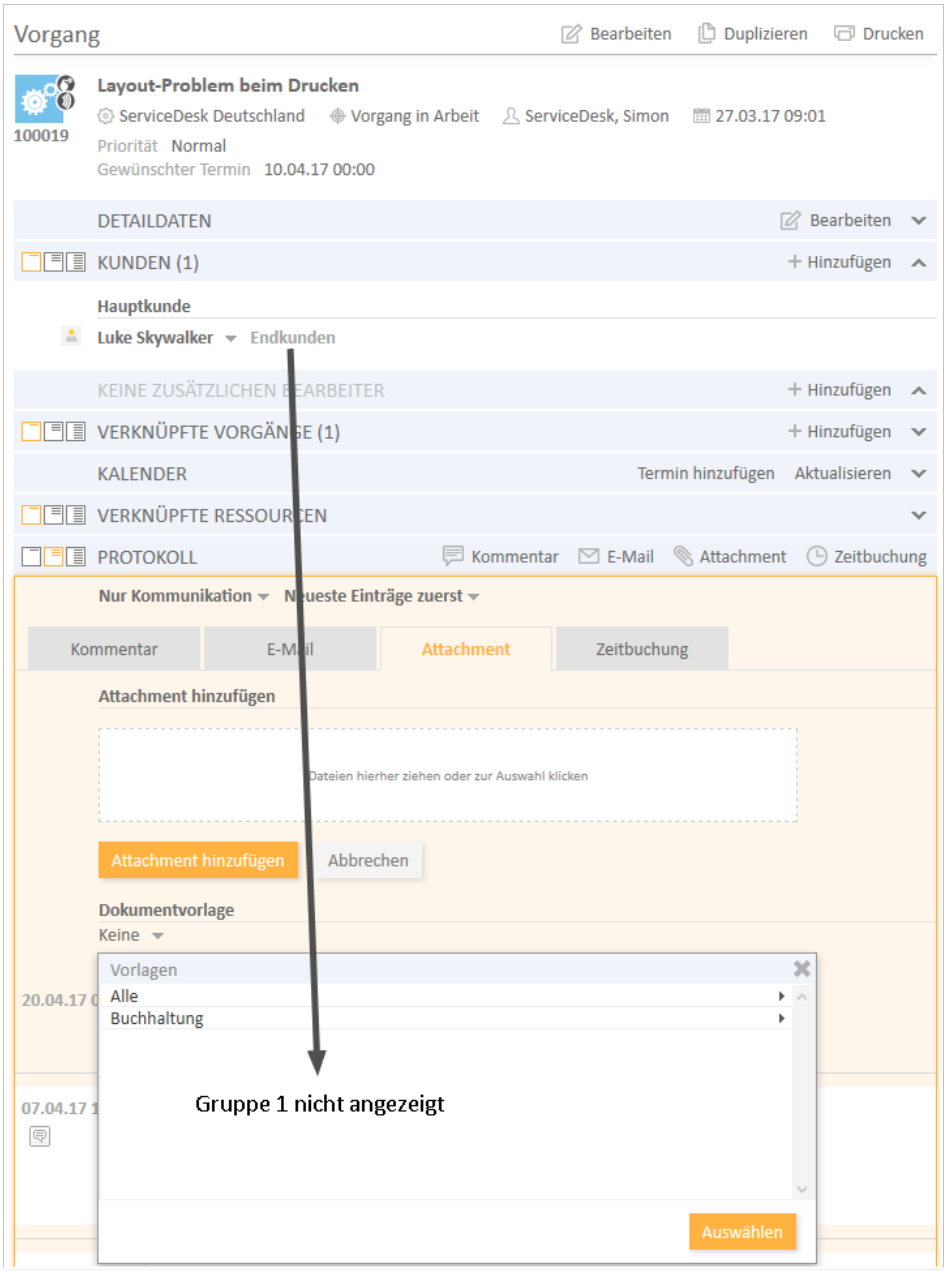


Abbildung 543: *ConSol CM Web Client - Microsoft Word-Vorlage wird aufgrund von Queue und Kundengruppe NICHT angeboten*

Erstellen und Editieren des benötigten Microsoft Word-Dokuments

Wenn Sie alle Felder ausgefüllt und auf *OK* geklickt haben, wird die neue Vorlage im *Vorlagen-Archiv*, *Liste der Vorlagen* angezeigt. Sie können den folgenden Schritt direkt nach der Eingabe der neuen Vorlagendaten durchführen oder (in der Spalte *Dokumentvorlage*) den Namen der Vorlagendatei anklicken, die Sie herunterladen und editieren möchten.

Im nächsten Schritt können *MailMerge-Felder*, die Ticket- und Kundendatenfelder darstellen, zur Vorlage hinzugefügt werden. Wählen Sie im Abschnitt *Dokument MailMerge-Felder* ein Ticket aus, das alle benötigten Felder hat. Geben Sie dazu den Ticketnamen oder das Ticketthema in das Feld unter *Zeige verfügbare MailMerge-Felder für ein Ticket* ein und wählen Sie das gewünschte Ticket aus der Liste der Suchergebnisse.

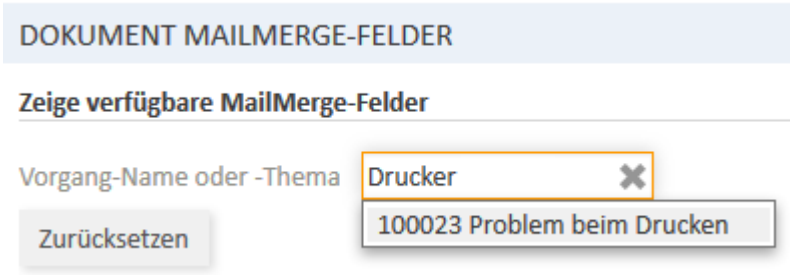


Abbildung 544: *ConSol CM Web Client - Auswahl eines Tickets, um alle benötigten Datenfelder anzuzeigen*

Alle verfügbaren *MailMerge-Felder* werden angezeigt.

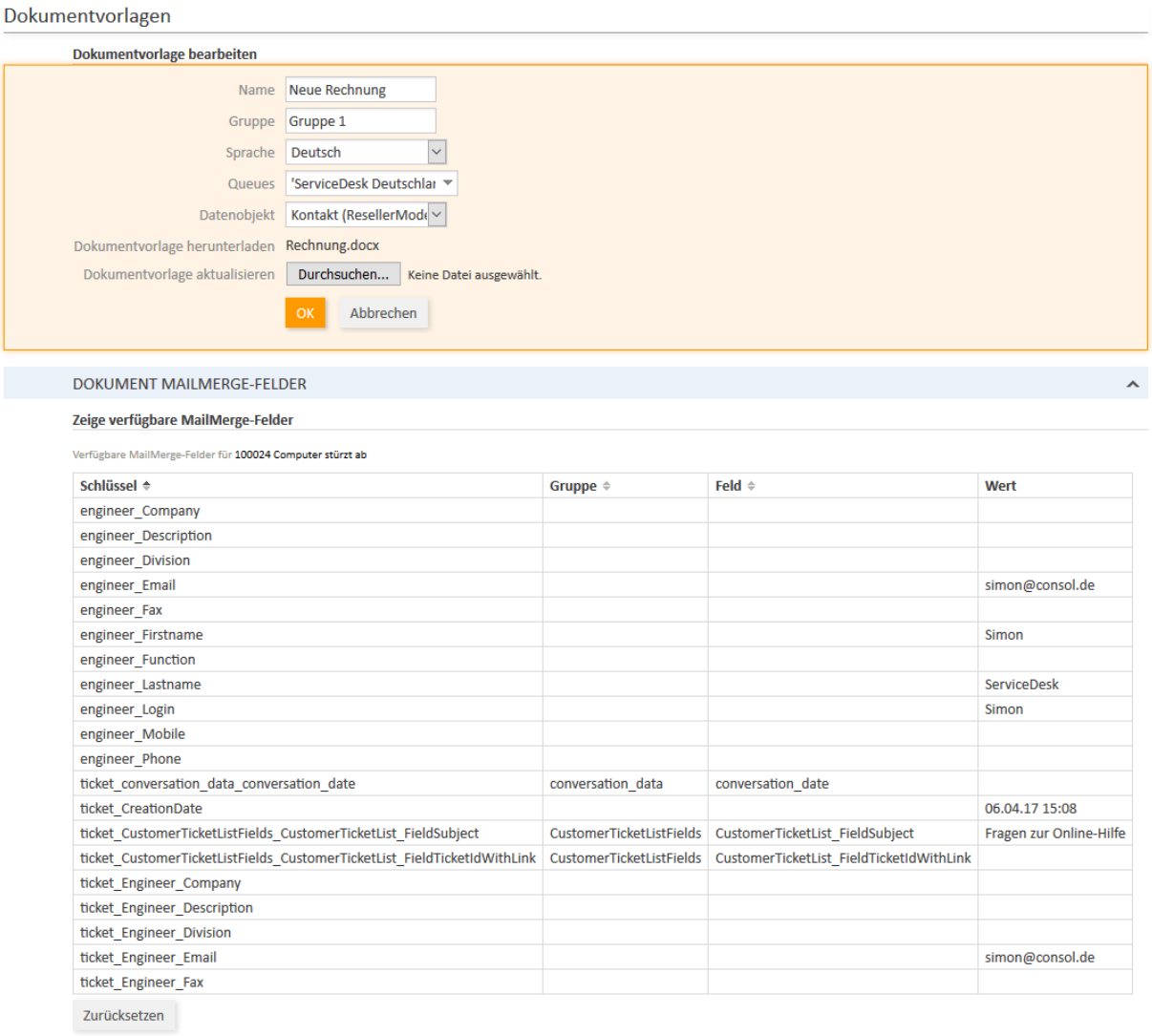


Abbildung 545: *ConSol CM Web Client - Verfügbare MailMerge-Felder anzeigen*

Die Liste hat folgende Spalten.

- **Schlüssel**
  Der Name des Felds (den Sie später für das MergeField verwenden müssen)
- **Gruppe**
  Wird für Kunden- und Ticketdaten angegeben. Für Kundendaten wird hier die Kundenfeldgruppe angezeigt. (Es werden nur die Kundenfeldgruppen des ausgewählten Kundenobjekts angeboten.) Für Ticketdaten wird die Ticketfeldgruppe angezeigt.
- **Feld**
  Wird für Kunden- und Ticketdaten angegeben. Für Kundendaten wird hier der Name des Kundenfeldes angezeigt. (Es werden nur die Kundenfelder des ausgewählten Kundenobjekts angeboten.) Für Ticketdaten wird der Name des Ticketfeldes angezeigt.

- **Wert**
  Der Wert, den das Feld im ausgewählten Ticket hat. Dieser muss in der Dokumentvorlage nicht verwendet werden.

Laden Sie die Microsoft Word-Vorlage aus einem der folgenden Orte herunter und öffnen Sie sie. In beiden Fällen sehen Sie den technischen Namen der Vorlage:

- aus dem Abschnitt *Dokumentvorlage bearbeiten*, in *Dokumentvorlage herunterladen*
- aus dem *Vorlagen-Archiv*, Spalte *Dokumentvorlage*

Klicken Sie in diesem Dokument an die Stelle, wo Sie das erste Feld einfügen möchten. In unserem Beispiel ist das der Kundenname. Über *Einfügen -> Schnellbausteine -> Feld* können Sie das *MergeField* einfügen. Kopieren Sie den Schlüssel des gewünschten MergeField aus dem Web Client und fügen Sie ihn in Microsoft Word in das entsprechende Feld ein (*Feldeigenschaften*, *Feldname*):

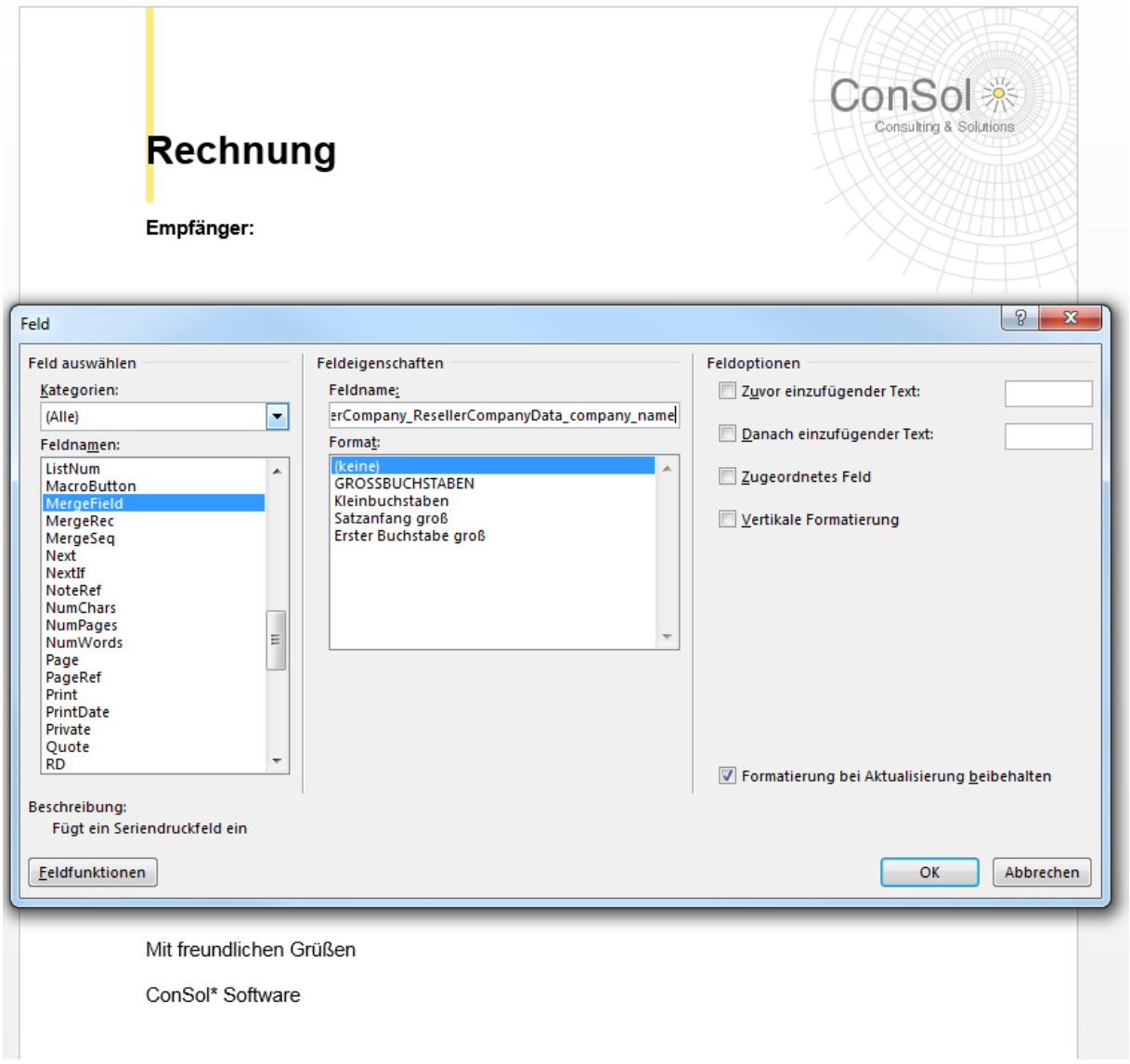


Abbildung 546: *Microsoft Word - MergeField in Dokument einfügen*

Wiederholen Sie diesen Schritt für alle Felder, die beim Öffnen der Microsoft Word-Vorlage vorausgefüllt sein sollen. Am Ende kann Ihre Vorlage zum Beispiel so aussehen, wie in der folgenden Abbildung gezeigt.

Sie können alle im Ticket angebotenen Felder verwenden, d. h.

- **Firmenfelder**
  Kundenfelder des ausgewählten Kundenobjekts, wenn es sich um eine Firma handelt, oder aus der Firma des ausgewählten Kundenobjekts, wenn das ausgewählte Kundenobjekt ein Kontakt ist. Die Schlüssel beginnen mit *ticket_queue_fields_* und haben als *Gruppe* den Namen der Kundenfeldgruppe.)
- **Kontaktfelder**
  Kundenfelder des ausgewählten Kundenobjekts, wenn es sich um einen Kontakt handelt, oder aus dem Kontakt des ausgewählten Kundenobjekts, wenn das ausgewählte Kundenobjekt eine Firma ist. Die Schlüssel beginnen mit *ticket_queue_fields_* und haben als *Gruppe* den Namen der Kundenfeldgruppe.)
- **Ticketdatenfelder**
  Ticketfelder aus dem Ticket. Die entsprechenden Schlüssel beginnen mit *ticket_* und haben als *Gruppe* den Namen der Ticketfeldgruppe.
- **Bearbeiterdaten**
  - Daten des aktuellen Bearbeiters des Tickets. Die entsprechenden Schlüssel beginnen mit *ticket_Engineer* und haben keine *Gruppen-* und *Feldnamen*.
  - Daten des aktuell angemeldeten Bearbeiters. Dies muss **nicht** unbedingt der Bearbeiter des Tickets sein! Die entsprechenden Felder beginnen mit *engineer_* und haben keine *Gruppen-* und *Feldnamen*.)



Abbildung 547: *Beispielvorlage in Microsoft Word*

Speichern Sie das Word-Dokument im lokalen Dateisystem und laden Sie es über das Feld *Dokumentvorlage aktualisieren* hoch. So wird die Vorlage im Dokumentvorlagen-Manager gespeichert.

Details über die ConSol CM-Schlüssel für MailMergeFields

Bei Kundendaten gibt für jeden Wert zwei Schlüssel, d. h. zwei Schlüssel rufen den Wert desselben Kundenfelds ab. Welchen der beiden Sie verwenden, macht hinsichtlich des Verhaltens der Vorlage keinen Unterschied.

Die beiden Schlüssel sind:

1. Ein allgemeines Feld, z. B. ticket_queue_fields_contacts_member_companyReferenceField_**company_name**
2. Ein Feld aus einem bestimmten Kundendatenmodell entsprechend der folgenden Syntax xxx_[Kundenobjekt]_[Kundenfeldgruppe]_[Kundenfeld], z. B. ticket_queue_fields_contacts_member_companyReferenceField_ResellerCompany_ResellerCompanyData**_company_name**

F.12.2.7 Verwendung von Microsoft Word-Vorlagen im Web Client

Erstellen eines neuen Microsoft Word-Attachments mithilfe einer Dokumentvorlage

Wenn Microsoft Word-Vorlagen für eine Queue verfügbar sind, kann ein Bearbeiter sie verwenden, indem er im Protokollbereich eines Tickets auf *Attachments* klickt und die gewünschte Vorlage auswählt. Microsoft Word wird gestartet (wenn es nicht bereits geöffnet ist) und es wird ein Dokument erstellt, das auf der ausgewählten Vorlage basiert. Das Dokument wird geöffnet. Alle Werte/Parameter sind bereits an der richtigen Stelle ausgefüllt. Dies kann zum Beispiel so aussehen, wie in den folgenden Abbildungen dargestellt.

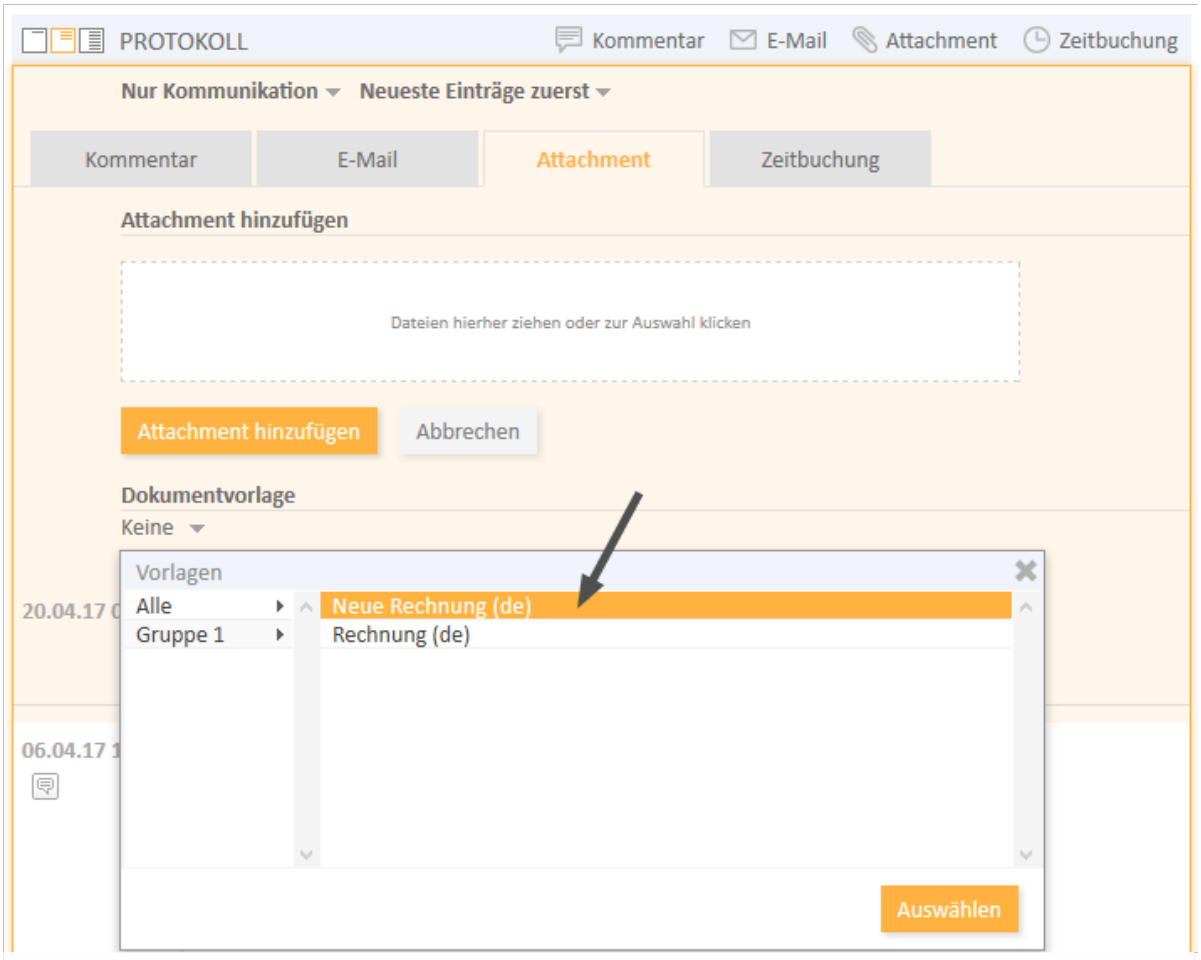


Abbildung 548: *ConSol CM Web Client - Microsoft Word-Vorlage für Attachments verfügbar*



Rechnung

Empfänger:

Luke Skywalker

MyNewSpaceCompany
Milkyway 77
Alderaan

Datum der Rechnung: <bitte ausfüllen>

Servicezeit: 24.07.15 09:21 bis < bitte ausfüllen>

Services: < bitte ausfüllen>

Durchgeführt von: Susan ServiceDesk

Bitte zahlen Sie den Betrag von < bitte ausfüllen> entsprechend des Vertrags.

Vielen Dank.

Mit freundlichen Grüßen

ConSol* Software

Abbildung 549: *Beispiel für Microsoft Word-Dokument*

Der Bearbeiter kann das Dokument editieren, falls dies erforderlich ist, und speichern. Damit wird die neue Version des Dokuments automatisch an das Ticket angehängt.

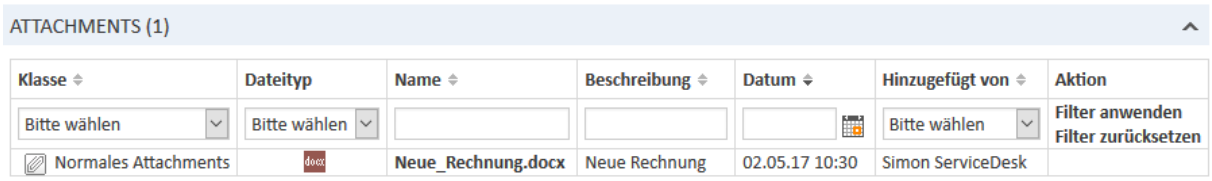


Abbildung 550: *ConSol CM Web Client - Editiertes Word-Dokument als Attachment im Ticket*

Ab hier wird das Word-Dokument genauso bearbeitet wie ein normales Attachment im Word-Format (.doc, .docx). Siehe nächster Abschnitt.

Arbeiten mit vorhandenen Microsoft Word-Attachments

Als Bearbeiter können Sie auch ein Microsoft Word-Dokument öffnen, das an das Ticket angehängt wurde. Klicken Sie dazu auf den Namen des Attachments. Dadurch wird die Datei in Microsoft Word geöffnet. Editieren Sie das Dokument und speichern Sie es. Eine neue Version des Dokuments wird automatisch an das Ticket angehängt.

F.12.2.8 Verwenden von CM/Doc mit OpenOffice

Um CM/Doc mit OpenOffice zu verwenden, führen Sie folgende Schritte durch:

Schritt 1: Aktivieren von CM/Doc in Ihrem CM-System

Siehe Abschnitt Konfigurieren des ConSol CM-Systems für CM/Doc.

Schritt 2: Erstellen der Rolle mit Zugangsberechtigungen für CM/Doc

Siehe Abschnitt Erstellen einer Bearbeiterrolle mit Berechtigungen für den Dokumentvorlagen-Manager.

Schritt 3: Erstellen / Setzen der System-Properties

Setzen Sie die folgende(n) System-Property / System-Properties im Abschnitt *System-Properties*, Navigationsgruppe *System*:

- Der Pfad zum Programmverzeichnis von OpenOffice: cmweb-server-adapter, cmoffice.oo.path.<NUMMER>
  Diese Properties sind nummeriert (beginnend mit 0), sodass mehrere Pfade für verschiedene OpenOffice-Installationen auf unterschiedlichen Betriebssystemen und unterschiedliche Systemkonfigurationen abgebildet werden können. Eine mögliche Liste der Properties und Werte für die Pfadkonfiguration könnte folgendermaßen aussehen:
  - `cmoffice.oo.path.0`: C:\Programme (x86)\openoffice\program
  - `cmoffice.oo.path.1`: /usr/lib/openoffice/program
  - `cmoffice.oo.path.2`: /usr/lib64/openoffice/program

Der Umgang mit OpenOffice-Dokumenten im ConSol CM Web Client ist identisch mit dem Umgang mit Microsoft Word-Dokumenten. Die Vorbereitung einer Dokumentvorlage mit ConSol CM-Daten in OpenOffice funktioniert ebenfalls ähnlich wie in Microsoft Office. Die detaillierte Erklärung in den obigen Abschnitten (beginnend mit Erstellen und Bereitstellen von Microsoft Word-Vorlagen) gilt also

grundsätzlich auch hier. Wenn Sie ein neues Feld zu einer OpenOffice-Vorlage hinzufügen möchten, können Sie das Dialogfenster *Feldbefehle* öffnen, indem Sie das Menüelement *Einfügen -> Feldbefehl -> Andere* auswählen und dann den Tab *Variablen* auswählen.

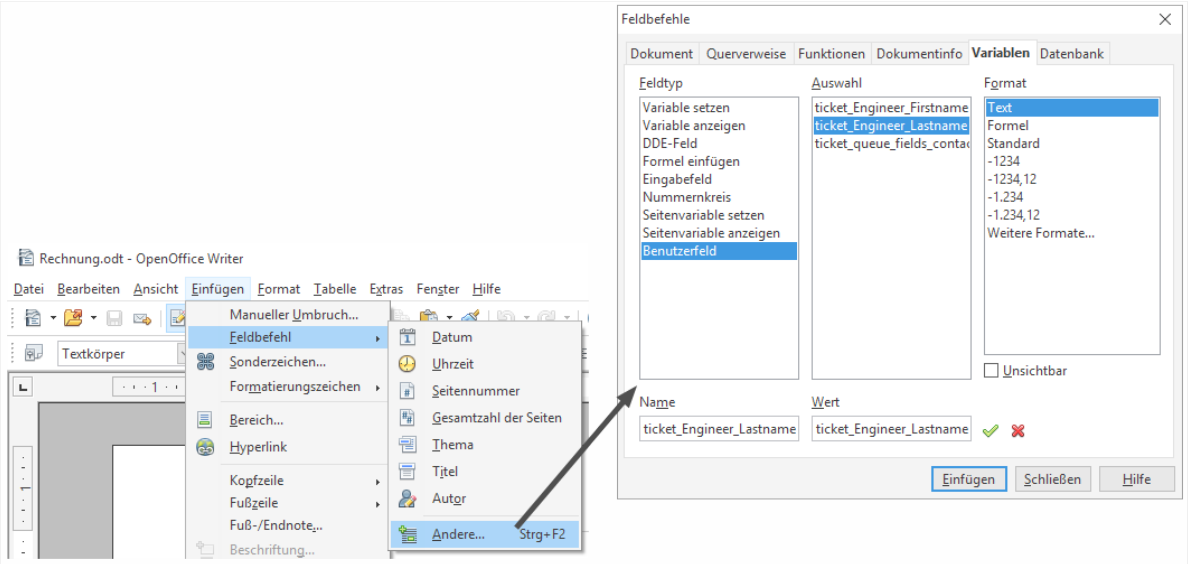


Abbildung 551: *Erstellen einer Dokumentvorlage mit OpenOffice*

In diesem Tab können Sie ein ConSol CM-Feld hinzufügen. Wählen Sie dazu den Feldtyp *Benutzerfeld* und das Format *Text* aus. Im nächsten Schritt geben Sie den Feldnamen (aus der Spalte *Schlüssel* des Abschnitts *Dokument MailMerge-Felder*) in das Feld *Name* unter dem *Feldtyp* ein. Fügen Sie einen passenden *Wert* ein, mit dem Sie das Feld im Dokument wiedererkennen können. Klicken Sie auf das grüne Häkchen rechts neben dem Feld *Wert*, damit das Feld im mittleren Bereich unter *Auswahl* angezeigt wird, und vergewissern Sie sich, dass es dort ausgewählt ist. Klicken Sie unten rechts im Dialogfenster auf *Einfügen*, um das Feld an der aktuellen Position des Cursors in das Dokument einzufügen. Dieser Vorgang muss für alle Felder, die im Dokument enthalten sein sollen, wiederholt werden.

F.12.2.9 Konfigurieren der Speicher-Strategie von CM/Doc

Bis ConSol CM-Version 6.10.5.3 wird bei jedem Speichervorgang im Word- bzw. OpenOffice-Dokument ein neues Word- bzw. OpenOffice-Attachment gespeichert und an das Ticket angehängt. Auf diese Weise erhält man ein gut gepflegtes Protokoll aller angehängten Word- bzw. OpenOffice-Dokumente. Auf der anderen Seite führt es allerdings auch zu einer großen Anzahl an Attachments, wodurch das Ticket deutlich größer wird.

Deshalb kann ab CM-Version 6.10.5.4 die Speicher-Strategie mit der CM-System-Property `cmweb-server-adapter`, `cmoffice.strict.versioning.enabled`, ein Boolean mit Standardwert „true“, konfiguriert werden. Die beiden möglichen Werte haben folgende Bedeutung:

- **true** (Standardwert, Verhalten bis 6.10.5.3)
  Bei jedem Speichervorgang im Dokument wird eine neue Version des Word- bzw. OpenOffice-Dokuments gespeichert und an das Ticket angehängt.

- **false**
  Bei einem Speichervorgang im Word- bzw. OpenOffice-Dokument wird das vorhandene Attachment mit dem gleichen Namen überschrieben, solange Word- bzw. OpenOffice zwischendurch nicht neu gestartet wurde.



Abbildung 552: *ConSol CM Web Client, an ein Ticket angehängte Dokumente, cmoffice.strict.versioning.enabled=false*

F.13 Zeitbuchung mit ConSol CM

In diesem Kapitel werden folgende Themen behandelt:

F.13.1 Allgemeine Einführung in die Zeitbuchung mit ConSol CM 860
F.13.2 Manuelle Zeitbuchungen 860
F.13.3 Automatische Zeitbuchung (verfügbar in CM-Versionen 6.9.4.2 und höher) 866
F.13.4 DWH-Reports 869
F.13.5 Seitenanpassung für Zeitbuchungen 869
F.13.6 Verwenden von Zeitbuchungsdaten in Skripten 869

F.13.1 Allgemeine Einführung in die Zeitbuchung mit ConSol CM

In ConSol CM können Arbeitsstunden auf Tickets gebucht werden. Es gibt zwei Buchungsarten:

1. Manuelle Buchungen, siehe Abschnitt Manuelle Zeitbuchungen
2. Automatische Buchungen (verfügbar in ConSol CM-Versionen 6.9.4.2 und höher), siehe Abschnitt Automatische Zeitbuchung (verfügbar in CM-Versionen 6.9.4.2 und höher)

F.13.2 Manuelle Zeitbuchungen

In ConSol CM können die Bearbeiter Arbeitsstunden auf ein Ticket buchen. Diese Arbeitsstunden können in Reports ausgewertet werden.

Bei manuellen Zeitbuchungen werden Arbeitsstunden immer auf Projekte gebucht, die einer oder mehreren Queues zugewiesen sein müssen. Wenn Ihr Unternehmen zum Beispiel eine Migration der Clients von Windows 7 auf Windows 8 plant und alle Arbeitsstunden für das Migrationsprojekt erfasst werden sollen, muss der ConSol CM-Administrator ein Migrationsprojekt erstellen und es allen Queues zuweisen, in denen Aufgaben für dieses Projekt anfallen können. Die Bearbeiter können Zeit auf das Projekt buchen und ihre eigenen Reports für das Projekt sehen. Zusätzlich kann mit dem DWH (Data Warehouse, siehe Abschnitt Verwaltung des Data Warehouse (DWH)) ein Report über alle Zeitbuchungen (aller Bearbeiter) implementiert werden.

F.13.2.1 Konfigurieren von manuellen Zeitbuchungen mit dem Admin Tool

Damit die Bearbeiter Arbeitsstunden auf Projekte buchen können, muss der ConSol CM-Administrator im Admin Tool die beiden folgenden Schritte durchführen:

1. Erstellen des Projekts im Navigationselement *Projekte* (Navigationsgruppe *Globale Konfiguration*), siehe Abschnitt Projekte.
2. Zuweisen von einem oder mehreren Projekten zu den gewünschten Queues in der Queue-Verwaltung.

Im folgenden Beispiel werden fünf Projekte erstellt. Bearbeiter sollen in der Queue *HelpDesk_1st_Level Germany* auf zwei der Projekte Zeit buchen können. Deshalb müssen die zwei Projekte der Queue *HelpDesk_1st_Level Germany* zugewiesen werden.

Sie gelangen auf diese Seite, indem Sie in der Navigationsgruppe *Globale Konfiguration* das Navigationselement *Projekte* auswählen.

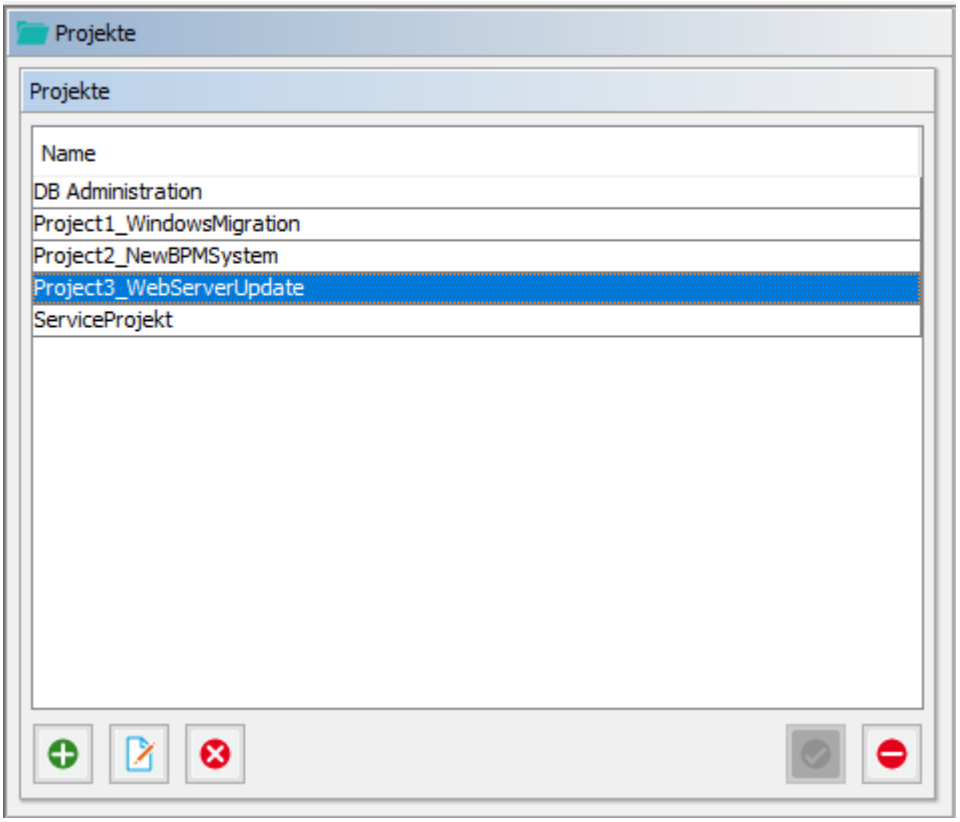


Abbildung 553: *ConSol CM Admin Tool - Globale Konfiguration, Projekte: Projektverwaltung*

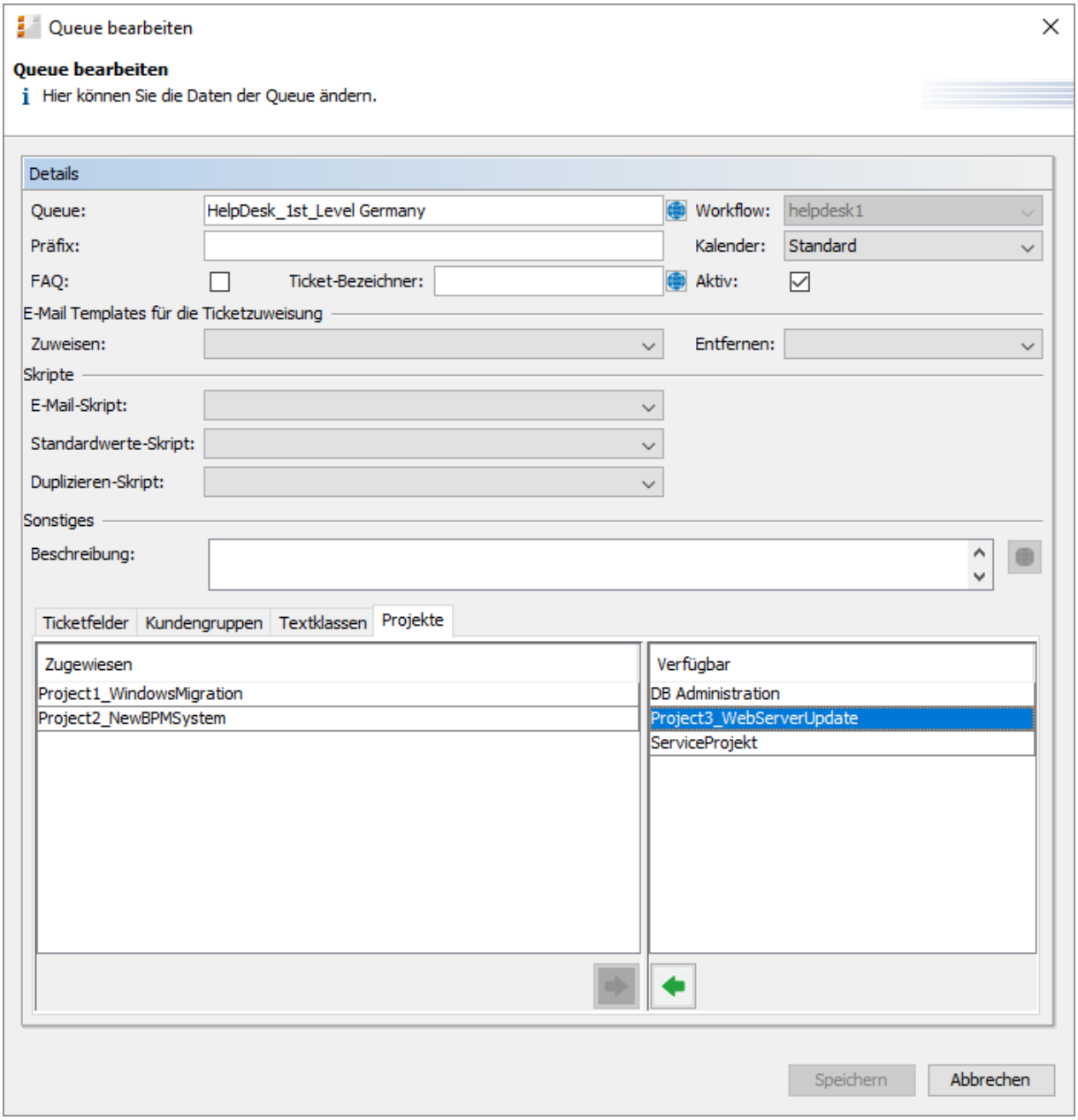


Abbildung 554: *ConSol CM Admin Tool - Queue-Verwaltung: Zuweisen von Projekten zu einer Queue*

F.13.2.2 Manuelle Zeitbuchung aus der Sicht eines Bearbeiters (Web Client)

Im *ConSol CM Benutzerhandbuch* finden Sie eine detaillierte Erklärung der Zeitbuchungsfunktion. An dieser Stelle steht nur eine kurze Übersicht.

Der Benutzer (Bearbeiter) kann auf zwei Arten Arbeitsstunden auf ein Ticket buchen:

1. Verwenden des Bereichs *Zeitbuchung* eines **Tickets**, um Arbeitsstunden direkt auf dieses Ticket zu buchen.
2. Manuelle Zeitbuchungen auf der Bearbeiterprofilseite

Verwenden des Bereichs *Zeitbuchung* eines Tickets, um Arbeitsstunden direkt auf dieses Ticket zu buchen

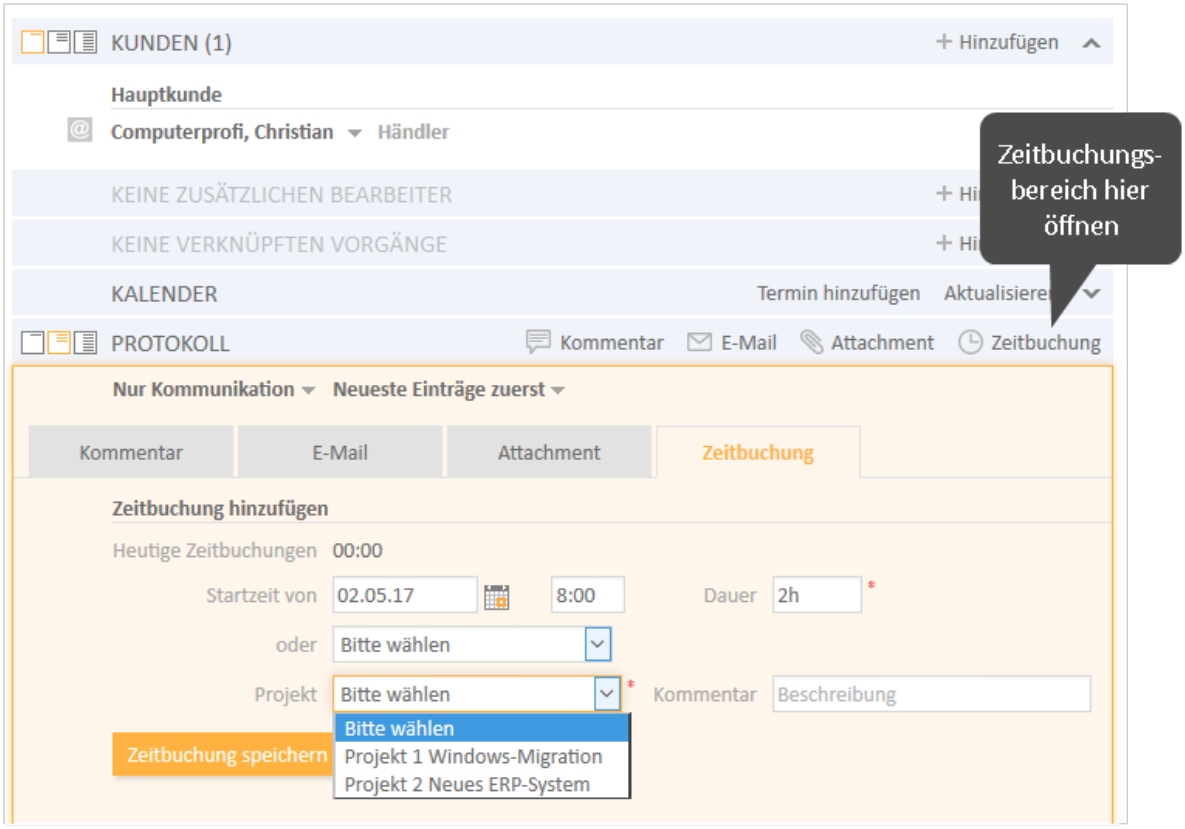


Abbildung 555: *ConSol CM Web Client - Zeitbuchung in einem Ticket*

Manuelle Zeitbuchungen auf der Bearbeiterprofilseite

Im Bereich *Zeitbuchung* des **Bearbeiterprofils** können Arbeitsstunden direkt auf ein Ticket gebucht werden. Es können nur Tickets ausgewählt werden, die dem Bearbeiter zugewiesen sind und bei denen er bestimmte Aktivitäten durchgeführt hat. Außerdem muss ein Projekt in der Liste ausgewählt werden.

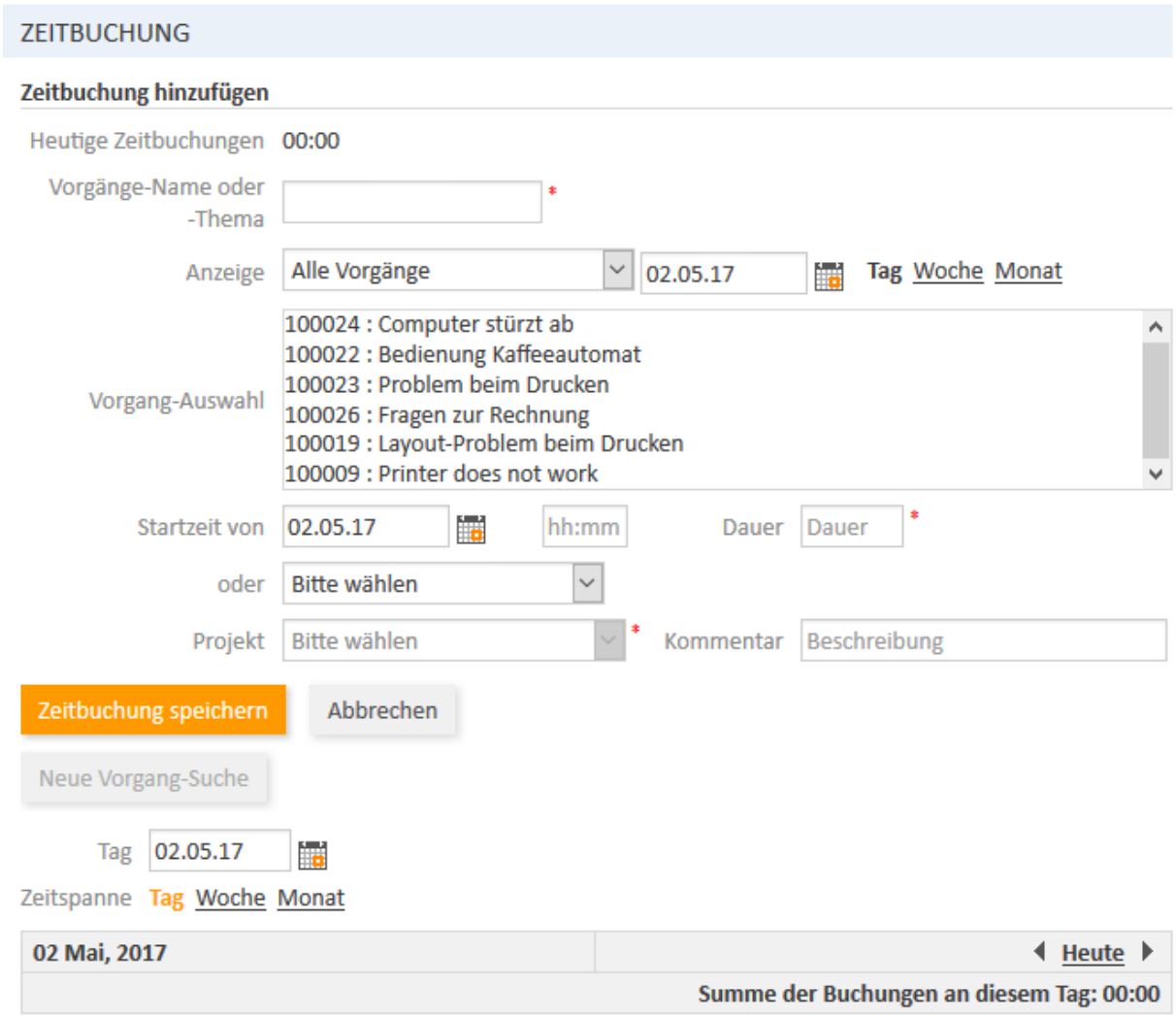


Abbildung 556: *ConSol CM Web Client - Zeitbuchung im Bearbeiterprofil*

Bearbeiter können eine Liste ihrer Zeitbuchungen auf ihren Bearbeiterprofilseiten sehen. Die folgende Abbildung zeigt ein Beispiel dafür.

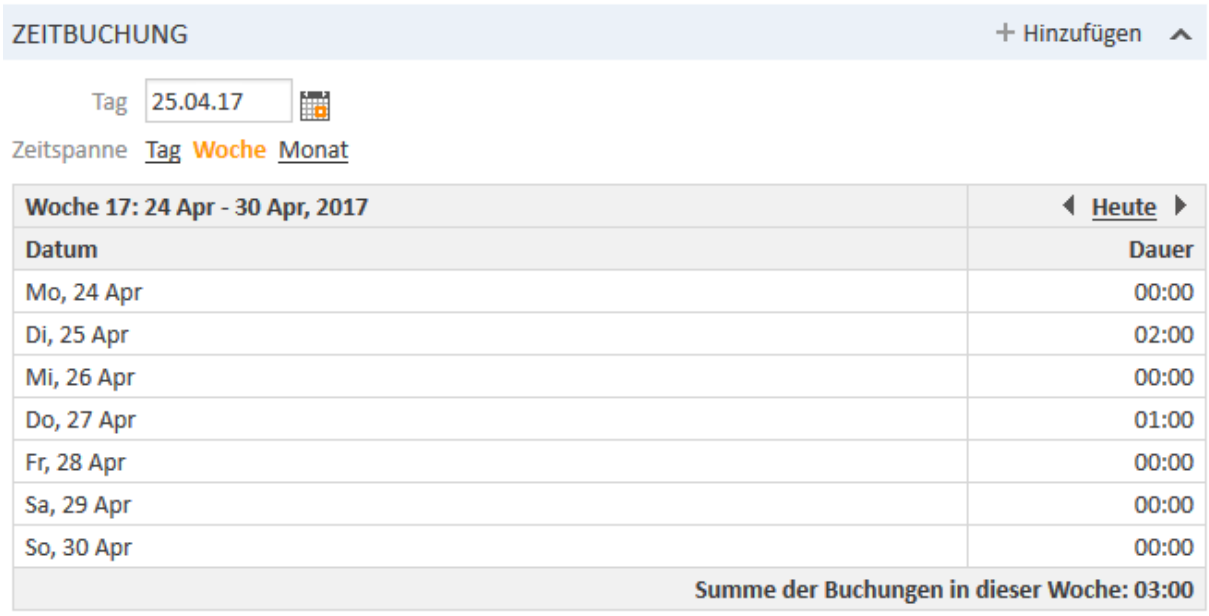


Abbildung 557: *ConSol CM Web Client - Zeitbuchungsübersicht im Bearbeiterprofil*

Als Bearbeiter können Sie auswählen, ob Sie die Buchungen für den aktuellen Tag, die aktuelle Woche oder den aktuellen Monat sehen möchten. In der Ansicht *Tag* sind die einzelnen Projekte angegeben. In den Ansichten *Woche* und *Monat* sehen Sie nur die Summe der pro Tag/Woche gebuchten Zeiten.

F.13.3 Automatische Zeitbuchung (verfügbar in CM-Versionen 6.9.4.2 und höher)

F.13.3.1 Einführung in die automatische Zeitbuchung

ConSol CM kann so konfiguriert werden, dass die Arbeitsstunden automatisch nachverfolgt und auf Tickets gebucht werden. Diese Buchungen beziehen sich immer auf Tickets und können nicht mit Projekten verknüpft werden.

Es werden folgende Zeiten erfasst:

- **Dauer der Arbeit mit dem Rich-Text-Editor**
  Wird beim Öffnen des Rich-Text-Editors gestartet und beim Klicken auf den Button *Hinzufügen* angehalten.
- **Dauer der Ticketerstellung**
  Wird beim Klicken auf *Neues Ticket* gestartet und beim Klicken auf den Button *Erstellen* angehalten.

Die Zeitbuchung wird ausgesetzt, wenn ein Ticket in den Workspace verschoben wird, und wieder aufgenommen, wenn das Ticket wieder aktiv bearbeitet wird.

In folgenden Fällen wird keine Zeit auf das Ticket gebucht ...

- der Vorgang wird abgebrochen
- der Bearbeiter meldet sich manuell ab
- die Session des Bearbeiters endet automatisch

Zeiten werden immer in Minuten gebucht und auf die nächste Minute aufgerundet.

F.13.3.2 Konfigurieren der automatischen Zeitbuchung

Um diese Funktion in Ihrem ConSol CM-System zu aktivieren, setzen Sie die System-Property cmweb-server-adapter, automatic.booking.enabled auf „true“.

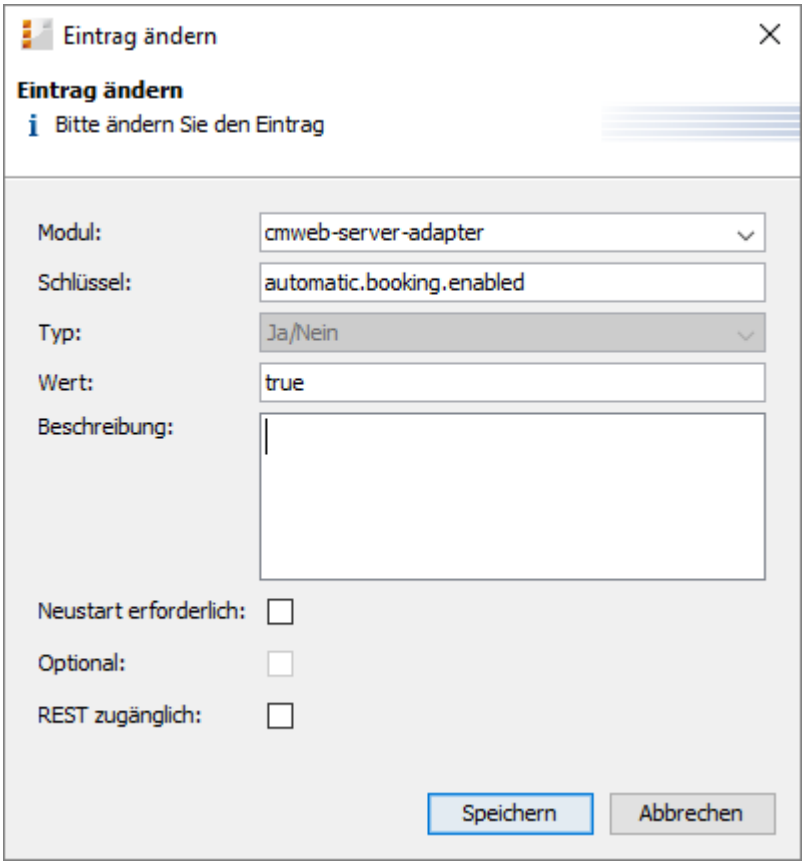


Abbildung 558: *ConSol CM Admin Tool - Setzen der System-Property zum Aktivieren der automatischen Zeitbuchung*

F.13.3.3 Automatische Zeitbuchungen aus der Sicht eines Bearbeiters (Web Client)

Der Bearbeiter muss nichts weiter tun, um mit der automatischen Zeitbuchung zu arbeiten. Wenn er einen Kommentar in ein Ticket eingibt oder ein Ticket erstellt, wird die Zeit automatisch auf dieses Ticket gebucht. Sie ist danach in der **Zeitbuchungsübersicht** auf der Bearbeiterprofilseite zu sehen. Die folgende Abbildung zeigt ein Beispiel dafür.

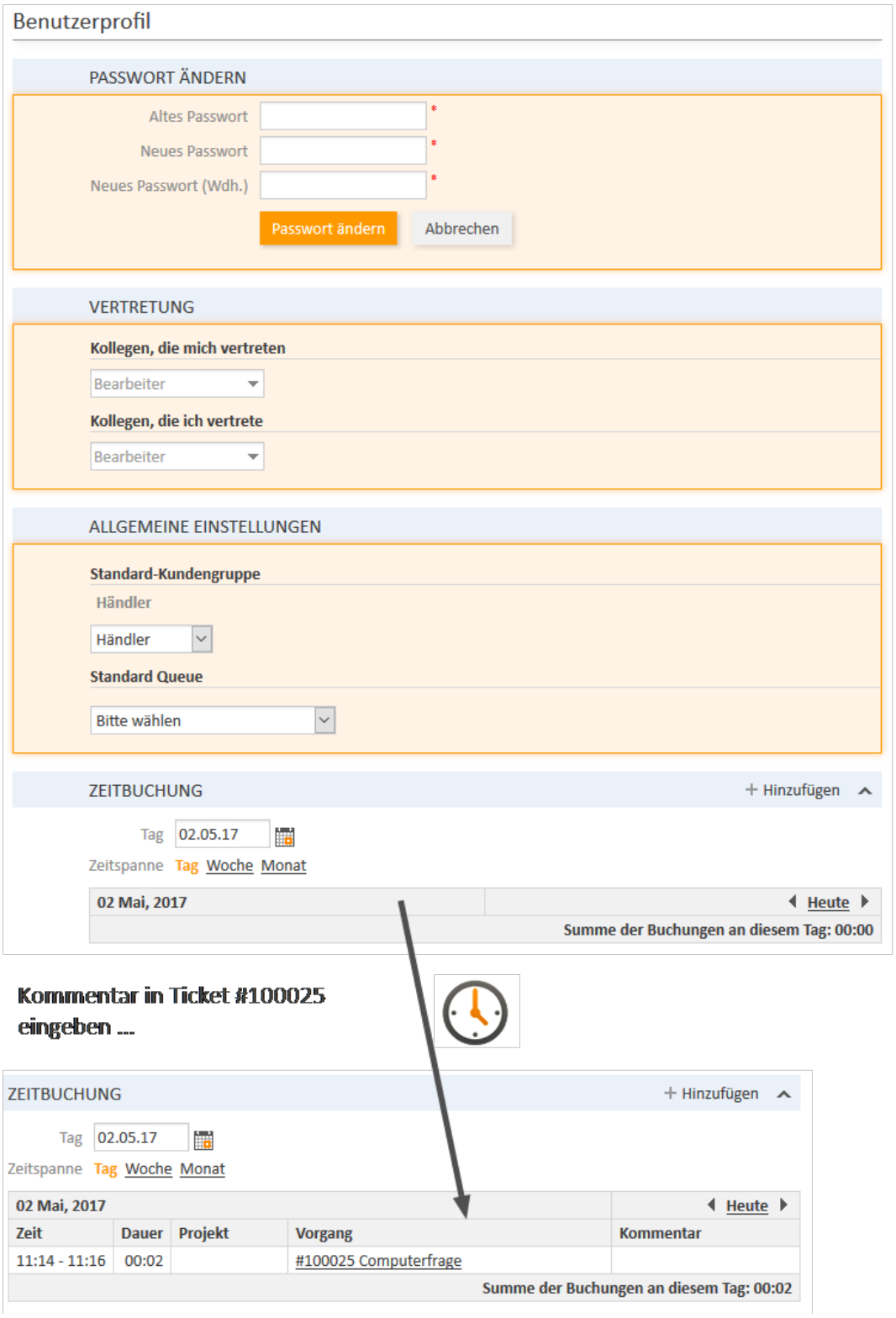


Abbildung 559: *ConSol CM Web Client - Automatisch gebuchte Zeit in der Zeitbuchungsübersicht auf der Bearbeiterprofilseite*

F.13.4 DWH-Reports

Das DWH bietet eine gute Grundlage für detailliertere Reports, wenn diese von Ihrem Unternehmen gewünscht werden. Es können Reports entwickelt werden, die anhand von DWH-Daten zum Beispiel die von allen Bearbeitern auf ein bestimmtes Projekt gebuchten Zeiten liefern.

F.13.5 Seitenanpassung für Zeitbuchungen

Falls die Zeitbuchungsfunktion nicht benötigt wird, können Sie diese Funktion in der *Seitenanpassung*, siehe Abschnitt Seitenanpassung abschalten.

In diesem Zusammenhang sind die folgenden beiden Parameter relevant:

- timeBookingSection
- timeBookingFeature

F.13.6 Verwenden von Zeitbuchungsdaten in Skripten

Ab ConSol CM-Version 6.11 enthält die CM-API eine Methode zum Abrufen der auf ein Ticket gebuchten Zeiten. Verwenden Sie dazu folgende Methode des Objekts `workflowApi`.

```
Set<TimeBooking> getBookings(Ticket pTicket)
```

Sie können diese Methode in Admin-Tool-Skripten und Workflow-Skripten verwenden. Beachten Sie, dass der Workflow-Kontext in Admin-Tool-Skripten möglicherweise nicht verfügbar ist.

Das folgende Beispiel zeigt ein Skript, das aus einer Workflow-Aktivität aufgerufen werden kann. Sie können den Code direkt im Skript schreiben oder in ein Admin-Tool-Skript einfügen, das aus dem Workflow aufgerufen wird. Details über dieses Prinzip finden Sie im *ConSol CM Process Designer Handbuch*.

F.13.6.1 Beispiel für ein Zeitbuchungsskript

Das Skript:

```
import com.consol.cmas.common.model.ticket.Ticket
import com.consol.cmas.common.model.time.TimeBooking

log.info 'Running script printTimeBookingsToTicket ...'

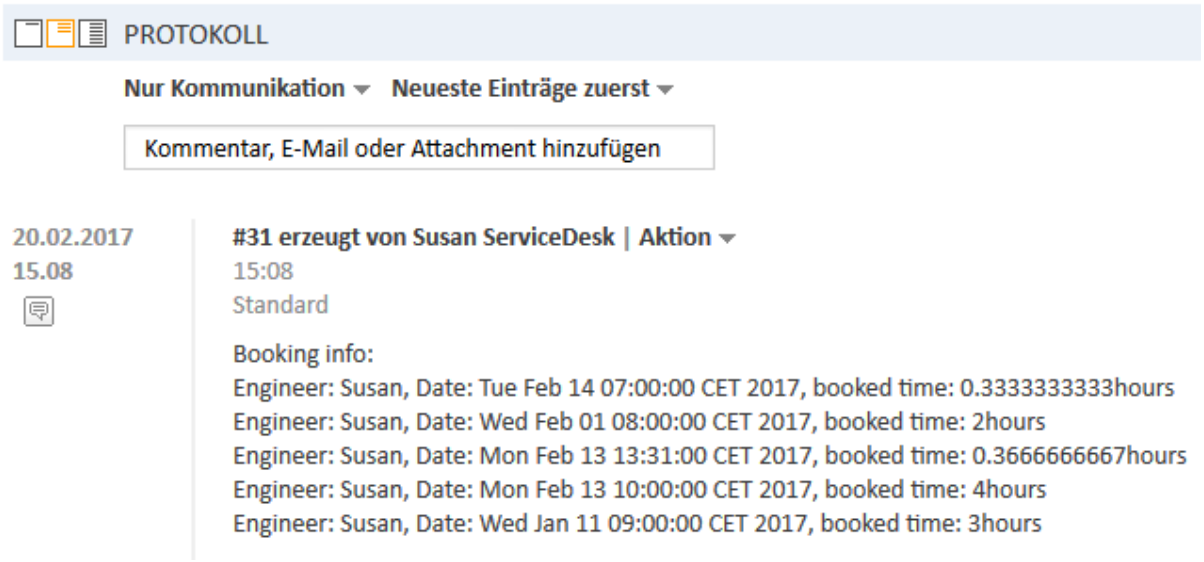
def ticket = workflowApi.ticket
def bookings = workflowApi.getBookings(ticket)
def eng
def per
def per_date
def per_time
def text = 'Booking info: \n'

if (bookings.size() > 0){
   log.info 'Time bookings found in ticket ' + ticket.name
   bookings.each() { bk ->
      eng = bk.engineer.name
      per = bk.timePeriod
      per_date = per.bookingDate
      per_time = per.bookedTime /1000 / 60 / 60
      def text1 = 'Engineer: ' + eng + ', Date: ' + per_date + ', booked time: ' +
       per_time + 'hours'
      text += text1 + '\n'
   }
   workflowApi.addTicketText(text, 'Time Booking report for this ticket', false)
} else {
   workflowApi.addValidationError('INFO', 'No booking on this ticket.')
}

return
```

Code-Beispiel 98: *Aus dem Workflow aufgerufenes Admin-Tool-Skript: alle auf das Ticket gebuchten Zeiten werden als Kommentar ins Ticketprotokoll geschrieben*

Die Ausgabe im entsprechenden Ticket:



F.14 Authentifizierungsmethoden in ConSol CM

Für die Authentifizierung der Benutzer in ConSol CM können Sie folgende drei Methoden konfigurieren:

1. **Standard / Datenbank**
   Alle Benutzerdaten werden in der ConSol CM-Datenbank gespeichert. Diese Authentifizierungsmethode ist für Bearbeiter (siehe Datenbank-Authentifizierung für Bearbeiter) und Kunden (siehe Datenbank-Authentifizierung für Kunden) verfügbar.
2. **LDAP**
   Die Anmeldeinformationen werden auf einem LDAP-Server gespeichert. ConSol CM sendet zur Authentifizierung der Bearbeiter eine Anfrage an diesen Server. Diese Authentifizierungsmethode ist für Bearbeiter (siehe LDAP-Authentifizierung für Bearbeiter im Web Client) und Kunden (siehe LDAP-Authentifizierung für Kunden in CM/Track) verfügbar.
3. **Kerberos SSO**
   Die Anmeldeinformationen werden mittels Kerberos aus einer gültigen Session abgerufen, siehe Single Sign-On für ConSol CM mittels Kerberos (in einer Windows-Domäne). Diese Authentifizierungsmethode ist nur für Bearbeiter im Web Client verfügbar.

F.14.1 Authentifizierungsmethoden für Bearbeiter im Web Client

Für die Authentifizierung der Bearbeiter im Web Client können Sie folgende drei Methoden konfigurieren:

- **Datenbank-Authentifizierung**
  Benutzername und Passwort werden in der CM-Datenbank gespeichert, siehe Datenbank-Authentifizierung für Bearbeiter.
- **LDAP-Authentifizierung**
  Der Benutzername wird in der CM-Datenbank, in der Bearbeiterverwaltung, gespeichert. Das Passwort wird auf einem LDAP-Server gespeichert, siehe LDAP-Authentifizierung für Bearbeiter im Web Client.
- **Kerberos-Authentifizierung**
  Der Benutzername wird in der CM-Datenbank, in der Bearbeiterverwaltung, gespeichert. Das Passwort wird in Kerberos gespeichert, siehe Single Sign-On für ConSol CM mittels Kerberos (in einer Windows-Domäne).

F.14.1.1 Datenbank-Authentifizierung für Bearbeiter

Setzen von Benutzernamen und Passwort

Der Benutzername und das Passwort werden in den Bearbeiterdaten (Navigationsgruppe *Zugriff und Rollen*, Navigationselement *Bearbeiter*, siehe Bearbeiterverwaltung) gesetzt. Die folgenden Felder sind für die Datenbank-Authentifizierung relevant:

- **Login:**
  Pflichtfeld. Dieses Feld enthält den Namen des Kontos, der auf der Login-Seite des Web Clients eingegeben wird. Verwenden Sie hier nur internationale alphanumerische Zeichen, keine Leerzeichen, Satzzeichen oder Sonderzeichen wie Umlaute, Bindestriche oder Ähnliches.
- **E-Mail:**
  Pflichtfeld. Die E-Mail-Adresse des Bearbeiters. Verwenden Sie bitte nur internationale alphanumerische Zeichen, Bindestriche, Unterstriche, Punkte und das @-Zeichen. Die Eingabe von mehreren E-Mail-Adressen in einer Zeile ist nicht zulässig.
- **Passwort:**
  Pflichtfeld. Das Passwort des Bearbeiters ist obligatorisch. Verwenden Sie hier nur internationale alphanumerische Zeichen und Satzzeichen. Verwenden Sie **keine** Sonderzeichen wie Umlaute. Das eingegebene Passwort wird als Reihe mit Punkten angezeigt. Informationen über eine optionale Passwort-Richtlinie finden Sie in Abschnitt Konfiguration der Passwort-Richtlinie.

Konfiguration der Passwort-Richtlinie

Diese Konfiguration ist optional.

Die folgenden System-Properties können verwendet werden, um eine bestimmte Passwort-Richtlinie zu implementieren. Alle folgenden System-Properties befinden sich im Modul `cmas-core-security`. (Details über die System-Properties finden Sie im Abschnitt System-Properties.

- policy.password.pattern (String)
  RegEx-Ausdruck für das Passwort, Beispiel und Standardwert: „^.3,$.“

- policy.password.age (Integer)
  Maximaler Gültigkeitszeitraum, in Tagen, Beispiel „183“ (6 Monate), Standardwert: „5500“ (= 15 Jahre, d. h. es wird keine Passwortänderung erzwungen). Falls Sie möchten, dass der Bearbeiter sein Passwort sofort ändert, verwenden Sie einen der folgenden Werte:
  - **0**
    Der Bearbeiter muss sein Passwort bei der nächsten Anmeldung ändern.
  - **1**
    Der Bearbeiter muss sein Passwort am nächsten Tag ändern.
- policy.rotation.ratio (Integer)
  Zahl, definiert die Anzahl der vorherigen Passwörter, die nicht identisch sein dürfen, Beispiel und Standardwert: „1“.
- policy.username.case.sensitive (Boolean)
  Legt fest, ob Groß- und Kleinschreibung bei der Validierung berücksichtigt wird. Beispiel und Standardwert: „true“. Beachten Sie, dass diese Einstellung von der Collation in MySQL beeinflusst wird. Damit das CM-System korrekt mit MySQL arbeitet, muss in MySQL die richtige Collation verwendet werden.

Zurücksetzen des Passworts eines Bearbeiters

Wenn ein Bearbeiter sein Passwort vergessen hat, kann er über den Link *Passwort vergessen?* auf der Login-Seite ein neues Passwort anfordern. Der Bearbeiter erhält eine E-Mail mit einem Link zu einer URL, über die er ein neues Passwort setzen kann.

Beachten Sie, dass dieser Vorgang nur funktioniert, wenn der Bearbeiter ein gültiges E-Mail-Konto hat und der entsprechende Wert in den Bearbeiterdaten im Feld *E-Mail* eingetragen ist!

Die E-Mail, die an den Bearbeiter gesendet wird, basiert auf dem Template `password-reset-template`, das im Abschnitt *Templates* des Admin Tools gespeichert ist. Eine detaillierte Beschreibung der Templates im Allgemeinen finden Sie im Abschnitt Admin-Tool-Templates. Details über das Zurücksetzen von Bearbeiter-Passwörtern sind im Abschnitt Zurücksetzen des Passworts für Bearbeiter im Web Client beschrieben.

Das Zurücksetzen von Passwörtern im Web Client ist nur möglich, wenn der Standardmodus für die Authentifizierung genutzt wird. Bei LDAP- oder Kerberos-Authentifizierung ist dies nicht möglich. Eine Beschreibung aller möglichen Authentifizierungsmethoden finden Sie im Abschnitt Authentifizierungsmethoden in ConSol CM.

F.14.1.2 LDAP-Authentifizierung für Bearbeiter im Web Client

Einführung in die LDAP-Authentifizierung in ConSol CM

ConSol CM bietet LDAP-Authentifizierung für den Web Client als Standardfunktion an, d. h. die Passwörter der ConSol CM-Bearbeiter werden nicht in der ConSol CM-Datenbank verwaltet, sondern von einem LDAP-Server (wie z. B. ein ***Microsoft Active Directory***-Server) abgerufen.

Wenn sich die Bearbeiter im ConSol CM Web Client anmelden möchten, geben sie ihren Benutzernamen und ihr Passwort ein und drücken die Eingabetaste. Im Hintergrund sendet der ConSol CM-Server eine Anfrage mit dem Benutzernamen und dem Passwort an den LDAP-Server und dieser überprüft, ob die Anmeldeinformationen gültig sind.

Wenn die Anmeldeinformationen gültig sind, wird die Genehmigung zurück an den ConSol CM-Server gesendet und der Bearbeiter wird im Web Client angemeldet.

> Beachten Sie, dass die LDAP-Verbindung nur zur Authentifizierung des Benutzers (Bestätigung seiner Identität) verwendet wird. Die Autorisierung (d. h. die Zuweisung von Zugangsberechtigungen im System) erfolgt im Admin Tool über die Verwaltung der Bearbeiter und Rollen. Es muss für jeden Benutzer, der als Bearbeiter im System arbeiten soll, ein Bearbeiterkonto in der Bearbeiterverwaltung angelegt werden!

Der CM-Authentifizierungsprozess mittels LDAP ist in der folgenden Abbildung erklärt.

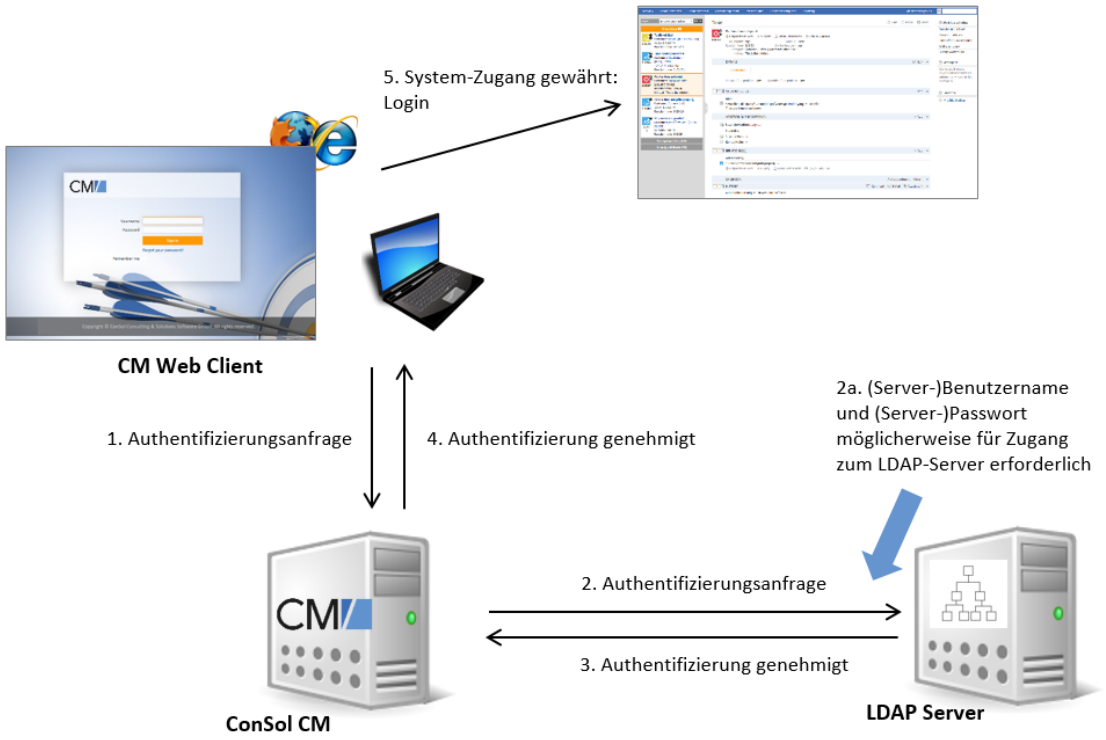


Abbildung 560: *ConSol CM - LDAP-Authentifizierungsprozess*

Konfigurieren von LDAP-Authentifizierung

Es gibt zwei Möglichkeiten, um das ConSol CM-System für die Verwendung der LDAP-Authentifizierung für Bearbeiter im Web Client zu konfigurieren:

- Wählen Sie bei der Systemeinrichtung LDAP als Authentifizierungsmethode aus und geben Sie nach der Einrichtung die erforderlichen Parameter (System-Properties) ein.
- Richten Sie das System mit der Standardauthentifizierungsmethode ein und wechseln Sie danach zu LDAP, d. h. geben Sie alle erforderlichen System-Properties später ein.

Konfigurieren von LDAP bei der Ersteinrichtung

Bei der Systemeinrichtung können Sie *LDAP* als Authentifizierungsmethode auswählen. Dadurch wird die System-Property cmas-core-security, authentication.method „LDAP“ gesetzt. Es werden keine weiteren Parameter eingegeben. Sie müssen die LDAP-Parameter manuell setzen. Dies wird im nächsten Abschnitt erklärt.

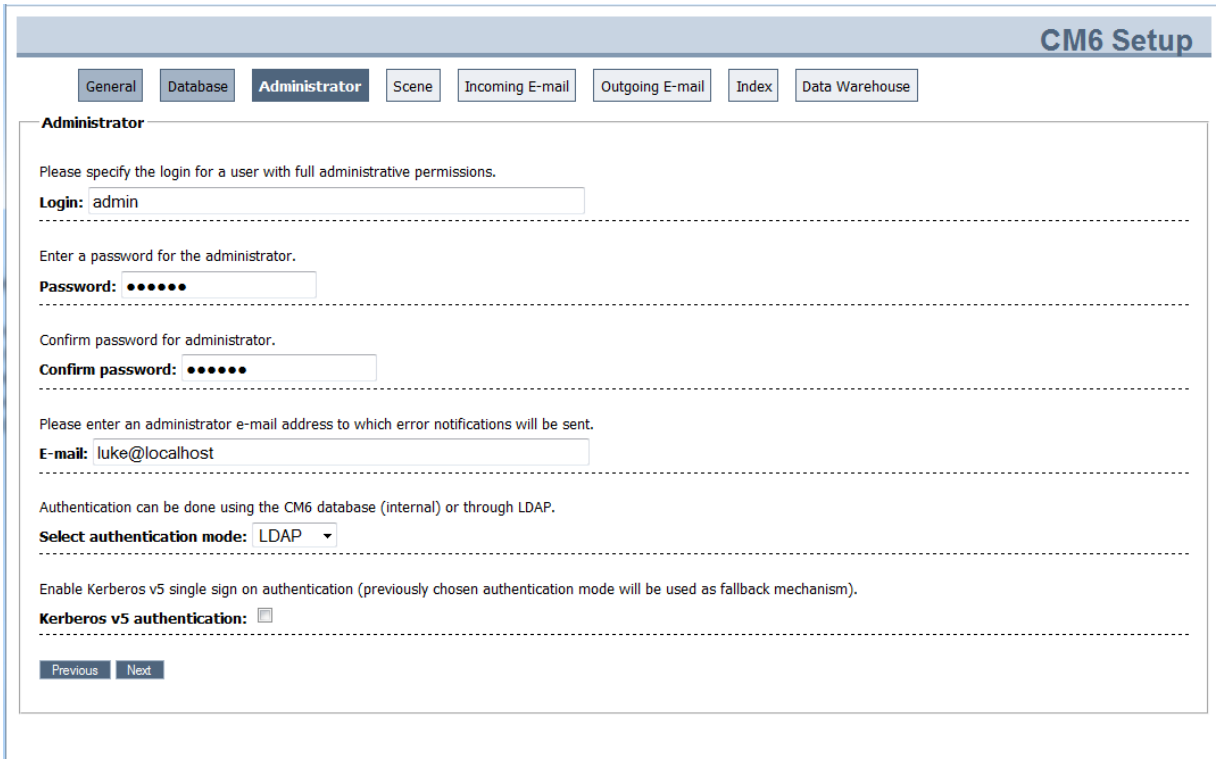


Abbildung 561: *ConSol CM-Systemeinrichtung - Authentifizierungsmethode LDAP*

Wechseln zu LDAP als Authentifizierungsmethode in einem laufenden System

Um die Authentifizierungsmethode in LDAP zu ändern, müssen Sie die erforderlichen Werte in den System-Properties setzen (Navigationsgruppe *System*, Navigationselement *System-Properties*, siehe System-Properties):

- authentication.method
  LDAP
- ldap.authentication
  simple
- ldap.basedn
  Der DN (Distinguished Name) der LDAP-Baumstruktur, in der sich die erforderlichen Attribute befinden.

- ldap.initialcontextfactory
  Der Name der Java-Klasse für die Initial Context Factory der LDAP-Implementierung bei der Verwendung von LDAP-Authentifizierung. Dies sollte normalerweise `com.sun.jndi.ldap.LdapCtxFactory` sein.
- ldap.password
  Das Passwort für die Verbindung zum LDAP-Server zum Nachschlagen der Benutzer. Wird nur benötigt, wenn die Suche nicht anonym ausgeführt werden kann.
- ldap.userdn
  Der LDAP-Benutzer für die Verbindung zum LDAP-Server zum Nachschlagen der Benutzer. Wird nur benötigt, wenn die Suche nicht anonym ausgeführt werden kann.

  ⚠ Möglicherweise sind die Angaben Benutzername/Passwort des Servers erforderlich, um auf den LDAP-Server zuzugreifen. Wenn Sie sich nicht sicher sind, können Sie dies zuerst mit einem LDAP-Browser überprüfen.

- ldap.providerurl
  Die komplette URL des LDAP-Servers:

  ```
  ldap://<HOSTNAME>:<LDAP PORT>
  ```

- ldap.searchattr
  Suchattribut zum Nachschlagen eines LDAP-Eintrags für ein CM-Login, d. h. das Attribut, das bei der Authentifizierung als Benutzername verwendet wird.

Verwendung von LDAPS (LDAP über SSL)

Einleitung

Standardmäßig werden die Daten im Klartext übertragen, wenn ein LDAP-Client auf einen LDAP-Server zugreift. Wenn Sie möchten, dass der Benutzername und das Passwort verschlüsselt an den LDAP-Server übertragen werden, müssen Sie die LDAP-Authentifizierung mit LDAPS aufsetzen.

Vorbereitungen

Sie müssen den Rechner des CM-Servers (Java) so konfigurieren, dass er Zertifikate verwenden kann. Eine Methode dafür ist im folgenden Abschnitt für einen Linux-Rechner beschrieben.

1. Rufen Sie das Zertifikat ab:
   `openssl s_client -connect dc2.mydomain.com:ldaps`
2. Die Antwort enthält einen Abschnitt, der mit "---BEGIN CERTIFICATE " beginnt und mit "END CERTIFICATE ---" endet.
   Kopieren Sie diesen Abschnitt in eine Datei, z. B. /tmp/certificate2_dc2_mydomain_com.txt
3. Importieren Sie das Zertifikat in den Truststore des Rechners, z. B. /home/mydirectory/mytruststore
   `$JAVA_HOME/bin/keytool -import -alias <arbitrary> -`

`trustcacerts -keystore /home/mydirectory/mytruststore -file`/tmp/certificate2_dc2_mydomain_com.txt
Sie müssen ein Passwort eingeben (setzen).

4. Fügen Sie den Truststore in die ConSol CM-Konfigurationsdatei unter JAVA_OPTS ein:
`-Djavax.net.ssl.trustStore=/home/mydirectory/mytruststore -Djavax.net.ssl.trustStorePassword=<see above>`

LDAPS-Konfiguration mit dem ConSol CM Admin Tool (System-Properties)

Konfigurieren Sie den ConSol CM-Server so, wie im folgenden Beispiel gezeigt:

- cmas-core-security, ldap.authentication = simple
- cmas-core-security, ldap.basedn = OU=myOU,DC=myDC
- cmas-core-security, ldap.initialcontextfactory = com.sun.jndi.ldap.LdapCtxFactory
- cmas-core-security, ldap.password = myLDAPpw
- cmas-core-security, ldap.searchattr = sAMAccountName
- cmas-core-security, ldap.userdn = myLDAP_UserDN

Je nach Konfiguration des LDAP-Servers, verwenden Sie einen der folgenden Werte für die Server-URL:

- **Standard LDAPs port**
cmas-core-security, ldap.providerurl = ldaps://dc2.mydomain.com:636
- **LDAPs port Global Catalogue**
cmas-core-security, ldap.providerurl = ldaps://dc2.mydomain.com:3269

Konfigurieren von Bearbeiterkonten für LDAP-Authentifizierung

Verwenden Sie die Bearbeiterverwaltung (Navigationsgruppe *Zugriff und Rollen*, Navigationselement *Bearbeiter*) im Admin Tool, um die Bearbeiterkonten zu konfigurieren.

Wenn LDAP als Authentifizierungsmethode verwendet wird, ist es nicht möglich, das ConSol CM-Password innerhalb der Bearbeiterverwaltung zu setzen. Das Popup-Fenster für die Bearbeiterverwaltung enthält folgende Felder, die für die LDAP-Authentifizierung relevant sind. Details zu den anderen (nicht für LDAP relevanten) Datenfeldern finden Sie in Bearbeiterverwaltung.

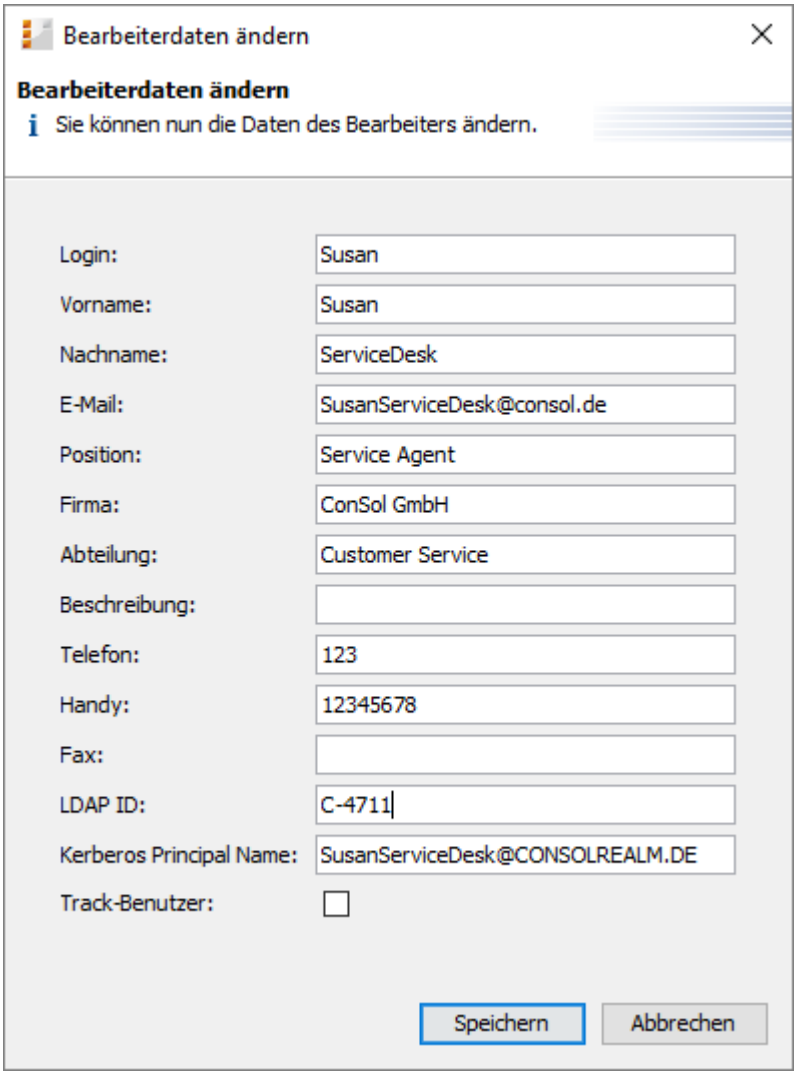


Abbildung 562: *ConSol CM Admin Tool - Zugriff und Rollen: Editieren eines Bearbeiters*

- **Login:**
  Wenn keine LDAP-ID angegeben wurde, wird dieses Feld beim LDAP-Authentifizierungsprozess als Benutzername, der im LDAP-Verzeichnis im Knoten `ldap.searchattr` gesucht wird, verwendet.
- **LDAP ID:**
  Füllen Sie dieses Feld aus, wenn Sie in ConSol CM spezielle Benutzernamen verwenden möchten, die nicht mit den im LDAP-Verzeichnis gespeicherten Werten übereinstimmen. Beim LDAP-Authentifizierungsprozess wird die LDAP-ID als Benutzername, der im LDAP-Verzeichnis im Knoten `ldap.searchattr` gesucht wird, verwendet.

F.14.1.3 Single Sign-On für ConSol CM mittels Kerberos (in einer Windows-Domäne)

Kurze Einführung in Kerberos

Kerberos ist ein Netzwerkprotokoll, das zur Authentifizierung eines Benutzers oder eines Systems verwendet wird, um zu überprüfen, ob der Benutzer bzw. das System die Identität hat, die es vorgibt zu haben. Der Authentifizierungsmechanismus basiert auf sogenannten *Tickets*. Wenn ein Benutzer oder System von einem Kerberos-System authentifiziert wurde, können andere Systeme, z. B. der ConSol CM-Server, dieser Authentifizierung vertrauen und dem Benutzer bzw. System Zugriff auf die eigenen Ressourcen gewähren. Einer der Vorteile der Verwendung von Kerberos ist, dass keine Passwörter (verschlüsselt oder im Klartext) über das Netzwerk gesendet werden. Es werden nur Informationen, die mit dem Passwort verschlüsselt wurden, gesendet. Sobald er authentifiziert ist, hat ein Client Zugriff auf alle Ressourcen innerhalb der Kerberos-Domäne (Realm), ohne sich erneut anmelden zu müssen (Single Sign-On, SSO).

Die aktuelle Version von Kerberos ist V5, sie wurde zuerst 1993 entwickelt. Wenn Sie sich genauer in das Thema einarbeiten möchten, können Sie RFC 1510 (initialer RFC) oder RFC 4120 (V5) lesen.

Folgende Komponenten sind in einer Kerberos-Infrastruktur wichtig:

Das **Key Distribution Center**, das zwei aktive Komponenten enthält:

- **Authentication Service**
  Führt die Authentifizierung durch (z. B. mittels Microsoft Active Directory wie in unserem Beispiel) und erstellt ein *Ticket Granting Ticket (TGT)*, sofern der Benutzer oder das System erfolgreich authentifiziert wurde.
- **Ticket Granting Service**
  Erstellt Tickets, die Zugriff auf andere Systeme gewähren. Der Benutzer oder das System, das ein *Service Ticket* erhalten möchte, muss ein TGT vorlegen, um dieses *Service Ticket* zu bekommen.

Kurze Einführung in ConSol CM mit Kerberos in einer Windows-Domäne

Die **Single Sign-On-Funktion** ermöglicht es, die Benutzer automatisch in ConSol CM zu authentifizieren, z. B. mit ihren Windows-Anmeldeinformationen.

Diese Authentifizierungsmethode ...

- funktioniert vollständig transparent. Es ist keine Benutzerinteraktion (d. h. Ausfüllen der Anmeldedaten auf der Anmeldeseite) erforderlich.
- beeinträchtigt vorhandene Authentifizierungsmethoden nicht. Wenn die Kerberos-Authentifizierung fehlschlägt, wird die konfigurierte Authentifizierungsmethode verwendet (z. B. LDAP- oder Datenbank-Authentifizierung, siehe CM-System-Property cmas-core-security, authentication.method).

Die Single-Sign-On-Funktion basiert auf dem Protokoll ***Kerberos V5***, das in ***Windows Active Directory*** integriert ist.

Der ConSol CM-Server fungiert als *Nicht-Windows-Kerberos-Dienst* und kann auf einem beliebigen Betriebssystem / Application Server installiert werden.

Der ConSol CM-Server ist Teil einer Windows-Domäne, in der das Key Distribution Center der Domänen-Controller ist, und in der Active Directory ausgeführt wird, um die Informationen zur Benutzerauthentifizierung zu speichern.

Die folgende Abbildung zeigt eine (vereinfachte) Übersicht über die Schritte, die erforderlich sind, um eine gültige Client/Server-Session für einen ConSol CM-Client und den entsprechenden ConSol CM-Server bereitzustellen. Die Erklärungen in diesen Abschnitten führen Sie durch den gesamten Konfigurationsprozess, der notwendig ist, um ConSol CM mit Kerberos-Authentifizierung zu betreiben.

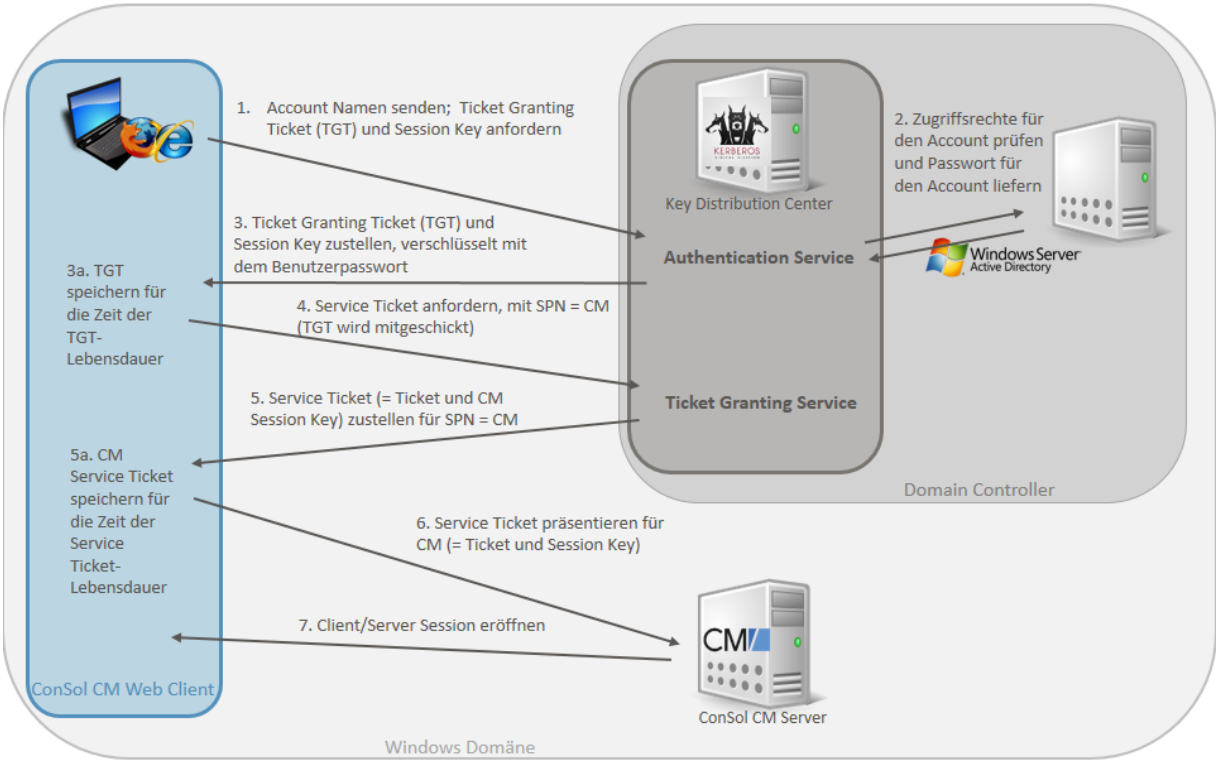


Abbildung 563: *Kerberos-Authentifizierung in einer Windows-Domäne für den Zugang von ConSol CM*

Schritte des Authentifizierungsprozesses

Schritt 1: Senden des Kontonamens, Anfordern eines Ticket Granting Tickets (TGT) und Sitzungsschlüssels

Der Benutzer meldet sich an seinem PC/Laptop mit dem Benutzernamen und Passwort an. Der PC/Laptop sendet eine Nachricht an den Authentifizierungsserver und fordert ein Ticket Granting Ticket (TGT) an.

Sofern die Präauthentifizierung aktiviert wurde, wird ein Zeitstempel mit dem Hash des Benutzerpassworts als Verschlüsselungsschlüssel verschlüsselt. Wenn der Authentifizierungsserver eine gültige Zeit erkennt, wenn er den Zeitstempel mit dem (in Active Directory gespeicherten) Hash des Benutzerpassworts entschlüsselt, weiß er, dass die Anforderung keine Erneuerung einer vorherigen Anforderung ist.

In diesem Beispiel ist die Präauthentifizierung deaktiviert.

Schritt 2: Überprüfen der Rechte für das Konto, Angeben des Passworts für das Konto

Der Autorisierungsserver überprüft die Zugriffsrechte des Benutzers in Active Directory und AD stellt das Passwort des Benutzers für Schritt 3 zur Verfügung.

Schritt 3: Zustellen eines Ticket Granting Tickets (TGT) und Sitzungsschlüssels (Nachricht mit dem Client-Passwort verschlüsselt)

Der Authentication Service erstellt ein Ticket Granting Ticket (TGT) und einen Sitzungsschlüssel. Diese werden in einer Nachricht an den Client gesendet und sind mit einem aus dem Benutzerpasswort abgeleiteten Schlüssel verschlüsselt.

Der PC/Laptop fragt den Benutzer nach einem Passwort und verwendet dieses Passwort, um die eingehende Nachricht zu entschlüsseln. Wenn die Entschlüsselung erfolgreich ist, kann der Benutzer mit dem TGT ein Service Ticket anfordern.

Schritt 3a: Das TGT wird während seiner Gültigkeitsdauer für den Benutzer gespeichert und kann während dieses Zeitraums für alle Anforderungen von Service Tickets verwendet werden.

Schritt 4: Anfordern eines Service Tickets für CM

Wenn ein Benutzer Zugang zu einem Dienst haben möchte, d. h. wenn sich ein Bearbeiter in ConSol CM anmelden möchte, sendet die Client-Applikation seiner Workstation (ConSol CM Web Client) eine Anforderung an den Ticket Granting Service, die den SPN (Service Principal Name, Dienstprinzipalname) des angefragten Dienstes, den Client-Namen, den Realm-/Domänennamen und einen Zeitstempel (mit dem Sitzungsschlüssel verschlüsselter Name und Zeitstempel) enthält. Der Benutzer weist sich durch Senden des in Schritt 3 erhaltenen Authentikators (TGT) aus.

Schritt 5: Zustellen eines Service Tickets für CM (= Ticket und Sitzungsschlüssel)

Der Ticket Granting Service entschlüsselt das Ticket und den Authentifikator, überprüft die Anforderung und erstellt ein Service Ticket für den angefragten Dienst (SPN = ConSol CM). Das Service Ticket enthält den Namen des Benutzers (Bearbeiters). Es enthält auch den Realm-/Domänennamen und die Gültigkeitsdauer des Service Tickets. Das Service Ticket wird mit dem geheimen Schlüssel des Dienstes (CM) verschlüsselt. Der Ticket Granting Service sendet das Service Ticket an den Benutzer.

Schritt 5a: Das Service Ticket wird während seiner Gültigkeitsdauer für den Benutzer gespeichert und kann während dieses Zeitraums für alle Anforderungen an den SPN verwendet werden.

Schritt 6: Vorlegen eines Service Tickets für CM (= Ticket und Sitzungsschlüssel)

Die Client-Applikation (CM Web Client) sendet jetzt eine Dienstanforderung, die das in Schritt 5 erhaltenen Ticket und einen Authentifikator enthält, an den CM-Server. Der Dienst (CM-Server) authen-

tifiziert die Anforderung durch Entschlüsseln des Sitzungsschlüssels. Der CM-Server überprüft, dass das Service Ticket und der Authentifikator übereinstimmen und gewährt dann Zugang zum Dienst (ConSol CM-Applikation).

Schritt 7: Öffnen der Client/Server-Session

Die Client/Server-Session läuft.

Verschlüsselung im Prozess

Normalerweise wird Kerberos mit symmetrischer Verschlüsselung ausgeführt (d. h. derselbe Schlüssel wird zum Verschlüsseln und zum Entschlüsseln der Nachricht verwendet). In Kerberos basiert die Verschlüsselung auf dem Passwort des Benutzers. Optional kann in einer Kerberos-Umgebung Kryptographie mit öffentlichen Schlüsseln eingesetzt werden.

Microsoft Windows verwendet Kerberos als Standardauthentifizierungsmethode mit RC4 für die Verschlüsselung und HMAC (Hash-based Message Authentication Code).

Kerberos-Terminologie

Kerberos-Prinzipal

Ein *Kerberos-Prinzipal* ist eine eindeutige Identität, der Kerberos Tickets zuweisen kann. Dies kann ein Benutzerkonto einer Person oder eines Systems sein. Kerberos-Prinzipale bestehen aus mehreren Komponenten. Die Komponenten sind durch ein Trennzeichen voneinander getrennt, normalerweise "/". Die letzte Komponente ist der Realm/Domäne, der vom Rest des Prinzipals durch das Realm-/Domänen-Trennzeichen, normalerweise "@", getrennt ist. Wenn der Kerberos-Prinzipal keinen Realm/Domäne enthält, wird angenommen, dass der Prinzipal zum Standard-Realm/-Domäne für die aktuelle Umgebung gehört.

Normalerweise besteht ein Kerberos-Prinzipal aus drei Teilen: Primary, Instanz und Realm/Domäne. Das Format eines typischen Kerberos V5-Prinzipals ist `primary/instanz@REALM`.

- Der Primary ist der erste Teil des Prinzipals, dies ist der Benutzername (falls es sich um das Konto einer Person handelt), oder der Name/das Konto eines Systembenutzers oder ein Hostname.
- Die Instanz ist ein optionaler String, der den Primary näher bestimmt. Die Instanz ist durch einen Schrägstrich (/) vom Primary getrennt. Im Fall eines Benutzers ist die Instanz normalerweise `null`, aber ein Benutzer kann auch einen zusätzlichen Prinzipal mit einer Instanz `admin` haben, die er für die Verwaltung einer Datenbank verwendet. Der Prinzipal `mycmuser@CONSOL.DE` ist völlig unabhängig vom Prinzipal `mycmuser/administrator@CONSOL.DE`. Er hat ein eigenes Passwort und eigene Berechtigungen. Bei einem Host ist die Instanz der vollqualifizierte Hostname, z. B. `mymachine.consol.com`.
- Der Realm ist der Kerberos-Realm, normalerweise der Domänenname in Großbuchstaben. Der Rechner `myserver@consol.com` befindet sich zum Beispiel im Realm `CONSOL.COM`.

Keytab-Datei

Eine keytab-Datei ist eine Datei, die Paare von Kerberos-Prinzipalen und die entsprechenden verschlüsselten Schlüssel (die aus dem Kerberos-Passwort abgeleitet sind) enthält. Wenn ein Passwort geändert wird, müssen Sie die keytab-Datei neu bauen. Im Kontext von ConSol CM tritt dies ein, wenn das Passwort des Systembenutzers (in unserem Beispiel ***tomcat***) geändert wurde, **nicht** wenn ein CM-Bearbeiter sein Passwort geändert hat.

Konfiguration von Single Sign-On mit Kerberos für ConSol CM

Anforderungen

Für auf Kerberos basierendes Single Single Sign-On in ConSol CM benötigen Sie:

- Domänen-Controller für die Windows-Domäne
- ConSol CM-Server
- Windows-Clients



Da in einer Kerberos-Umgebung Zeitstempel verwendet werden, um die Gültigkeitsdauer der Tickets zu berechnen und zu überprüfen, ist es unbedingt notwendig, dass alle Komponenten mit der gleichen Systemzeit arbeiten! Es kann zum Beispiel ein auf NTP basierender Zeitserver/-dienst verwendet werden. (Einige Win DCs lassen maximal eine Differenz von fünf Minuten zu.)

Einrichten des Systems

Grundprinzip und erforderliche Schritte

Um als CM-Bearbeiter mit SSO arbeiten zu können, muss der entsprechende CM-Server zu einer Windows-Domäne gehören. Der Domänen-Controller fungiert als *Kerberos Key Distribution Center* (KDC) und die Benutzer-/Kontonamen werden in Active Directory verwaltet. Deshalb müssen Sie

- den CM-Server als Mitglied der Windows-Domäne registrieren.

ConSol CM muss als Kerberos-Dienst registriert sein, sodass Kerberos Service Tickets für Benutzer, die mit CM arbeiten möchten, erstellt werden können. Deshalb muss CM bei Kerberos registriert sein. Da CM unter einem bestimmten Systembenutzer ausgeführt wird (in unserem Beispiel *tomcat*), muss dieser Systembenutzer für den entsprechenden Rechner registriert sein. Deshalb müssen Sie

- eine keytab-Datei für den Systembenutzer *tomcat* für den CM-Rechner anlegen.

Da nicht der Systembenutzer (*tomcat)*, sondern die CM-Bearbeiter mit dem CM arbeiten werden,

- muss die Option *Computer bei Delegierungen aller Dienste vertrauen* in KDC gesetzt sein. Auf diese Art und Weise kann der Systembenutzer (in unserem Beispiel *tomcat*) die Identität von anderen Benutzern (den CM-Bearbeitern) annehmen.

Auf dem CM-Rechner muss der Application Server so konfiguriert werden, dass Kerberos verwendet werden kann, und CM weiß, wo die keytab-Datei mit dem Systembenutzer und Prinzipalnamen zu finden ist, um Kerberos-Verschlüsselung und -Kommunikation einzusetzen. Deshalb müssen Sie

- das entsprechende Modul auf dem Application Server definieren
- die keytab-Datei auf dem CM-Server speichern
- die Datei `cm6-kerberos.properties` konfigurieren

Für die Benutzer, die SSO mit Kerberos und CM verwenden möchten, muss ein CM-Bearbeitername dem Kerberos-Prinzipalnamen zugewiesen werden. Deshalb müssen Sie

- Kerberos-Prinzipalnamen für die CM-Bearbeiter definieren. Dies wird im Admin Tool in der Bearbeiterverwaltung gemacht.

Auf dem Domänen-Controller durchzuführende Schritte

Der erste Schritt ist die Konfiguration des Domänen-Controllers, damit dieser den ConSol CM-Server kennt, d. h. die Integration des CM-Servers in der Windows-Domäne. In unserem Beispiel heißt der Domänen-Controller `mywin2003srv` und die Domäne ist `MYSSODOMAIN.COM`.

Registrieren des ConSol CM-Serverrechners

Zuerst muss der ConSol CM-Serverrechner im Active Directory des Domänen-Controllers registriert werden. In unserem Beispiel ist der Domänen-Controller der Computer *MyComputer*.

> ⚠ Der Radio-Button *Computer bei Delegierungen aller Dienste vertrauen (nur Kerberos)* muss aktiviert sein!

Führen Sie folgenden Befehl aus: `dsamsc`, machines

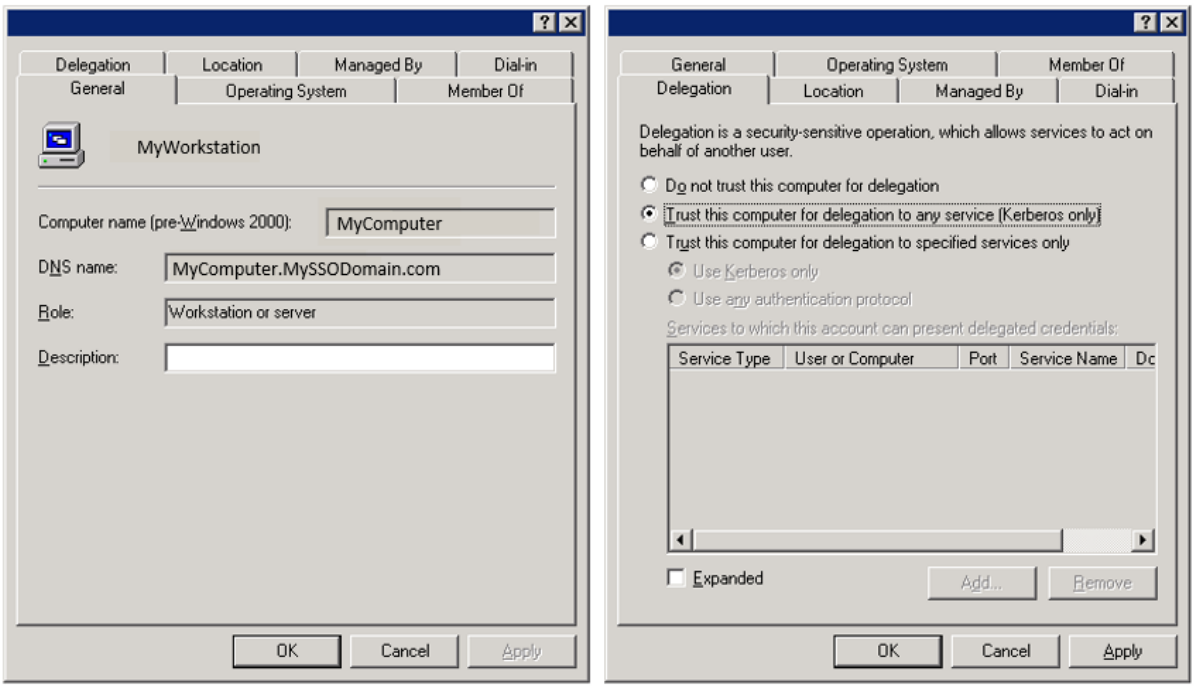


Abbildung 564: *Registrieren des ConSol CM-Serverrechners*

Registrieren des ConSol CM-Serverbenutzers

Als Zweites muss der Benutzer, unter dem der ConSol CM-Serverprozess ausgeführt wird, in Active Directory (als Key Distribution Center fungierend) erstellt und registriert werden, in unserem Beispiel der Benutzer *tomcat*.

Folgende Kontooptionen müssen aktiviert sein:

- Konto wird für Delegierungszwecke vertraut
- Keine Kerberos-Präauthentifizierung erforderlich

Führen Sie folgenden Befehl aus: `dsamsc`, users

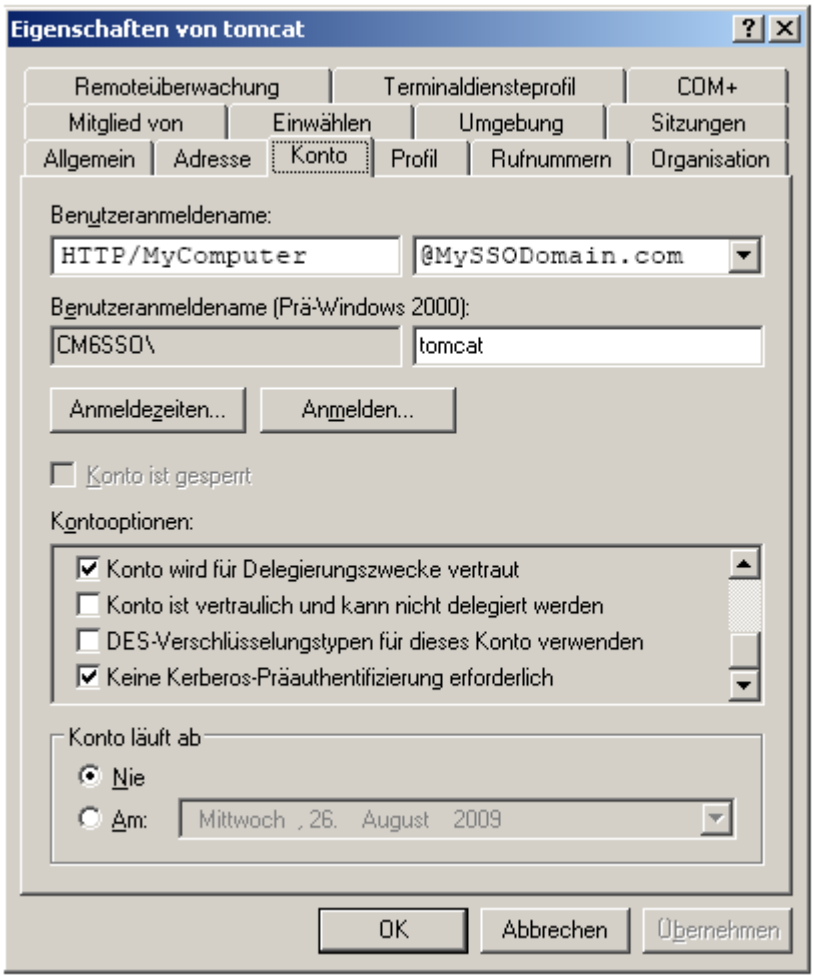


Abbildung 565: *Registrieren des ConSol CM-Serverbenutzers*

Stellen Sie sicher, dass der Prozess des CM-Application-Servers (z. B. JBoss) unter dem angegebenen Benutzer auf dem CM-Rechner läuft.

Erzeugen der keytab-Datei

Auf dem Domänen-Controller wird der ConSol CM-Server als neuer (nicht-Windows) Kerberos-Dienst erstellt und eine Kerberos-keytab-Datei erzeugt. Diese Datei wird später auf dem ConSol CM-Serverrechner benötigt. Die keytab-Datei enthält den gemeinsamen geheimen Schlüssel des Dienstes (SPN, Service Principal Name).

Verwenden Sie folgenden ktpass-Befehl:

```
C:\Program Files\Support Tools>ktpass /out tomcat.keytab /ptype KRB5_NT_PRINCIPAL
 /princ HTTP/MyComputer.MySSODomain.com@MYSSODOMAIN.COM /pass consol.123 /mapuser
 tomcat /crypto rc4-hmac-nt
```

⚠ Wenn `ktpass` nicht verfügbar ist, müssen die ***Windows Server 2003 Support Tools*** installiert werden. Diese sind hier verfügbar.

Erklärung des ktpass-Befehls

```
/out tomcat.keytab
```

=> Verwendet `tomcat.keytab` als Ausgabedatei.

```
/ptype KRB5_NT_PRINCIPAL
```

=> Gibt den Prinzipaltyp an.

- **KRB5_NT_PRINCIPAL** ist der allgemeine Prinzipaltyp (empfohlen).
- **KRB5_NT_SRV_INST** ist die Instanz des Benutzerdienstes.
- **KRB5_NT_SRV_HST** ist die Instanz des Hostdienstes.

```
/princ HTTP/MyComputer.MySSODomain.com@MYSSODOMAIN.COM
```

=> Gibt den Kerberos-Prinzipalnamen in der Form host/computer.<domäne>@<realm> an. Siehe Benutzeranmeldename in der Konfiguration (obige Abbildungen).

```
/pass consol.123
```

=> Setzt das Passwort für den Benutzer *princ*, der mit `/princ` angegeben wird

```
/mapuser tomcat
```

=> Ordnet den Namen des Kerberos-Prinzipals (Parameter `/princ`) dem angegebenen Domänenkonto zu.

```
/crypto rc4-hmac-nt
```

=> Gibt die Schlüssel an, die in der keytab-Datei erzeugt wurden:

- **DES-CBC-CRC** wird aus Kompatibilitätsgründen verwendet.
- **DES-CBC-MD5** hält sich enger an die MIT-Implementierung und wird aus Kompatibilitätsgründen verwendet.
- **RC4-HMAC-NT** setzt 128-bit-Verschlüsselung ein.
- **AES256-SHA1** setzt AES256-CTS-HMAC-SHA1-96-Verschlüsselung ein.
- **AES128-SHA1** setzt AES128-CTS-HMAC-SHA1-96-Verschlüsselung ein.
- **Alle** bedeutet, dass alle unterstützten Kryptographietypen verwendet werden können.

Auf dem ConSol CM-Server durchzuführende Schritte

Führen Sie den Application Server (z. B. JBoss) unter dem Systembenutzer aus, der in der Windows-Domäne registriert ist (in unserem Beispiel *tomcat*).

Installieren Sie ConSol CM wie gewohnt und aktivieren und konfigurieren Sie Kerberos wie in den nächsten Schritten beschrieben.

Aktivieren von Kerberos in ConSol CM

Wenn Sie eine Ersteinrichtung vornehmen, können Sie entscheiden, ob Kerberos aktiviert sein soll. Beachten Sie, dass dies nur ein Hinweis ist und zusätzliche Konfigurationen erforderlich sind (siehe nächste Schritte).

Wenn Ihr ConSol CM schon ohne Kerberos konfiguriert ist, können Sie Kerberos im Admin Tool aktivieren, indem Sie die System-Property cmas-core-security, kerberos.v5.enabled auf „true“ setzen. Ein Neustart des Servers ist erforderlich, damit die neue Einstellung wirksam wird.

Konfigurieren von Kerberos

Ein ConSol CM-Server liest die Konfigurationsparameter aus der Datei `cm6-kerberos.properties` aus folgendem Klassenpfad: (Die einzelnen Cluster-Nodes brauchen möglicherweise eigene Konfigurationen, weshalb jeder Node seine Datei `cm6-kerberos.properties` aus dem Klassenpfad liest.)

- **Unter JBoss 5:**
  `../jboss/server/{domain}/conf/cm6-kerberos.properties`
- **Unter WebLogic:**
  `../{domain}/cm6-kerberos.properties`

- **Unter JBoss 7 und WildFly 8.2:**
  1. mkdir -p ${jboss}/modules/system/layers/base/com/consol/cmas/main/
  2. Erstellen und editieren Sie ${j-boss}/modules/system/layers/base/com/consol/cmas/main/module.xml

```
<module xmlns="urn:jboss:module:1.1" name="com.consol.cmas">
   <resources>
      <resource-root path="."/>
   </resources>
</module>
```

  3. Legen Sie die Properties-Datei hier ab: ${j-boss}/modules/system/layers/base/com/consol/cmas/main/cm6-kerberos.properties
  4. Editieren Sie die Konfiguration, z. B. ${jboss}/standalone/configuration/cm6.xml

```
<subsystem xmlns="urn:jboss:domain:ee:1.1">
   ...
   <global-modules>
      <module name="com.consol.cmas" slot="main" />
   </global-modules>
</subsystem>
```

Code-Beispiel 99: *JBoss 7*

```
<subsystem xmlns="urn:jboss:domain:ee:2.0">
   ...
   <global-modules>
      <module name="com.consol.cmas" slot="main" />
   </global-modules>
</subsystem>
```

Code-Beispiel 100: *WildFly 8.2*

Falls Sie ein Cluster aus mehr als einem ConSol CM-Server in Betrieb haben, muss jeder Server eine eigene Properties-Datei haben.

Die .properties-Datei sollte Folgendes enthalten:

- Referenz auf eine Kerberos-Konfigurationsdatei (z. B. `krb5.ini` oder `krb5.conf`)
- Einen oder mehrere Dienstprinzipale, d. h. Referenz auf die keytab-Datei

```
# path to kerberos configuration
kerberos.config.location=C:\\conf\\krb5.ini

# one or more service principals (principal = path to keytab file)
HTTP/MyComputer.MySSODomain.com@MYSSODOMAIN.COM=C:\\conf\\tomcat.keytab
```

Code-Beispiel 101: *cm6-kerberos.properties*

```
[libdefaults]
   default_realm = MYSSODOMAIN.COM
   default_tkt_enctypes = rc4-hmac des-cbc-md5 des-cbc-crc des3-cbc-sha1
   default_tgs_enctypes = rc4-hmac des-cbc-md5 des-cbc-crc des3-cbc-sha1

[realms]
   MYSSODOMAIN.COM = {
      kdc = mywin2003srv
      admin_server = mywin2003srv:8888
   }

[domain_realm]
   .mywin2003srv = MYSSODOMAIN.COM
   mywin2003srv = MYSSODOMAIN.COM
```

Code-Beispiel 102: *krb5.ini*

keytab-Datei

Kopieren Sie die keytab-Datei, die Sie auf dem Domänen-Controller erzeugt haben, an den in der Konfigurationsdatei `cm6-kerberos.properties` angegebenen Speicherort.

⚠ Sie müssen den ConSol CM-Serverprozess neu starten, damit diese Änderung wirksam wird.

Konfigurieren der CM-Bearbeiter für Kerberos

Wenn die System-Property cmas-core-security, kerberos.v5.enabled auf „true“ gesetzt wurde, ist das Feld *Kerberos Principal Name* in den Bearbeiterdaten verfügbar.

Sie gelangen auf diese Seite, indem Sie das Navigationselement *Bearbeiter* der Navigationsgruppe *Zugriff und Rollen* öffnen und auf den Button *Bearbeiten* eines Bearbeiter klicken.

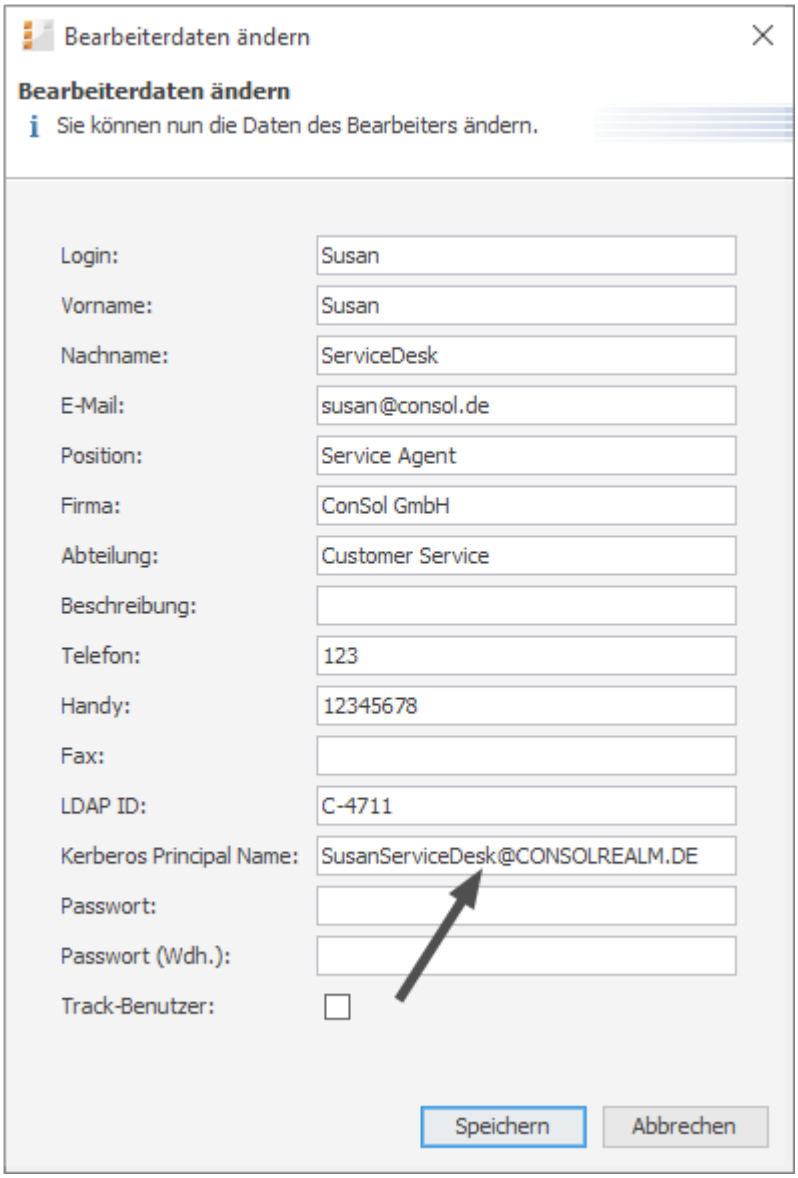


Abbildung 566: *ConSol CM Admin Tool - Zugriff und Rollen, Bearbeiter: Editieren des Kerberos Principal Name*

Der *Kerberos Principal Name* ist ein Benutzername zusammen mit dem entsprechenden Realm-/Domänennamen, zum Beispiel SusanServiceDesk@consol.de. Damit können sich Benutzer mit dem Anmeldenamen SusanServiceDesk authentifizieren, die zu unterschiedlichen Kerberos-Realms gehören (SusanServiceDesk@consol.de und SusanServiceDesk@consol.pl sind zwei unterschiedliche Benutzer). Wenn das Feld *Kerberos Principal Name* für einen Bearbeiter leer ist, wird stattdessen der Benutzername (*Login*) verwendet. Wenn die Kerberos-Authentifizierung fehlschlägt, wird der reguläre Authentifizierungsmechanismus (DATABASE, LDAP) verwendet. In diesem Fall ist Single Sign-On nicht möglich und dem Bearbeiter wird die Anmeldeseite angezeigt.

Auf dem Client-Rechner durchzuführende Schritte

Internet Explorer

Internet Explorer muss so konfiguriert werden, dass die automatische Anmeldung aktiviert ist. Standardmäßig ist dies bei der Sicherheitseinstellung *mittel bis niedrig*, der Standardeinstellung für die Zone *Lokales Intranet*, erlaubt.

Es sind folgende Einstellungen für das Anmeldeverhalten verfügbar.

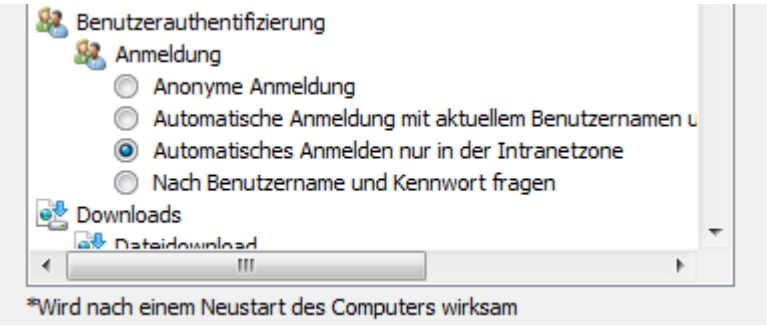


Abbildung 567: *Anmeldekonfiguration von Internet Explorer*

Einstellungen und resultierendes Verhalten:

- **Anonyme Anmeldung**
  Single Sign-On ist nicht möglich, der Benutzer (CM-Bearbeiter) bekommt die ConSol CM-Anmeldeseite angezeigt.
- **Automatisches Anmelden nur in der Intranetzone**
  Single Sign-On erfolgt automatisch, aber nur wenn die Website zur Zone *Lokales Intranet* gehört.
- **Automatische Anmeldung mit aktuellem Benutzernamen und Kennwort**
  Single Sign-On erfolgt automatisch mit den aktuellen Anmeldeinformationen des Benutzers.
- **Nach Benutzername und Kennwort fragen**
  Das Betriebssystem zeigt ein Dialogfenster an, in dem der Benutzer die Anmeldeinformationen beim Betriebssystem eingibt, die dann für die Kerberos-Authentifizierung verwendet werden.

Firefox

Bei Verwendung der Standardeinstellungen unterstützt ***Firefox*** Single Sign-On mit Kerberos nicht. Um Single Sign-On zu aktivieren, müssen Sie die URI des ConSol CM Web Clients in der Firefox-Konfiguration eintragen.

Führen Sie dazu folgende Schritte aus:

- Öffnen Sie `about:config`.
- Fügen Sie den Webserver zur Property `network.automatic-ntlm-auth.trusted-uris` hinzu
  (zum Beispiel `http://mycm6server`, wenn das die URI ist).
  Wenn das nicht funktioniert, führen Sie folgenden Schritt aus:

- Fügen Sie den Web-Server zur Property `network.negotiate-auth.trusted-uris` hinzu
  (zum Beispiel `http://mycm6 server`, wenn das die URI ist)

Sie können diese Property auch im Dateisystem setzen. Öffnen Sie dazu die Datei

`C:\Benutzer\[BENUTZER]\AppData\Roaming\Mozilla\Firefox\Profiles\[XYZ].default\prefs.js`

und fügen Sie folgende Zeile hinzu bzw. ersetzen Sie sie:

```
user_pref("network.automatic-ntlm-auth.trusted-uris", "http://mycm6server");
```

oder

```
user_pref("network.negotiate-auth.trusted-uris", "http://mycm6server");
```

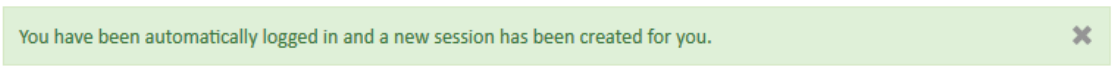
⚠ Sie müssen Firefox nach dieser Änderung neu starten.

Verwendung des Systems

Single Sign-On aus der Benutzerperspektive

Ein Bearbeiter, der sich mit Single Sign-On in ConSol CM anmeldet, wird Folgendes bemerken:

- Es wird keine ConSol CM-Anmeldeseite angezeigt.
- Stattdessen kann (kurz) eine Zwischenseite mit Text angezeigt werden (mit der über JavaScript einige Client-Daten gesammelt werden), die den Benutzer sofort zur Hauptseite des ConSol CM Web Clients weiterleitet.
  Hier wird folgende Meldung angezeigt:



Es ist trotzdem möglich, sich als anderer ConSol CM-Benutzer anzumelden. Klicken Sie dazu auf den Button zum Abmelden, wodurch Sie auf die Anmeldeseite weitergeleitet werden, oder rufen Sie explizit die URL `.../cm-client/login` auf.

Single Sign-On in mehreren Domänen

Erstellen Sie für jede Domäne, in der Sie Single Sign-On aktivieren, eine neue Domäne/Benutzer und einen neuen Kerberos-Prinzipal und tragen Sie diese in die Datei `cm6-kerberos.properties` ein:

```
# path to kerberos configuration (think krb5.conf or krb5.ini)
kerberos.config.location=/etc/krb5.conf

# one or more service principals (principal = path to keytab file)
HTTP/MyServerMyDomain.com@MYDOMAIN.COM=/etc/krb5_mycompanycom.keytab
HTTP/MyServer.MyDEDomain@MYDOMAIN.DE=/etc/krb5_mycompanyde.keytab
```

Zuordnen des Kerberos-Benutzernamens zum Bearbeiternamen

Um auf Kerberos basierendes Single Sign-On verwenden zu können, muss der Kerberos-Prinzipal (d. h. der Anmeldename des Benutzers im Betriebssystem) mit dem Namen eines ConSol CM-Bearbeiters verknüpft werden.

Standardmäßig findet die Zuordnung auf einem der beiden folgenden Wege statt:

- **Explizite Zuordnung**
  Es wird der Prinzipalname genommen und ein ConSol CM-Bearbeiter gesucht, bei dem dieser als *Kerberos Principal Name* gespeichert ist. Wenn ein solcher Bearbeiter gefunden wird, wird dieser verwendet.
- **Zuordnung über regulären Ausdruck**
  Der reguläre Ausdruck, der in der System-Property cmas-core-security, kerberos.v5.username.regex definiert ist, wird genommen und auf den Prinzipal angewendet. Das Ergebnis wird verwendet, um einen ConSol CM-Bearbeiter mit diesem Login zu suchen:
  - Die erste passende Gruppe des regulären Ausdrucks (in Klammern) wird als Anmeldename des Bearbeiters verwendet,
    mit dem Standardwert der Property „(.*)@.*“ wird z. B. *Huber@MySSODomain.com* in *Huber* konvertiert.

Wenn weitere Anpassungen erforderlich sind, lesen Sie bitte *UsernameAdapter interface javadoc*.

Starten und Stoppen der Kerberos-Authentifizierung

Die Kerberos-Authentifizierung kann im Admin Tool in der Navigationsgruppe *Dienste* -> Navigationselement *CM-Services -> Kerberos v5 authentication provider*, siehe Abschnitt CM-Services, gestartet bzw. gestoppt werden.

F.14.2 Authentifizierungsmethoden für Kunden in CM/Track

F.14.2.1 Verfügbare Authentifizierungsmethoden

Es gibt drei mögliche Authentifizierungsmethoden:

- Gegen die ConSol CM-Datenbank
  Diese Methode heißt *Datenbank-Modus*, siehe Datenbank-Authentifizierung für Kunden.
- Gegen einen LDAP-Server
  Diese Methode heißt *LDAP-Modus*, siehe LDAP-Authentifizierung für Kunden in CM/Track.
- Gegen einen LDAP-Server und die ConSol CM-Datenbank
  Die Reihenfolge kann konfiguriert werden. Diese Methode heißt *Gemischter Modus*, siehe Gemischte Authentifizierungsmethode

F.14.2.2 Definieren der Authentifizierungsmethode

Die Authentifizierungsmethode wird durch die System-Property cmas-core-security, contact.authentication.method bestimmt. Eine Änderung dieser Property erfordert keinen Neustart des Servers und wird für alle Cluster-Nodes übernommen.

Die möglichen Werte (siehe auch Abschnitt System-Properties) und ihr entsprechendes Systemverhalten sind:

- **DATABASE**
  Zuerst wird die Anmeldung über die Datenbank versucht, sofern der Kunde ein Datenbankpasswort hat. d. h. wenn Login und Passwort in der ConSol CM-Datenbank gespeichert sind und damit von den ConSol CM-Bearbeitern, oder indirekt von den Kunden selber verwaltet werden, wenn diese ihr Passwort zurücksetzen, siehe Abschnitt Template zum Zurücksetzen des Passworts durch Kunden in CM/Track V2.
- **LDAP**
  Es wird eine Anmeldung über die verfügbaren LDAP-Server versucht, wenn eine LDAP-ID angegeben wurde, d. h. das Passwort ist im LDAP-Verzeichnis gespeichert und kann nicht über ConSol CM geändert werden, weder vom Kunden noch von einem Bearbeiter.
- **LDAP,DATABASE**
  Beim ersten Authentifizierungsversuch werden die verfügbaren LDAP-Server verwendet, sofern eine LDAP-ID angegeben ist. Sollte dieser fehlschlagen, wird eine Anmeldung über die Datenbank versucht, sofern der Kunde ein Datenbankpasswort hat.
- **DATABASE,LDAP**
  Zuerst wird die Anmeldung über die Datenbank versucht, sofern der Kunde ein Datenbankpasswort hat. Wenn diese fehlschlägt, wird die Authentifizierung über einen verfügbaren LDAP-Server versucht, sofern eine LDAP-ID angegeben wurde.

Groß- und Kleinschreibung, Kommas und Leerzeichen werden bei den Werten ignoriert.

F.14.2.3 Datenbank-Authentifizierung für Kunden

Die Datenbank-Authentifzierung durch das Setzen der System-Property cmas-core-security, contact.authentication.method auf „DATABASE“ (Standardwert) aktiviert.

Es gibt zwei Schritte, die durchgeführt werden müssen, um die Datenbank-Authentifizierung für Kunden mit CM/Track einzurichten:

- Erstellen Sie Kundenfelder für den Benutzernamen (Login) und das Passwort im Admin Tool (siehe Definieren der Kundenfelder für das CM/Track-Login und -Passwort)
- Geben Sie den Benutzernamen und das Passwort für die jeweiligen Kunden im Web Client ein (siehe Gewähren von Zugang zu CM/Track für Kunden)

Wenn die Datenbank-Authentifizierung verwendet wird, können Sie es Ihren Kunden ermöglichen, ihre eigenen Passwörter zu ändern, siehe Konfigurieren von CM/Track für das Zurücksetzen des Passworts durch den Kunden.

Definieren der Kundenfelder für das CM/Track-Login und -Passwort

Die Felder für das Login und Passwort für einen Kunden sind normale Kundenfelder auf der Kontaktstufe. Eine Einführung in die Verwaltung von Kundenfeldern und die GUI-Konfiguration für Kundendaten finden Sie im Abschnitt Einrichten des Kundendatenmodells.

Editieren Sie die Felder, die Kundendaten enthalten (wenn es zwei Stufen gibt: **nicht** auf der Firmenstufe sondern auf der Kontaktstufe!) wie im folgenden Beispiel gezeigt. Sie gelangen auf die folgende Seite, indem Sie in der Navigationsgruppe *Kunden* das Navigationselement *Datenmodelle* öffnen.

- Erstellen Sie ein Feld für das **Login** mit der Annotation `username` = „true“.

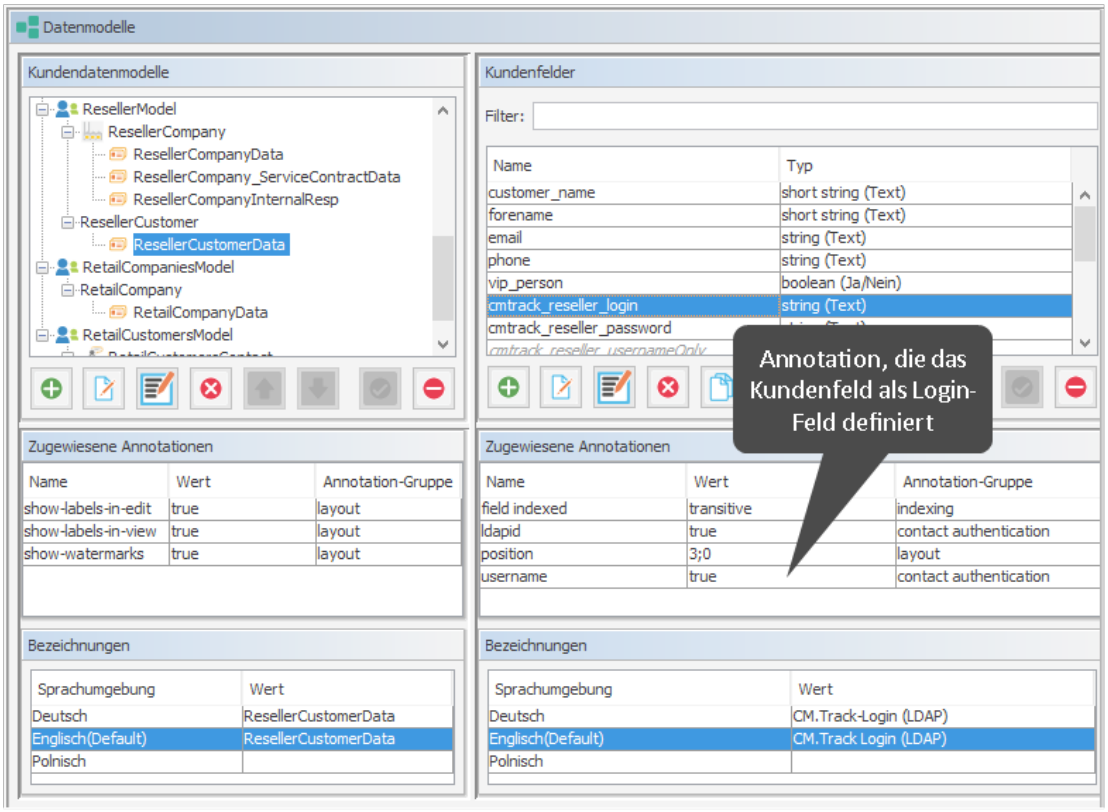


Abbildung 568: *ConSol CM Admin Tool - Kunden, Datenmodelle, CM/Track: Annotation „username“ für Login*

> ⚠ Die Zuweisung der Annotation `username` zu einem Kundenfeld ist nur möglich, wenn diese Annotation noch nicht zugewiesen wurde. Andernfalls ist sie nicht erlaubt. Beim Zuweisen der Annotation muss zuerst eine Warnung bestätigt werden, bevor der Vorgang ausgeführt wird, da dieser länger dauern kann. Das Entfernen der Annotation muss ebenfalls bestätigt werden, da es nicht rückgängig gemacht werden kann: Beim Entfernen werden die Werte für die Benutzernamen unwiderruflich aus dem internen Speicher gelöscht.

- Erstellen Sie ein Feld für das Passwort mit der Annotation `password` = „true“. Die Annotation `text-type` = „password“ sorgt dafür, dass nur Sternchen/Punkte im Web Client angezeigt werden und das Passwort nicht im Klartext angezeigt wird.

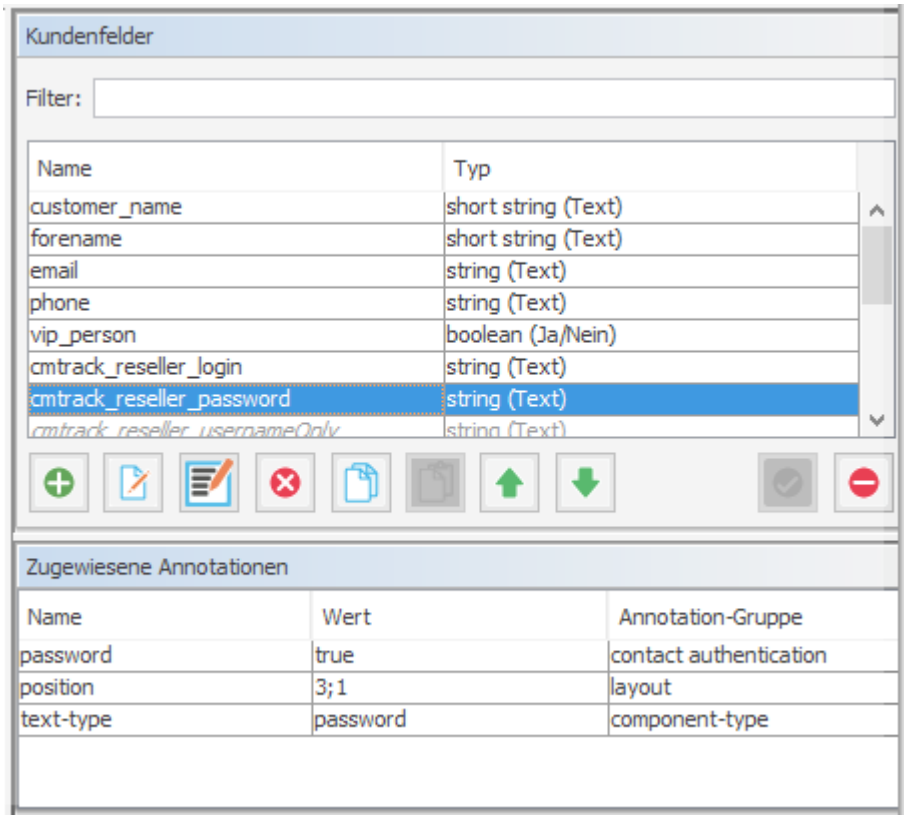


Abbildung 569: *ConSol CM Admin Tool - Kunden, Datenmodelle, CM/Track: Annotation für Passwort*

> ⚠ Die Annotation `password` muss vor der Zuweisung bestätigt werden.
>
> Bei einer Aktualisierung von einer CM-Version unter 6.11 auf 6.11 und höher: Wenn diese Annotation gesetzt wird, liest das System die Passwörter im Klartext aus den ursprünglichen Feldwerten, verschlüsselt sie und speichert sie verschlüsselt im internen Speicher. Die ursprünglichen Werte werden gelöscht, sodass die Werte im Klartext nicht mehr wiederhergestellt werden können.
>
> Beim Versuch, die Annotation `password` zu entfernen, muss der Vorgang ebenfalls bestätigt werden, da die verschlüsselten Passwörter aus dem internen Speicher gelöscht werden. Nach dem Entfernen der Annotation sind die Passwort-Informationen nicht mehr vorhanden und können nicht mehr wiederhergestellt werden.

Wenn ein Szenario aus einer CM-Version unter 6.11 in ein System mit CM 6.11 oder höher importiert wird, wird die Transformation der Benutzernamen und Passwörter automatisch ausgeführt. Dies ist detailliert in Transformation von Benutzernamen- und Passwortfeldern bei einem Import nach CM 6.11 beschrieben.

Gewähren von Zugang zu CM/Track für Kunden

Der Bearbeiter, der mit dem Web Client arbeitet, kann jedem Kunden, der Zugang zum Portal CM/Track haben soll, einen Benutzernamen, ein initiales Passwort und ein CM/Track-Benutzerprofil zuweisen. Der Benutzername muss eindeutig sein. Dies wird vom System überprüft. Sie können einen Benutzernamen nicht ein zweites Mal eingeben, wenn dieser bereits für einen anderen Kunden verwendet wird. Das Passwort wird verschlüsselt in der CM-Datenbank gespeichert. Das bedeutet, dass ein Bearbeiter ein neues Passwort setzen kann, z. B. wenn ein Kunde anruft und darum bittet. Es ist allerdings nie möglich, das Passwort aus dem System auszulesen.

> Als Administrator können Sie festlegen, ob bei den CM/Track-Benutzernamen zwischen Groß- und Kleinbuchstaben unterschieden werden soll. Verwenden Sie dazu die CM-System-Property cmas-core-security, policy.track.username.case.sensitive. Dabei handelt es sich um eine Boolean-Variable. Wenn sie auf „true“ gesetzt ist, wird bei den CM/Track-Benutzernamen zwischen Groß- und Kleinschreibung unterschieden. Stellen Sie sicher, dass die verwendete Datenbank-Collation Strings mit Groß- und Kleinschreibung unterstützt.

Das folgende Beispiel zeigt die Kundendaten eines Beispielkontaktes im ConSol CM Web Client. Sie gelangen auf diese Seite, indem Sie einen Kontaktdatensatz im Bearbeitungsmodus öffnen.

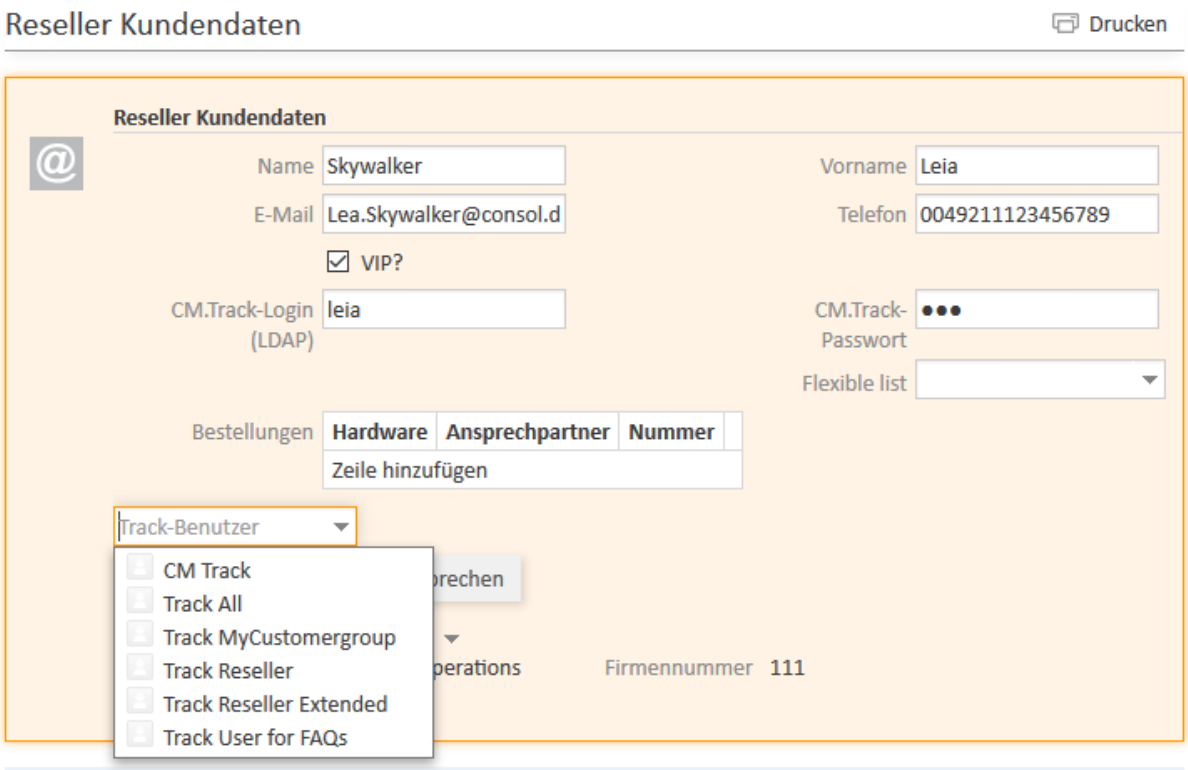


Abbildung 570: *ConSol CM Web Client, Kontaktseite - CM/Track-Benutzerdaten*

Konfigurieren von CM/Track für das Zurücksetzen des Passworts durch den Kunden

CM/Track kann so konfiguriert werden, dass den Kunden ein Link angezeigt wird, mit dem sie ihr Passwort zurücksetzen können. Diese Funktion basiert auf dem Template `track-password-reset-template`. Eine detaillierte Erklärung dazu finden Sie im Abschnitt Template zum Zurücksetzen des Passworts durch Kunden in CM/Track V1 / Template zum Zurücksetzen des Passworts durch Kunden in CM/Track V2.Das Zurücksetzen des Passworts in CM/Track ist nur möglich, wenn der Datenbank-Modus verwendet wird. Bei LDAP-Authentifizierung ist dies nicht möglich. Eine Beschreibung aller verfügbaren Authentifizierungsmethoden finden Sie im Abschnitt Authentifizierungsmethoden für Kunden in CM/Track.

Beachten Sie, dass die From-Adresse der E-Mail, die an einen Kunden gesendet wird, der ein neues Passwort angefordert hat, mit der CM-System-Property cmas-core-security, password.reset.mail.from gesetzt werden kann.

F.14.2.4 LDAP-Authentifizierung für Kunden in CM/Track

Einführung in die LDAP-Authentifizierung in ConSol CM

ConSol CM bietet LDAP-Authentifizierung für CM/Track als Standardfunktion an, d. h. die Passwörter der ConSol CM-Kunden werden nicht in der ConSol CM-Datenbank verwaltet, sondern von einem LDAP-Server (wie z. B. ein ***Microsoft Active Directory***-Server) abgerufen.

Wenn sich die Kunden in CM/Track anmelden möchten, geben sie ihren Benutzernamen und ihr Passwort ein und drücken die Eingabetaste. Im Hintergrund sendet der ConSol CM-Server eine Anfrage mit dem Benutzernamen und dem Passwort an den LDAP-Server und dieser überprüft, ob die Anmeldeinformationen gültig sind.

Wenn die Anmeldeinformationen gültig sind, wird die Genehmigung zurück an den ConSol CM-Server gesendet und der Kunde wird in CM/Track angemeldet.

> Beachten Sie, dass die LDAP-Verbindung nur zur Authentifizierung des Kunden (Bestätigung seiner Identität) verwendet wird. Die Autorisierung (d. h. die Zuweisung von Zugangsberechtigungen im System) erfolgt über Zuweisung eines CM/Track-Benutzerprofils im Web Client. Die CM/Track-Benutzerprofile werden in der Bearbeiter- und Rollenverwaltung im Admin Tool verwaltet.

Der CM-Authentifizierungsprozess mittels LDAP ist in der folgenden Abbildung erklärt.

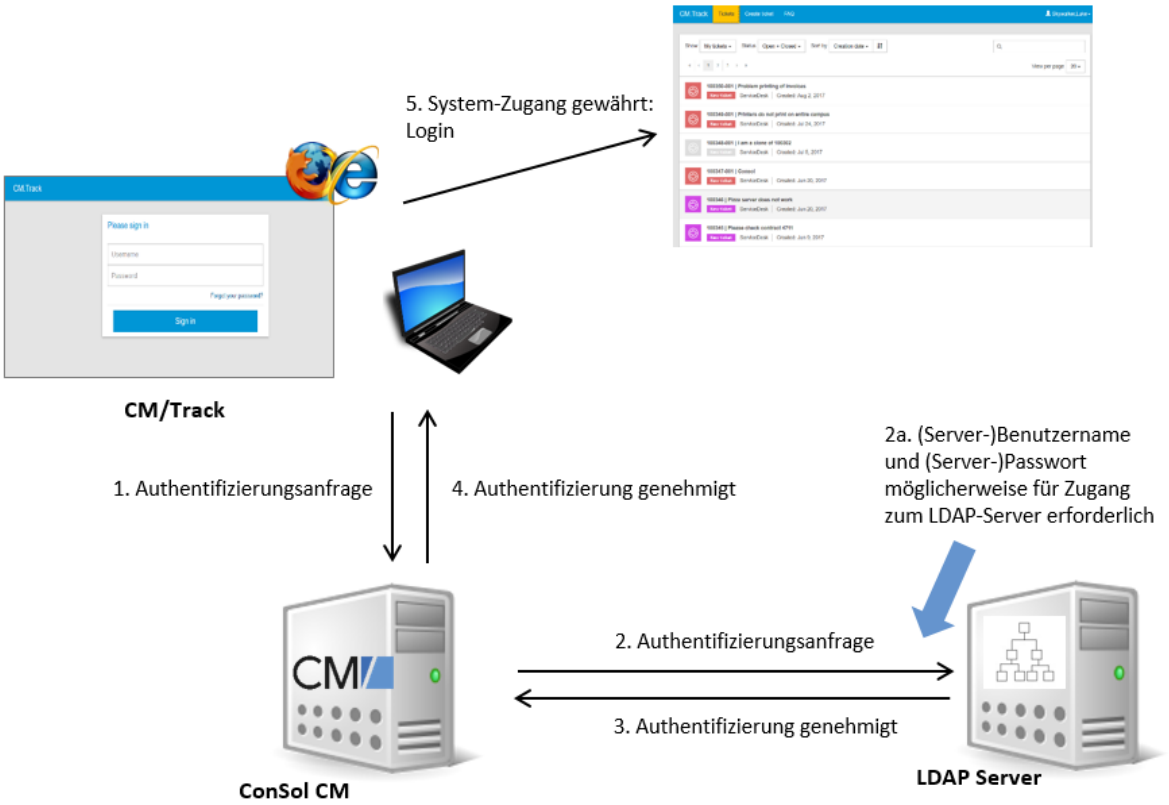


Abbildung 571: *ConSol CM - LDAP-Authentifizierungsprozess für CM/Track*

Konfigurieren von LDAP-Authentifizierung

Die LDAP-Authentifizierung wird durch Setzen der System-Property cmas-core-security, contact.authentication.method auf „LDAP“ aktiviert.

Als nächstes müssen Sie die erforderlichen Werte in den System-Properties setzen (Navigationsgruppe *System*, Navigationselement *System-Properties*, siehe System-Properties).

Die LDAP-Server können mit folgenden System-Properties aus dem Modul `cmas-core-security` definiert werden.

{name} ist ein String, den Sie frei wählen können, um die LDAP-Server zu unterscheiden. Er muss immer angegeben werden, auch wenn nur ein LDAP-Server konfiguriert wird. Sie sollten einen einfachen String für {name} verwenden, der keine Schlüsselwörter wie *internal* oder *external* und keine Sonderzeichen enthält.

- contact.authentication.method
  LDAP
- ldap.contact.{name}.providerurl
  Der Wert der Property ist die Adresse des LDAP-Servers in der Form *ldap[s]://host:port*.

- ldap.contact.{name}.userdn
  Der Wert ist der Benutzer-DN, der verwendet wird, um den Kontakt-DN über die LDAP-ID nachzuschlagen. Wenn der Wert nicht gesetzt ist, wird ein anonymes Konto verwendet.
- ldap.contact.{name}.password
  Die Property enthält das Passwort, mit dem der Kontakt-DN über die LDAP-ID nachgeschlagen wird. Wenn der Wert nicht gesetzt ist, wird ein anonymes Konto verwendet.
- ldap.contact.{name}.basedn
  Dies ist der Stammpfad, in dem der Kontakt-DN über die LDAP-ID gesucht wird, z. B. *ou=accounts,dc=mycompany,dc=de*.
- ldap.contact.{name}.searchattr
  Der Wert dieser Property steht für das Attribut, mit dem der Kontakt-DN über die LDAP-ID gesucht wird, z. B. *uid*.

Diese System-Properties sind möglicherweise initial nicht in Ihrem CM-System vorhanden. Fügen Sie sie einfach manuell hinzu. Nach einer Änderung einer der obigen System-Properties ist kein Neustart des Servers erforderlich. Die Änderung wird für alle Cluster-Nodes übernommen. Der Platzhalter {name} ermöglicht die Definition von Konfigurationen für mehrere unterschiedliche LDAP-Server.

- ldap.initialcontextfactory
  Dies ist eine vordefinierte globale Property. Wenn sie nicht gesetzt ist, wird *com.sun.jndi.ldap.LdapCtxFactory* als Wert verwendet.

Authentifizierungsversuche gegen LDAP-Server finden bis zum ersten erfolgreichen Versuch statt, wobei die Reihenfolge der Server durch die Werte von {name} bestimmt wird (aufsteigende alphabetische Reihenfolge der Werte).

Gemischte Authentifizierungsmethode

Setzen Sie die System-Property cmas-core-security, contact.authentication.method entsprechend der gewünschten Reihenfolge der Authentifizierungsinstanzen:

- **LDAP,DATABASE**
  Beim ersten Authentifizierungsversuch werden die verfügbaren LDAP-Server verwendet, sofern eine LDAP-ID angegeben ist. Sollte dieser fehlschlagen, wird eine Anmeldung über die Datenbank versucht, sofern der Kunde ein Datenbankpasswort hat
- **DATABASE,LDAP**
  Zuerst wird die Anmeldung über die Datenbank versucht, sofern der Kunde ein Datenbankpasswort hat. Wenn diese fehlschlägt, wird die Authentifizierung über einen verfügbaren LDAP-Server versucht, sofern eine LDAP-ID angegeben wurde.

Das CM-System kontaktiert zuerst die Instanz, die an erster Stelle steht, und danach die zweite Instanz. Wenn die Authentifizierungsmethode für Kontakte zum Beispiel auf „LDAP,DATABASE“ gesetzt ist, und der Kunde (Kontakt) ein Passwort verwendet, das nur in der Datenbank gültig ist, ist die Anmeldung erfolgreich.

Im `server.log` wird dann folgende Meldung angezeigt:

```
LDAP login failed: [LDAP: error code 49 - Invalid Credentials]; nested exception is
 javax.naming.AuthenticationException: [LDAP: error code 49 - Invalid Credentials]
```

Logging aller LDAP-Anmeldeversuche in CM/Track V2

Alle aufgetretenen LDAP-Fehler werden ohne Stack Trace von Loggern mit folgendem Präfix protokolliert:

- `com.consol.cmas.core.security.contact`

Der Stack Trace von LDAP-Fehlern wird nicht protokolliert, da die Log-Dateien ansonsten mit Details zu fehlgeschlagenen Anmeldeversuchen auf dem ersten LDAP-Server vollgeschrieben würden, obwohl die darauf folgende Anmeldung beim zweiten LDAP-Server erfolgreich war.

Verwendung von LDAPS (LDAP über SSL)

Einleitung

Standardmäßig werden die Daten im Klartext übertragen, wenn ein LDAP-Client auf einen LDAP-Server zugreift. Wenn Sie möchten, dass der Benutzername und das Passwort verschlüsselt an den LDAP-Server übertragen werden, müssen Sie die LDAP-Authentifizierung mit LDAPS aufsetzen.

Vorbereitungen

Sie müssen den Rechner des CM-Servers (Java) so konfigurieren, dass er Zertifikate verwenden kann. Eine Methode dafür ist im folgenden Abschnitt für einen Linux-Rechner beschrieben.

1. Rufen Sie das Zertifikat ab:
   `openssl s_client -connect dc2.mydomain.com:ldaps`
2. Die Antwort enthält einen Abschnitt, der mit "---BEGIN CERTIFICATE " beginnt und mit "END CERTIFICATE ---" endet.
   Kopieren Sie diesen Abschnitt in eine Datei, z. B. /tmp/certificate2_dc2_mydomain_com.txt
3. Importieren Sie das Zertifikat in den Truststore des Rechners, z. B. /home/mydirectory/mytruststore
   `$JAVA_HOME/bin/keytool -import -alias <arbitrary> -trustcacerts -keystore /home/mydirectory/mytruststore -file`/tmp/certificate2_dc2_mydomain_com.txt
   Sie müssen ein Passwort eingeben (setzen).
4. Fügen Sie den Truststore in die ConSol CM-Konfigurationsdatei unter JAVA_OPTS ein:
   `-Djavax.net.ssl.trustStore=/home/mydirectory/mytruststore -Djavax.net.ssl.trustStorePassword=<see above>`

LDAPS-Konfiguration mit dem ConSol CM Admin Tool (System-Properties)

Konfigurieren Sie den ConSol CM-Server so, wie im folgenden Beispiel gezeigt:

- cmas-core-security, ldap.authentication = simple
- cmas-core-security, ldap.contact.name.basedn = OU=myOU,DC=myDC
- cmas-core-security, ldap.initialcontextfactory = com.sun.jndi.ldap.LdapCtxFactory

- cmas-core-security, ldap.contact.name.password = myLDAPpw
- cmas-core-security, ldap.contact.name.searchattr = sAMAccountName
- cmas-core-security, ldap.contact.name.userdn = myLDAP_UserDN

Je nach Konfiguration des LDAP-Servers, verwenden Sie einen der folgenden Werte für die Server-URL:

- **Standard LDAPs port**
  cmas-core-security, ldap.contact.name.providerurl = ldaps://dc2.mydomain.com:636
- **LDAPs port Global Catalogue**
  cmas-core-security, ldap.contact.name.providerurl = ldaps://dc2.mydomain.com:3269

Einrichten von Kundenkonten für LDAP

Sie müssen zwei Schritte ausführen, um LDAP-Authentifizierung für Kunden in CM/Track einzurichten:

- Setzen Sie die erforderliche Annotation für das Kundenfeld, in dem die LDAP-ID im Admin Tool gespeichert werden soll.
- Geben Sie die LDAP-IDs für die jeweiligen Kunden im Web Client ein.

Wenn der LDAP-Modus genutzt wird, muss das Kundenfeld, das für den CM/Track-Benutzernamen (Login) verwendet wird, zwei Annnotationen haben:

- `username` = true
- `ldapid` = true

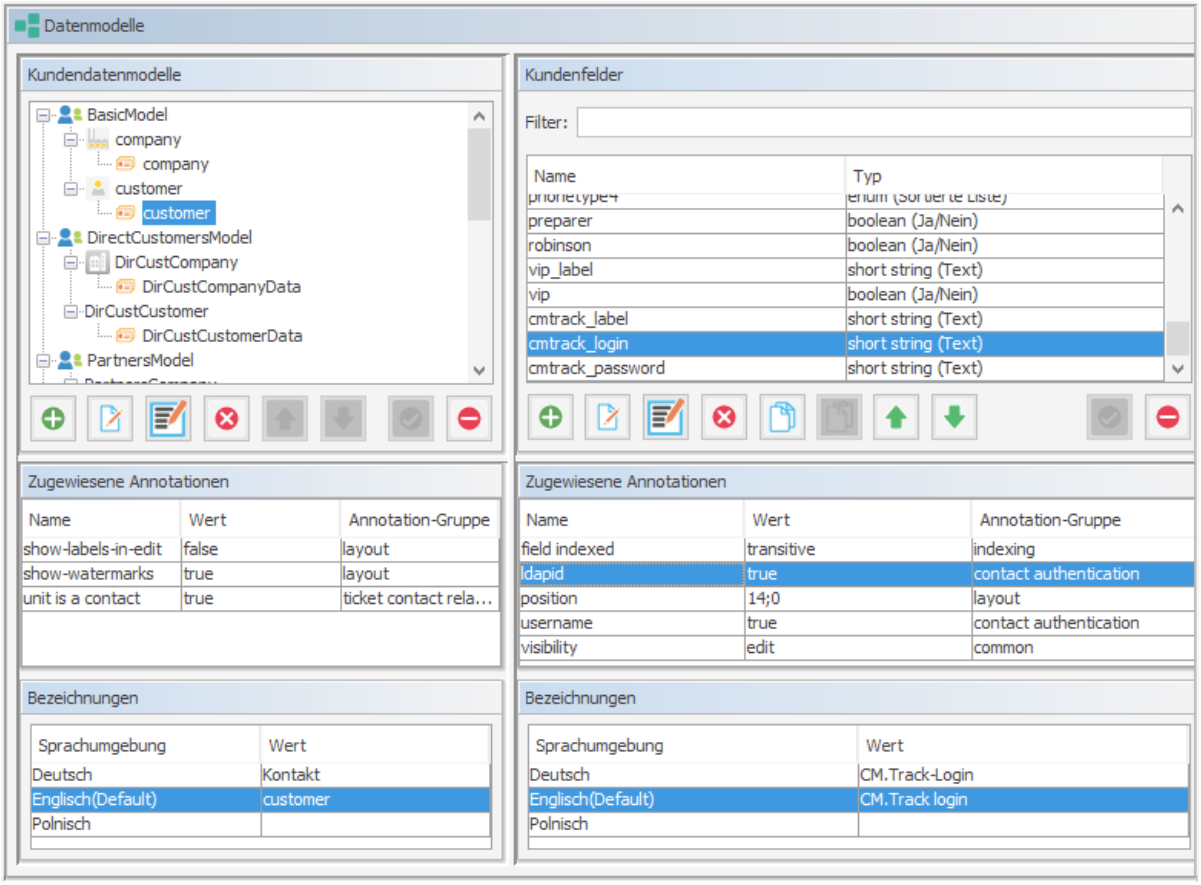


Abbildung 572: *ConSol CM Admin Tool - Kunden, Datenmodelle: Kundenfeld für die LDAP-Authentifizierung von CM/Track-Benutzern*

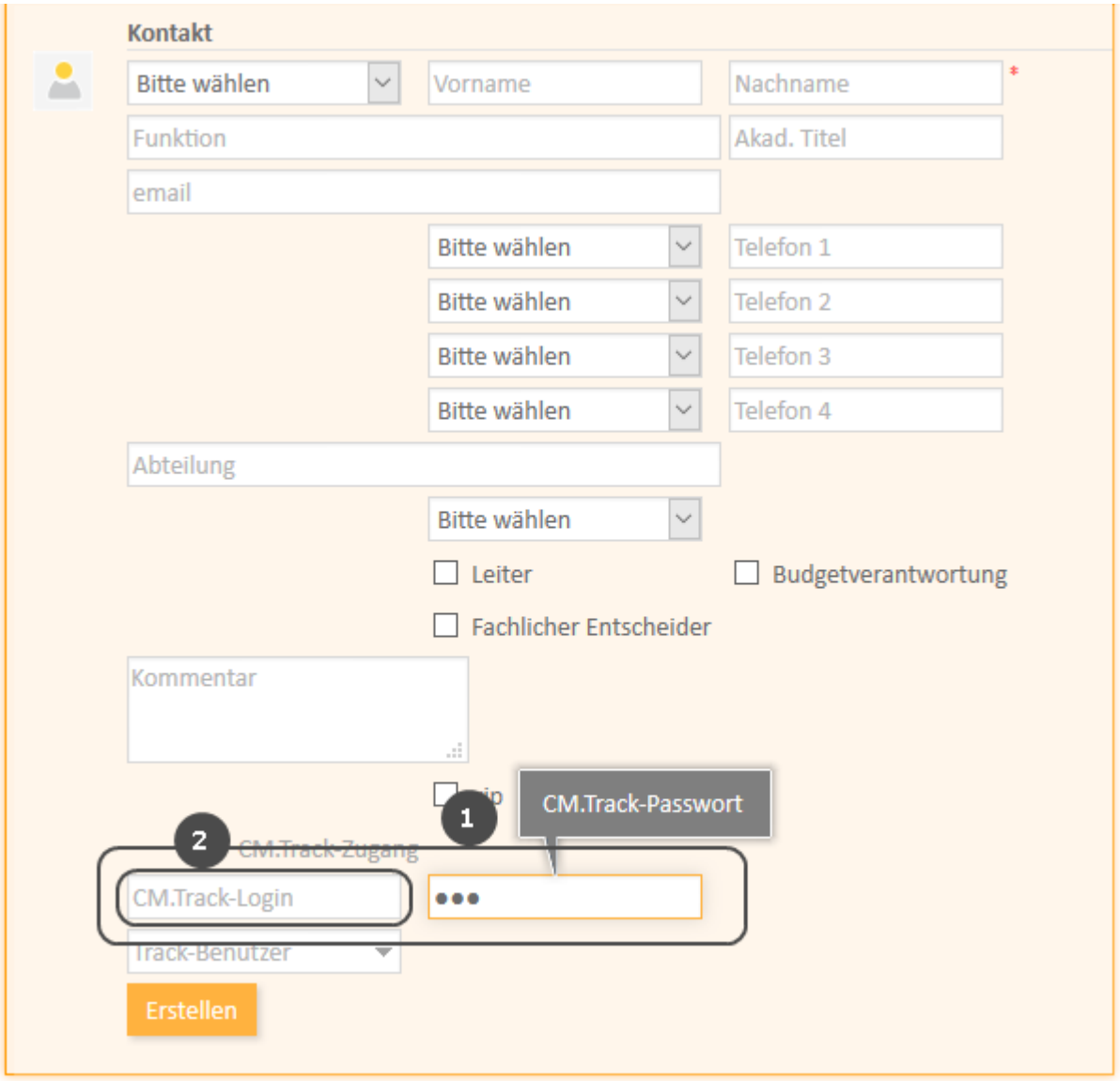


Abbildung 573: *ConSol CM Web Client - Feld für die LDAP-ID in den Kontaktdaten*

- Für Datenbankauthentifizierung im gemischten Modus verwendet (1)
- Für LDAP-Authentifizierung verwendet (2)

F.15 Externe Schnittstellen in ConSol CM

ConSol CM ist kein alleinstehendes System, sondern kann in die IT-Infrastruktur Ihres Unternehmens integriert werden. Eine Möglichkeit ist die Integration von CM mithilfe einer der externen Schnittstellen:

- **REST-API**
  - Diese wird für die Implementierung von CM/Track verwendet, siehe Abschnitt Das Kundenportal: CM/Track.
  - Wenn Sie eine eigene Applikation entwickeln möchten, die die REST-API verwendet, wenden Sie sich an Ihren ConSol CM-Consultant, der Ihnen die REST-API-Dokumentation zur Verfügung stellt.
- **Webhooks-Schnittstelle**
  - Diese bietet die Möglichkeit, CM einfach mit Applikationen wie Chat-Tools, Verkaufssystemen oder Social-Media-Plattformen zu integrieren.
  - Details dazu finden Sie im Abschnitt Webhooks.
- **ETL-Schnittstelle**
  - Dies ist die Schnittstelle für ETL-Tools (Extract- Transform - Load) . Diese Tools können verwendet werden, um eine große Anzahl an Objekten in die ConSol CM-Datenbank zu importieren, z. B. Kunden. Eine detaillierte Erklärung dazu finden Sie im *ConSol CM ETL Manual*.

F.15.1 Webhooks

F.15.1.1 Einleitung

ConSol CM Webhooks ermöglichen Ihnen eine leichte Integration von ConSol CM mit Applikationen von Drittanbietern, zum Beispiel Einkaufs-Applikationen, Social-Media-Plattformen oder Chat-Tools. Um mit diesen Tools zu kommunizieren, kann das CM-Webhooks-Modul mehrere Services bereitstellen, sodass Sie CM gleichzeitig mit mehreren Tools integrieren können.

Die folgenden Schritte bilden die Grundlage für die Übertragung von Informationen. Eine detaillierte Erklärung finden Sie im Abschnitt Hintergrundinformationen über ConSol CM Webhooks.

- Die externe Applikation sendet eine HTTP-Anfrage an ConSol CM, genauer gesagt, an den speziellen Service, der auf Webhooks-Technologie basiert.
- Die Anfrage wird vom CM-Webhooks-Modul verarbeitet. Die Funktionalitäten des speziellen Services werden in einem Admin-Tool-Skript implementiert.
- Die Antwort wird zurück an die anfragende Site gesendet, die diese dann verarbeiten muss.

F.15.1.2 Beispielhafter Anwendungsfall

Das folgende Beispiel zeigt, wie Sie einfach einen neuen Kontakt in ConSol CM erstellen können, indem Sie Kundendaten aus einer (fiktiven) Social-Media-Plattform abrufen.

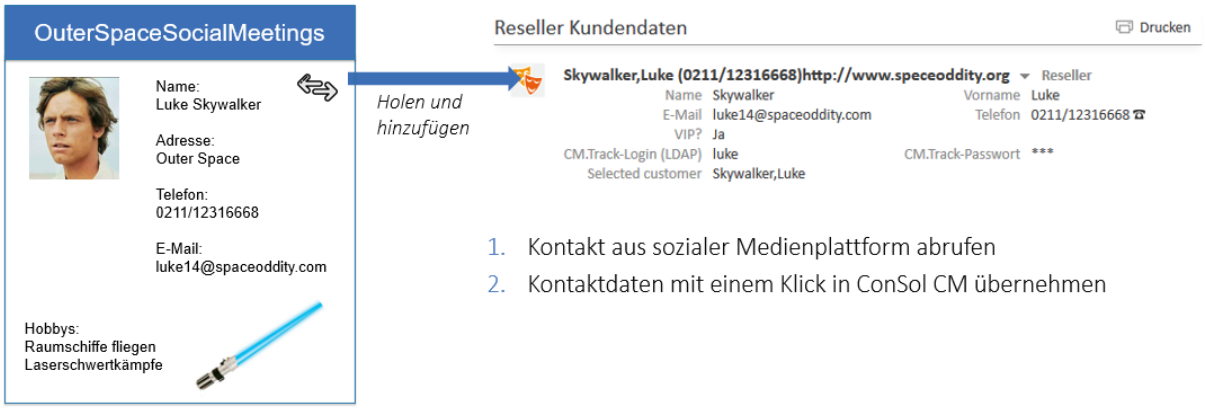


Abbildung 574: *Erstellen eines neuen ConSol CM-Kontakts mit dem Webhooks-Service, fiktives Beispiel einer Anwendung mit einer Social-Media-Plattform*

F.15.1.3 Hintergrundinformationen über ConSol CM Webhooks

Grundprinzip

Das Grundprinzip der Webhooks-Schnitstelle kann folgendermaßen zusammengefasst werden:

- Die Webhoooks-Schnittstelle kann eine beliebige Anzahl an Services bereitstellen. Die einzige Begrenzung ist die System-Performance.
- Ein Service wird in einem Admin-Tool-Skript des Typs *Integration* implementiert.

- Für jeden Service muss ein Security Provider definiert werden, der die Anfragen überprüft. Hintergrundinformationen zu diesem Thema finden Sie im Abschnitt Secret Token. Lesen Sie Abschnitt Schritt 2: Neuen Service, genauer gesagt, die Security Providers für den Service, konfigurieren, um herauszufinden, wie Sie den benötigten Security Provider konfigurieren können.
- Wenn der Service und (mindestens ein) Security Provider korrekt eingerichtet wurden, ist der neue Service über die entsprechende URL verfügbar:

```
http://[CM-Server-Adresse]/intg/<Admin-Tool-Skriptname>/service
```

- Wenn der Service korrekt eingerichtet wurde, können Anfragen verarbeitet werden:
  - Die anfragende Applikation sendet einen JSON-Ausdruck in einer HTTP-POST-Anfrage an die Service-URL. Der Content-Type muss „application/json“ sein.
  - Der Security Provider prüft die Anfrage. Wenn IP- und Token-Prüfung konfiguriert sind:
    - Zuerst wird die IP-Adresse überprüft. Wenn sie sich im zulässigen Bereich befindet, kann die Anfrage verarbeitet werden. Siehe Abschnitt Definition des gültigen IP-Adressbereichs.
    - Wenn sich die IP nicht im erforderlichen Bereich befindet, wird danach der Secret Token geprüft. Wenn er korrekt ist, kann die Anfrage ebenfalls verarbeitet werden. Siehe Abschnitt Secret Token.
    - Wenn keines der beiden Kriterien erfüllt ist, wird die Anfrage abgewiesen und es wird eine Meldung „authorization failed“ angezeigt.
  - Wenn die Sicherheitskriterien erfüllt sind, kann die Anfrage weiterbearbeitet werden. Sie wird dann vom Admin-Tool-Skript, in dem der spezielle Service implementiert ist, verarbeitet.
  - Die Antwort wird als Objekt der Klasse `IntgServiceResponse` zurück an die anfragende Applikation gesendet.

    ⚠ Beachten Sie die Änderung zwischen CM-Version 6.11.1.0 und 6.11.1.1:
    In CM 6.11.1.0 gibt das Webhook-Integrationsskript einen JSON-Ausdruck zurück. In den CM-Versionen 6.11.1.1 und höher gibt das Webhook-Integrationsskripts ein Objekt der Klasse `IntgServiceResponse` zurück. Dieses Objekt kann im Meldungstext einen JSON-Ausdruck enthalten.

  - Die anfragende Applikation muss die Antwort entsprechend verarbeiten.

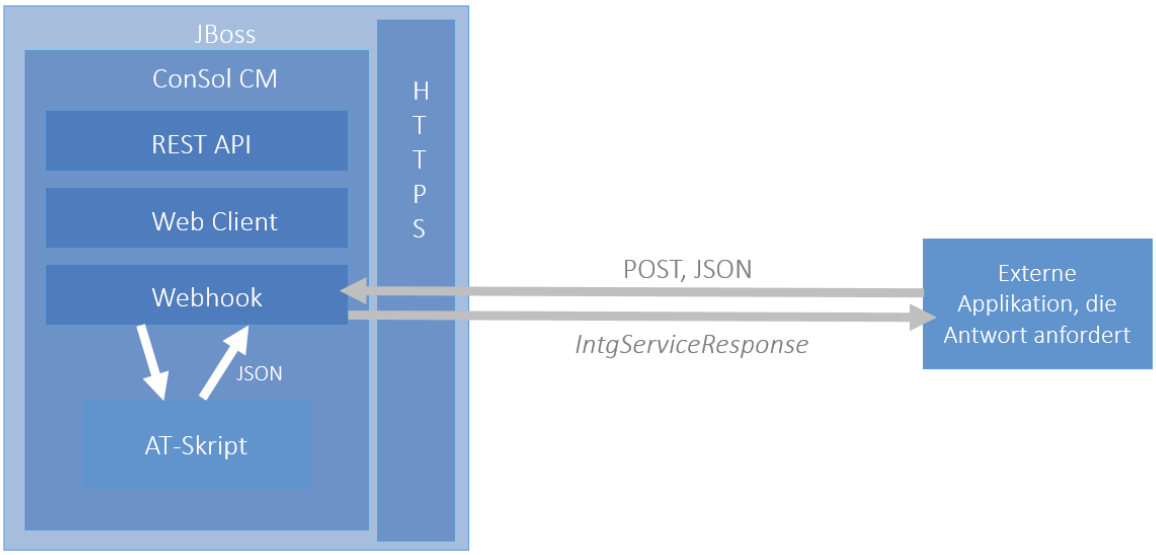


Abbildung 575: *Grundprinzip vom ConSol CM Webhooks*

Die Antwort kann benutzerdefinierte Objekte der folgenden Typen enthalten:

- httpStatusCode
- httpHeaders
- body

Details dazu finden Sie im folgenden Abschnitt.

Admin-Tool-Skript des Typs *Integration*

Das im Admin Tool definierte Skript muss den Typ *Integration* haben. Im Skript wird die Logik für die Verarbeitung der gesendeten Anfragen implementiert. Der Skriptname ohne die Erweiterung „.groovy“ ist zudem der Servicename für die URL für den Zugriff auf die Schnittstelle. Für die URL `http://10.20.3.40:8080/intg/myWebHook/service` muss zum Beispiel ein Skript des Typs *Integration* mit dem Namen „myWebHook.groovy“ im Admin Tool vorhanden sein, das die Logik der Schnittstelle implementiert. Wenn das Skript zur Verarbeitung einer Anfrage ausgeführt wird, wird es im Administratorkontext ausgeführt und hat vollen Systemzugriff.

Ein Skript des Typs *Integration* gibt ein Objekt des Typs `IntgServiceResponse` zurück. Wenn Sie ein neues Admin-Tool-Skript dieses Typs erstellen, wird der Editor mit einem einfachen Skript-Template gefüllt, das den grundlegenden Code zum Zurückgeben von `IntgServiceResponse` enthält.

`IntgServiceResponse` kann folgende Objekte enthalten:

- **httpStatusCode**
  Definiert den HTTP-Statuscode. Optional, wenn nicht angegeben, werden die Standard-Statuscodes verwendet (siehe Abschnitt unten).

- **httpHeaders**
  Optional, definiert den Antwort-Header. Nur erforderlich, wenn Sie einen speziellen httpHeader definieren möchten, d. h. einen Header, der nicht zu den unterstützten Standard-Headern gehört (siehe Abschnitt unten).
- **body**
  Definiert den Inhalt der Antwort. Optional, wenn nicht angegeben, ist die Antwort leer. Dies ist das Objekt, das einen JSON-Ausdruck enthalten kann, siehe folgende Codebeispiele.

Die folgenden Standard-HTTP-Statuscodes werden unterstützt:

- **200 (OK)**
- **401 (Unauthorized)**
  Es sind ein oder mehrere Security-Provider aktiviert, aber die vom Client angegebenen Daten entsprechen nicht den erwarteten Daten (z. B. falsches Token oder IP-Adresse).
- **404 (Not Found)**
  Es wurde kein Skript für den Webhook mit dem angegebenen Namen gefunden / der Webhook ist deaktiviert / es ist kein Security Provider für den Webhook aktiviert.
- **405 (Method Not Allowed)**
  Eine andere HTTP-Methode als POST
- **500 (Internal Server Error)**
  Probleme/Exceptions auf der Serverseite (CM)

Das Beispielskript unten (Abschnitt Schritt 1: Admin-Tool-Skript schreiben) gibt ein einfaches JSON-Objekt mit Ticketnummer, -betreff und -bearbeiter als Antwort auf eine Anfrage mit der Ticketnummer zurück. Wenn die Ticketnummer ungültig ist oder kein Bearbeiter zugewiesen ist, gibt es „n/a“ zurück. Bevor die Antwort zurückgegeben wird, werden die Anfrage und der Inhalt der Antwort in die Log-Datei geschrieben.

Definition des gültigen IP-Adressbereichs

Der IP-basierte Filter verwendet, sofern er durch Markieren der Checkbox aktiviert ist, eine Liste von in der CIDR-Notation (Classless Inter-Domain Routing) definierten IP-Adressbereichen. Ein gültiger Eintrag in dieser Notation ist eine IP-Adresse gefolgt von einem Schrägstrich, der die Adresse von der Anzahl der Bits, die die Subnetzmaske definieren, trennt. Das Beispiel „10.20.30.40/24“ steht für die IP-Adresse „10.20.30.40“ und die Subnetzmaske „255.255.255.0“. Die Adressliste für die Servicekonfiguration lässt mehrere IP-Bereiche in dieser Notation zu, wobei jeder Bereich in einer eigenen Zeile steht.

Secret Token

Der Schnittstellenzugriff über Secret Token muss durch Markieren der Checkbox *Enabled* aktiviert werden. Jedes Token, das verwendet werden soll, muss als String in einer eigenen Zeile in die Liste *Tokens* im letzten Eingabefeld auf der Seite eingefügt werden. Die Checkbox *HMAC* aktiviert die Validierung der Anfrage durch einen obligatorischen Hash, der auf dem Token basiert. Der HMAC-Wert (Hash-based Message Authentication Code) wird aus dem unverschlüsselten Payload und dem Shared Secret abgeleitet. Dieser Code wird im Header der Anfrage übermittelt. Der einzige von ConSol CM für diesen Fall unterstützte Hashing-Algorithmus ist SHA-1; MD5 wird nicht unterstützt.

Ein entsprechendes Hashing der Antwort wird aktuell nicht unterstützt.

Das Eingabefeld *Header* ermöglicht die Definition des HTTP-Header-Feldnamens, der für die Übermittlung des gemeinsamen Shared Secret verwendet werden soll. Der Standardwert ist „SECRET_TOKEN“. Es wird allerdings empfohlen, diesen Wert, wenn möglich, zu ändern. Ein Name, der nicht der Standardname ist, kann die Sicherheit verbessern.

Beachten Sie, dass es für eine Webhook-Integration keine weitere obligatorische Zugriffsprüfung gibt, die über die Security Providers hinausgeht. Es gibt insbesondere keine auf einer Anmeldung basierende Authentifizierung wie für Bearbeiter, die zusätzliche Zugriffsbegrenzungen erzwingt. Die Schnittstelle hat vollen Systemzugriff, sodass eine zusätzliche Absicherung über einen Proxy für die Verwendung in der Praxis verpflichtend ist.

Wenn Shared Secret als Security Provider verwendet wird, ist das Token für das Skript über die Headers-Map in einem Feld verfügbar, das so heißt, wie es in der Konfigurations-Webschnittstelle unter *Headers* definiert ist.

Sofern der Standardwert „SECRET_TOKEN“ verwendet wird, ist er im Skript aus diesem HTTP-Header-Feld verfügbar.

Falls gewünscht, kann er auch explizit in der Antwort zurückgegeben werden, siehe Code-Beispiel.

```
...
token = headers.get('secret_token')
...
JsonOutput.toJson([ticketId: ticket.getId(), token: headers.get('secret_token')])

return response
```

Beachten Sie, dass das Verhalten des Dienstes ganz von der Implementierung des entsprechenden Skripts abhängt, d. h. die auf den angegebenen Token basierende Sicherheit muss implementiert werden. Sie ist nicht systemimmanent.

Konfiguration von Webhook-Services

Bevor der neue Service verwendet werden kann, muss er konfiguriert werden. Dies muss auf der Seite *Configuration*, die unter folgender URL zur Verfügung steht, erfolgen:

```
http://[CM-Server-Adresse]/intg/<Admin-Tool-Skriptname>/config
```

Melden Sie sich mit einem Administratorkonto an (ein ConSol CM-Bearbeiter mit Administratorberechtigungen).

Nach der erfolgreichen Anmeldung, wird die Konfigurationsseite mit einem Formular zum Eingeben der relevanten Informationen angezeigt. Ein Beispiel für eine Konfigurationsseite finden Sie im Abschnitt Schritt 2: Neuen Service, genauer gesagt, die Security Providers für den Service, konfigurieren.

Folgende Parameter können konfiguriert werden:

- Allgemeine Informationen:
  - Anzeige der relativen URL für den Zugriff auf den Webhooks-Service
  - Checkbox zum allgemeinen Aktivieren/Deaktivieren des Services. Muss markiert sein, damit die Schnittstelle funktioniert. Beachten Sie, dass mindestens ein Security Provider konfiguriert sein muss, damit der Service aktiviert wird. Die Checkbox alleine aktiviert ihn nicht!
- Konfiguration des Security Providers:
  - IP-basierte Filter zum Begrenzen des Zugriffs auf definierte IP-Adressbereiche:
    - Checkbox zum Aktivieren/Deaktivieren des IP-basierten Filters. Muss markiert sein, um die auf der IP-Adresse basierende Zugriffskontrolle zu aktivieren.
    - Liste mit IP-Adressbereichen in CIDR-Notation (siehe unten)
- Konfiguration von Secret Token, um ein Shared Secret für den Zugriff anzufordern:
  - Checkbox *Enabled* zum Aktivieren/Deaktivieren des Zugriffs über Token. Muss für den Zugriff über Token markiert sein.
  - Checkbox für HMAC zur Verwendung eines sicheren Hash für die Authentifizierung der Nachricht (siehe unten)
  - HTML-Header-Name für das Shared Secret, standardmäßig „SECRET_TOKEN“
  - Liste mit Zugriffs-Token

Berechtigungen

Das Webhook-Skript wird mit Administratorberechtigungen ausgeführt. Dies bedeutet, dass auch alle im Skript durchgeführten Operationen mit diesen Berechtigungen ausgeführt werden. Falls dies nicht erwünscht ist, kann das Skript die Anmeldeinformationen eines Bearbeiters und Code enthalten, um die Session auf die Berechtigungen des Bearbeiters herunterzustufen. Dafür wird die Methode `executeWithUserPermissions` der Klasse `SecurityTemplate` verwendet.

Das folgende Code-Beispiel zeigt die Verwendung dieser Methode in einem Skript. Die Session wird auf die Berechtigungen des Benutzers zurückgestuft, der über das im JSON-Payload enthaltene Login und Passwort identifiziert wird. Dann wird mit diesen Berechtigungen eine Kundensuche nach dem Kunden mit der ID „12345“ durchgeführt.

```
import com.consol.cmas.common.security.template.SecurityCallbackWithoutResult;
import com.consol.cmas.common.security.template.SecurityTemplate;
import groovy.json.JsonSlurper

def jsonSlurper = new JsonSlurper()
def message = jsonSlurper.parseText(payload);
SecurityTemplate.executeWithUserPermissions(message.login, message.password, new
 SecurityCallbackWithoutResult() {
   @Override
   public void doInSecurityContextWithoutResult() {
      unitService.getById(12345)
   }
});
```

Code-Beispiel 103: *Beispiel für das Zurückstufen der Session auf die Berechtigungen eines Bearbeiters*

F.15.1.4 Beispiel einer Service-Implementierung auf Basis von CM Webhooks

Das folgende Beispiel zeigt die Implementierung eines Services mit dem Namen „myWebHook“.

Schritt 1: Admin-Tool-Skript schreiben

Das Skript mit dem Namen „myWebHook.groovy“ wird im Bereich *Skripte* des Admin Tools gespeichert.

```
////////////////////////////////////////////////////////////////////
//// Variables available in script: ////
//// headers - headers as map ////
//// payload - request payload as json ////
//// Script should return json as result ////
////////////////////////////////////////////////////////////////////
import com.consol.cmas.intg.service.IntgServiceResponse
import groovy.json.JsonSlurper
import groovy.json.JsonOutput
import com.consol.cmas.common.service.TicketService
def jsonSlurper = new JsonSlurper()
def message = jsonSlurper.parseText(payload)
def ticketname
def ticketsubject
def engineername
log.info 'PAYLOAD is NOW ' + payload
def OK = false
if (message.ticket) {
   ticket = ticketService.getByName(message.ticket)
   if (ticket) {
      ticketname = message.ticket
      log.info 'TICKETNAME is now ' + ticketname
      ticketsubject = ticket?.getSubject()
      if (ticketsubject) {
         log.info 'TICKET SUBJECT is now ' + ticketsubject
      } else {
         log.info 'NO SUBJECT'
      }
      if (ticket.engineer) {
         engineername = ticket.engineer.getFirstname() + " " +
          ticket.engineer.getLastname()
      }
      OK = true
   }
}
def response = new IntgServiceResponse()
if (OK) {
   log.info "Webhook [Search engineer for ticket] returned: {ticket: " + ticketname
    + ", subject: " + ticketsubject + ", engineer: " + engineername + "} on request
    [" + message + "]"
   response.httpStatusCode = 200
   response.body = JsonOutput.toJson([ticket: ticketname, subject: ticketsubject,
    engineer: engineername]) // optional
} else {
   log.info 'No hits found in webhook request ... '
   response.httpStatusCode = 404
   response.httpHeaders = ['Content-Language':'en', 'Warning':'Required Objects not
    found']
}
return response
```

Code-Beispiel 104: *Admin-Tool-Skript, das einen Webhooks-Service implementiert (mit einigen log.-info-Anweisungen zu Testzwecken)*

Sobald das Skript implementiert ist, kann die Konfiguration des neuen Services erfolgen. Siehe folgenden Abschnitt.

Schritt 2: Neuen Service, genauer gesagt, die Security Providers für den Service, konfigurieren

Konfigurieren Sie den Service über folgende URL:

```
http://[CM-Server-Adresse]/intg/<Admin-Tool-Skriptname>/config
```

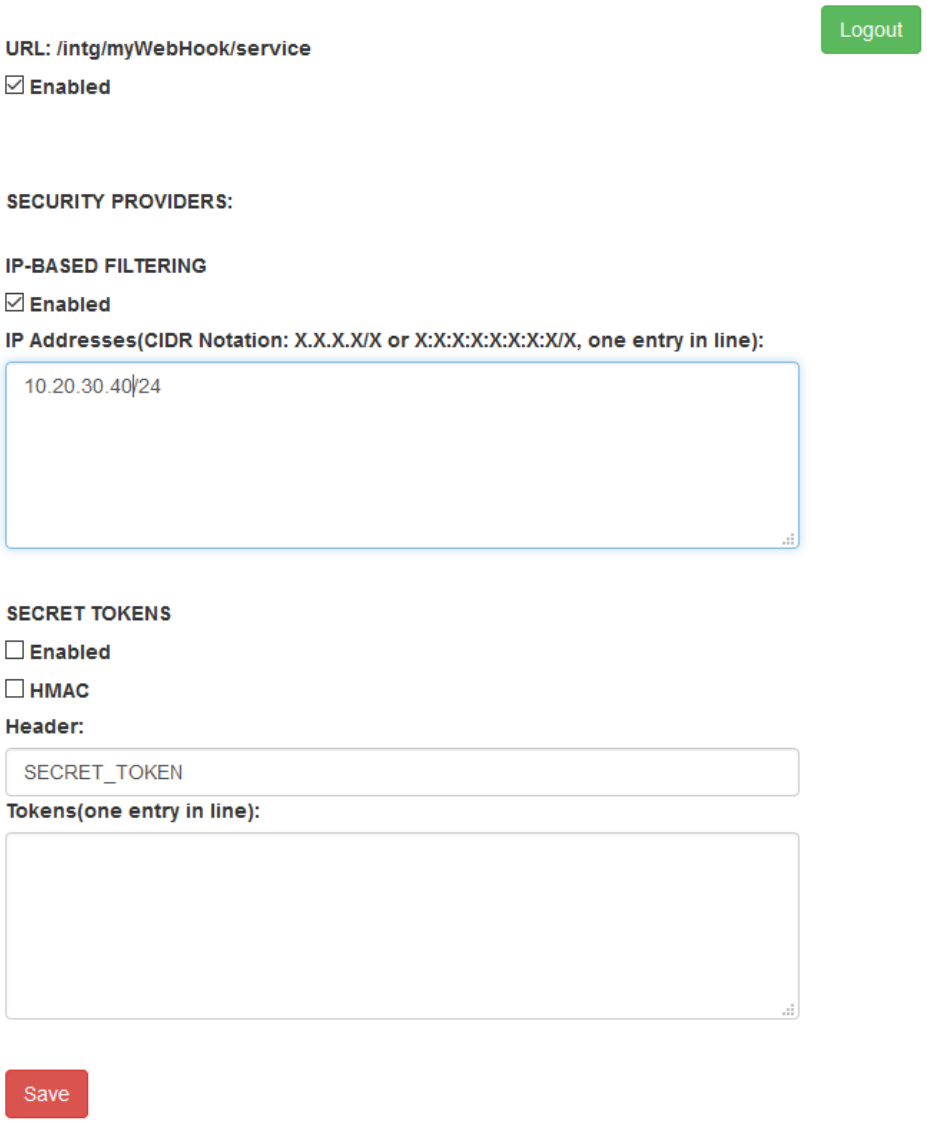


Abbildung 576: *Konfigurationsseite eines auf Webhooks basierenden Services*

In diesem Beispiel wird nur der IP-Bereich geprüft.

Schritt 3: Neuen Service testen

Wenn ein Rechner mit einer IP-Adresse innerhalb des gültigen Bereichs eine Anfrage an den neuen Service sendet, ist die Antwort eine einfache JSON-Ausgabe. Sie können zum Beispiel ein REST-Plugin im Webbrowser verwenden, um den Service zu testen. Der *content-type* muss „application/JSON“ sein.

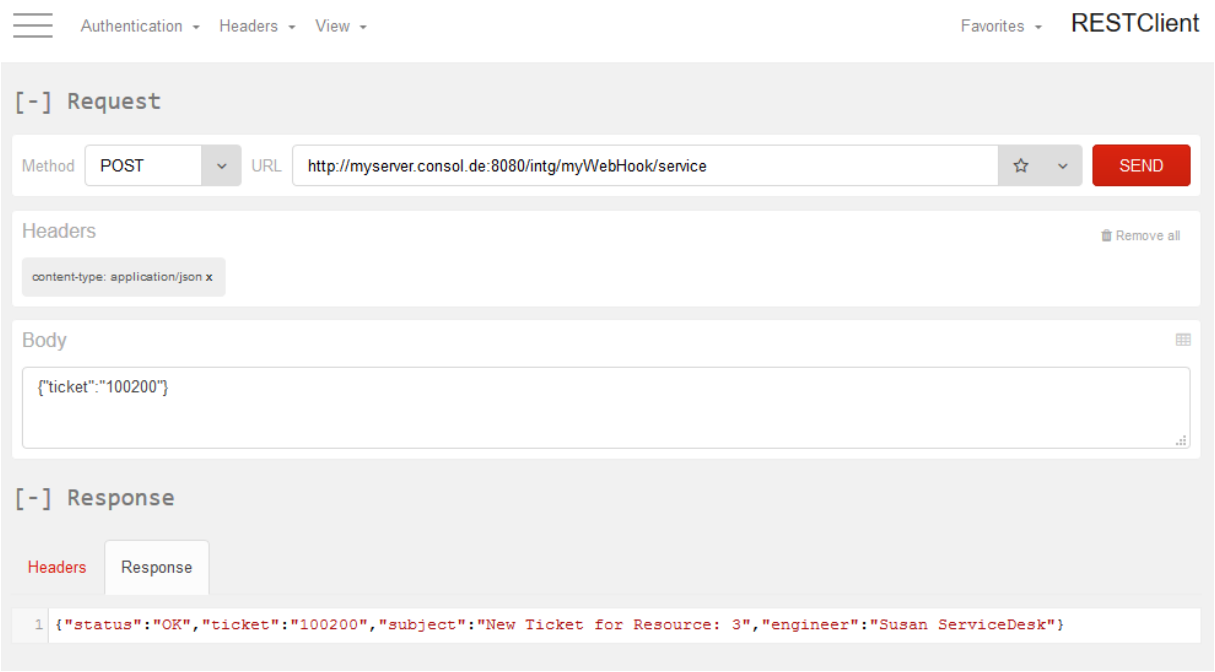


Abbildung 577: *Testen des Webhooks-Services myWebHook mit einem REST-Plugin im Webbrowser, Ticket wurde im System gefunden*

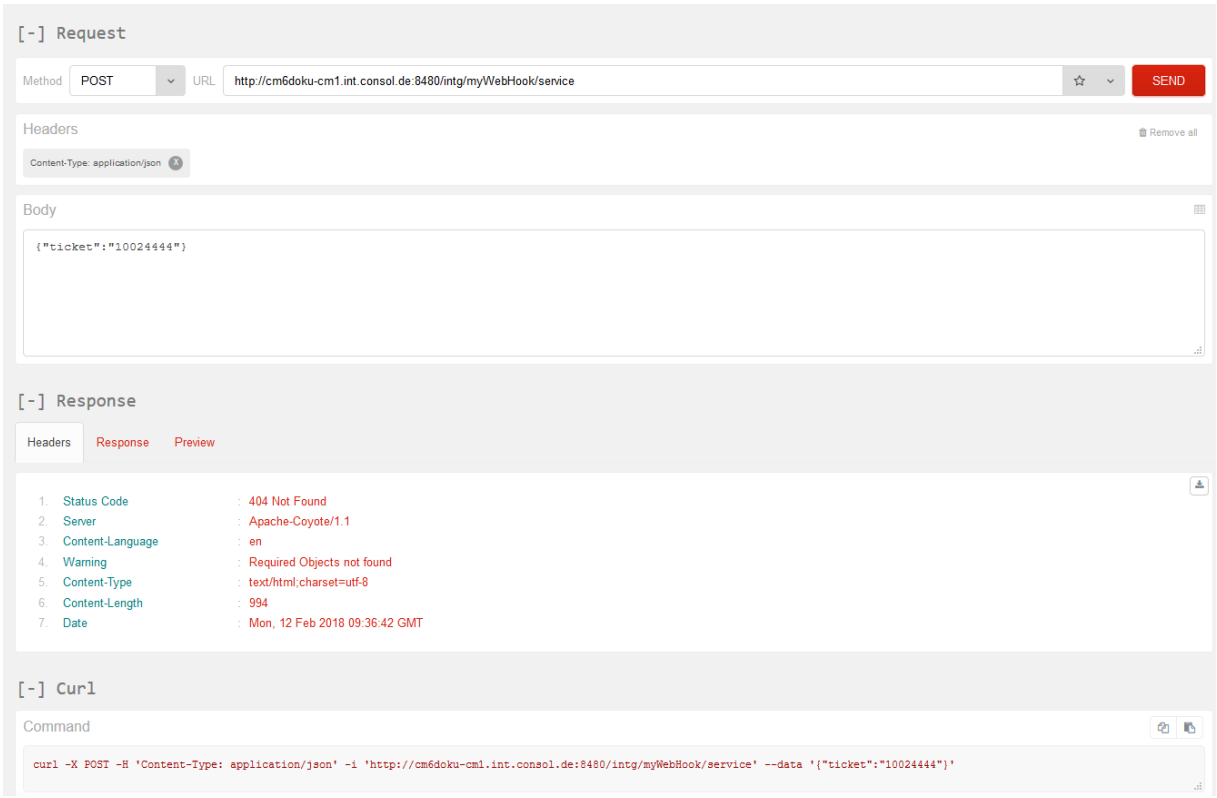


Abbildung 578: *Test des Webhooks-Services myWebHook über ein REST-Plugin im Webbrowser, Ticket wurde nicht gefunden, es wird ein benutzerdefinierter Fehlercode zurückgegeben*

Sie können auch ein Kommandozeilen-Tool verwenden:

```
# Linux:
curl -X POST http://myserver.consol.de:8080/intg/myWebHook/service //
-H 'content-type: application/json' //
-d '{"ticket": "100200"}'

# Windows (note the different quoting/escaping for the -d option):
curl -X POST http://myserver.consol.de:8080/intg/myWebHook/service //
-H "content-type: application/json" //
-d "{\"ticket\": \"100200\"}"
```

Der Payload der Anfrage in diesem Beispiel ist der Wert der Option `-d`.

Die vom Skript erzeugte und an den Anfragenden zurückgegebene Antwort ist ein bisschen ausführlicher. (Beachten Sie, dass das JSON zur besseren Lesbarkeit formatiert wurde.)

```
{
   "status" : "OK" ,
   "ticket" : "100200" ,
   "subject" : "New Ticket for Resource 3" ,
   "engineer" : "Susan ServiceDesk"
}
```

F.16 Systemarchitektur

ConSol CM ist eine Java EE-Applikation. Sie kann auf zwei Arten betrieben werden:

- **Nur ConSol CM-System**, siehe Abschnitt Grundlegende Systemarchitektur

oder

- **ConSol CM-System mit Reporting-Infrastruktur**, siehe Systemarchitektur mit Reporting-Infrastruktur

Weiterhin kann ConSol CM auch in einem Application Server Cluster installiert und betrieben werden. Dies ist im *ConSol CM Cluster Manual* detailliert beschrieben.

F.16.1 Architektur eines CM-Systems

F.16.1.1 Einführung in die Systemarchitektur von ConSol CM

ConSol CM ist eine Java EE-Applikation (Java Enterprise Edition), die auf einem Standard-Application-Server auf Unix/Linux- oder Windows-Systemen ausgeführt werden kann. Als Application Server werden JBoss und Oracle WebLogic unterstützt.

Dieses Kapitel enthält einen kurzen Überblick über die ConSol CM-Systemarchitektur.



Eine detaillierte Liste der unterstützten Betriebssysteme, Application Server, Datenbanksysteme und anderen Systeme, sowie der Anforderungen an CPU und Speicherplatz ist in den aktuellen *Systemanforderungen* enthalten.

F.16.1.2 Grundlegende Systemarchitektur

ConSol CM ist eine Java EE-Applikation, die auf einer klassischen dreistufigen (**three-tier**) Architektur basiert. Der ConSol CM-Server wird auf einem Application Server installiert und greift auf eine relationale Datenbank zu. Es stehen zwei Weboberflächen als Clients zur Verfügung: Die Standardoberfläche ist der ConSol CM Web Client, der von den Bearbeitern für die Arbeit mit Tickets verwendet wird. Der andere Web-Client ist das ConSol CM-Portal, CM/Track. Damit erhalten Kunden, die grundlegende Daten zum Status ihrer Tickets wissen möchten, Zugang zum System. Die beiden Java-Applikationen, mit denen ConSol CM konfiguriert wird, sind das Admin Tool und der Process Designer. Diese beiden Applikationen können von der ConSol CM-Startseite mit Java Web Start (JWS) heruntergeladen werden. JWS ist Bestandteil jeder aktuellen Java-Edition, sodass keine zusätzliche Installation auf den PCs oder Laptops, die Sie zum Verwalten des Systems verwenden möchten, erforderlich ist. Im Gegenteil, Sie können dies von jedem regulären Web-Client mit einem unterstützten Webbrowser tun. Stellen Sie sicher, dass die in Ihrer Firma verwendeten Versionen aller Komponenten die Systemvoraussetzungen erfüllen.

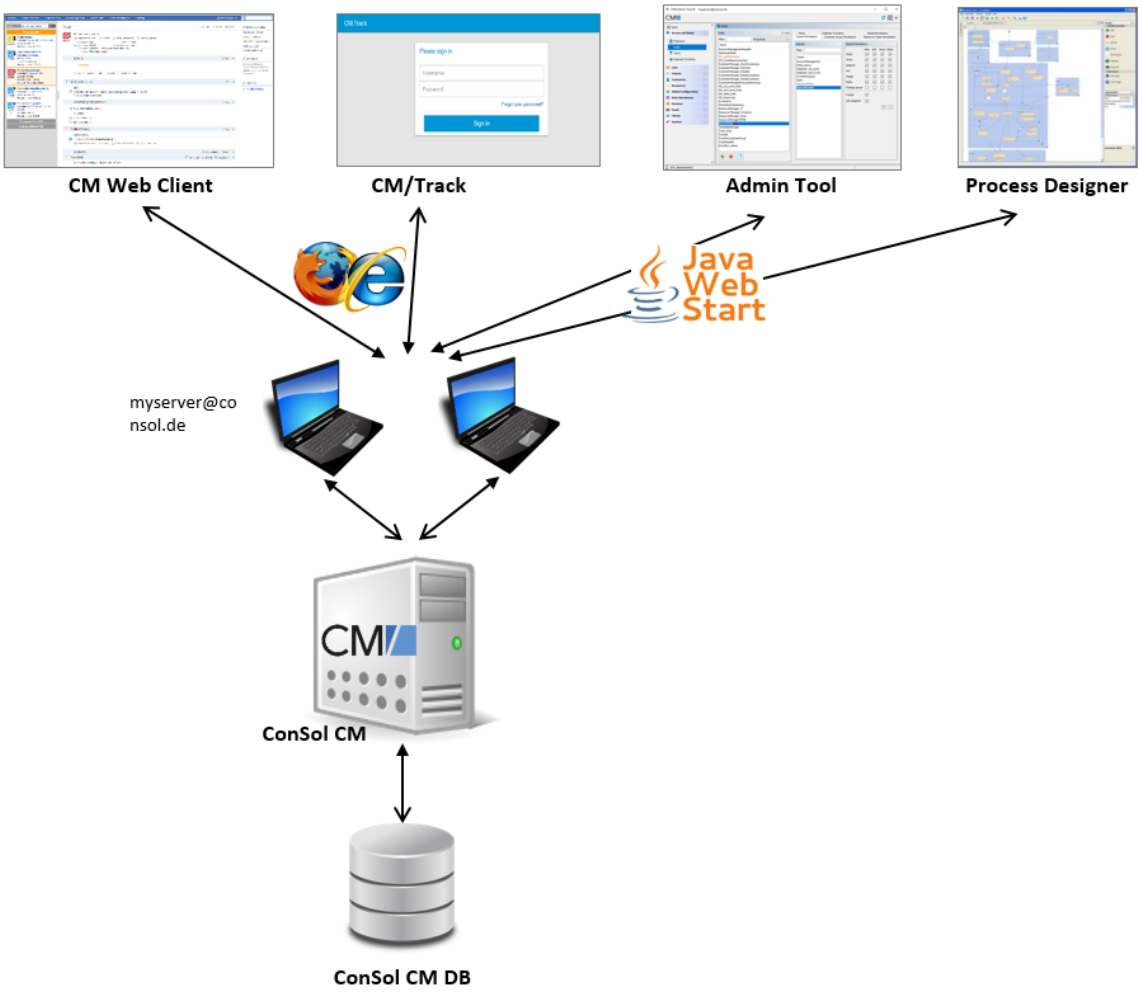


Abbildung 579: *ConSol CM - Grundlegende Systemarchitektur*

CM-Datenbank

Die ConSol CM-Datenbank ist eine relationale Datenbank, die als Oracle-, Microsoft SQL Server- oder MySQL-System betrieben werden kann.

> Eine detaillierte Liste der unterstützten Betriebssysteme, Application Server, Datenbanksysteme und anderen Systeme, sowie der Anforderungen an CPU und Speicherplatz ist in den aktuellen *Systemanforderungen* enthalten.

- **Oracle**
  ConSol CM verwendet ein Datenbankschema mit einem Datenbankbenutzer.
- **Microsoft SQL**
  ConSol CM verwendet ein Datenbankschema mit einem Datenbankbenutzer.
- **MySQL**
  ConSol CM verwendet eine Datenbank mit einem Datenbankbenutzer.

F.16.1.3 Systemarchitektur mit Reporting-Infrastruktur

Eine der Standardkomponenten von ConSol CM ist ein Data Warehouse (DWH), das es Business Intelligence (BI) Tools oder anderen Applikationen ermöglicht, spezielle Reports, OLAP-Cubes und andere Analysen zu erstellen. Das DWH ist eine eigene Datenbank (oder ein eigenes Datenbankschema, siehe unten). Das DWH wird von der Java EE-Applikation ConSol CM Reporting Framework (CMRF) befüllt.

Der Standardfunktionsumfang von ConSol CM umfasst daher zwei Komponenten, die das Reporting ermöglichen:

- **CMRF** (ConSol CM Reporting Framework)

  Dies ist eine Java EE-Applikation, die die ConSol CM-Datenbank mit dem ConSol CM Data Warehouse (DWH) synchronisiert. Das CMRF kann auf demselben Application Server installiert werden wie ConSol CM oder auf einem separaten Application Server laufen. Wir empfehlen die Verwendung von zwei Application Servern, wenn Sie mit JBoss-Systemen arbeiten, und die Verwendung von einem Application Server, wenn Sie mit Oracle WebLogic arbeiten. Die Synchronisierung der CM-Daten mit dem DWH kann auf JMS-Queues (Java Messaging Service) oder Direct Messaging basieren. Eine detaillierte Erklärung dazu finden Sie im *ConSol CM Betriebshandbuch*.

- **DWH** (Data Warehouse)

  Das ConSol CM-DWH ist eine relationale Datenbank, die als Oracle-, Microsoft SQL Server- oder MySQL-System betrieben werden kann. Es speichert die integrierten/vorverarbeiteten Daten aus der ConSol CM-Datenbank.

> Eine detaillierte Liste der unterstützten Betriebssysteme, Application Server, Datenbanksysteme und anderen Systeme, sowie der Anforderungen an CPU und Speicherplatz ist in den aktuellen *Systemanforderungen* enthalten.

Separate Application Server für ConSol CM und CMRF (Standalone-Modus):

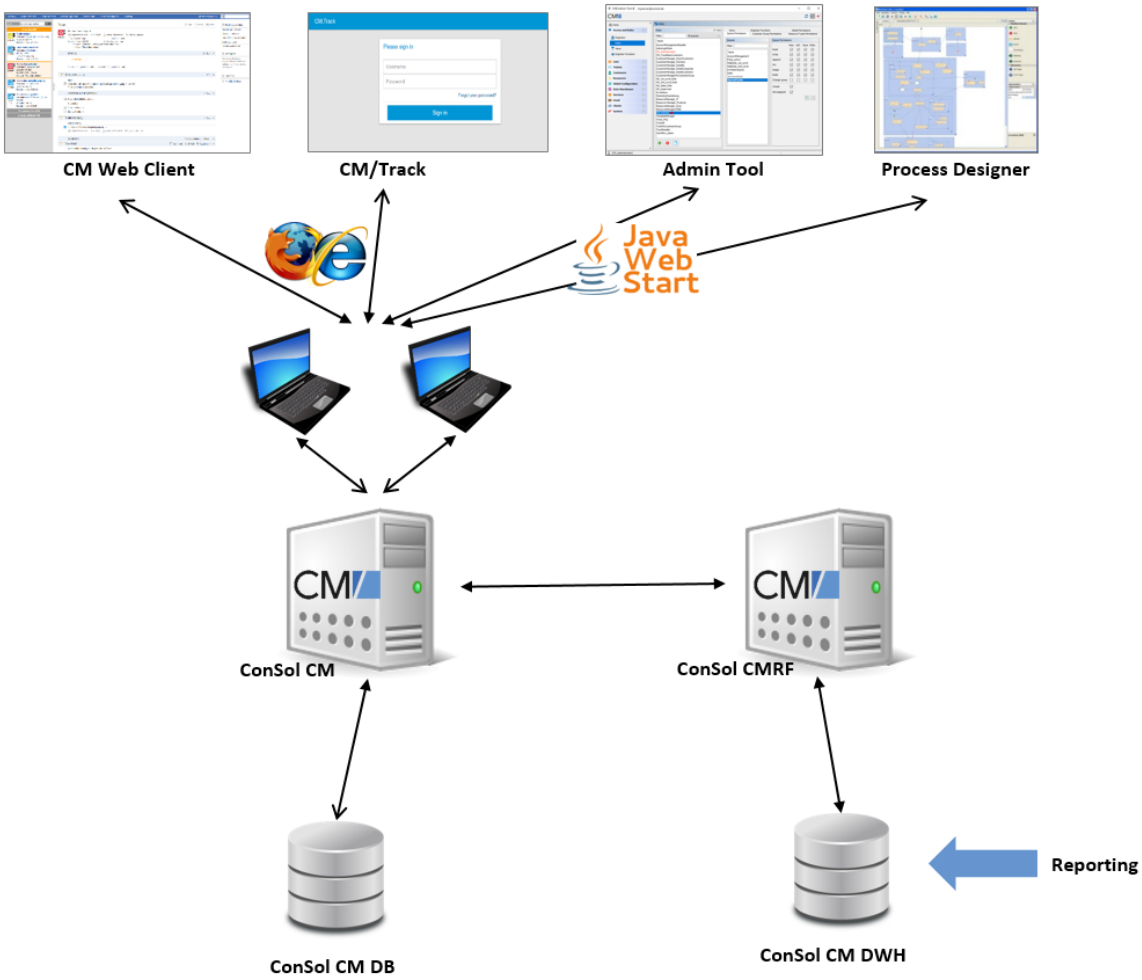


Abbildung 580: *ConSol CM - Infrastruktur mit CMRF und DWH (2 Server)*

Ein Application Server für ConSol CM und CMRF (Overlay-Modus):

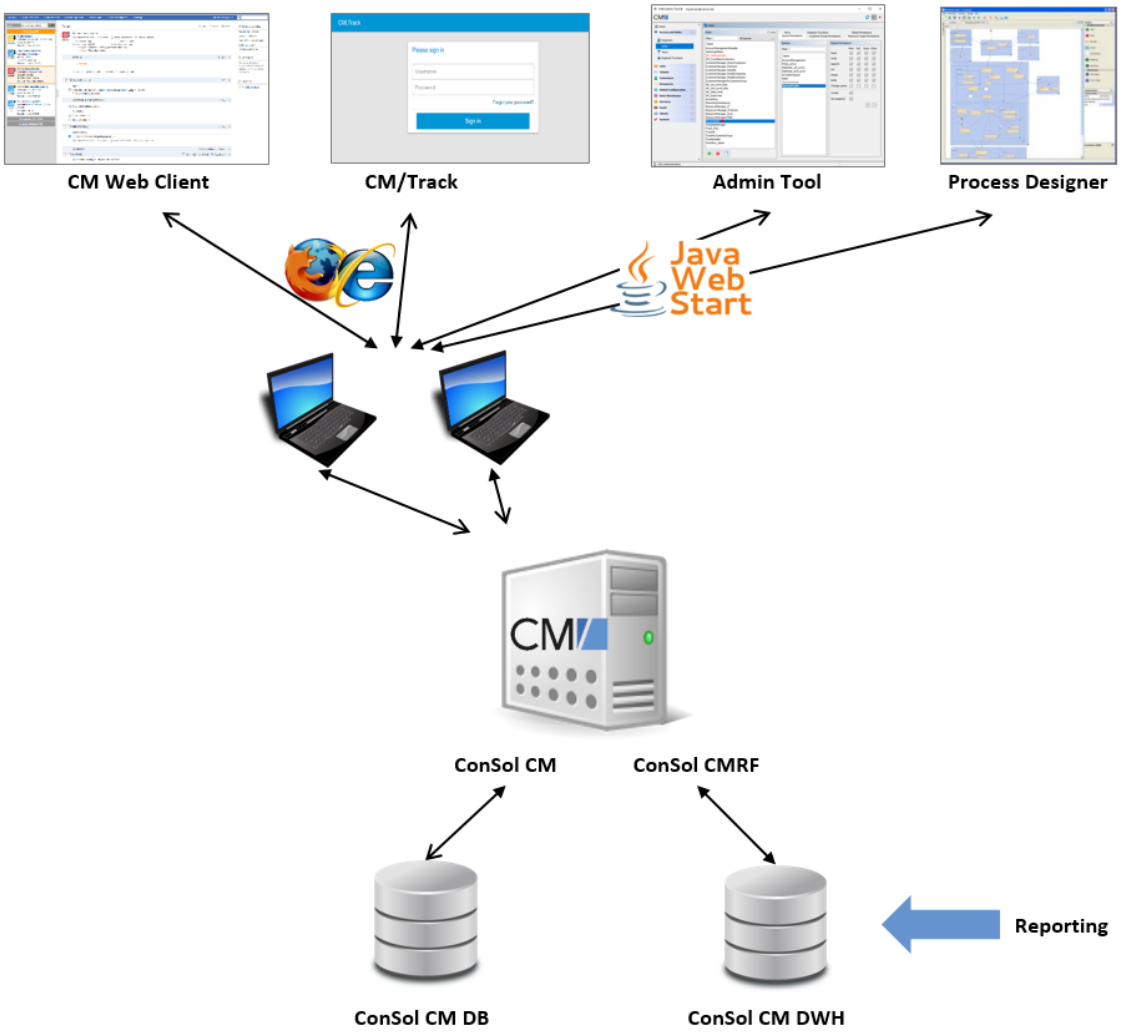


Abbildung 581: *ConSol CM - Infrastruktur mit CMRF und DWH (1 Server)*

Wenn das DWH eingerichtet wurde, können mithilfe von BI-Applikationen Reports, Daten-Cubes und andere Reporting-Ausgabeformate erstellt werden. Im folgenden Beispiel wird Pentaho™ BI Suite verwendet.

Separate Application Server für ConSol CM und CMRF (Standalone-Modus, Beispiel beide JBoss):

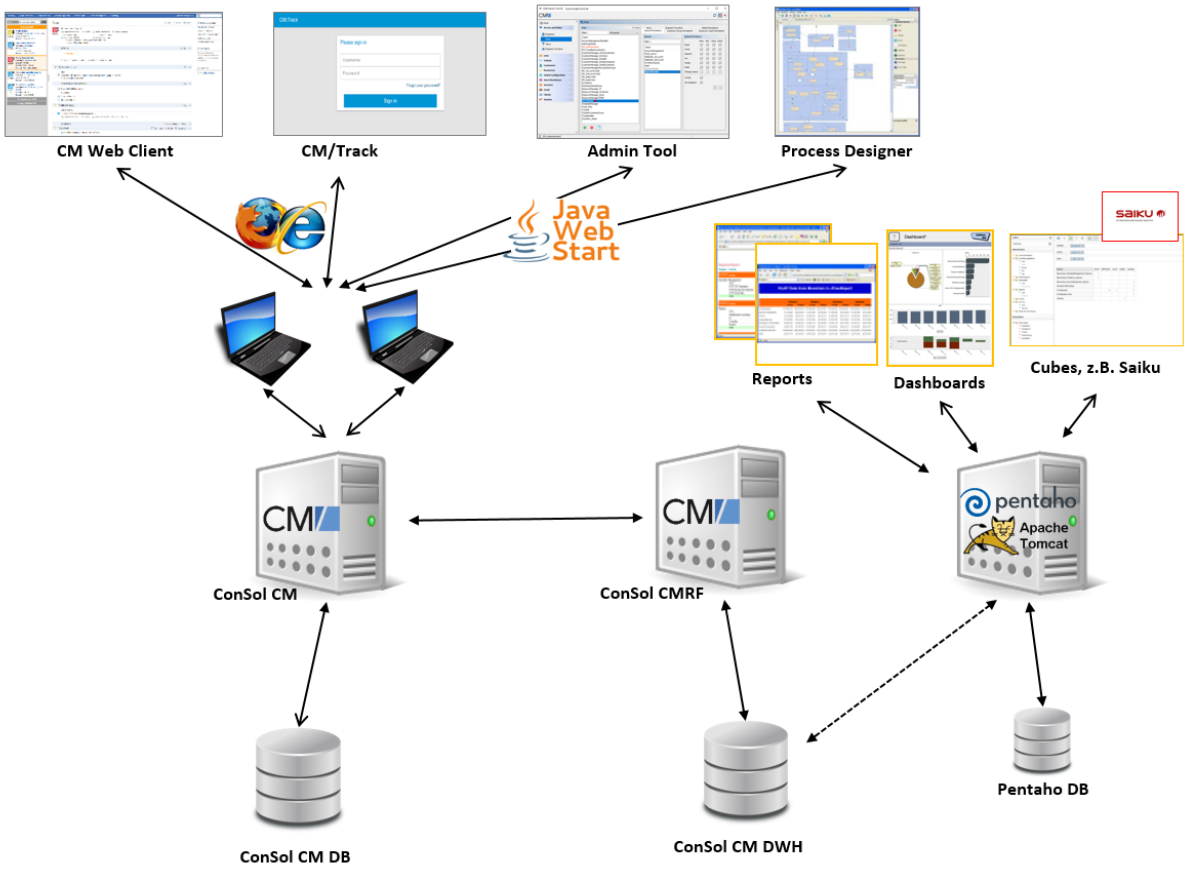


Abbildung 582: *ConSol CM - Reporting-Infrastruktur (2 Server, beide JBoss)*

Ein Application Server für ConSol CM und CMRF (Overlay-Modus, Beispiel Oracle WebLogic):

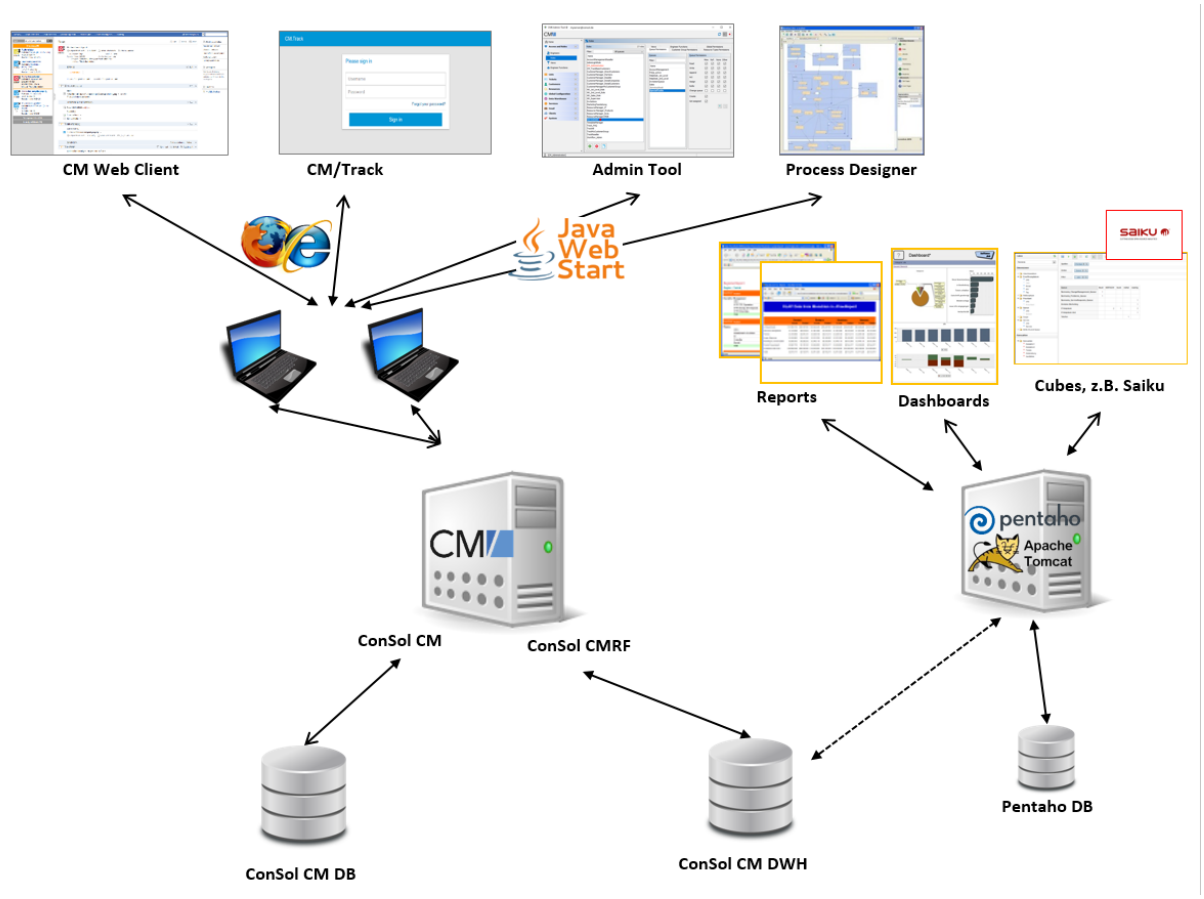


Abbildung 583: *ConSol CM - Reporting-Infrastruktur (1 Server, WebLogic)*

DWH-Datenbank

- **Oracle**
  Das DWH verwendet ein Datenbankschema mit einem Datenbankbenutzer.
- **Microsoft SQL**
  Das DWH verwendet ein Datenbankschema mit einem Datenbankbenutzer.
- **MySQL**
  Das DWH verwendet eine Datenbank mit einem Datenbankbenutzer.

F.16.1.4 Komponenten für die E-Mail-Funktionen

Eine der Kernfunktionen von ConSol CM ist die Integration mit E-Mail-Servern. Damit kann ConSol CM E-Mails senden und empfangen. Für die Bearbeiter bedeutet dies, dass neue Tickets einfach über E-Mail geöffnet werden können und dass die gesamte Kommunikation über einen Vorfall im entsprechenden Ticket gespeichert ist, einschließlich aller eingehenden und ausgehenden E-Mails.

Um E-Mails empfangen zu können, stellt ConSol CM eine Verbindung zu einem E-Mail-Server her und ruft die E-Mails aus einem oder mehreren Postfächern ab. ConSol CM verhält sich dabei wie ein regulärer E-Mail-Client (z. B. Thunderbird, Microsoft Outlook) und verwendet Standard-E-Mail-Protokolle wie IMAP oder POP3. Für den Fall, dass Sie die sichere Version verwenden möchten, werden auch IMAPs und POPs unterstützt. In diesem Fall müssen die benötigten Zertifikate auf dem Server gespeichert werden.

Zum Senden von E-Mails verwendet ConSol CM einen SMTP-Server.

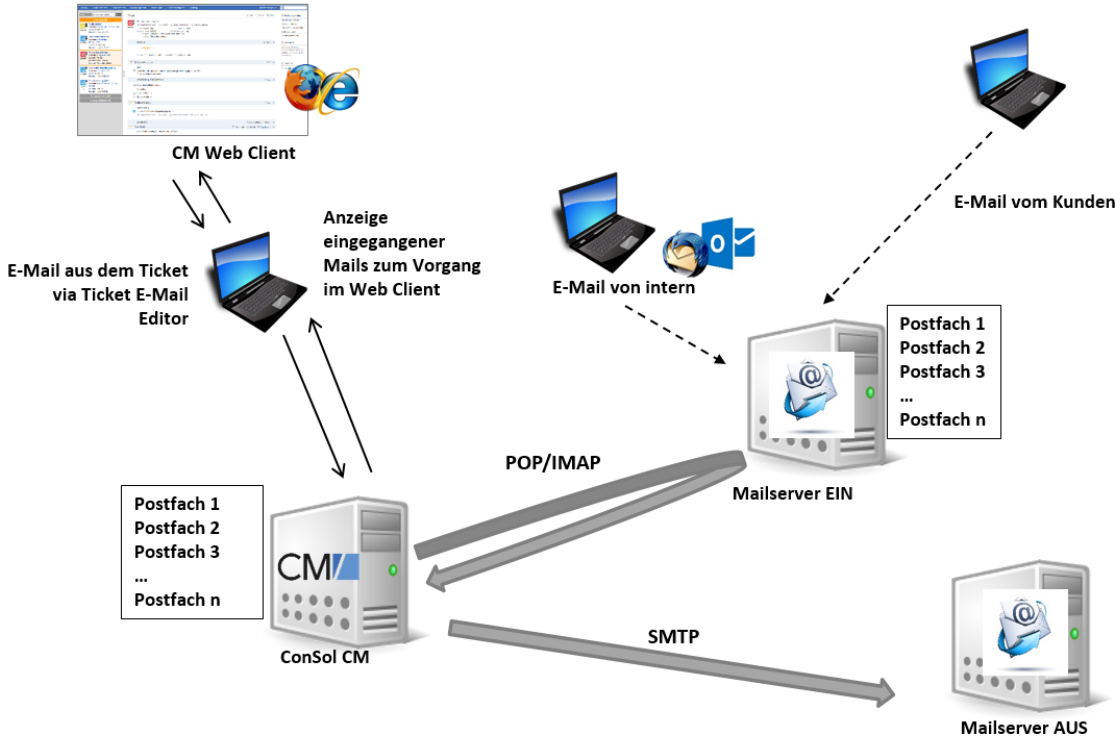


Abbildung 584: *ConSol CM - Zusammenspiel mit dem E-Mail-Server*

F.16.1.5 Indexer

Um in der Datenbank effektiv suchen zu können, erstellt ConSol CM für jedes Ticketfeld, Kundenfeld oder Ressourcenfeld, das in der Suche berücksichtigt werden soll, einen Index. Außerdem werden Bearbeiterdaten, Ticketkommentare und Attachments standardmäßig indiziert. Die Indizes werden im Dateisystem gespeichert. Im Abschnitt ConSol CM-Datenverzeichnis finden Sie eine Erklärung der Struktur des Indexverzeichnisses. Eine detaillierte Einführung in das gesamte Thema finden Sie im Abschnitt ConSol CM-Indexer.

F.16.1.6 LDAP-Authentifizierung

Als Standardfunktion kann in ConSol CM die LDAP-Authentifizierung im Web Client und/oder Portal (CM/Track) genutzt werden. Abhängig von der Konfiguration Ihres LDAP-Servers (z. B. Microsoft

Active Directory), sind möglicherweise ein Benutzername und ein Passwort erforderlich, um eine LDAP-Verbindung herzustellen. Alle LDAP-Parameter sind als ConSol CM-System-Properties gespeichert.

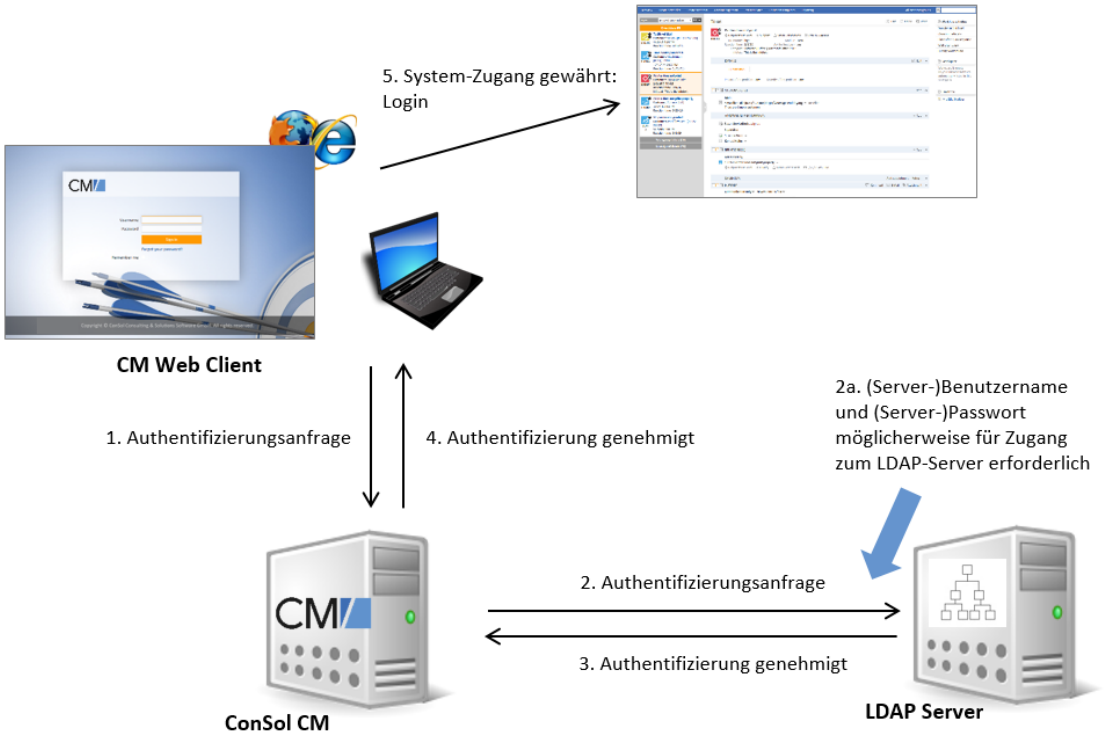


Abbildung 585: *ConSol CM - LDAP-Authentifizierung (Web Client)*

F.16.1.7 ConSol CM-Dateisystemstruktur

Die meisten Daten für die Konfiguration und den Betrieb von ConSol CM sind in der ConSol CM-Datenbank gespeichert. Einige Daten werden allerdings im Dateisystem gespeichert, und zwar in dem Datenverzeichnis, das Sie bei der Einrichtung des Systems angegeben haben.

ConSol CM-Datenverzeichnis

Die folgenden Abbildungen zeigen Beispiele aus einem Windows- und einem Linux-System:

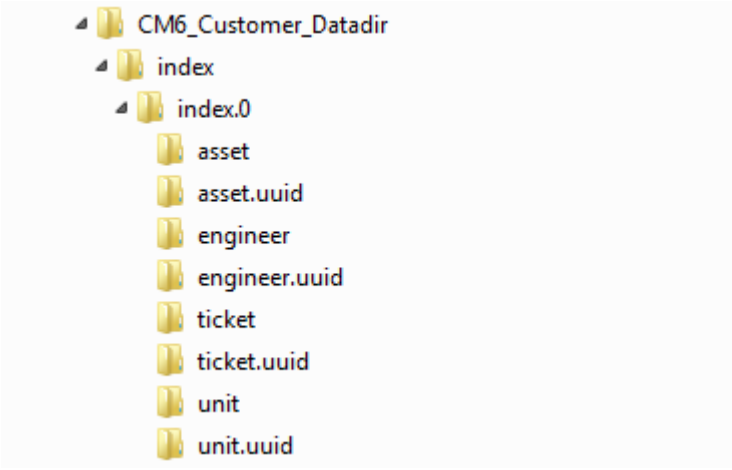


Abbildung 586: *ConSol CM - Datenverzeichnis (Windows)*

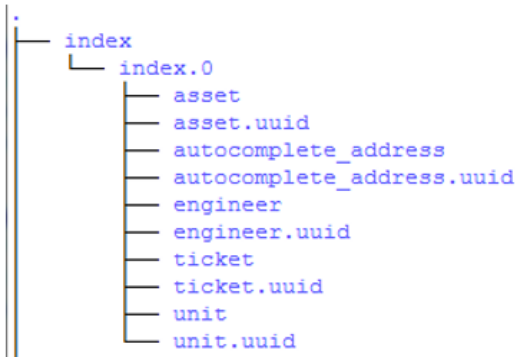


Abbildung 587: *ConSol CM - Datenverzeichnis (Linux)*

Beispielverzeichnisse:

- *index*

  In diesem Verzeichnis, werden alle Indizes gespeichert (siehe auch Abschnitt ConSol CM-Indexer). Stellen Sie sicher, dass es beim täglichen Backup des Dateisystems gesichert wird.

  - **index.0**
    In diesem Verzeichnis befindet sich ein Unterverzeichnis für jeden benötigten Index.

Dateistruktur bei JBoss 7 als Application Server

Die folgenden Verzeichnisse sind in einer JBoss 7-Installation von ConSol CM verfügbar:

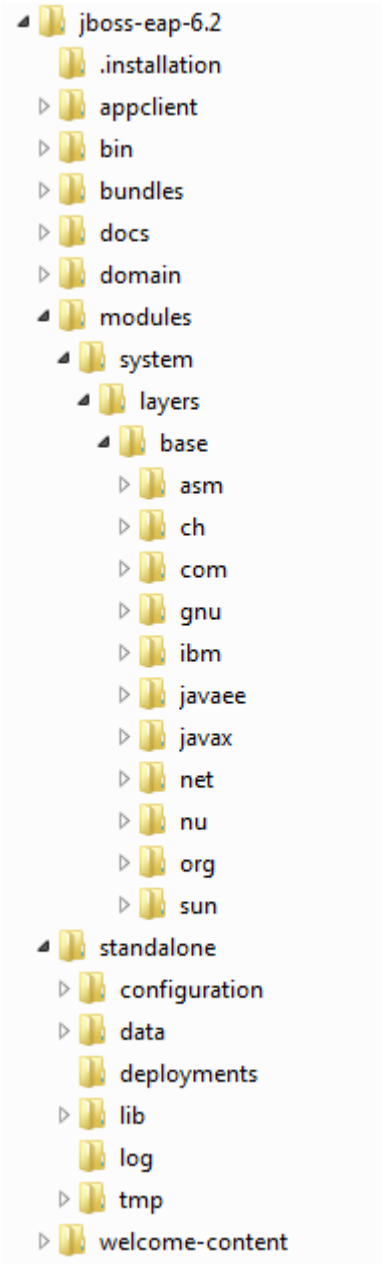


Abbildung 588: *ConSol CM - Dateistruktur in einem JBoss 7 Application Server*

Beispielverzeichnisse:

- **modules\system\layers\base**
  Unterordner enthalten die JDBC-Treiber:

  - com\microsoft\sqlserver\jdbc\main\sqljdbc4.jar
    (Microsoft SQL)
  - oracle\jdbc\main\ojdbc6-11.2.0.3.jar
    (Oracle)
  - com\mysql\jdbc\main\
    (Zielpfad des JDBC-Treibers von MySQL, muss manuell installiert werden)
- **standalone**
  Konfiguration in Umgebungen mit einem Server:
  - **configuration**
    Konfiguration von Datenbankverbindung und Logging in der Datei `cm6.xml`
  - **data**
    Betriebsdaten, z. B. `tx-operation-Keys`
  - **deployments**
    Installierte Applikationen, zum Beispiel `cm6.ear` und `cm-track.war`
  - **log**
    Log-Dateien, siehe Abschnitt Logging und Log-Dateien.
  - **tmp**
    Temporäre Daten und Arbeitskopie der Application-Server-Dateien. Kann geleert werden, z. B. zur Fehleranalyse oder -behebung.
- **domain**
  Konfiguration der Domänenumgebungen
  - **configuration**
    Konfiguration von Datenbankverbindung und Logging in der Datei `domain.xml`
  - **servers/<server-name>**/log
    Log-Dateien

Dateistruktur bei Oracle WebLogic als Application Server

In einer Oracle WebLogic-Umgebung wird ConSol CM als eigene Domäne installiert. ConSol CM und CMRF sind sogenannte *managed servers*. Details dazu finden Sie im *ConSol CM Betriebshandbuch*.

Hier werden einige Verzeichnisse erklärt. Wenn Sie ConSol CM als WebLogic-Applikation verwalten möchten, finden Sie weitere Informationen auch in den allgemeinen Weblogic-Anleitungen.

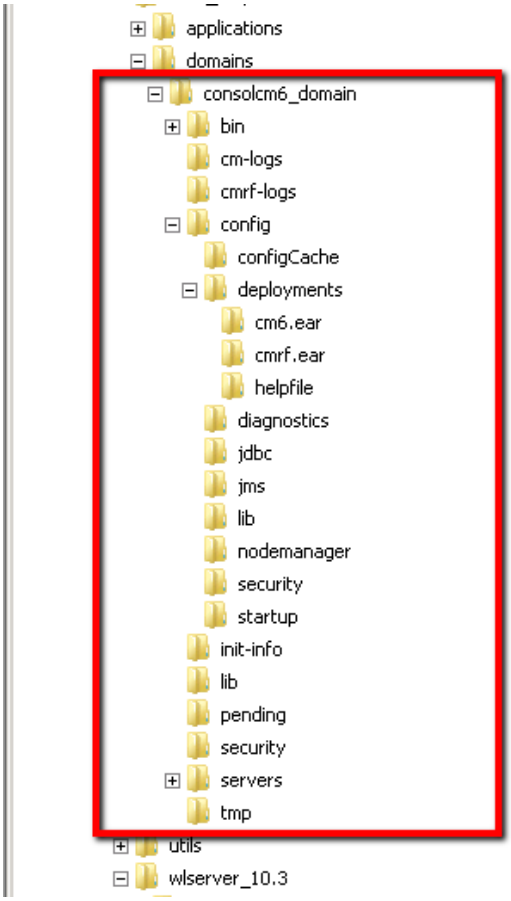


Abbildung 589: *ConSol CM - Dateistruktur in einem Oracle WebLogic Application Server*

Beispielverzeichnisse:

- **bin**
  Skripte für Start/Stopp
- **cm-logs**
  Alle Log-Dateien außer `cmrf.log`
- **cmrf-logs**
  - **Datei cmrf.log**
    Log-Meldungen für das CMRF (ConSol CM Reporting Framework)
- **config**
  Konfigurationsdateien
- **deployments**
  Installierte Applikationen, d. h. hier liegen die Verzeichnisse für ConSol CM und CMRF

F.16.1.8 Logging und Log-Dateien

Einleitung

Die Log-Dateien sind die Hauptinformationsquelle für den Administrator bezüglich der Aktivitäten und potentiellen Probleme des Systems. Der Administrator sollte sich die Log-Dateien regelmäßig ansehen. Es können Probleme auftreten, die auf der Benutzeroberfläche nicht zu sehen sind, aber in den Log-Dateien gemeldet werden.

Die Standardpfade für die Log-Dateien sind:

- **JBoss**: <JBOSS_HOME>/standalone/log
- **WebLogic**: <WEBLOGIC_HOME>\user_projects\domains\<WEBLOGIC_DOMAIN>\cm-logs

Log-Dateien

Das Logging-Verhalten kann durch Editieren der folgenden Dateien konfiguriert werden:

- **log4j.xml**
  (in WebLogic, wenn *Log4J* als Logging-Framework verwendet wird)
- **cm6.xml** und/oder **cm6-cmrf.xml**
  (in JBoss 7 im Standalone-Modus, wenn das in JBoss 7 integrierte Logging-Modul verwendet wird)
- **domain.xml**, Tag `<subsystem xmlns="urn:jboss:domain:logging:1.3">`
  (in JBoss 7 im Domänenmodus, wenn das in JBoss 7 integrierte Logging-Modul verwendet wird)

Arten von Log-Dateien

Es werden folgende Log-Dateien verwendet:

- **access.log**
  Protokolliert den Zugriff auf und die Benutzung vom Web Client. In der aktuellen Konfiguration wird jeden Tag eine eigene Datei `access.log` geschrieben.

Info zur Konfiguration: So aktivieren Sie den access.log

In einer Standardinstallation sind die folgenden Zeilen der Konfigurationsdatei, `cm6.xml` oder `cm6-cmrf.xml`, auskommentiert. Starten Sie die Zugriffsprotokollierung in Ihrem ConSol CM-System, indem Sie die Zeilen aktivieren.

```
<subsystem xmlns="urn:jboss:domain:web:1.5" default-
 virtualserver="default-host" native="false">
  <connector name="http" protocol="HTTP/1.1" scheme="http"
   socketbinding="http"/>
  <virtual-server name="default-host" enable-welcome-root="false">
    <alias name="localhost"/>
      <!--<access-log pattern='%h %l %u %t %r %s %b %{Referer}i %
       {User-Agent}i %S %T'>-->
        <!--<directory path="./" relative-
         to="jboss.server.log.dir"/>-->
      <!--</access-log>-->
  </virtual-server>
</subsystem>
```

- **boot.log**
  Meldungen bezüglich des Systemstarts (z. B. wird die Java-Version angegeben).
- **cmrf.log**
  Meldungen bezüglich des CMRF (ConSol CM Reporting Framework), d. h. Meldungen über Operationen zur Datenübertragung von der ConSol CM-Datenbank zur CMRF-Datenbank (DWH). Diese erfolgt über JMS (Java Messaging Service).
- **cmweb.log**
  Meldungen bezüglich des ***ConSol CM Web Clients***.
- **ctx.log**
  Enthält Meldungen des Spring-Frameworks.
- **errors.log**
  Enthält nur Meldungen, die mindestens das Log-Level `ERROR` haben.
- **index.log**
  Meldungen bezüglich des Indexers.
- **mail.log**
  Enthält Log-Meldungen des E-Mail-Subsystems.
- **operationtimes.log**
  Wird nur verwendet, wenn sie aktiviert wurde. Enthält die Laufzeiten der Anfragen, um mögliche Performance-Engpässe zu erkennen.

Info zur Konfiguration: So erzeugen Sie eine gute Ausgabe im operationtimes.log
Um eine kurze und informative Ausgabe in dieser Log-Datei zu erhalten, muss das entsprechende ConSol CM-Modul im DEBUG-Modus ausgeführt werden. Die folgenden Code-Schnipsel enthalten Konfigurationsbeispiele.

JBoss 7-Konfiguration:

```
<!-- JBoss7, add these parts to your cm6.xml or cm6-cmrf.xml file -->
<size-rotating-file-handler name="OPERATION_TIMES" autoflush="true">
   <file relative-to="jboss.server.log.dir"
    path="operationtimes.log"/>
   <append value="true"/>
   <rotate-size value="300m"/>
   <max-backup-index value="6"/>
   <formatter>
      <pattern-formatter pattern="%d %-5.5p [%30.-30c] [%X{username}-%X
       {sessionId}] %m%n"/>
   </formatter>
</size-rotating-file-handler>

<logger
 category="com.consol.cmweb.client.webapp.timemeasure.log.Log4jOperati
 onLogger">
   <level name="DEBUG"/>
   <handlers>
      <handler name="OPERATION_TIMES"/>
   </handlers>
</logger>
```

Oracle Weblogic Server-Konfiguration:

```
<!-- Oracle WLS -->
<appender name="OPERATION_TIMES" class="
 org.apache.log4j.RollingFileAppender">
  <errorHandler class="
   org.apache.log4j.helpers.OnlyOnceErrorHandler"/>
  <param name="File" value=" @DOMAIN_HOME/cm-
   logs/operationtimes.log"/>
  <param name="Append" value="true"/> <param name="MaxFileSize"
   value="500MB"/>
  <param name="MaxBackupIndex" value="6"/>
  <layout class="org.apache.log4j.PatternLayout">
     <param name="ConversionPattern" value="%m\n"/>
  </layout>
</appender>

<logger
 name="com.consol.cmweb.client.webapp.timemeasure.log.Log4jOperationLo
 gger">
  <level value="DEBUG"/> <appender-ref ref="OPERATION_TIMES"/>
</logger>
```

- **operationtimes-db.log**
  Enthält Informationen über die Datenbankzugriffszeiten von atomaren Operationen, um mögliche Performance-Engpässe zu erkennen. Die Protokollierung wird mit den ConSol CM-System-Properties cmas-core-server, dao.log.threshold.milliseconds und cmas-core-server, dao.log.username konfiguriert.

Info zur Konfiguration: So aktivieren Sie operationtimes-db.log nach einem Update auf 6.11.1.0
Fügen Sie die folgenden Zeilen an der richtigen Stelle in die Konfigurationsdatei `cm6.xml` oder `cm6-cmrf.xml` ein. Bitten Sie einen CM-Consultant um Hilfe, wenn Sie sich mit der Logging-Konfiguration nicht auskennen!

```
<subsystem xmlns="urn:jboss:domain:logging:1.5">
[...]
<size-rotating-file-handler name="DAO_OPERATIONS_LOG"
 autoflush="true">
  <level name="DEBUG"/>
  <formatter>
     <pattern-formatter pattern="%d %-5.5p [%30.-30c] [%X{context}-%X
      {sessionId}] %m%n"/>
  </formatter>
  <file relative-to="jboss.server.log.dir" path="operationtimes-
   db.log"/>
  <rotate-size value="300m"/>
  <max-backup-index value="6"/>
  <append value="true"/>
</size-rotating-file-handler>
[...]
<logger
 category="com.consol.cmas.core.server.service.aspect.DaoMetricsAspec
 t" use-parent-handlers="false">
<level name="TRACE"/>
  <handlers>
     <handler name="DAO_OPERATIONS_LOG"/>
  </handlers>
</logger>
```

- **server.log**
  Die allgemeine Log-Datei, die alle Meldungen enthält, die als Standardeinstellung mindestens das Log-Level `INFO` haben. Es wird empfohlen, den `DailyRollingFileAppender` zu verwenden, um zu vermeiden, dass die Datei zu lang wird.
- **session.log**
  Enthält Meldungen bezüglich Anmeldungen (Session-Starts) und Session-Timeouts von ConSol CM-Benutzern.
- **sql.log**
  Enthält Log-Einträge zu SQL-Statements aus Hibernate, wenn das Level `DEBUG` gesetzt ist (Standardmäßig ist `INFO` gesetzt).
- **support_libs_errors.log**
  Enthält Fehler aus unterstützenden Bibliotheken, die von der ConSol CM-Applikation korrekt behandelt wurden (auf diese Weise bleibt die `server.log` übersichtlich).

- **timer-manager.log**
  Enthält zusätzliche Log-Meldungen mit dem Log-Level `DEBUG`, wenn Workflow-Timer aktiviert oder deaktiviert werden. Informationen über das Eskalationsdatum werden ebenfalls protokolliert.
- **tx.log**
  Enthält Log-Meldungen bezüglich Transaktionen des Spring-Frameworks.
- **unit-deletion.log**
  Enthält Informationen über die Löschung von Kunden.
- **workflow.log**
  Informationen über aktivierte / neu initialisierte / deaktivierte Timer werden mit Log-Level `INFO` protokolliert und die gesamte Debug-Ausgabe im Zusammenhang mit der Workflow-Engine wird in diese dedizierte Datei geschrieben.

Admin-Tool-spezifische Log-Dateien

Es gibt zwei Log-Dateien, die spezielle Informationen über das Admin Tool enthalten. Bei einem Update von einer CM-Version unter 6.11 auf CM-Version 6.11 muss möglicherweise die CM-Logging-Konfiguration angepasst werden.

- **audit.log**
  Verfügbar in ConSol CM-Versionen 6.11 und höher. In neuen 6.11-Installationen ist die Log-Datei automatisch verfügbar. Wenn ein System von einer CM-Version unter 6.11 aktualisiert wird, muss die entsprechende Konfiguration für den Logger in die Konfigurationsdatei (`cm6.xml` oder `cm6-cmrf.xml`) eingetragen werden. Siehe Info zur Konfiguration Nr. 1.
  - Protokolliert Anmeldungen und Abmeldungen und alle Operationen, die mit dem Admin Tool vorgenommen werden:
    - Konfigurationsänderungen
    - Erstellen von neuen Objekten
    - Löschen von Objekten
    - Wiedereröffnen von Tickets
  - Protokolliert Anmeldungen und Abmeldungen und alle Operationen, die mit dem Process Designer vorgenommen werden, z. B.:
    - Installation eines Workflows
- **transfer.log**
  Verfügbar in ConSol CM-Versionen 6.11 und höher. In neuen 6.11-Installationen ist die Log-Datei automatisch verfügbar. Wenn ein System von einer CM-Version unter 6.11 aktualisiert wird, muss die entsprechende Konfiguration für den Logger in die Konfigurationsdatei (`cm6.xml` oder `cm6-cmrf.xml`) eingetragen werden. Siehe Info zur Konfiguration Nr. 2.
  - Protokolliert Informationen zum Export/Import von Szenarien.

Info zur Konfiguration Nr. 1: So integrieren Sie den audit.log in Ihre CM-Konfiguration. Nur beim Update von CM-Versionen vor 6.11 auf 6.11 erforderlich.

Fügen Sie die folgenden Zeilen an der richtigen Stelle in die Konfigurationsdatei `cm6.xml` oder `cm6-cmrf.xml` ein. Bitten Sie einen CM-Consultant um Hilfe, wenn Sie sich mit der Logging-Konfiguration nicht auskennen!

```
<size-rotating-file-handler name="AUDIT" autoflush="true">
   <file relative-to="jboss.server.log.dir" path="audit.log"/>
   <append value="true"/>
   <rotate-size value="300m"/>
   <max-backup-index value="6"/>
   <formatter>
      <pattern-formatter pattern="%d %-5.5p [%30.-30c] [%X{username}-%X
       {context}-%X{sessionId}] %m%n"/>
   </formatter>


</size-rotating-file-handler>
<logger
 category="com.consol.cmas.core.server.history.method.MethodExecutionJourna
 lAspect" use-parent-handlers="false">
   <level name="TRACE"/>
   <handlers>
      <handler name="AUDIT"/>
   </handlers>
</logger>
```

Info zur Konfiguration Nr. 2: So integrieren Sie den transfer.log in Ihre CM-Konfiguration. Nur beim Update von CM-Versionen vor 6.11 auf 6.11 erforderlich.

Fügen Sie die folgenden Zeilen an der richtigen Stelle in die Konfigurationsdatei `cm6.xml` oder `cm6-cmrf.xml` ein. Bitten Sie einen CM-Consultant um Hilfe, wenn Sie sich mit der Logging-Konfiguration nicht auskennen!

```
<size-rotating-file-handler name="TRANSFER_FILE" autoflush="true">
   <file relative-to="jboss.server.log.dir" path="transfer.log"/>
   <append value="true"/>
   <rotate-size value="300m"/>
   <max-backup-index value="6"/>
   <formatter>
      <pattern-formatter pattern="%d %-5.5p [%X{username}-%X{context}-%X
       {sessionId}] %m%n"/>
   </formatter>
</size-rotating-file-handler>


<logger category="TRANSFER" use-parent-handlers="false">
   <level name="INFO"/>
   <handlers>
      <handler name="TRANSFER_FILE"/>
   </handlers>
</logger>
```

Struktur der Log-Datei

In der Standardkonfiguration haben die Einträge in der Log-Datei folgende Syntax:

```
Date Timestamp Loglevel [Logger] Message
```

Beispiel für einen Eintrag in einer Log-Datei (erfolgreicher Start von ***ConSol CM*** unter JBoss):

```
2012-11-06 14:22:12,685 INFO [e.coyote.http11.Http11Protocol] Starting Coyote
 HTTP/1.1 on http-0.0.0.0-8080
```

Die Komponenten der Meldung:

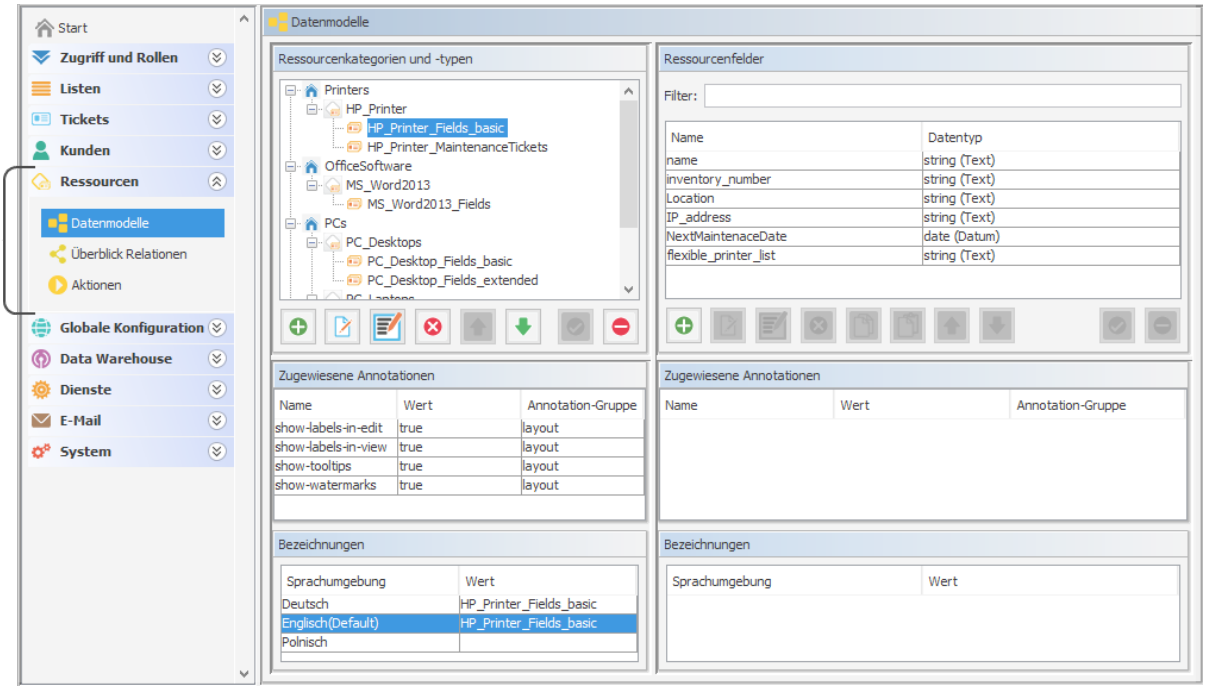
- **Date:**
  6. November 2012
- **Timestamp:**
  14:22:12
- **Loglevel:**
  INFO

- **Logger:**
  e.coyote.http11.Http11Protocol
  Name der Java-Klasse, nicht der vollständige Name sondern die letzten 30 Zeichen. Der komplette Name wäre `org.apache.coyote.http11.HttpProtocol`.
- **Message:**
  Starting Coyote HTTP/1.1 on http-0.0.0.0-8080

Einfache Meldungen und Meldungen zu einer erfolgreichen Operation haben häufig nur eine Zeile.

Wenn Fehler auftreten (Log-Level „ERROR“), sehen Sie möglicherweise Stack Traces in den Logs. Wenden Sie sich an einen unserer ConSol CM-Consultants oder das ConSol CM-Supportteam, wenn Sie Hilfe benötigen.

G - Abschnitt Add-Ons



Dieser Abschnitt enthält Informationen über die optionalen Add-ons von ConSol CM:

- CM/Resource Pool
- CM/Doc
- Das Kundenportal: CM/Track
- CM/Phone: CTI mit ConSol CM

G.1 CM/Resource Pool

Ab Version 6.10 bietet ConSol CM das Add-on CM/Resource Pool. Mit diesem Modul können Sie die ConSol CM-Datenbank erweitern, um Datenobjekte zu speichern, die eine beliebige Art von Asset oder Objekt Ihres Unternehmens darstellen können. Ein Dienstleistungsunternehmen kann den Ressourcenpool zum Beispiel verwenden, um SLAs zu speichern und zu verwalten. Ein IT-Servicedesk kann ihn zum Implementieren einer Datenbank mit IT-Assets einsetzen. Eine Marketingabteilung kann dort Design- und Textvorlagen speichern. Und ein Händler kann sein Produktportfolio mit dem Ressourcenpool abbilden. Die Implementierung des Ressourcenpools kann daher für Ihr Unternehmen einen wichtigen Schritt bei der Anpassung von ConSol CM darstellen. In den folgenden Abschnitten lernen Sie, wie man mit diesem Modul arbeitet:

Eine allgemeine Einführung in CM/Resource Pool finden Sie im Abschnitt Einführung in CM/Resource Pool.

Alle Elemente des Admin Tools, mit denen Sie bei der Konfiguration des Ressourcenpools arbeiten, sind im Abschnitt CM/Resource Pool - Admin-Tool-Elemente erklärt.

Um Ihnen als Administrator einen Überblick darüber zu geben, wie die Bearbeiter die Objekte des Ressourcenpools verwalten, haben wir den Abschnitt Kurze Einführung in die Funktionen von CM/Resource Pool im Web Client geschrieben.

Einer der ersten Schritte bei der Arbeit mit dem Ressourcenpool ist das Einrichten des Ressourcenmodells. Dies wird im Abschnitt CM/Resource Pool - Einrichten des Ressourcenmodells behandelt.

Ähnlich wie die Kunden-Templates können Sie auch Templates definieren, die die Darstellung von Ressourcendaten im Web Client steuern. Diese Templates sind im Abschnitt CM/Resource Pool - Templates für Ressourcendaten erklärt.

Ressourcen können Relationen haben. Im Abschnitt CM/Resource Pool - Ressourcenrelationen erfahren Sie, wie man Ressourcenrelationen definiert und mit ihnen arbeitet.

Für jeden Ressourcentyp können Aktivitäten ausgeführt werden. Diese basieren auf Ressourcenaktionen und sind im Abschnitt CM/Resource Pool - Ressourcenaktionen beschrieben. Einige spezifischere Informationen über die Implementierung von Skripten für das Action Framework finden Sie auch im Abschnitt Skripte für das Action Framework.

Entsprechend dem Standardverhalten von ConSol CM brauchen die Bearbeiter bestimmte Zugangsberechtigungen auf die Ressourcen. Diese Berechtigungen basieren auf Rollen. Details dazu finden Sie im CM/Resource Pool - Zuweisen von Berechtigungen für Ressourcen.

Das Ressourcenpool-Dashboard bietet einen praktischen Einstiegspunkt für alle Objekte des Ressourcenpools. Die Konfiguration des Dashboards wird im Abschnitt CM/Resource Pool - Das Ressourcenpool-Dashboard erklärt.

G.1.1 Einführung in CM/Resource Pool

G.1.1.1 Einführung in CM/Resource Pool

Ab ConSol CM-Version 6.10 ist das Add-on ***CM/Resource Pool*** verfügbar. Der Ressourcenpool bietet eine Datenbankerweiterung, die es ConSol CM ermöglicht, andere Objekte als Tickets und Kunden zu speichern. Zum Beispiel können Sie Ihre IT-Landschaft abbilden, indem Sie jedes Asset (wie PCs, Laptops, Bildschirme, Drucker) als Ressource im Ressourcenpool anlegen. Ähnliche Ressourcen (z. B. HP-Drucker) werden in einem *Ressourcentyp* verwaltet und ähnliche Ressourcentypen bilden eine *Ressourcenkategorie* (z. B. Drucker). Jedes einzelne Objekt (z. B. Drucker Nr. 4711) ist eine Instanz eines Ressourcentypen (z. B. Ressourcentyp *HP-Drucker*). Andere Beispiel für den Einsatz von CM/Resource Pool sind Produkte, Anlagen, Maschinen, Räume, Fahrzeuge, Verträge (z. B. SLAs) oder wissenschaftliche Proben.

CM/Resource Pool erweitert die Basisobjekte von ConSol CM:

1. Ein Ticket ist eine Instanz einer Prozessausführung.
2. Ein realer Kunde (z. B. eine Person, die den Servicedesk anruft, um ein Ticket zu öffnen) ist eine Instanz einer Kundendefinition (Unit-Definition).
3. **Eine reale Ressource (z. B. ein Drucker oder eine SLA) ist eine Instanz einer Ressourcendefinition.**

G.1.1.2 CM/Resource Pool auf einen Blick

Struktur von CM/Resource Pool

Die Struktur des Ressourcenpools basiert auf einer zweistufigen Hierarchie und umfasst eine beliebige Anzahl an Ressourcenkategorien. Jede Ressourcenkategorie ist eine Unterstruktur innerhalb der Baumstruktur des Ressourcenpools. Sie können eine beliebige Anzahl an Ressourcenkategorien, d. h. eine beliebige Anzahl an Unterstrukturen, innerhalb des Ressourcenpools verwalten. Auf diese Weise können Sie zum Beispiel die IT-Landschaft sowie Ihr Produktportfolio als Ressourcen anlegen, jede in einer oder mehreren eigenen Ressourcenkategorien (Unterstrukturen).

Relationen zwischen Ressourcen und anderen CM-Objekten können verschiedene Konstellationen widerspiegeln, so kann zum Beispiel eine Ressource der Ressourcenkategorie *Drucker* und des Ressourcentyps *HP-Drucker* eine Relation zu einem Ticket haben, in dem ein Problem mit genau diesem Drucker gelöst werden soll. Oder ein Drucker ist mit allen Kontakten, die diesen Drucker nutzen, verknüpft. Relationen zwischen Ressourcen sind ebenfalls möglich. Eine detaillierte Beschreibung der Arbeit mit Ressourcenrelationen finden Sie im Abschnitt CM/Resource Pool - Ressourcenrelationen.

Die Datenfelder für die einzelnen Ressourcentypen werden so ähnlich definiert, wie die Felder für Ticketdaten (Ticketfelder) und Kundendaten (Kundenfelder). Die Datenfelder heißen *Ressourcenfelder* und werden immer in den *Ressourcenfeldgruppen* genannten Gruppen verwaltet. Das Einrichten des Datenmodells für den Ressourcenpool, d. h. das *Ressourcenmodell*, wird detailliert im Abschnitt CM/Resource Pool - Einrichten des Ressourcenmodells erklärt.

Für jeden Ressourcentyp können Ressourcenaktionen definiert werden. Diese sind so ähnlich wie die Kundenaktionen für Kundenobjekte (Firmen und Kontakte). Die Ressourcenaktionen starten Skripte, die ressourcenspezifische Aktivitäten ausführen, wie die Aktualisierung von Ressourcendaten (z. B.

Erstellen einer Liste aller Firmen, die mit Drucker Nr. 4711 verknüpft sind, oder Erstellen eines Wartungstickets für eine Firmware-Aktualisierung). Ressourcenaktionen sind Teil des Action Frameworks und werden im Abschnitt CM/Resource Pool - Ressourcenaktionen detailliert beschrieben.

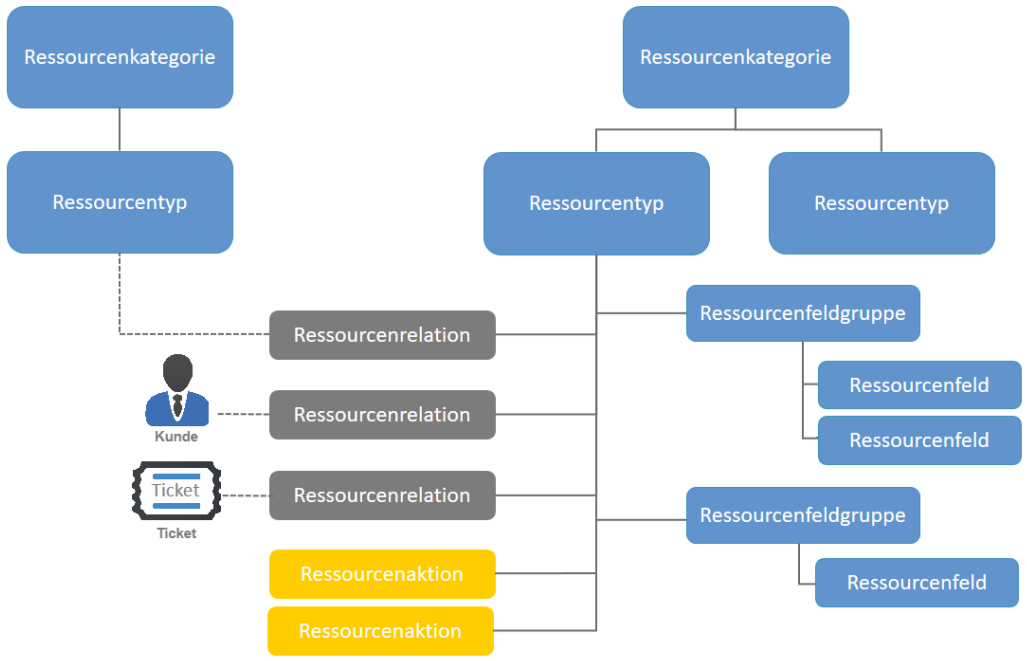


Abbildung 590: *CM/Resource Pool - Objekte in einem Ressourcenmodell*

Wichtige Begriffe

- Eine **Ressourcenkategorie**
  stellt die höchste Hierarchiestufe im Ressourcenpool dar. Sie kann als Stammknoten einer Unterstruktur im Ressourcenpool betrachtet werden. Der Ressourcenpool eines Unternehmens könnte zum Beispiel eine Ressourcenkategorie *Drucker* haben. Eine Ressourcenkategorie umfasst einen oder mehrere Ressourcentypen, z. B. HP-Drucker, Kyocera-Drucker, Lexmark-Drucker.
- Ein **Ressourcentyp**
  stellt ähnliche Ressourcen dar, die die gleichen Datenfelder haben, die aber mit anderen Werten gefüllt sind. Der Ressourcentyp *HP-Drucker* hat zum Beispiel die Datenfelder (Ressourcenfelder) *Modell*, *IP-Adresse*, *Inventarnummer* und *Standort*. Jeder reale HP-Drucker wird durch ein Objekt dieses Ressourcentyps abgebildet. Lexmark-Drucker können wiederum andere Datenfelder (Ressourcenfelder) haben, wie *Modell*, *IP-Adresse*, *Standort* und *SLA-Typ*. Jeder reale Lexmark-Drucker wird durch ein Objekt dieses Ressourcentyps abgebildet.

> Innerhalb der Hierarchie des Ressourcenpools gibt es **keine Vererbung**. Datenfelder (Ressourcenfelder) werden immer als Ressourcenfeldgruppe für den einzelnen Ressourcentyp definiert!

- In einer **Ressourcenfeldgruppe**
  sind ein oder mehrere Ressourcenfelder innerhalb eines Ressourcentyps gruppiert. Auf der Ressourcenseite kann eine Ressourcenfeldgruppe als Tab angezeigt werden, so wie eine Ticketfeldgruppe für Ticketdaten auf der Ticketseite. Ressourcenfeldgruppen können annotiert werden.
- Ein **Ressourcenfeld**
  ist ein Datenfeld für die Daten eines Ressourcentyps, z. B. *Modell*, *IP-Adresse* oder *Standort*. Ein Ressourcenfeld hat immer einen bestimmten Datentyp. Ressourcenfelder können Ressourcendaten enthalten, so wie Ticketfelder Ticketdaten enthalten. Ressourcenfelder können annotiert werden, um ihr logisches Verhalten oder ihre grafische Darstellung zu verändern.
- Das **Ressourcenmodell**
  ist das komplette Datenmodell, mit dem alle Objekte im Ressourcenpool definiert werden. Es wird im Admin Tool in der Navigationsgruppe *Ressourcen*, Navigationselement *Datenmodelle* erstellt.
- Eine **Ressourcenrelation**
  ist eine Relation von einer Ressource zu einem anderen ConSol CM-Objekt. Dies kann ein Ticket, ein Kunde oder eine andere Ressource sein.
- Eine **Ressourcenaktion**
  ist eine Aktion, die für einen Ressourcentyp definiert wird und für die realen Ressourcenobjekte dieses Typs zur Verfügung steht. Eine Aktion kann manuell oder automatisch angestoßen werden. Manuelle Ressourcenaktionen können im Web Client auf der Ressourcenseite ausgewählt werden, so wie Workflow-Aktivitäten für Tickets.
- Die **Ressourcenseite**
  ist die Seite im Web Client, auf der alle Daten für eine reale Ressource angezeigt werden, z. B. die Ressourcenseite für den HP-Drucker Nr. 4711.
- Die **Ressourcentypseite**
  ist die Seite im Web Client, auf der Informationen über einen Ressourcentyp und eine Übersicht (Liste) aller Ressourcen dieses Typs angezeigt werden.
- Das **Ressourcenpool-Dashboard**
  ist eine Übersichtsseite im Web Client, auf der alle Ressourcenkategorien und Ressourcentypen angezeigt werden, für die der aktuelle Bearbeiter die entsprechenden Zugangsberechtigungen hat.

G.1.1.3 Definieren von Ressourcenmodellen mit dem Admin Tool

Um reale Ressourcen (z. B. HP-Drucker Nr. 4711) erstellen zu können, muss zuerst ein Ressourcenmodell erstellt werden, d. h. eine Definition aller Datenfelder, die für alle Ressourcen dieses Typs erforderlich sind, z. B. für den Typ *HP-Drucker*. Alle Ressourcenmodelle müssen im Admin Tool definiert werden. Das gilt für:

- Ressourcenkategorien (z. B. die Ressourcenkategorie *Drucker*)
- Ressourcentypen (z. B. der Ressourcentyp *HP-Drucker*)
- Ressourcenrelationen (z. B. eine mögliche Relation zwischen dem Ressourcentyp *HP-Drucker* und einem *Ticket*)

- Ressourcenaktionen (z. B. die Ressourcenaktion *Daten des Wartungsvertrags aus dem ERP-System abrufen*)

Die folgende Abbildung zeigt ein Beispiel einer einfachen Ressourcenpool-Konfiguration mit vier Ressourcenkategorien. Alle Schritte, die Sie zum Konfigurieren dieser Konstellation durchführen müssen, werden in den folgenden Abschnitten erklärt. Die für die Konfiguration des Ressourcenpools verwendeten Bereiche des Admin Tools sind im Abschnitt CM/Resource Pool - Admin-Tool-Elemente beschrieben.

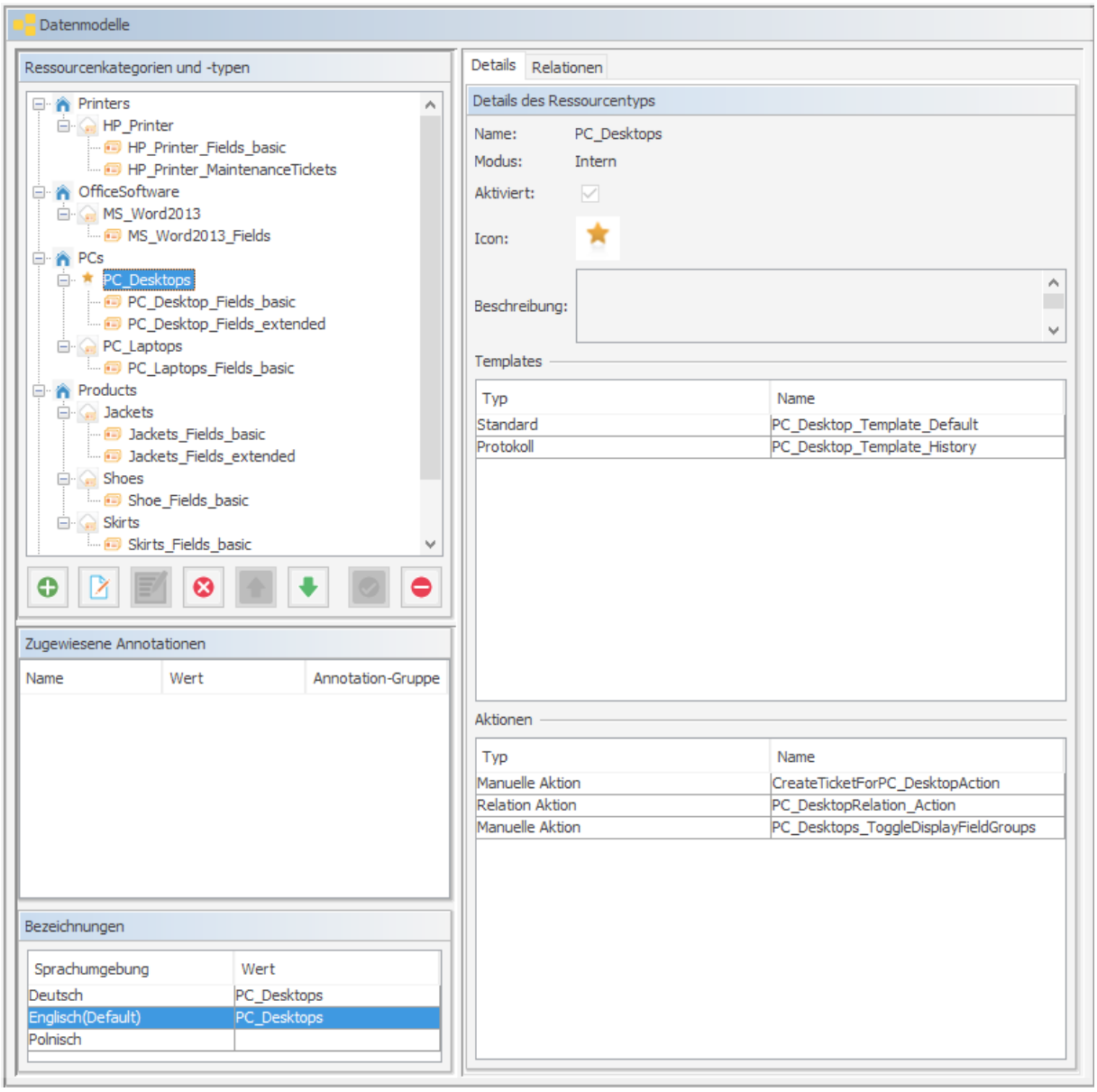


Abbildung 591: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Einfache Konfiguration des Ressourcenpools*

Sobald Sie die Ressourcen (d. h. die Datenmodelle für die realen Ressourcen) im Admin Tool definiert haben, können die Bearbeiter im Web Client mit der Arbeit mit realen Ressourcenobjekten beginnen. Dies wird kurz im Abschnitt Kurze Einführung in die Funktionen von CM/Resource Pool im Web Client erklärt. Eine detaillierte Beschreibung finden Sie im *ConSol CM Benutzerhandbuch*.

G.1.1.4 Lizenzierung

CM/Resource Pool muss separat lizenziert werden. Nähere Informationen dazu erhalten Sie vom ConSol CM-Vertrieb.

G.1.1.5 Programmierung mit Ressourcenpoolobjekten

Die ConSol CM Java API wurde im Zuge der Implementierung des Ressourcenpools deutlich erweitert. Die Erklärung der wichtigen Groovy-Klassen und mehrere Programmierbeispiele finden Sie im *ConSol CM Process Designer Handbuch*, CM-Version 6.10.

G.1.2 CM/Resource Pool - Admin-Tool-Elemente

Für die Konfiguration von CM/Resource Pool müssen Sie mit mehreren Bereichen des Admin Tools arbeiten:

- **Navigationsgruppe *Zugriff und Rollen***
  Zum Zuweisen von Ressourcenpool-Berechtigungen zu Rollen
- **Navigationsgruppe *Listen***
  Falls Sie MLAs und/oder Sortierte Listen in Ressourcenfeldern verwenden.
- **Navigationsgruppe *Ressourcen*** (siehe folgende Abbildung)
  - **Datenmodelle**
    Definition des Ressourcendatenmodells, einschließlich der Ressourcenrelationen
  - **Überblick Relationen**
    Enthält eine Liste aller Ressourcenrelationen im schreibgeschützten Modus. Hier können keine Definitionen/Konfigurationen vorgenommen werden.
  - **Aktionen**
    Definition der Ressourcenaktionen
- **Navigationsgruppe *Globale Konfiguration***
  - **Bezeichnungen**
    Falls Sie die Bezeichnungen der Elemente des Ressourcenpools im Web Client ändern möchten, oder Ihre eigenen Bezeichnungen definieren möchten, um sie in Skripten zu verwenden.
- **Navigationsgruppe *System***
  - **Skripte und Templates**
    - Skripte für Ressourcenaktionen
    - Templates für die Anzeige von Ressourcendaten im Web Client
  - **Lizenz**
    Zum Aktivieren einer neuen Lizenz für die ConSol CM-Version 6.10 mit CM/Resource Pool

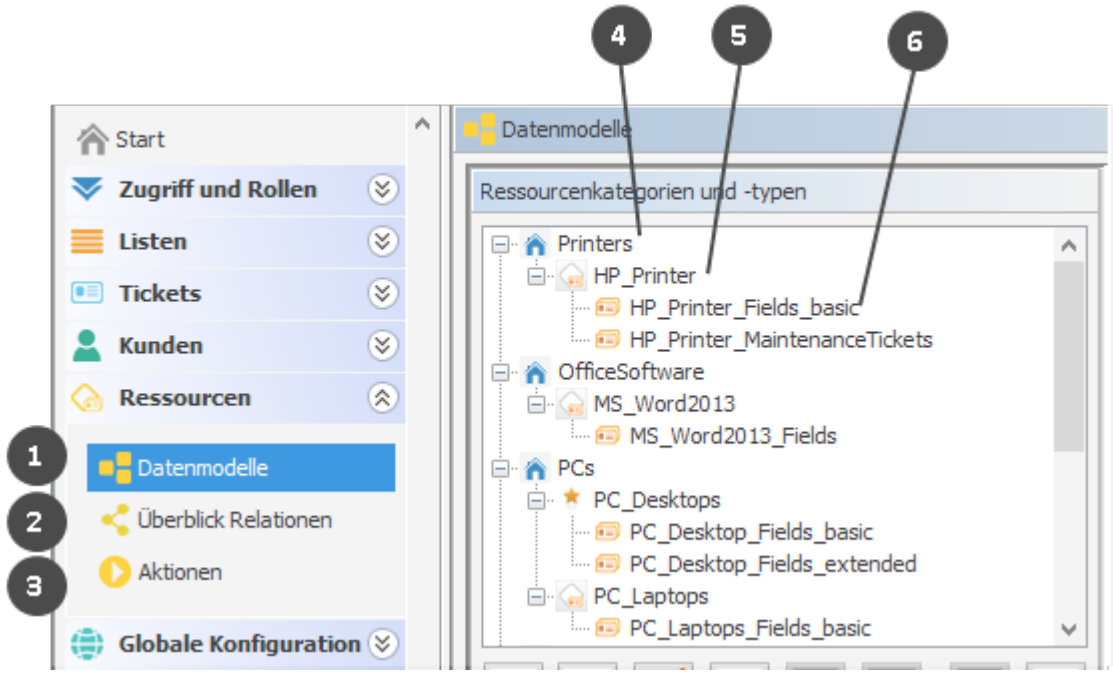


Abbildung 592: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Hauptelemente für die Konfiguration von CM/Resource Pool*

Die Hauptelemente von CM/Resource Pool sind:

- Navigationselement *Datenmodelle* zum Definieren von Ressourcen und Ressourcenrelationen (1)
- Navigationselement *Überblick Relationen* zum Anzeigen der Relationsliste (2)
- Navigationselement *Aktionen* zum Verwalten von Ressourcenaktionen (3)
- Ressourcenkategorie (4)
- Ressourcentyp (5)
- Ressourcenfeldgruppe (6)

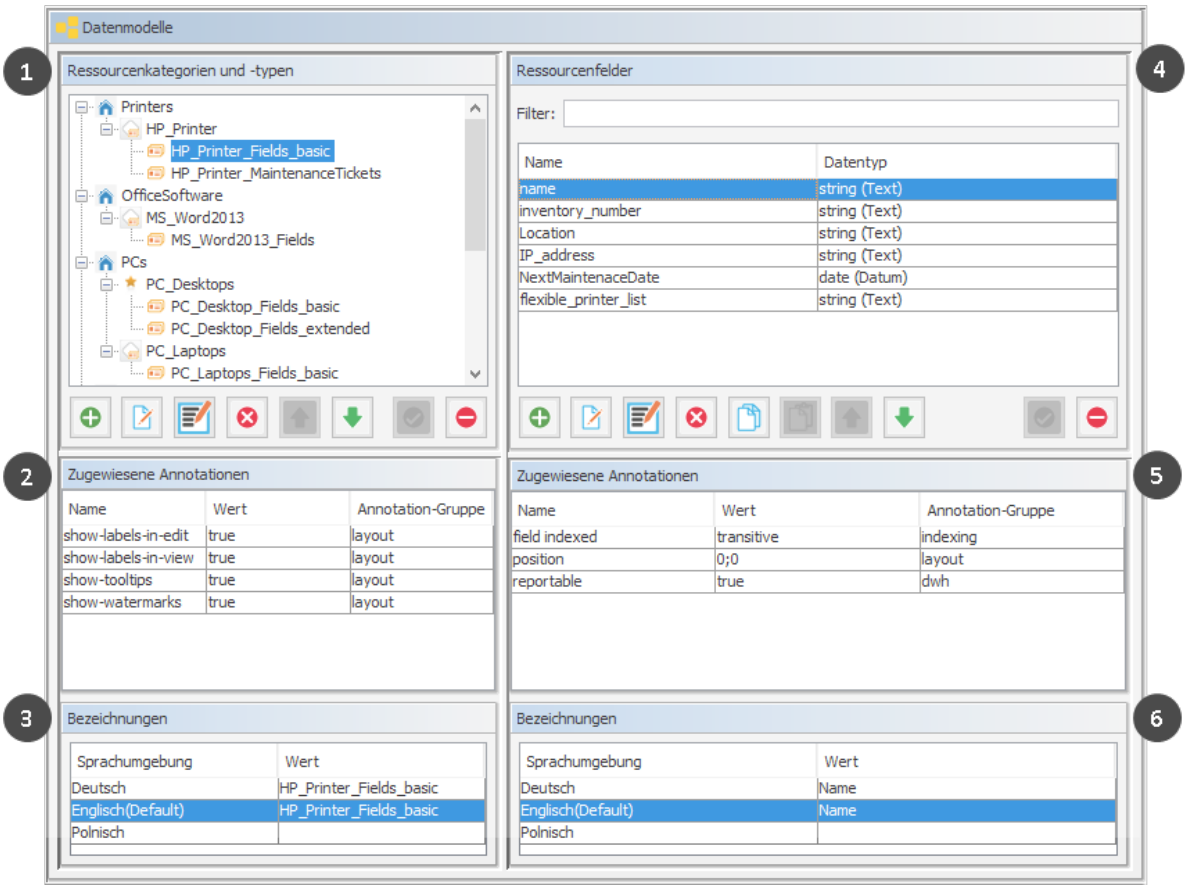


Abbildung 593: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Konfiguration von CM/Resource Pool*

Die Seite *Datenmodelle* hat sechs Abschnitt:

- Ressourcendatenmodell (1)
- Annotationen für die ausgewählte Ressourcenfeldgruppe (2)
- Bezeichnungen für die ausgewählte Ressourcenfeldgruppe (3), d. h. die lokalisierten Werte des Namens der ausgewählten Ressourcenfeldgruppe
- Ressourcenfelder (4) der ausgewählten Ressourcenfeldgruppe
- Annotationen für das ausgewählte Ressourcenfeld (5)
- Bezeichnungen für das ausgewählte Ressourcenfeld (6), d. h. die lokalisierten Werte des Namens des ausgewählten Ressourcenfelds

G.1.3 Kurze Einführung in die Funktionen von CM/Resource Pool im Web Client

G.1.3.1 Übersicht der Ressourcenpool-Operationen im Web Client

Wenn Sie als Administrator im Admin Tool die Ressourcendatenmodelle definiert haben (Ressourcenkategorien und die erforderlichen Ressourcentypen mit ihren Ressourcenfeldgruppen und Ressourcenfeldern), können die Bearbeiter mit der Arbeit mit realen Ressourcenpool-Objekten beginnen, sofern sie über die nötigen Zugangsberechtigungen verfügen. Die Berechtigungen werden im Abschnitt CM/Resource Pool - Zuweisen von Berechtigungen für Ressourcen. Im aktuellen Abschnitt erhalten Sie eine Übersicht über die folgenden Operationen:

- Erstellen von Ressourcen im Web Client
- Arbeiten mit der Ressourcentypseite
- Hinzufügen einer Ressource zu den Favoriten
- Definieren und Verwenden von Ressourcenrelationen
- Definieren und Verwenden von Ressourcenaktionen
- Finden von Ressourcen mit der Schnellsuche
- Finden von Ressourcen mit der Detailsuche

G.1.3.2 Erstellen von Ressourcen im Web Client

Im Admin Tool können Sie nur Datenmodelle für die Objekte definieren (für die Entwickler unter uns: dies ist vergleichbar mit der Definition von Klassen in der objektorientierten Programmierung). Die realen Objekte (die Instanzen der definierten Datenmodelle) müssen im Web Client erstellt werden. Das Ressourcenpool-Dashboard bietet die dafür benötigte grafische Oberfläche. Die Verwaltung der Berechtigungen für die Ressourcen basiert entsprechend dem Standardverhalten von ConSol CM auf Rollen. Der Bearbeiter sieht nur die Ressourcentypen im Ressourcenpool-Dashboard, für die er die erforderlichen Berechtigungen hat. Die Rollenverwaltung für Ressourcen ist detailliert im Abschnitt CM/Resource Pool - Zuweisen von Berechtigungen für Ressourcen.

Die folgende Abbildung zeigt das Ressourcenpool-Dashboard mit den im Admin Tool definierten Ressourcenmodellen (siehe Abschnitt CM/Resource Pool - Admin-Tool-Elemente). Der angemeldete Bearbeiter hat in diesem Beispiel Zugangsberechtigungen für alle Ressourcentypen.

Die Bezeichnung des Ressourcenpools im Hauptmenü (siehe folgende Abbildung) kann an die Anforderungen des jeweiligen Systems angepasst werden. Beispiele für alternative Bezeichnungen sind *Assetmanagement*, *Produkte*, *Maschinen* oder *Inventar*. Eine Beschreibung, wie Sie die Bezeichnung ändern können, finden Sie im Abschnitt Bezeichnungen.

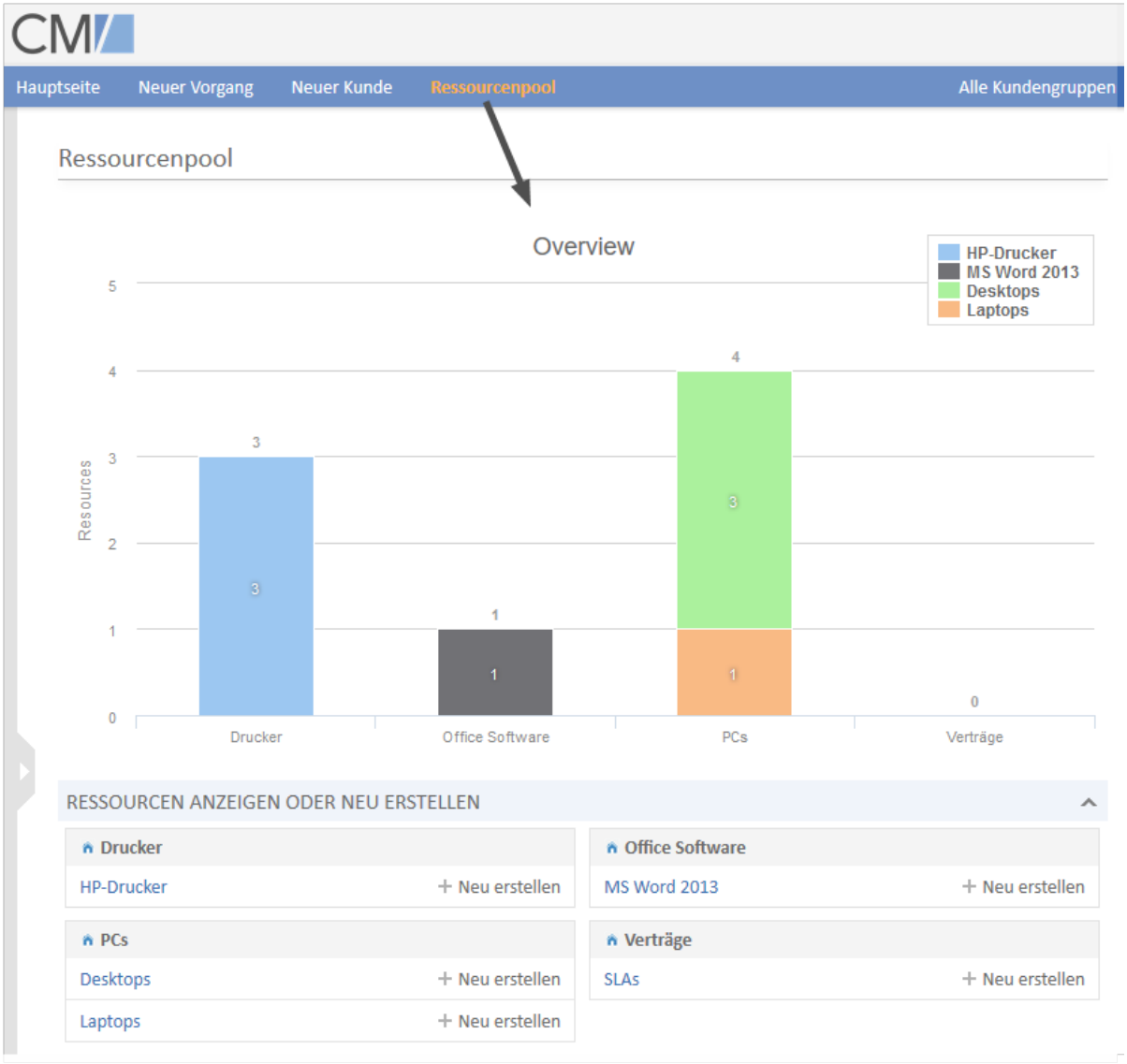


Abbildung 594: *ConSol CM Web Client - Ressourcenpool-Dashboard*

Wenn Datenmodelle (im Admin Tool) definiert wurden und Sie als Bearbeiter die erforderlichen Zugangsberechtigungen haben, können Sie ein reales Objekt erstellen, z. B. ein Objekt für den Drucker Nr. 4711 auf der obersten Etage.

Die folgenden Abbildungen zeigen die Schritte, die Sie dazu ausführen sollten.

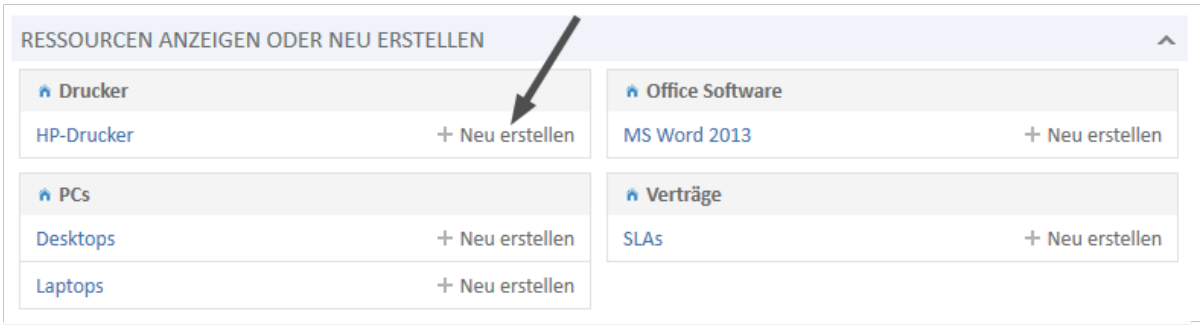


Abbildung 595: *ConSol CM Web Client - Ressourcenpool-Dashboard: Erstellen einer neuen Ressource des Typs HP_Printer, Schritt 1*

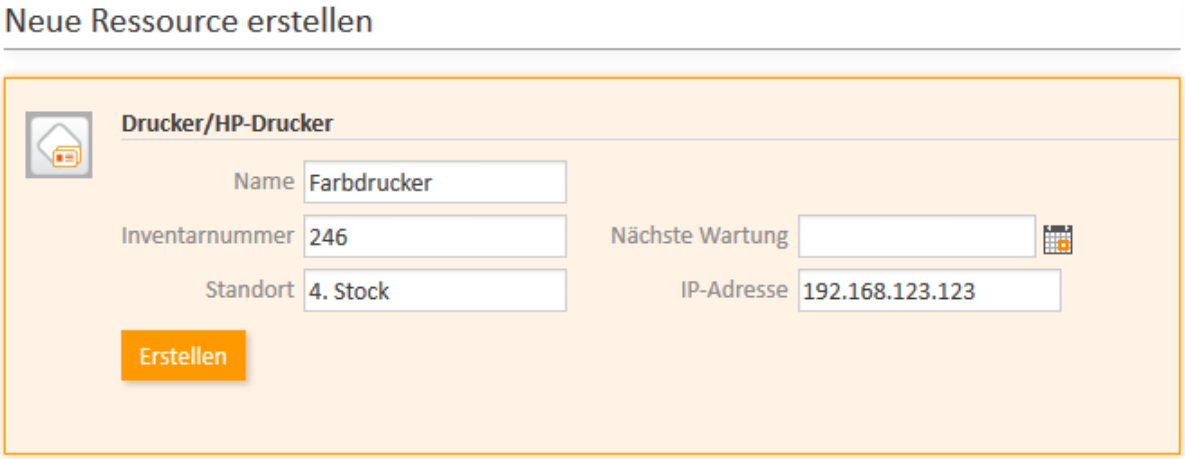


Abbildung 596: *ConSol CM Web Client - Erstellen einer neuen Ressource des Typs HP_Printer, Schritt 2*

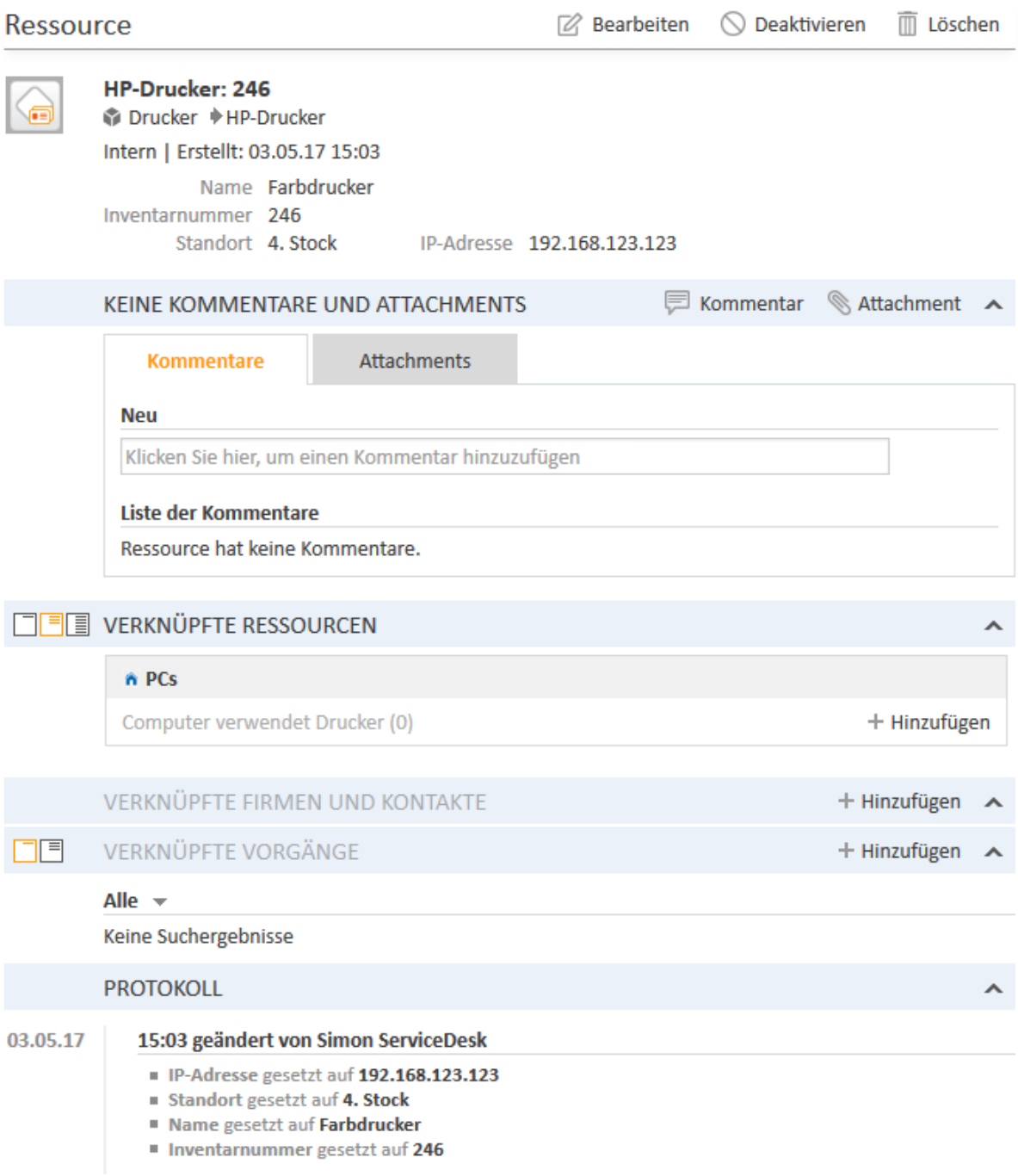


Abbildung 597: *ConSol CM Web Client - Ressourcenpool-Dashboard: Erstellen einer neuen Ressource des Typs HP_Printer, Schritt 3*

Jetzt haben Sie ein neues Objekt des Typs *HP-Drucker* erstellt, d. h. Sie haben den Drucker Nr. 246 angelegt, der sich im 4. Stock befindet. Die Ressourcenseite dieses speziellen Druckers wird angezeigt (siehe obige Abbildung).

Eine detaillierte Einführung in die Arbeit mit CM/Resource Pool im Web Client finden Sie im *ConSol CM Benutzerhandbuch*. Dieses Kapitel enthält lediglich eine kurze Erklärung, damit Sie als Administrator wissen, welche Auswirkungen die Admin-Tool-Konfiguration auf die ConSol CM-Ressourcen hat.

So verändern Sie als Administrator das Layout des Bereichs *Ressourcen anzeigen oder neu erstellen* im Ressourcenpool-Dashboard

- Die Rollen eines Bearbeiters bestimmen, ob er Zugangsberechtigungen für einen Ressourcentyp hat. Daher sieht der aktuell angemeldete Bearbeiter nur die Ressourcentypen, für die er Zugangsberechtigungen besitzt.
- Der Name des Ressourcentyps wird als Hyperlink angezeigt, der auf die Ressourcentypseite führt. Dieser Hyperlink wird nur aktiv, wenn der Bearbeiter die Zugangsberechtigung *Lesen* für den entsprechenden Ressourcentyp hat.
- Der Link *Neu erstellen* für einen Ressourcentyp wird nur angezeigt, wenn der Bearbeiter die Zugangsberechtigung *Erzeugen* für diesen Ressourcentypen hat.

So ändern Sie als Administrator das Layout der Ressourcenseite

- Im oberen Bereich der Ressourcenseite werden die lokalisierten Werte der Ressourcenfelder angezeigt (analog zu den Ticketdaten in Ticketfeldern eines Tickets). Daten der Ressourcenfelder können auch in Tabs angezeigt werden, die die Ressourcenfeldgruppen darstellen (ebenfalls analog zu den Tabs auf der Ticketseite, die es für Ticketfeldgruppen gibt). Die Ressourcenfelder werden entsprechend dem definierten Layout angezeigt. Siehe Abschnitt CM/Resource Pool - Einrichten des Ressourcenmodells.
- Das Icon für jede Ressource eines Ressourcentyps kann im Admin Tool für jeden einzelnen Ressourcentyp definiert werden.
- Das Template für die Anzeige des Ressourcennamens wird für den Ressourcentyp definiert. Hier wird auf der Ressourcenseite das Template *Standard* verwendet, siehe CM/Resource Pool - Templates für Ressourcendaten.
- Die Menüeinträge (Links) in der oberen rechten Ecke der Seite werden nur angezeigt, wenn der Bearbeiter die entsprechenden Zugangsberechtigungen hat:
  - Link *Bearbeiten*: Berechtigung *Schreiben*
  - Link *Deaktivieren*: Berechtigung *Deaktivieren / Aktivieren*
  - Link *Löschen*: Berechtigung *Löschen*
- Wenn Ressourcenaktionen für Ressourcen dieses Typs definiert sind (nicht in der obigen Abbildung gezeigt), werden diese ebenfalls auf der Ressourcenseite angezeigt, so wie die Workflow-Aktivitäten für Tickets. Eine detaillierte Erklärung dazu finden Sie im Abschnitt CM/Resource Pool - Ressourcenaktionen. Der Bearbeiter sieht die Ressourcenaktionen allerdings nur, wenn er die Berechtigung *Ausführen* für den entsprechenden Ressourcentyp hat.
- Der Bereich für Kommentare und Attachments wird nicht angezeigt, wenn die Berechtigung *Details lesen* nicht in den Rollen des Bearbeiters enthalten ist. Siehe Abschnitt CM/Resource Pool - Zuweisen von Berechtigungen für Ressourcen.

- Wenn im Admin Tool Relationen zu anderen Ressourcen, Kontakten, Firmen oder Tickets definiert wurden, können diese in den entsprechenden Bereichen der Ressourcenseite angezeigt werden, sofern reale Relationen für die Ressource von einem Bearbeiter erstellt wurden. Details finden Sie im Abschnitt CM/Resource Pool - Ressourcenrelationen.

G.1.3.3 Arbeiten mit der Ressourcentypseite

Die Ressourcentypseite bietet eine Übersicht über alle Ressourcen von einem bestimmten Ressourcentyp. Sie können die Ressourcentypseite öffnen, indem Sie im Ressourcenpool-Dashboard auf den Namen des Ressourcentyps klicken.



Abbildung 598: *ConSol CM Web Client - Öffnen einer Ressourcentypseite*

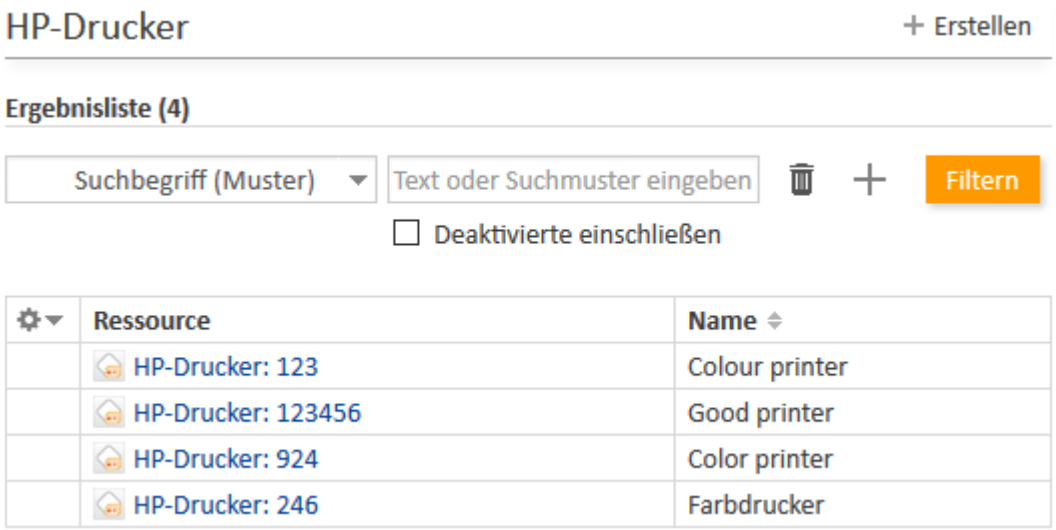


Abbildung 599: *ConSol CM Web Client - Ressourcentypseite*

Auf der Ressourcentypseite sind alle Ressourcen dieses Typs aufgeführt. Sie können diese nach den Werten in den Ressourcenfeldern filtern, indem Sie die Suche im oberen Bereich der Seite verwenden.

So ändern Sie als Administrator das Layout der Ressourcentypseite

- Im Drop-down-Menü der Suche werden nur Felder angeboten, die indiziert wurden. Siehe Abschnitt Konfiguration der Suche.
- Die Kopfzeilen der Spalten zeigen die lokalisierten Werte der Namen der Ressourcenfelder, die Sie (oder ein anderer Administrator) definiert haben. Siehe Abschnitt CM/Resource Pool - Einrichten des Ressourcenmodells.

G.1.3.4 Hinzufügen einer Ressource zu den Favoriten

Wie Tickets, Suchen und Kunden können auch Ressourcen mit Drag-and-Drop in die Favoriten übernommen werden, z. B. auf der Ressourcenseite.



Abbildung 600: *ConSol CM Web Client - Ressourcenseite: Hinzufügen einer Ressource zu den Favoriten mittels Drag-and-Drop*

So ändern Sie als Administrator das Anzeigeformat der Ressource beim Drag-and-Drop in die Favoriten

- Das Format für die Anzeige der Ressource in den Favoriten basiert auf dem Template *Standard*, siehe Abschnitt CM/Resource Pool - Templates für Ressourcendaten.
- Das Format für die Anzeige der Ressource im Drag-and-Drop-Feld basiert auf dem Template *Standard*, siehe Abschnitt CM/Resource Pool - Templates für Ressourcendaten.

G.1.3.5 Definieren und Verwenden von Ressourcenrelationen

Das gleiche Prinzip wie für Ressourcenkategorien und Ressourcentypen gilt auch für Ressourcenrelationen:

- Zuerst müssen die Ressourcenrelationen im Datenmodell im Admin Tool definiert werden.
- Danach können reale Relationen zwischen Ressourcen (Objekten eines bestimmten Ressourcentyps) und anderen Objekten (Tickets, Kunden und anderen Ressourcen) im Web Client erstellt werden.

Dies wird im Abschnitt CM/Resource Pool - Ressourcenrelationen näher erklärt. An dieser Stelle finden Sie nur ein kurzes Beispiel.

Um eine Ressource (z. B. einen PC-Desktop) mit Tickets in der Queue *ServiceDesk* verknüpfen zu können, muss dieser Relationstyp (d. h. Ressource-Ticket) zuerst im Admin Tool definiert werden.

Dann kann ein Bearbeiter eine Relation von einem PC zu einem Ticket erstellen, z. B. wenn ein Vorfallticket hereinkommt, das ein Problem mit diesem Rechner behandelt.

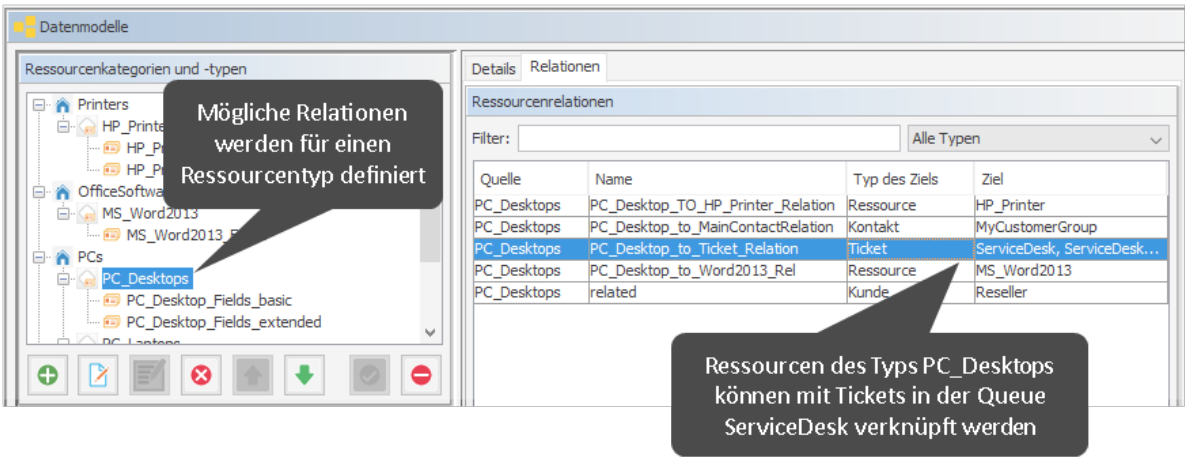


Abbildung 601: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Definieren einer neuen Ressourcenrelation*

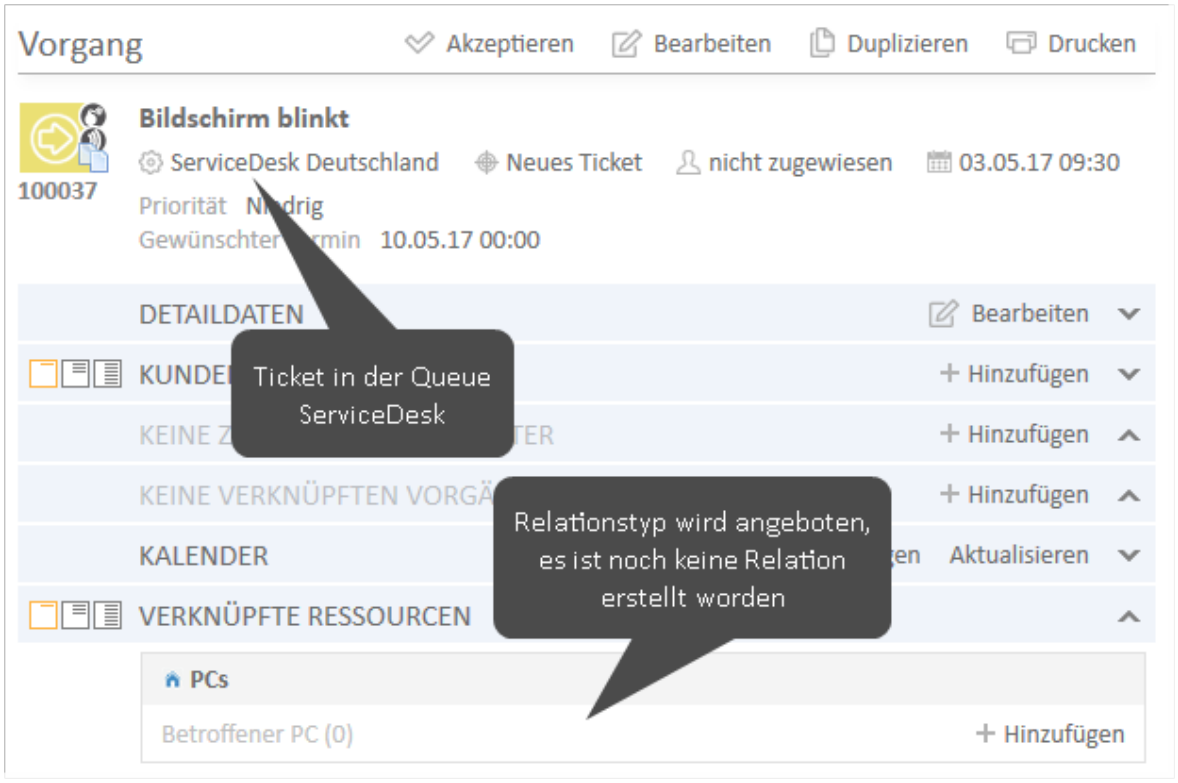


Abbildung 602: *ConSol CM Web Client - Ticket, in dem eine Relation zu einem PC erstellt werden kann*

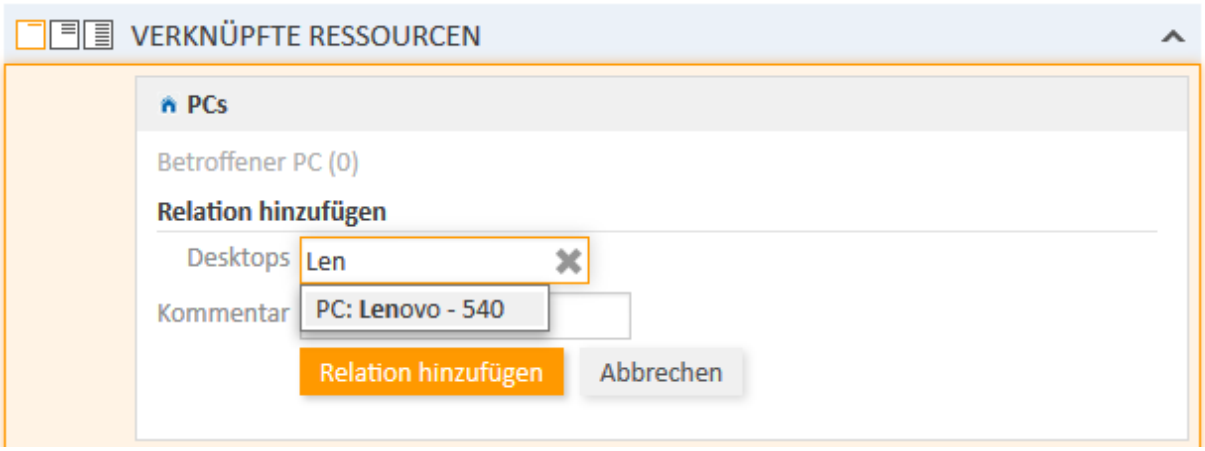


Abbildung 603: *ConSol CM Web Client - Erstellen einer Relation zwischen einem Ticket und einem PC*

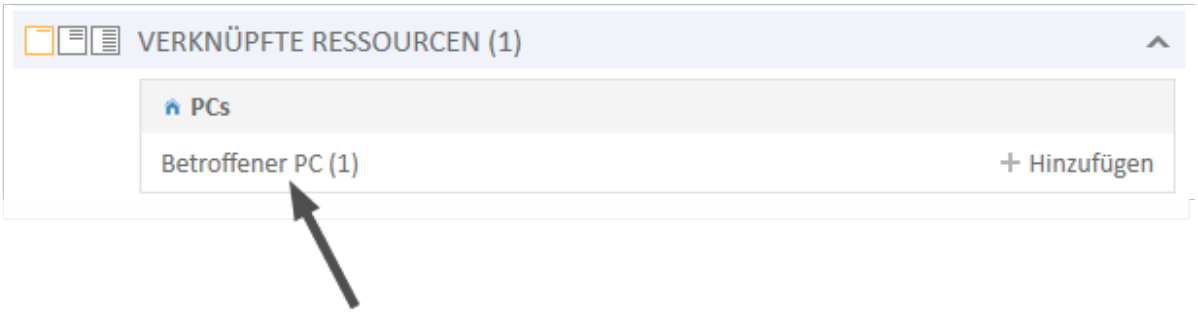


Abbildung 604: *ConSol CM Web Client - Ticket mit einer Relation zu einem PC*

Wenn eine Relation erstellt wurde, wird der Zähler der verknüpften Ressourcen auf 1 gesetzt. Der Bearbeiter kann dann die Details der Relation anzeigen, indem er auf den Namen der Relation klickt.

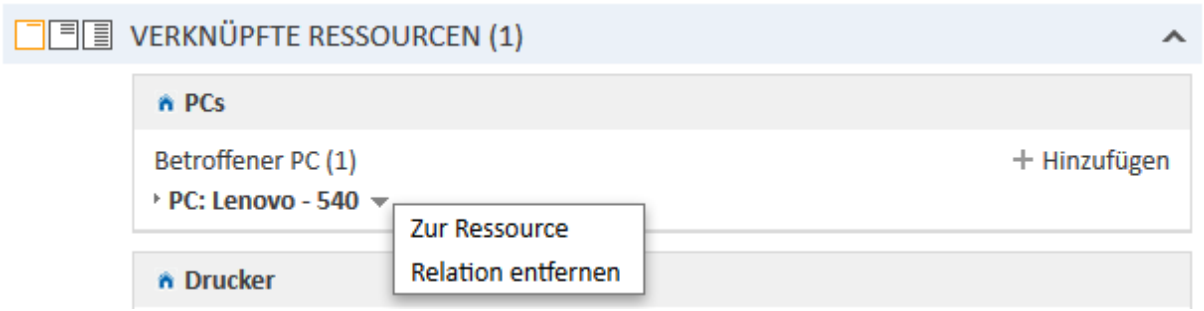


Abbildung 605: *ConSol CM Web Client - Details einer verknüpften Ressource und Kontextmenü*

Das Kontextmenü der verknüpften Ressource enthält zwei Optionen:

- **Zur Ressource**
  Öffnet die Ressourcenseite der verknüpften Ressource
- **Relation entfernen**
  Entfernt die Relation des Tickets zur Ressource

So ändern Sie als Administrator die Anzeige von Ressourcenrelationen

- Der Name und die Beschreibung, die für die Relation angezeigt werden, sind die lokalisierten Werte des im Admin Tool definierten Relationsnamens und der Beschreibung. Beachten Sie dabei, dass es eine Beschreibung für die Quellseite und eine Beschreibung für die Zielseite einer Relation gibt.
- Eine Anmerkung für eine Ressourcenrelation kann im Web Client nur hinzugefügt werden, wenn die Checkbox *Anmerkungsfeld verfügbar* in der Definition der Ressourcenrelation im Admin Tool aktiviert ist.
- Wenn bereits eine Ressource eines Typs zu einem Objekt (z. B. einem Ticket) hinzugefügt wurde, ist die Option *Hinzufügen* nur verfügbar, wenn die Kardinalität der Relation auf *m zu 1, 1 zu m* oder *m zu n* gesetzt ist (z. B. um eine weitere Ressource des Typs *PC* hinzuzufügen).

G.1.3.6 Definieren und Verwenden von Ressourcenaktionen

Die Ressourcenaktionen müssen ebenfalls im Admin Tool definiert werden. Jede Ressourcenaktion basiert auf einem Admin-Tool-Skript.

Ressourcenaktionen werden im Abschnitt CM/Resource Pool - Ressourcenaktionen näher erklärt. An dieser Stelle finden Sie nur ein kurzes Beispiel.

Eine Ressourcenaktion wird immer im Navigationselement *Aktionen* der Navigationsgruppe *Ressourcen* definiert und danach einem oder mehreren Ressourcentypen zugewiesen. Es wird zum Beispiel eine Aktion definiert, die die Möglichkeit bieten soll, direkt auf der Ressourcenseite ein neues Ticket in der Queue *ServiceDesk* zu erstellen. Wenn dann ein Kunde anruft, um sich über ein Problem mit dem PC zu beschweren, kann der ServiceDesk-Mitarbeiter zuerst die Detailseite des PCs aufrufen, um sich über das Gerät zu informieren. Falls nötig kann er danach direkt ein neues ServiceDesk-Ticket für den Kunden erstellen.

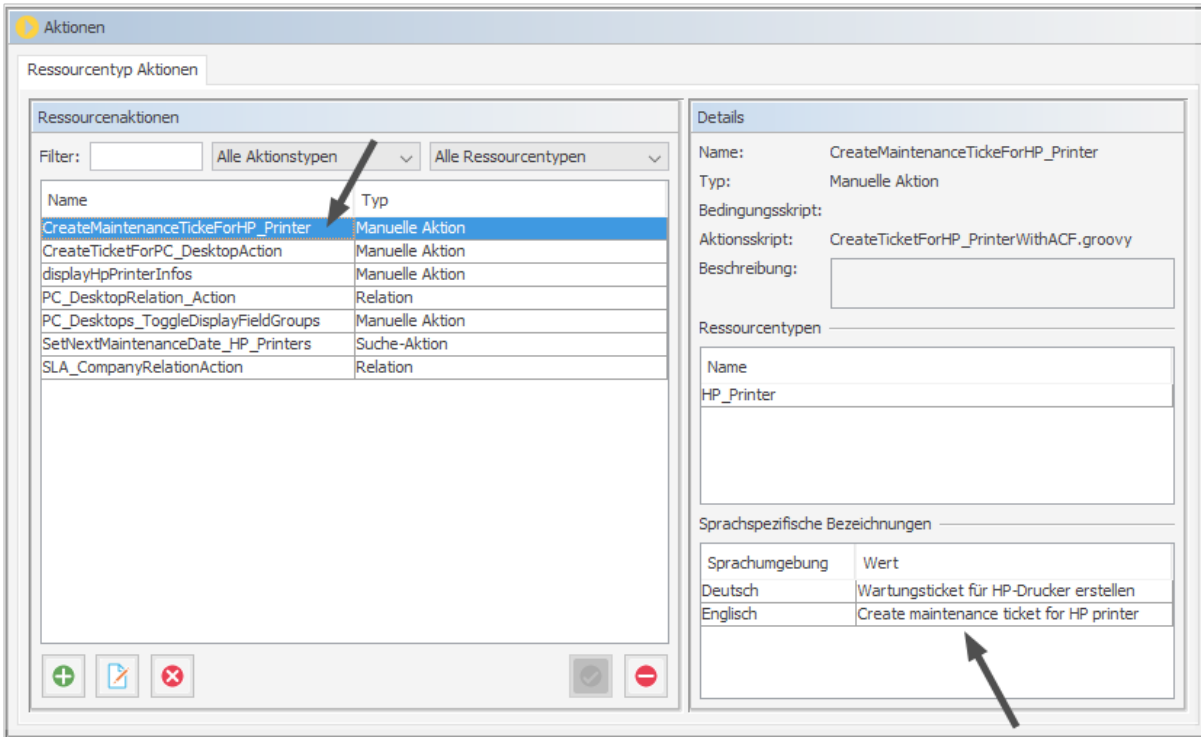


Abbildung 606: *ConSol CM Admin Tool - Ressourcen, Aktionen: Definition einer Ressourcenaktion*

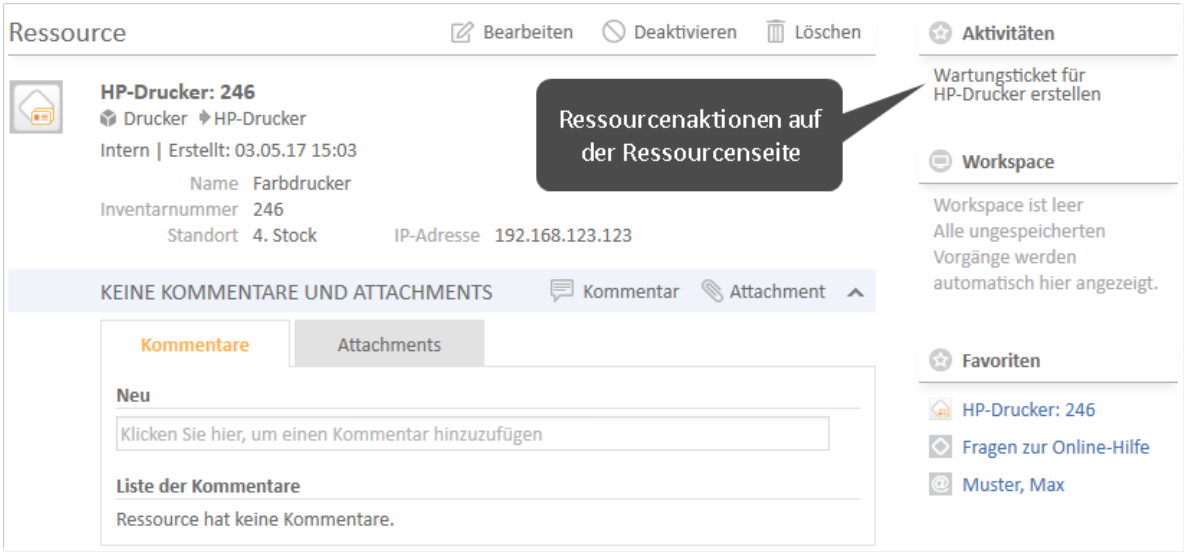


Abbildung 607: *ConSol CM Web Client - Ressourcenaktivität (basierend auf Ressourcenaktion)*

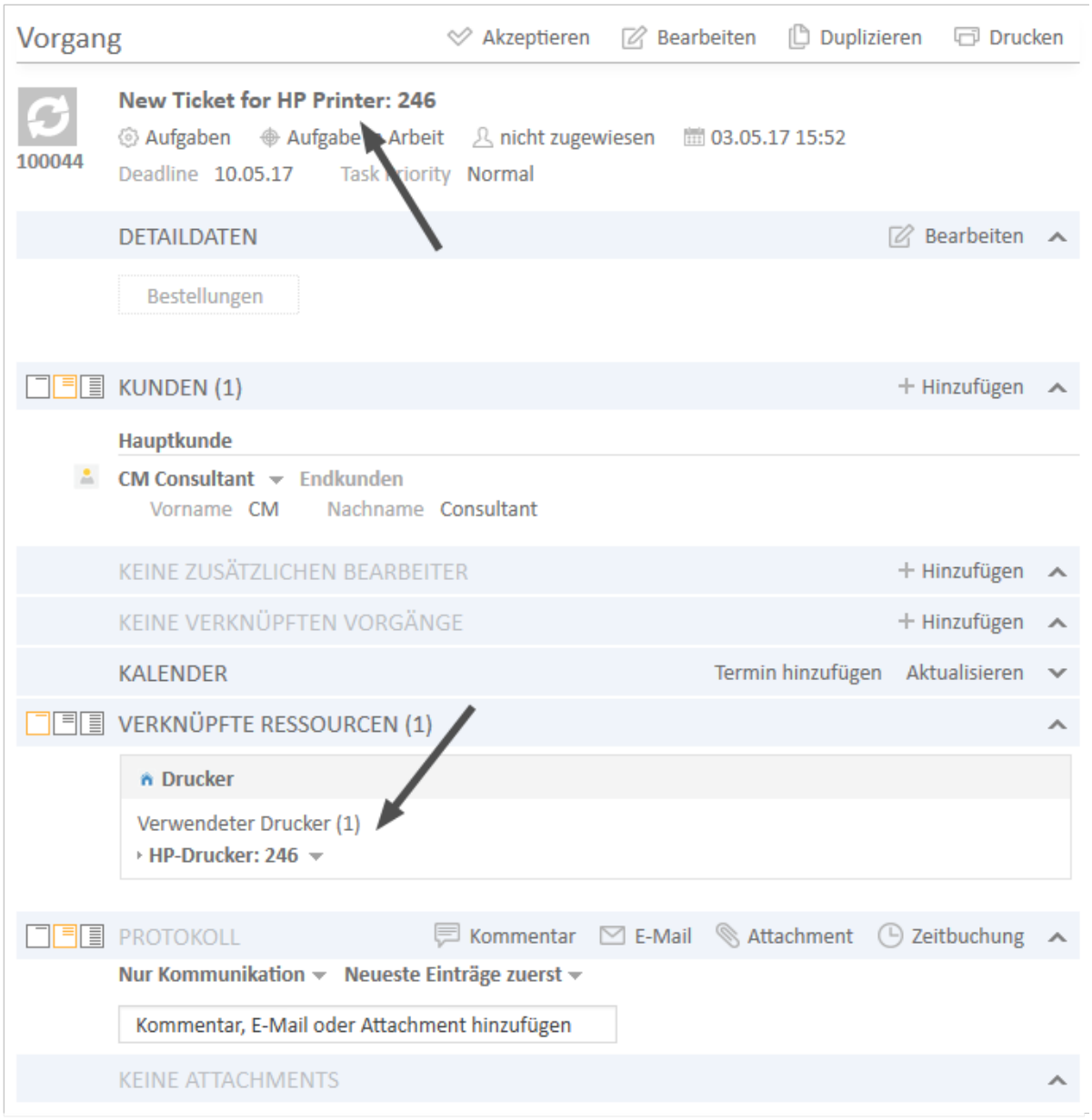


Abbildung 608: *ConSol CM Web Client - Mithilfe einer Ressourcenaktion erstelltes Aufgaben-Ticket*

So ändern Sie als Administrator die Anzeige von Ressourcenaktionen

- Der Name der im Web Client angezeigten Ressourcenaktivität ist der lokalisierte Name der Ressourcenaktion, die Sie als Administrator im Abschnitt *Ressourcenaktionen* definiert haben. Details dazu finden Sie im Abschnitt CM/Resource Pool - Ressourcenaktionen.
- Die lokalisierten Werte der Beschreibung der Ressourcenaktion werden als Mouseover für die Ressourcenaktivität im Web Client angezeigt.

G.1.3.7 Finden von Ressourcen mit der Schnellsuche

In der Ergebnisliste der Schnellsuche werden Ressourcen in einem eigenen Bereich angezeigt:



Abbildung 609: *ConSol CM Web Client - Ergebnisliste der Schnellsuche mit einem eigenen Ressourcenbereich*

So ändern Sie als Administrator die Anzeige von Ressourcen in der Schnellsuche

- Das Anzeigeformat der Ressource in der Ergebnisliste der Schnellsuche basiert auf dem Template *Schnellsuche*. Siehe Abschnitt CM/Resource Pool - Templates für Ressourcendaten.
- Der Wert der System-Property cmweb-server-adapter, globalSearchResultSizeLimit definiert die maximale Trefferanzahl in der Ergebnisliste.

G.1.3.8 Finden von Ressourcen mit der Detailsuche

In der Detailsuche kann ein Bearbeiter nach allen Ressourcen eines bestimmten Typs suchen, vorausgesetzt das entsprechende Ressourcenfeld ist indiziert (die Annotation `field indexed` ist gesetzt, normalerweise auf den Wert „transitive“).

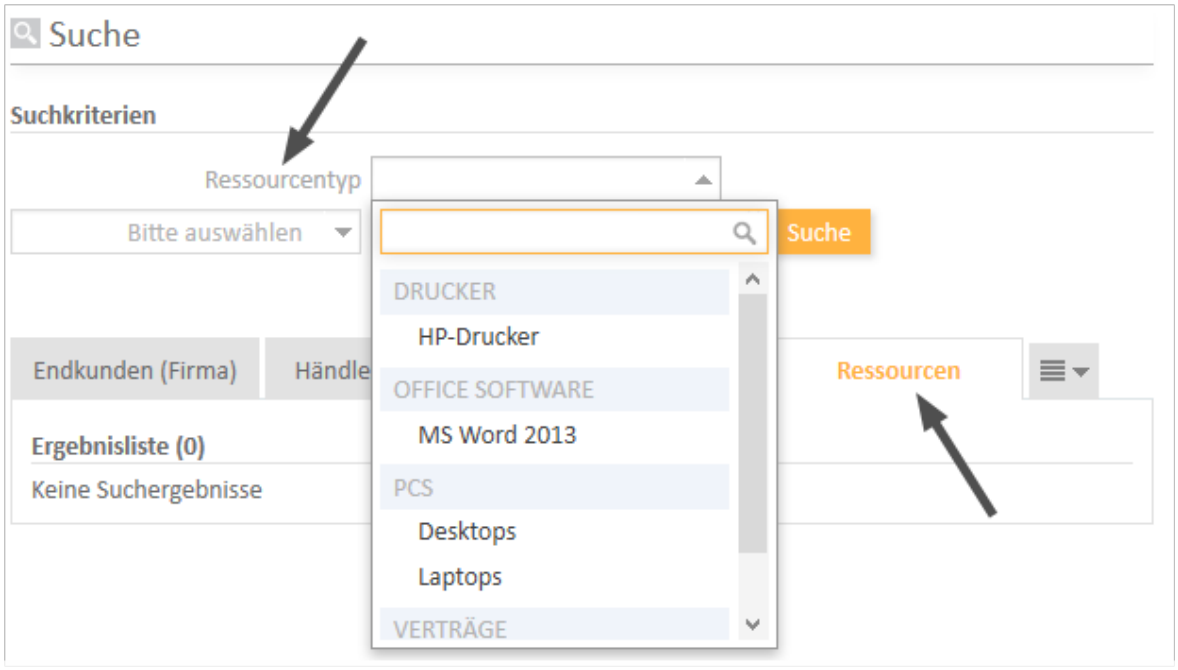


Abbildung 610: *ConSol CM Web Client - Detailsuche nach Ressourcen*

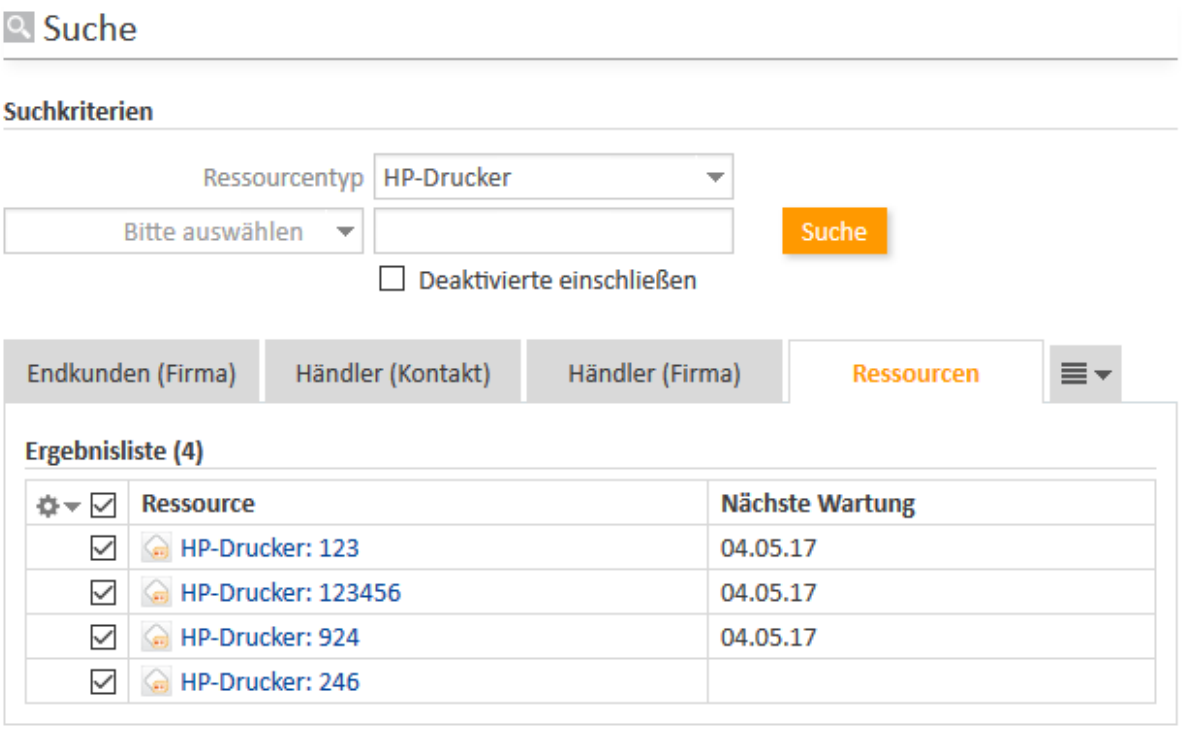


Abbildung 611: *ConSol CM Web Client - Ergebnisliste der Detailsuche nach Ressourcen*

So ändern Sie als Administrator die Anzeige von Ressourcen in der Detailsuche

- Auf der Seite der Detailsuche sind nur die Ressourcentypen im entsprechenden Drop-down-Menü aufgeführt, die die Annotation `field indexed`, normalerweise mit dem Wert „transitive“, haben.
- Der Wert der System-Property cmweb-server-adapter, searchPageSize definiert die Anzahl der Treffer, die am Anfang angezeigt werden.



Falls Suche-Aktionen für einen Ressourcentyp definiert sind, werden diese dem Bearbeiter als Aktivitäten im Web Client für Such-Ergebnislisten angeboten, sofern die Liste Ressourcen dieses Typs enthält. Details zu diesem Thema finden Sie im Abschnitt Action Framework - Suche-Aktionen.

G.1.4 CM/Resource Pool - Einrichten des Ressourcenmodells

G.1.4.1 Einleitung

Dieses Kapitel beschreibt die komplette Einrichtung eines Ressourcenmodells. Sie lernen dabei Folgendes:

- Erstellen einer Ressourcenkategorie
- Erstellen von zwei Ressourcentypen innerhalb dieser Ressourcenkategorie
- Erstellen von Datenfeldern für die Ressourcen, d. h. von Ressourcenfeldgruppen für die Ressourcentypen
- Erstellen von Ressourcenfeldern innerhalb der Ressourcenfeldgruppen, d. h. Erstellen der Datenfelder
- Definieren von Templates für das Anzeigeformat der Ressourcennamen im Web Client

In unserem Beispiel sind Sie ein Administrator, der ein Ressourcenmodell für ein Unternehmen aufsetzt, das seine IT-Assets und seine Produkte im Ressourcenpool verwalten möchte.

Das vollständige Modell ist in der folgenden Abbildung dargestellt. Anhand der Definition der Drucker und Office-Software zeigen wir Ihnen das Grundprinzip. Danach sollten Sie in der Lage sein, alle verbleibenden Ressourcen selbst zu konfigurieren.

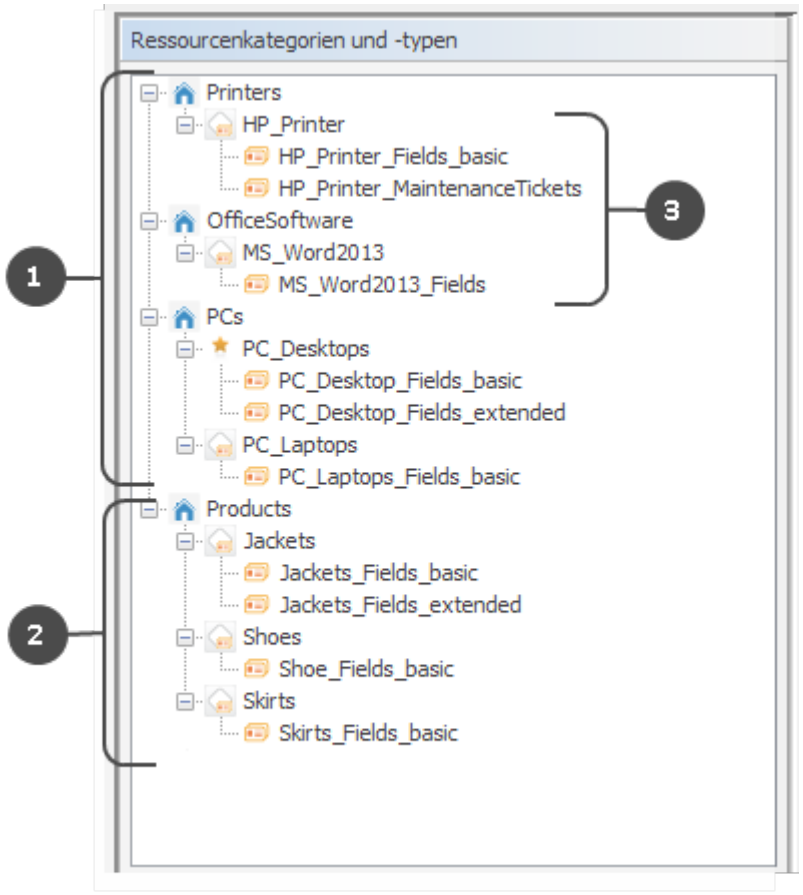


Abbildung 612: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Beispiel für Ressourcenmodell*

Die obige Abbildung zeigt ein Modell für Ressourcenpool-Daten mit folgenden Elementen:

- Ressourcen für das IT-Management (1)
- Ressourcen für das Produktmanagement (2)

Im folgenden Beispiel werden Drucker und Office-Software (3) behandelt.

G.1.4.2 Erstellen einer Ressourcenkategorie mit dem ersten Ressourcentyp

Eine Ressourcenkategorie stellt die Objekte auf der obersten Hierarchiestufe im Datenmodell des Ressourcenpools dar. In unserem Beispiel werden die Drucker in der Ressourcenkategorie *Printers* und die Office-Softwarepakete in der Ressourcenkategorie *OfficeSoftware* verwaltet. Die folgende Abbildung zeigt einige Ressourcenkategorien und Ressourcentypen im Web Client. Diese soll Ihnen dabei helfen, den Überblick über die Konfiguration im Admin Tool und ihre Auswirkungen auf die GUI zu behalten. Denken Sie daran, dass die im Admin Tool verwendeten technischen Namen normalerweise nicht die (lokalisierten) Namen sind, die im Web Client angezeigt werden.

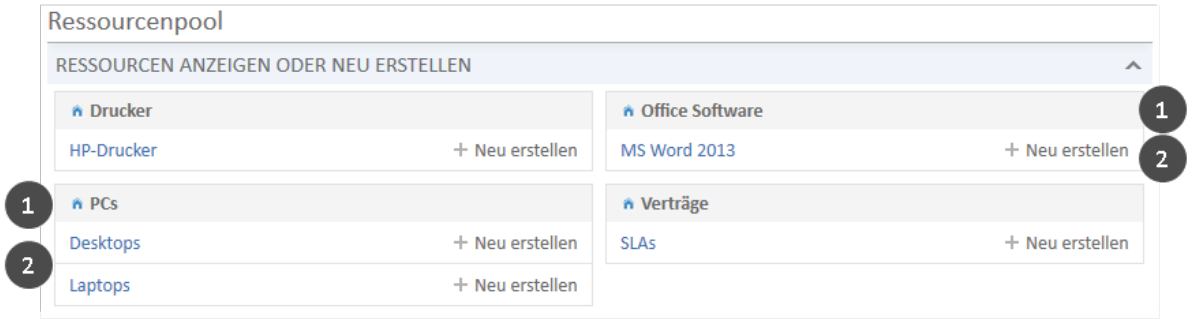


Abbildung 613: *ConSol CM Web Client - Ressourcen im Ressourcenpool-Dashboard*

- Ressourcenkategorien (1)
- Ressourcentypen (2)

Der erste Schritt bei einem neuen (leeren) Ressourcenmodell ist die Erstellung einer neuen Ressourcenkategorie, die bereits einen Ressourcentyp enthält. Der Ressourcentyp enthält wiederum bereits eine Ressourcenfeldgruppe.

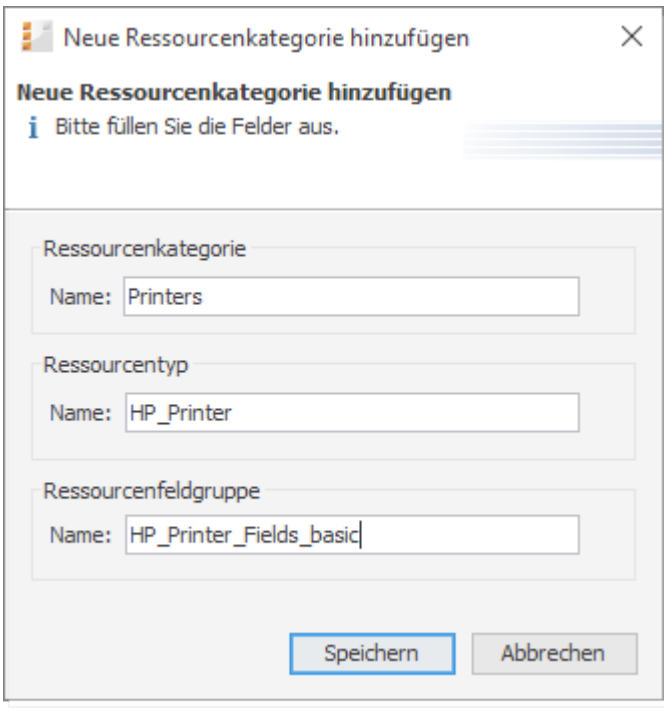


Abbildung 614: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Erstellen einer Ressourcenkategorie*

Füllen Sie zum Erstellen einer Ressourcenkategorie folgende Felder aus. Die Lokalisierung der Namen für die Ressourcenkategorie, den Ressourcentyp und die Ressourcenfeldgruppe erfolgt mithilfe der Felder im Navigationselement *Datenmodelle* und ist im Abschnitt Lokalisierung von Datenfeldern erklärt. Die Beschreibungen werden über den Button *Lokalisieren* lokalisiert, was im Abschnitt Lokalisierung von Objekten allgemein, Typ 1 erklärt ist.

- **Ressourcenkategorie**, Name
  Der technische Name der Ressourcenkategorie.
- **Ressourcentyp**, Name
  Der technische Name des ersten Ressourcentyps innerhalb der Ressourcenkategorie.
  Später können weitere Ressourcentypen für die Ressourcenkategorie hinzugefügt werden.
- **Ressourcenfeldgruppe**, Name
  Der Name der ersten Ressourcenfeldgruppe (ähnlich wie eine Ticketfeldgruppe für Ticketdaten) innerhalb des Ressourcentyps. Falls die Ressourcenfeldgruppe im Bereich für Detaildaten der Ressourcenseite angezeigt wird (Annotation `show-in-group-section` = „true“), wird der lokalisierte Name als Titel des entsprechenden Tabs verwendet.
  Später können weitere Ressourcenfeldgruppen für den Ressourcentyp hinzugefügt werden.

Klicken Sie auf *Speichern*, um die Ressourcenkategorie, den Typ und die erste Feldgruppe zu erstellen.

Wenn Sie die Ressourcenkategorie definiert haben, können Sie weitere Daten für die erstellten Objekte hinzufügen.

Editieren der Ressourcenkategorie

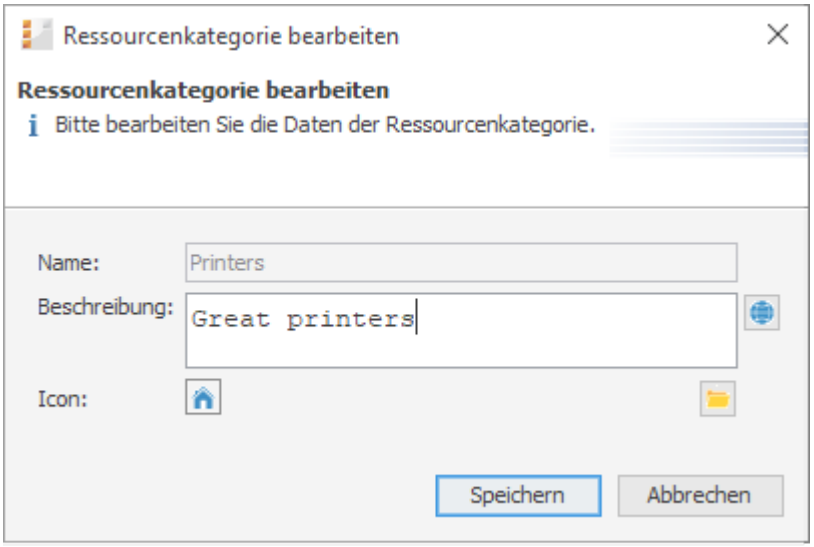


Abbildung 615: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Editieren einer Ressourcenkategorie*

Markieren Sie die Ressourcenkategorie und klicken Sie auf den Button *Bearbeiten*, um einige Parameter der Ressourcenkategorie zu editieren:

- **Beschreibung**
  Sie können direkt im Pop-up-Fenster eine Beschreibung eingeben, die danach als Standardbeschreibung dient. Alternativ können Sie über den Button *Lokalisieren* die lokalisierten Werte für die einzelnen Sprachen eingeben. Die Beschreibung der Ressourcenkategorie wird nur zu Informationszwecken im Admin Tool angezeigt, sie ist im Web Client nicht zu sehen.

- **Icon**
  Wählen Sie eines der CM-Standardicons, indem Sie auf das Icon klicken, oder laden Sie über den Dateibrowser ein Icon Ihrer Wahl hoch. Das Icon der Ressourcenkategorie wird im Web Client neben dem Namen der Ressourcenkategorie angezeigt. Auf diese Weise können Sie zum Beispiel für die Drucker ein Drucker-Icon auswählen, für die PCs ein PC-Icon usw. Die zulässigen Formate sind: jpg, jpeg, png, gif. Wir empfehlen eine Bildgröße von 32 x 32 px, um eine gute Darstellung auf der Ressourcentypseite zu erreichen. In Tabellen und an anderen Stellen, an denen die Ressource auf der GUI angezeigt wird, wird das Icon automatisch (kleiner) skaliert.

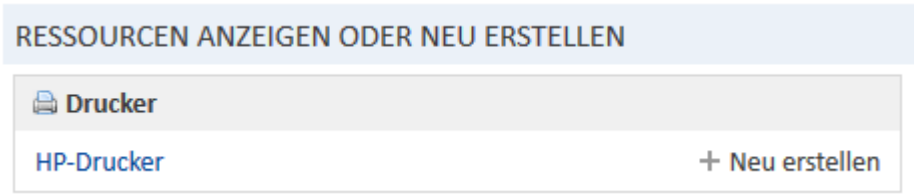


Abbildung 616: *ConSol CM Web Client - Angepasstes Icon für eine Ressourcenkategorie*

Klicken Sie auf *Speichern*, um Ihre Änderungen zu übernehmen.

Editieren des Ressourcentyps

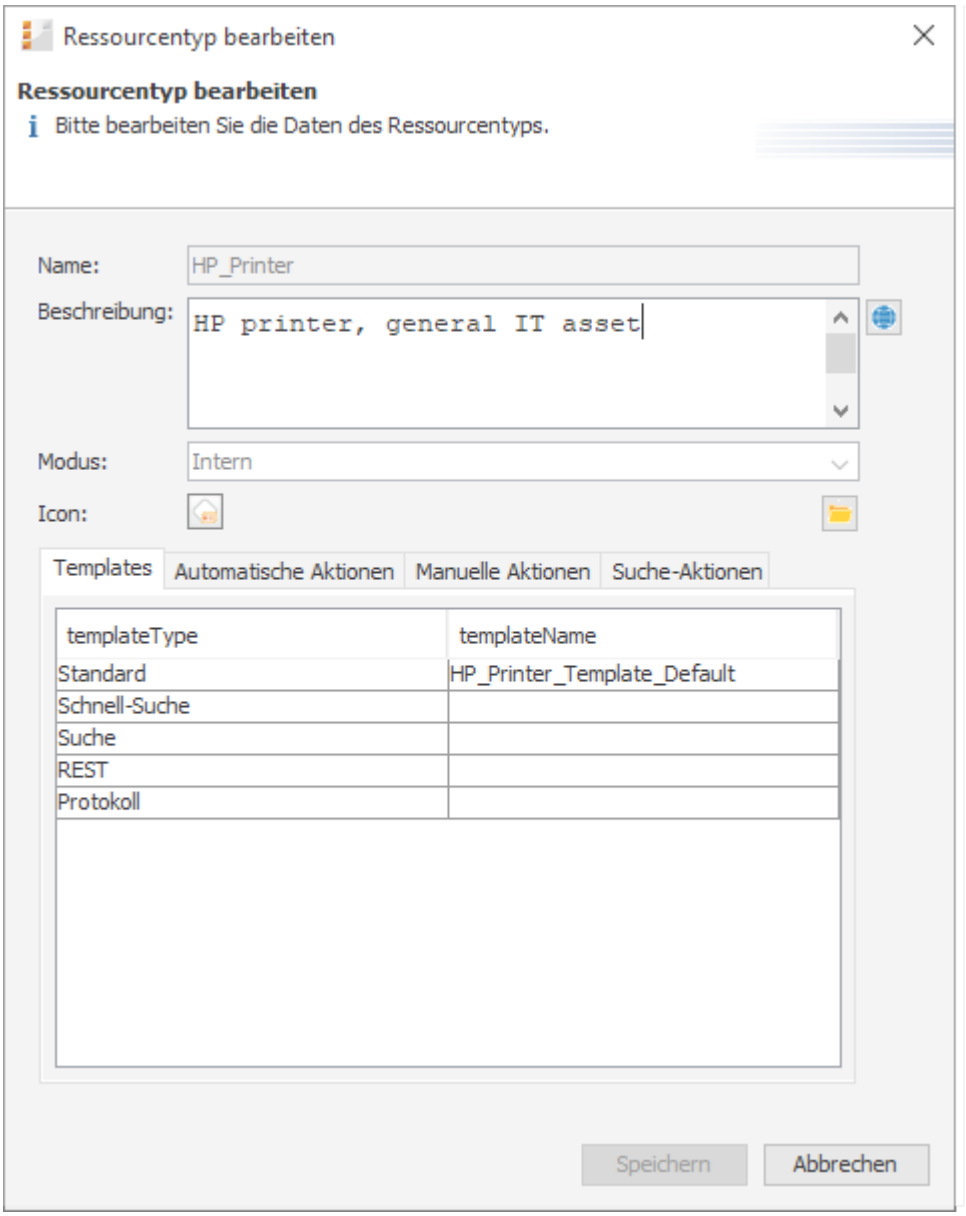


Abbildung 617: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Editieren eines Ressourcentyps*

Markieren Sie den Ressourcentyp und klicken Sie auf den Button *Bearbeiten*, um einige Parameter des Ressourcentyps zu editieren:

- **Beschreibung**
  Sie können direkt im Pop-up-Fenster eine Beschreibung eingeben, die danach als Standardbeschreibung dient. Alternativ können Sie über den Button *Lokalisieren* die lokalisierten Werte für die einzelnen Sprachen eingeben. Die Beschreibung des Ressourcentyps wird nur zu Informationszwecken im Admin Tool angezeigt, sie ist im Web Client nicht zu sehen.

- **Modus** (eine detaillierte Erklärung aller Modi finden Sie im Abschnitt Information über Ressourcenmodi)
  - **Intern**
    Die Ressourcendaten werden komplett in ConSol CM verwaltet. Details zu diesem Modus finden Sie im Abschnitt Interne Ressourcen.
  - **Auf Anforderung**
    Die Ressourcendaten werden auf Anforderung aus einem externen Backend-System abgerufen (verfügbar in ConSol CM 6.10.3.0 und höher). Details zu diesem Modus finden Sie im Ressourcen auf Anforderung.
  - **Cached**
    Die Ressourcendaten stammen aus einem externen Backend-System und werden in ConSol CM zwischengespeichert. Eine Aktualisierung eines Elements kann manuell angestoßen werden (verfügbar in ConSol CM 6.10.3.0 und höher). Details zu diesem Modus finden Sie im Abschnitt Ressourcen im Cache.
  - **Importiert**
    Die Ressourcendaten stammen aus einem externen Backend-System und werden in ConSol CM gespeichert. Für eine Aktualisierung ist ein neuer Massendatenimport erforderlich (verfügbar in ConSol CM 6.10.3.0 und höher). Details zu diesem Modus finden Sie im Abschnitt Importierte Ressourcen.
- **Icon**
  Wählen Sie eines der CM-Standardicons, indem Sie auf das Icon klicken, oder laden Sie über den Dateibrowser ein Icon Ihrer Wahl hoch. Das Icon des Ressourcentyps wird im Web Client neben dem Namen des Ressourcentyps angezeigt, z. B. auf der Ressourcentypseite. Auf diese Weise können Sie zum Beispiel für die Drucker ein Drucker-Icon auswählen, für die PCs ein PC-Icon usw. Die zulässigen Formate sind: jpg, png, gif. Wir empfehlen eine Bildgröße von 32 x 32 px, um eine gute Darstellung auf der Ressourcenseite zu erreichen. In Tabellen und an anderen Stellen, an denen die Ressource auf der GUI angezeigt wird, wird das Icon automatisch (kleiner) skaliert.
- **Templates**
  Hier können Sie die Templates für das Anzeigeformat der Ressourcendaten im Web Client definieren (analog zu Templates für Kundendaten). Dies wird im Abschnitt CM/Resource Pool - Templates für Ressourcendaten näher erklärt.
- **Automatische Aktionen**
  Hier können Sie dem Ressourcentyp automatische Ressourcenaktionen zuweisen. Die Ressourcenaktionen müssen zuerst im Bereich *Ressourcenaktionen* des Admin Tools definiert werden. Eine detaillierte Erklärung dazu finden Sie im Abschnitt CM/Resource Pool - Ressourcenaktionen.
- **Manuelle Aktionen**
  Hier können Sie dem Ressourcentyp manuelle Ressourcenaktionen zuweisen. Die Ressourcenaktionen müssen zuerst im Bereich *Ressourcenaktionen* des Admin Tools definiert werden. Eine detaillierte Erklärung dazu finden Sie im Abschnitt CM/Resource Pool - Ressourcenaktionen.

- **Suche-Aktionen**
  Hier können Sie dem Ressourcentyp Suche-Aktionen zuweisen. Die Suche-Aktionen müssen zuerst im Bereich *Skripte und Templates* des Admin Tools definiert werden. Eine detaillierte Erklärung dazu finden Sie im Abschnitt Action Framework - Suche-Aktionen.

Die Ressourcenrelationen werden ebenfalls für den Ressourcentyp definiert. Dies wird im Abschnitt CM/Resource Pool - Ressourcenrelationen näher erklärt.

Information über Ressourcenmodi

Der Modus eines Ressourcentypen definiert, ob die Ressourcendaten im ConSol CM-System gespeichert sind und, sofern erforderlich, wie die Übertragung von einem externen System an ConSol CM verwaltet wird. Wenn der Modus festgelegt wurde und es im System Ressourcen des entsprechenden Typs gibt, kann der Modus nicht mehr geändert werden.

In ConSol CM wird zwischen *internen Ressourcen* und *externen Ressourcen* unterschieden. Die letzteren umfassen *Ressourcen auf Anforderung*, *Ressourcen im Cache* und *Importierte Ressourcen*, d. h. alle Ressourcen, die nicht komplett in ConSol CM gespeichert und verwaltet werden.

Im Web Client können nur interne Ressourcen erstellt werden. Externe Ressourcen können mit dem Web Client weder erstellt, noch (permanent) geändert, noch gelöscht werden. Für alle externen Ressourcen muss das komplette Datenmodell im Admin Tool definiert werden, bevor die erste Datenübertragung ausgeführt werden kann. Die Verwendung von *Ressourcen auf Anforderung* und *Ressourcen im Cache* basiert auf der Implementierung eines speziellen Interface. Daher können diese Ressourcen nur mit einer kundenspezifischen .ear-Datei von ConSol CM verwendet werden. Das entsprechende Entwicklungsprojekt muss gebaut und installiert werden, bevor die erste Datenübertragung stattfindet. Die folgenden Abschnitte enthalten eine detaillierte Übersicht über alle vier Ressourcentypen.

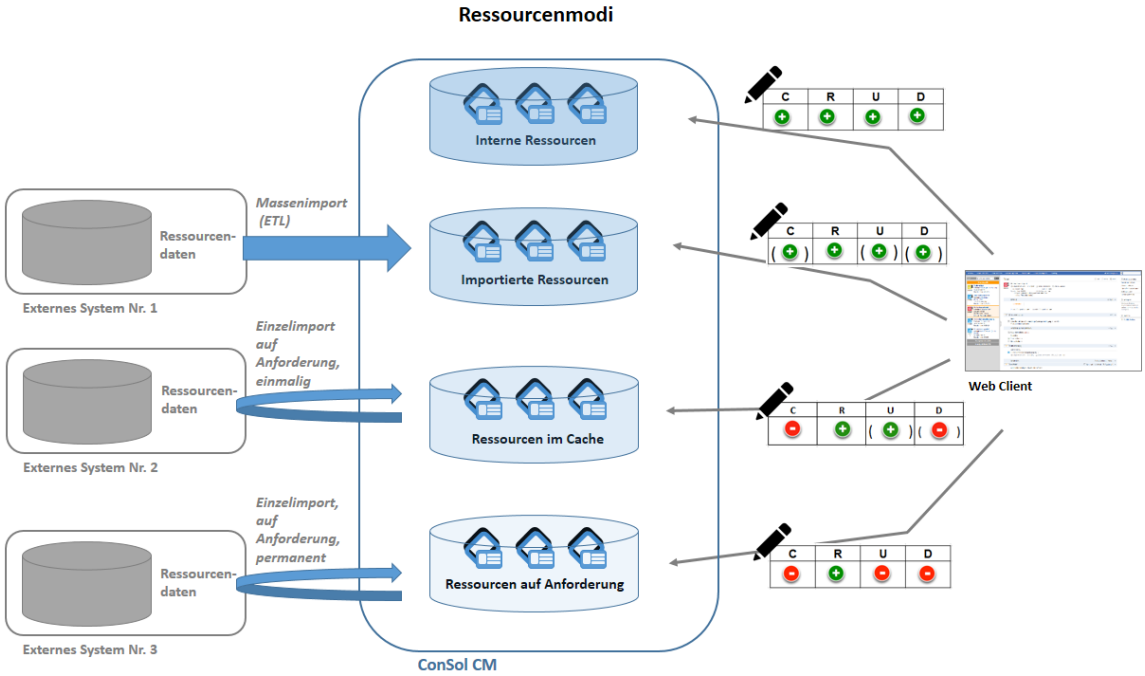


Abbildung 618: *Ressourcenmodi*

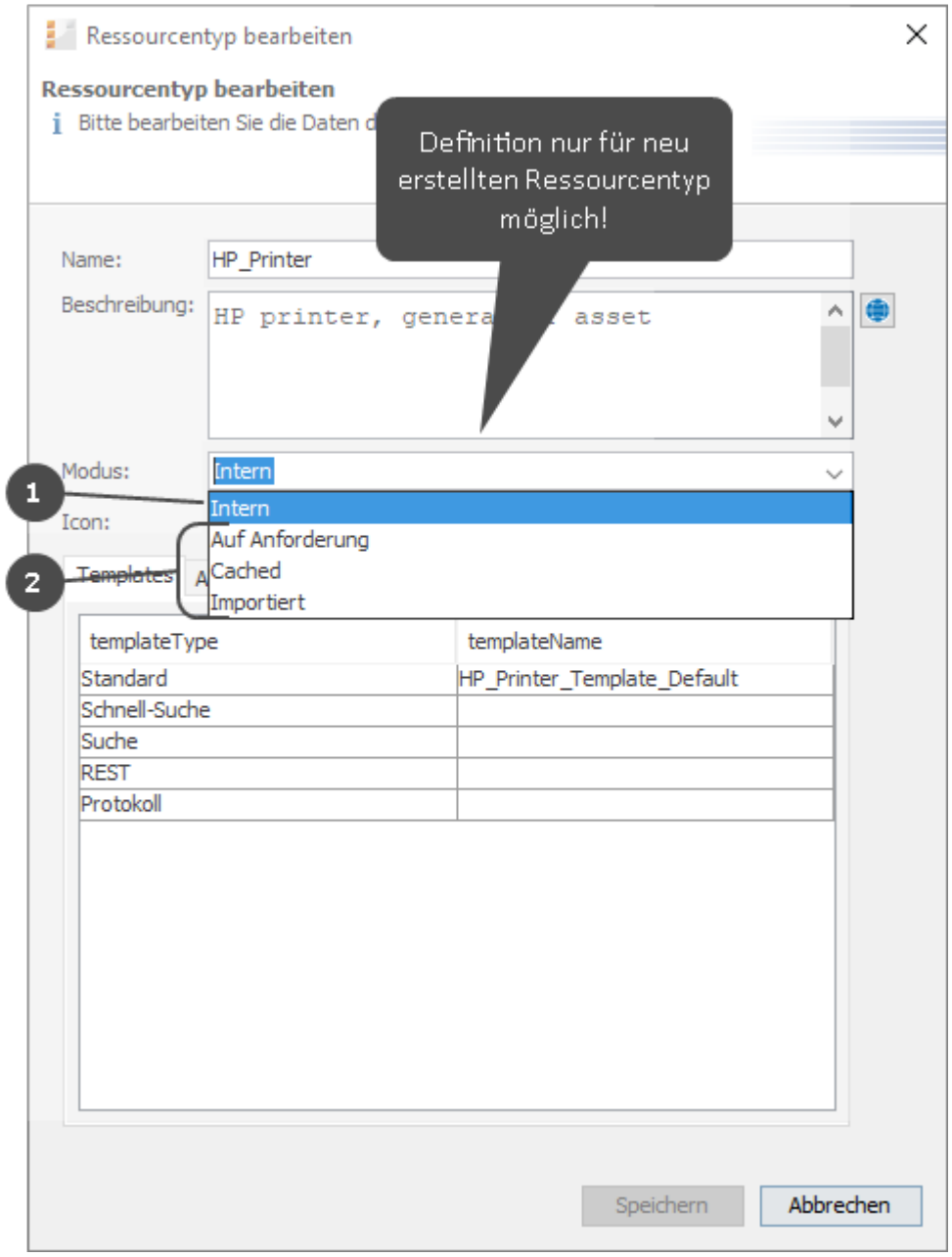


Abbildung 619: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Konfiguration des Ressourcentypmodus*

- (1) Intern
- (2) Extern

Interne Ressourcen

Die Ressourcendaten werden komplett von ConSol CM verwaltet. Bearbeiter mit den entsprechenden Zugangsberechtigungen können im Web Client Ressourcen erstellen, ändern und löschen.

Ressourcen auf Anforderung

Die Ressourcendaten werden auf Anforderung aus einem externen Backend-System abgerufen. Der Modus *auf Anforderung* benötigt eine kundenspezifische Implementierung des Java Interface `ResourceExternalSource`.

Für eine Ressource *auf Anforderung* sind in ConSol CM keine Daten gespeichert mit Ausnahme einer externen ID. Mit dieser ID werden die Daten auf Anforderung aus dem externen System abgerufen.

Im Web Client:

- Die Detailsuche ist nicht verfügbar.
- Für die Suche nach einer Ressource auf Anforderung kann nur die externe ID verwendet werden.
- Es gibt keine Ressourcenseite.
- Es gibt keine Ressourcentypseite.
- Die Ressource ist nur im Bereich für verknüpfte Ressourcen verfügbar in
    - Tickets
    - Kunden
    - anderen Ressourcen
- Die Ressourcendaten können nicht editiert werden.

Ressourcen im Cache

Die Ressourcendaten stammen aus einem externen Backend-System. Die Daten werden nach der ersten Anforderung an ConSol CM übertragen und dann in der ConSol CM-Datenbank zwischengespeichert (gecached). Die erste Anforderung an das externe System wird ausgeführt, sobald eine Ressource mit einem anderen Objekt verknüpft wird (einem Ticket, einem Kunden oder einer anderen Ressource), d. h. wenn eine Relation hergestellt wird. Der Modus *Cached* benötigt eine kundenspezifische Implementierung des Java Interface `ResourceExternalSource`.

Im Web Client:

- Die Detailsuche ist nur für Ressourcen verfügbar, die mindestens einmal angefordert wurden und deren kompletter Datensatz daher in ConSol CM vorhanden ist.
- Für in ConSol CM vorhandene Ressourcen (d. h. Ressourcen im Cache) gilt:
    - Es ist eine Ressourcenseite verfügbar.
    - Es ist eine Ressourcentypseite verfügbar.
    - Eine Aktualisierung der Ressourcendaten kann manuell auf der Ressourcenseite angestoßen werden.
- Um in ConSol CM nach einer Ressource *im Cache* zu suchen, die noch nicht angefordert wurde, kann nur die externe ID verwendet werden.
- Es gibt keine Ressourcenseiten für noch nicht zwischengespeicherte Ressourcen.

- Die Ressource ist nur im Bereich für verknüpfte Ressourcen von Tickets, Kunden und anderen Ressourcen verfügbar. Die Ressourcenfelder werden auch nur im Bereich für verknüpfte Ressourcen angezeigt.
- Die Ressourcendaten können nicht editiert werden.

Importierte Ressourcen

Die Ressourcendaten stammen aus einem externen Backend-System und werden in der ConSol CM-Datenbank gespeichert. Nachdem die Daten nach ConSol CM importiert wurden, werden sie fast genauso wie interne Daten behandelt. Für den Import nach ConSol CM wird normalerweise ein **ETL-Job** (Extract - Transform - Load) eingesetzt, der einmal oder nach einem bestimmten Zeitplan ausgeführt wird.

Im Web Client:

- Die Detailsuche ist für importierte Ressourcen komplett verfügbar.
- Es ist eine Ressourcenseite verfügbar.
- Es ist eine Ressourcentypseite verfügbar.
- Importierte Ressourcen können nicht im Web Client erstellt werden (nur über einen Import).
- Die Ressourcendaten können editiert werden, aber ein neuer Import kann die Änderungen überschreiben.



Um externe Ressourcen in Ihr CM-System zu importieren, müssen Sie eine spezielle Java- / Groovy-Klasse schreiben, die das Interface `ResourceExternalSource` mit folgenden drei Methoden implementiert:

- PageResult<Resource> **searchByPattern**(ResourceType pType, String pPattern, int pPageSize, int pPageNumber)
- Resource **importResource**(Resource pResource)
- Resource **getByExternalId**(ResourceType pType, String pExternalId)

Eine detaillierte Beispielklasse finden Sie in den *ConSol CM Release Notes* für Version 6.10.3. Sie müssen die Methoden für Ihr System implementieren, um echte Daten aus Ihrer IT-Infrastruktur bereitstellen zu können.

Editieren der Ressourcenfeldgruppe

Eine Ressourcenfeldgruppe ist eine Sammlung von Ressourcenfeldern, den Datenfeldern für Ressourcendaten (analog zur Ticketfeldgruppe, die eine Sammlung von Ticketfeldern für Ticketdaten darstellt). Ressourcenfelder können nie als alleinstehende Objekte verwaltet werden; sie werden immer als Ressourcenfeldgruppe verwaltet (z. B. eingeblendet oder ausgeblendet).

Eine Ressourcenfeldgruppe hat folgende gruppenspezifische Parameter. Markieren Sie die Ressourcenfeldgruppe und klicken Sie auf *Bearbeiten*, um diese Werte zu editieren (siehe nächster Abschnitt).

Editieren der spezifischen Parameter einer Ressourcenfeldgruppe

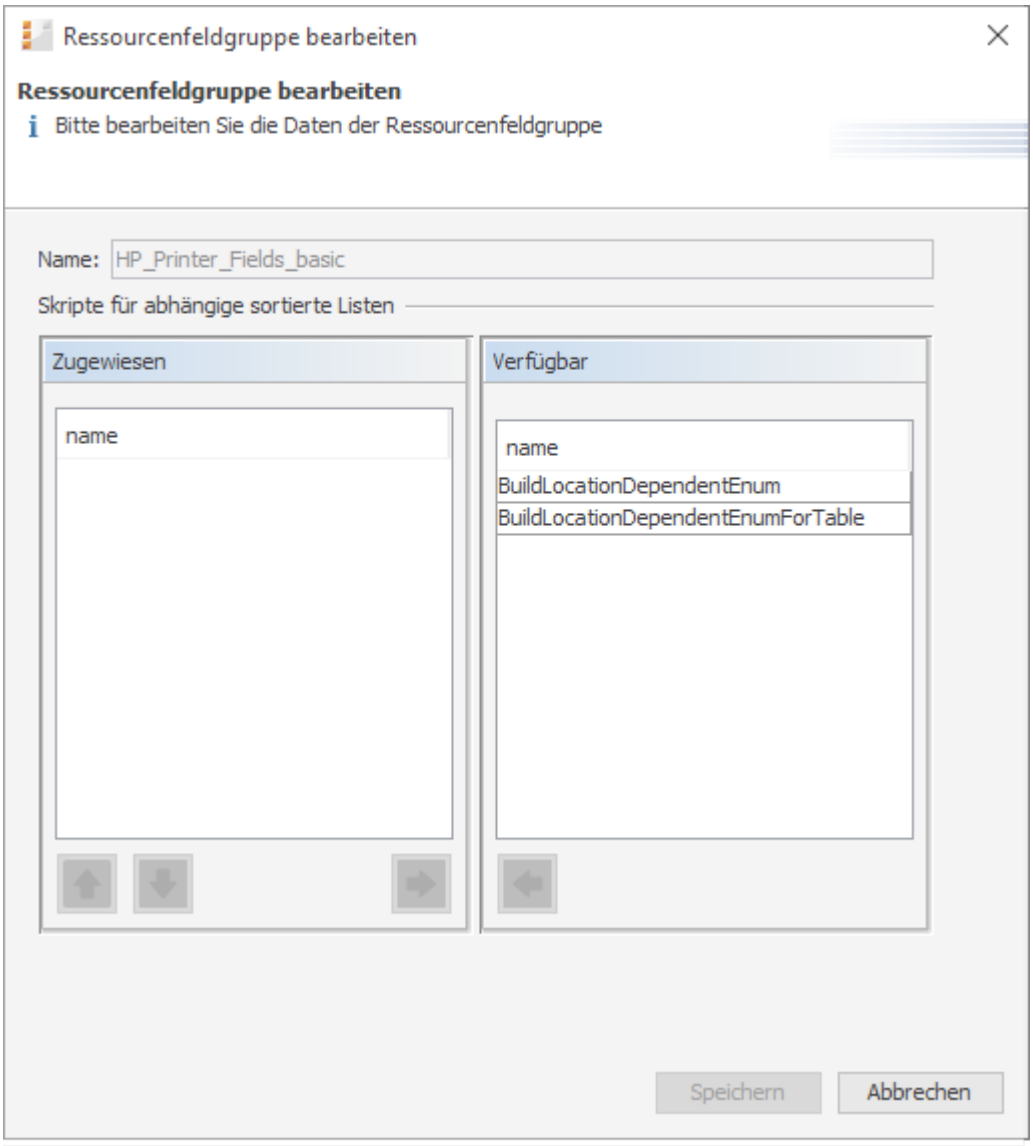


Abbildung 620: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Editieren der spezifischen Parameter einer Ressourcenfeldgruppe*

Hier können Sie folgende Felder editieren:

- **Name**
  Der technische Name der Ressourcenfeldgruppe. Dieser sollte nur geändert werden, wenn dies unumgänglich ist, da dies Nebeneffekte auf alle Skripte haben kann (Workflow und Admin Tool), die den Namen der Ressourcenkategorie verwenden. Um den auf der GUI für die Ressourcenfeldgruppe angezeigten Begriff zu ändern, setzen Sie den im Abschnitt Lokalisierung von Datenfeldern erklärten Mechanismus ein.

- **Skripte für abhängige sortierte Listen**
  Hier können Sie einer Ressourcenfeldgruppe eine abhängige sortierte Liste zuweisen. Die abhängige sortierte Liste muss zuerst als Admin-Tool-Skript definiert werden, siehe Abschnitt Skripte des Typs Abhängige sortierte Liste.

Annotieren von Ressourcenfeldgruppen

Eine Ressourcenfeldgruppe kann Annotationen haben. Wählen Sie die Ressourcenfeldgruppe und klicken Sie auf den Button *Annotieren*, um Annotationen zu setzen oder zu löschen. Es gibt die gleichen Annotationen wie für Ticketfeldgruppen. Alle Annotationen sind in Annotationen aufgeführt und beschrieben.

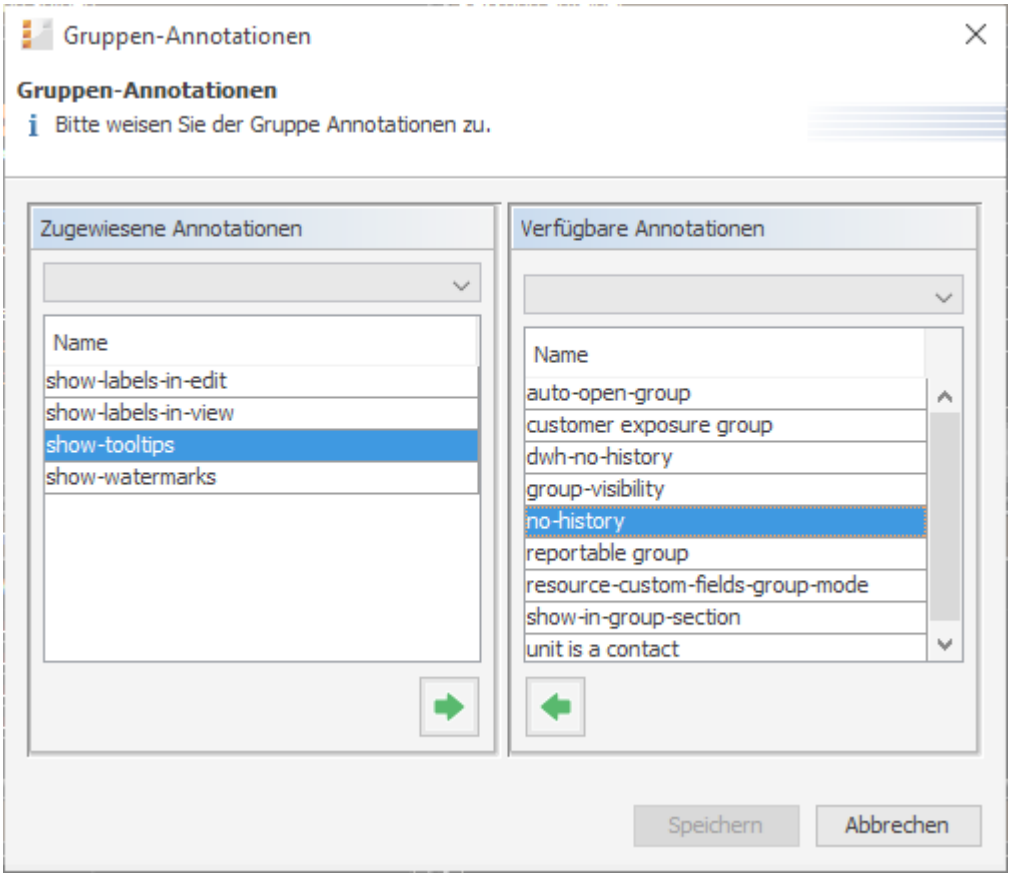


Abbildung 621: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Annotationen von Ressourcenfeldgruppen*

Erstellen und Editieren von Ressourcenfeldern

Ressourcenfelder sind Container für alle Daten der Ressourcen (analog zu den Ticketfeldern für die Ticketdaten). Ressourcenfelder gehören immer zu einer Ressourcenfeldgruppe. Um Ressourcenfelder zu einer Ressourcenfeldgruppe hinzuzufügen oder zu editieren, wählen Sie die Gruppe im Ressourcendatenmodell und geben Sie alle erforderlichen Daten auf der rechten Seite im Admin Tool ein.

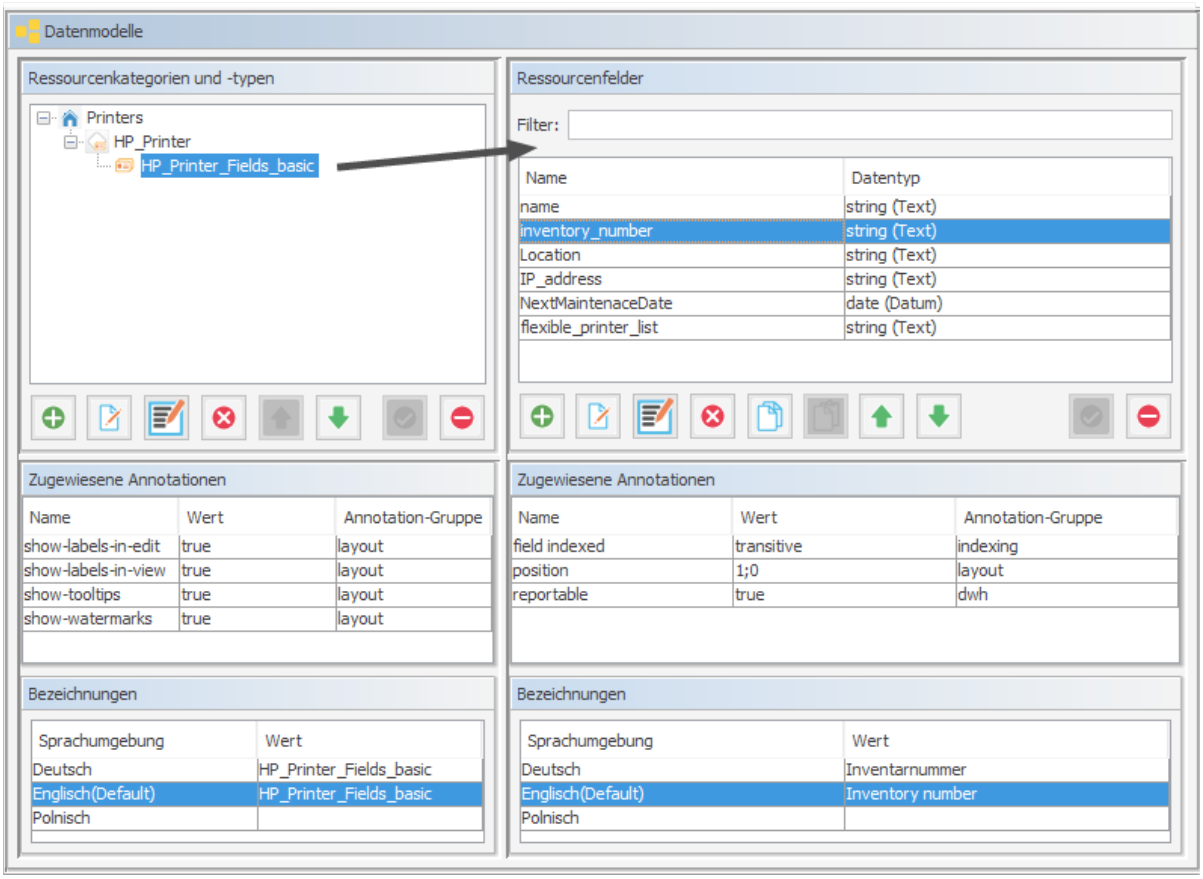


Abbildung 622: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Verwalten von Ressourcenfeldern*

Erstellen eines neuen Ressourcenfeldes

Um ein neues Ressourcenfeld zu erstellen, klicken Sie auf den Button *Hinzufügen* und geben Sie im Pop-up-Fenster alle erforderlichen Daten ein.

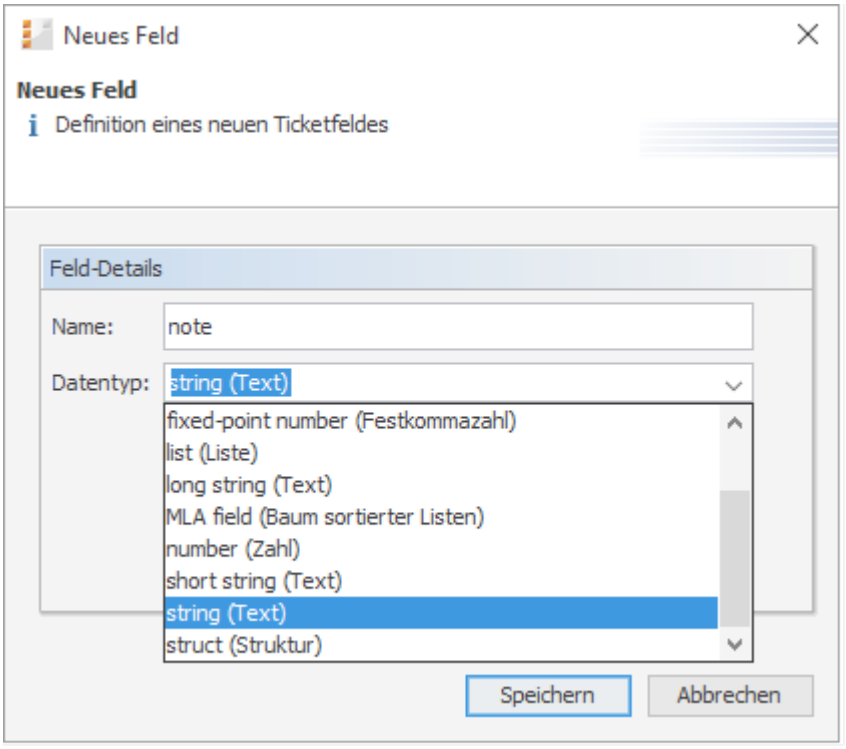


Abbildung 623: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Erstellen eines Ressourcenfeldes*

Geben Sie die folgenden Daten ein:

- **Name**
  Der technische Name des Ressourcenfelds. Dieser wird in Skripten und Templates verwendet. Geben Sie den lokalisierten Wert über den Button *Lokalisieren* ein.
- **Datentyp**
  Der Datentyp des Ressourcenfelds. Wie bei den Ticketfeldern oder Kundenfeldern können Sie einen Datentyp aus der Drop-down-Liste auswählen. Der folgende Abschnitt enthält detaillierte Informationen über die verfügbaren Datentypen.

Arten von Datenfeldern

Die folgenden Datentypen sind für Ticketfelder, Kundenfelder und Ressourcenfelder verfügbar:

- **autocomplete (Autovervollständigung)**
  Ein Datenfeld, das eine skriptbasierte Autocomplete-Liste enthält. Das ist eine dynamische Liste, die auf einem Skript des Typs *Text-Autovervollständigung* basiert. Eine detaillierte Beschreibung von skriptbasierten Autocomplete-Listen finden Sie im Abschnitt Skriptbasierte Autocomplete-Listen.
- **boolean (Ja/Nein)**
  Werte: true/false. Je nach Einstellung in der Annotation `boolean-type` wird der Wert als Checkbox, Radio-Button oder Drop-down-Liste angezeigt.

Wenn Sie in CM-Workflows oder im Admin Tool mit Skripten arbeiten, sollten Sie wissen, dass das Verhalten von Boolean-Feldern, die als Checkboxen dargestellt werden, d. h. mit der Annotation `boolean-type` = „checkbox “(Standardwert) versehen sind, anders ist als in früheren CM-Versionen!

- **In CM-Versionen vor 6.9.4.0:**
  Wenn ein Boolean-Feld noch nicht angefasst wurde, ist der Wert „false“. Wenn es aktiviert wird, ist der Wert „true“, und wenn es danach wieder deaktiviert wird, ist der Wert wieder „false“.
- **In Version 6.9.4.0 und höher:**
  Wenn ein Boolean-Feld noch nicht angefasst wurde, ist der Wert „NULL“. Wenn es aktiviert wird, ist der Wert „true“, und wenn es danach wieder deaktiviert wird, ist der Wert „false“.

Felder, die in der Datenbank bereits mit Werten gefüllt sind, werden während des Updates von einer Version vor 6.9.4.0 auf Version 6.9.4.0 und höher nicht verändert.

Boolean-Felder, die als Radio-Button (Annotation `boolean-type` = „radio“) oder Drop-down-Liste (Annotation `boolean-type` = „select“) dargestellt sind, hatten schon immer das Verhalten, das für Version 6.9.4.0 oder höher beschrieben ist, d. h. sie haben den Wert „NULL“, wenn sie noch nicht angefasst wurden.

- **date (Datum)**
  Format und Genauigkeit können über Annotationen festgelegt werden.
- **enum (Sortierte Liste)**
  Für Sortierte Listen. Die Bearbeiter können im Web Client zwischen einem der Listenwerte wählen. Die Sortierten Listen und ihre Werte müssen zuvor in der Verwaltung von Sortierten Listen erstellt worden sein. Wählen Sie den gewünschten *Listentyp* und die *Listengruppe* in den Feldern darunter.
- **list (Liste)**
  Ein Feld dieses Datentyps ist der erste Schritt zur Erstellung einer Liste (eine Spalte) oder einer Tabelle (mehrere Spalten) von Eingabefeldern im Web Client.
  - Für eine **Tabelle** wird im nächsten Schritt ein anderes Feld des Typs *struct* erstellt (siehe unten), das später die Eingaben der einzelnen Listenfelder enthält (die die Spalten der Tabelle sind). Wenn Sie also eine Tabelle erstellen möchten, müssen Sie zuerst ein Feld des Typs *struct* erzeugen (siehe unten), bevor Sie die Felder für die Tabellenspalten hinzufügen können.
  - Für eine **einfache Liste** ist der nächste Schritt die Erstellung der Felder, die zu der Liste gehören. In diesem Fall ist kein *struct* erforderlich.

  Für alle Felder, die zu einer Liste oder Tabelle gehören, müssen Sie die Abhängigkeiten im Feld *Gehört zu* angeben (siehe unten). Ein Tabellenfeld (ein normales Datenfeld), gehört zum Beispiel immer zu einem *struct* und ein *struct* gehört immer zu einer *list*.

- **struct (Struktur)**
  Mit einem Datenfeld dieses Typs wird eine Datenstruktur (Zeile einer Tabelle) definiert, die ein oder mehrere Felder umfasst. Das ist der zweite Schritt zur Erstellung einer Tabelle, nachdem Sie zuerst ein Feld des Typs *list* erstellt haben. Im nächsten Schritt fügen Sie die Felder für die Spalten der Tabelle hinzu. Die Abhängigkeiten müssen für jedes Datenfeld im Feld *Gehört zu* (siehe unten) gesetzt werden, d. h. ein *struct* gehört immer zu einer *list*.

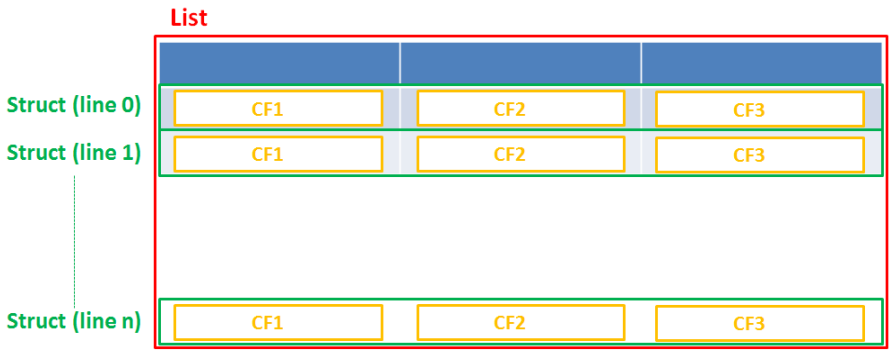


Abbildung 624: *Schema: List of Structs*

Technisch gesehen ist eine Liste ein Array, der in jedem Feld eine Map (= Paare von Schlüssel:Wert) enthält.

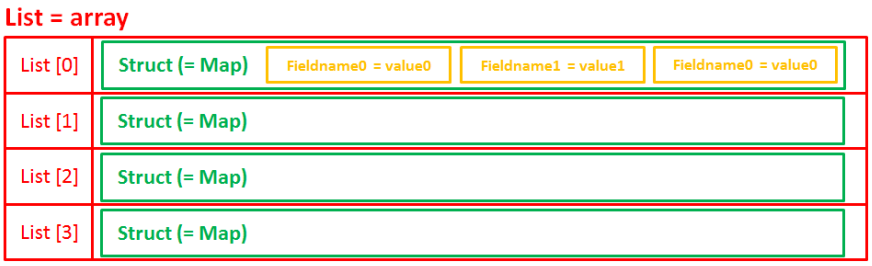


Abbildung 625: *List of structs, technisches Prinzip*

- **number (Zahl)**
  Für Ganzzahlen (Integer).
- **fixed point number (Festkommazahl)**
  Für Festkommazahlen, z. B. Währungen. Sie müssen die Gesamtanzahl der Ziffern (*Genauigkeit*) und die Anzahl der Ziffern nach dem Komma (*Skalierung*) in den entsprechenden Feldern eingeben.
- **string (Text)**
  Für bis zu 4000 alphanumerische Zeichen.

> ⚠ Beschränkung bei der Verwendung einer Oracle-Datenbank: Es können höchstens 4000 Bytes in UTF-Codierung gespeichert werden. Ab Oracle12c.

- **long string (Text)**
  Für große Objekte

  > ⚠ Beim Datentyp *long string (Text)* hängt die Grenze von dem für ConSol CM verwendeten Datenbanksystem ab: MS SQL Server: 2 GByte; MySQL: 4 GByte; Oracle: 16 - 64 GByte (je nach Page-Größe des Tablespace).

- **short string (Text)**
  Für bis zu 255 alphanumerische Zeichen.

  > ⓘ Bei Feldern des Datentyps *string (Text)* können Sie die Felddefinition mithilfe von Annotationen präzisieren. Ein Feld des Datentyps *string* kann zum Beispiel so definiert werden, dass es eine URL enthält und automatisch als Hyperlink angezeigt wird. Lesen Sie dazu den folgenden Abschnitt.

- **contact data reference (Referenz auf ein Kontaktdatenfeld)**
  Besonderer Datentyp, der intern als Referenz auf die mit dem Ticket verknüpften Kontakte verwendet wird. In FlexCDM (d. h. ab CM-Version 6.9.0), wird dieser Datentyp nicht mehr angezeigt, sondern nur noch intern im CM-System verwendet.
- **MLA field (Baum sortierter Listen)**
  Dieser Datentyp wird für Felder verwendet, die hierarchische Listen mit einer Baumstruktur enthalten, das sogenannte *MLA* (Multi Level Attributes). Der Name dieses Feldes ist der Name des neuen MLA, das in der MLA-Verwaltung angelegt werden muss. Die Gruppe des Feldes muss angegeben werden, nachdem das MLA erstellt wurde.

> ⚠ Der Datentyp, den Sie beim Erstellen eines Datenfeldes verwenden, kann danach nicht mehr verändert werden!

Details über String-Felder: Verwenden von Annotationen zum Anpassen von Strings

String-Felder werden häufig für Kunden-, Ticket- und Ressourcendaten verwendet und in Strings kann unterschiedlicher Inhalt gespeichert werden, zum Beispiel ein Textfeld mit einem Kommentar, ein einfaches Eingabefeld mit nur 20 Zeichen, eine URL oder ein Passwort. Das Anpassen von String-Feldern erfolgt mithilfe von speziellen Annotationen, die auf der Seite Annotationen des Anhangs aufgeführt sind. Da die Arbeit mit diesen Annotationen zu den alltäglichen Aufgaben eines CM-Administrators gehört, sind die wichtigsten und am häufigsten verwendeten Annotationen zusätzlich an dieser Stelle erklärt.

Wie kann ich ...

... statt einer einzelnen Zeile ein **Textfeld** einfügen?

Wert für Annotation `text-type`: „textarea“

Die Größe des Textfeldes kann angepasst werden. Es wird je nach Webbrowser als Standardtextfeld angezeigt. Verwenden Sie die Annotation `field-size`, wenn das Textfeld eine bestimmte Größe haben soll.

... die Eingabe des Feldes für **Passwörter** ausblenden?

Wert für Annotation `text-type`: „password“

Es werden nur Punkte angezeigt. Diese Annotation legt **nicht** fest, dass das Feld ein Passwort enthält! Es definiert nur den Anzeigemodus! Verwenden Sie die Annotation `password`, um ein String-Feld zu definieren, das das CM/Track-Passwort enthält.

... einen **Hyperlink** anzeigen, den Namen statt dem Link anzeigen?

Wert für Annotation `text-type`: „url“

Die Eingabe wird im Ansichtsmodus als Hyperlink angezeigt. Der String muss einem bestimmten URL-Muster entsprechen:

- "^((?:mailto\:|(?:(?:ht|f)tps?)\://)1\S+)(?: (?:\|)?(.*))?$"

Der erste Teil des Strings ist der Link (URL) und der zweite Teil ist der Name, der angezeigt werden soll.

Beispiel: "http://consol.de ConSol"

... einen **Dateilink** anzeigen?

Wert für Annotation `text-type`: „file-url“

Die Eingabe wird als Link auf eine Datei im Dateisystem angezeigt. Der Webbrowser muss solche Links zulassen/unterstützen!

Beispiel: Aktivieren von file://-URLs in einem Firefox-Browser

Fügen Sie die folgenden Zeilen entweder zur Konfigurationsdatei `prefs.js` oder zu `user.js` im Benutzerprofil hinzu. Auf einem Windows-System befindet sich diese normalerweise in einem Ordner wie `C:\Benutzer\<BENUTZERNAME>\AppData\Roaming\Mozilla\Firefox\Profiles\uvubg4fj.default`

- user_pref("capability.policy.localfilelinks.checkloaduri.enabled", "allAccess");
- user_pref("capability.policy.localfilelinks.sites", "http://cm-server.domain.com:8080");
- user_pref("capability.policy.policynames", "localfilelinks");

Alternativ kann ein Add-on des Firefox-Browsers wie ***Local Filesystem Links*** installiert werden, um einfacher auf die referenzierten Dateien und Ordner zugreifen zu können.

Dieser Link wird auch als Tooltip angezeigt.

Die URL ist korrekt formatiert, wenn folgende Bedingungen erfüllt sind:

- Sie beginnt mit `file:` gefolgt von normalen Schrägstrichen:
  - drei Schrägstriche "///" für Dateien, die sich auf demselben Computer befinden wie der Browser (alternativ "//localhost/") oder

    - zwei Schrägstriche gefolgt vom Servernamen und einem weiteren Schrägstrich für Dateien auf Dateiservern, die vom Rechner, auf dem der Webbrowser läuft, erreichbar sind.
- Danach folgt der vollständige Dateipfad, der mit dem Dateinamen endet.
- Auf Microsoft Windows-Systemen wird der Pfad ebenfalls mit normalen Schrägstrichen anstelle von umgekehrten Schrägstrichen geschrieben.
- Der Laufwerksbuchstabe eines lokalen Pfads auf Microsoft Windows-Systemen wird wie üblich verwendet, zum Beispiel `C:`.
- Pfade mit Leerzeichen und Sonderzeichen wie "{, }, ^, #, ?" müssen auf Microsoft Windows-Systemen mit Prozentzeichen kodiert werden (z. B. mit "%20" für ein Leerzeichen).

Beispiel-URLs:

- file://file-server/path/to/my/file.ext
- file:///linux/local/file.pdf
- file:///C:/Users/myuser/localfile.doc

Siehe auch die Erklärung von `file-url` im Abschnitt Liste der Feldannotationen

... ein **Label** definieren?

Wert für Annotation `text-type`: „label“

Dies ist ein schreibgeschütztes Feld, das grau angezeigt wird. Verwenden Sie die Annotation `label-group`, um das Label und die dazugehörigen Eingabefelder zu verknüpfen. Berücksichtigen Sie die Annotationen für Label (`show-label-in-edit`, `show-label-in-view`), bevor Sie spezielle Label-Felder implementieren!

... ein Feld für **gültige E-Mail-Adressen** definieren?

Wert für Annotation `email`: „true“

Das Feld darf nur gültige E-Mail-Adressen enthalten. Die Eingabe wird gemäß des Standardformats für E-Mail-Adressen validiert <name>@<domain>.

... eine skriptbasierte Autocomplete-Liste definieren?

Wert für die Annotation `text-type` = „autocomplete“

Optional: Wert für die Annotation `autocomplete-script` = <Name des entsprechenden Skripts>

Eine skriptbasierte Autocomplete-Liste wird verwendet, um ein Drop-down-Menü bereitzustellen, das anhand der Eingabe, die der Bearbeiter bereits vorgenommen hat, dynamisch gefüllt wird. Wenn der Benutzer zum Beispiel "Mei" eingibt, werden die möglichen Werte "Meier", "Meister" und "Meinert" als Liste angezeigt und der Bearbeiter kann den für das Feld erforderliche Wert auswählen. Sie kennen dieses Verhalten von anderen Autocomplete-Feldern, z. B. der Suche nach Bearbeitern für ein Ticket oder die Suche nach Kunden bei der Ticketerstellung. In diesen Fällen erzeugt CM die Liste allerdings automatisch und das Verhalten kann nicht beeinflusst oder angepasst werden. Im Gegensatz dazu können skriptbasierte Autocomplete-Listen vom CM-Administrator implementiert werden. Die Werte basieren auf einem Ergebnissatz, der dynamisch erstellt wird. Der Ergebnissatu kann Strings, Bearbeiter, Kunden (Units) und Ressourcen enthalten.

Eine detaillierte Beschreibung von skriptbasierten Autocomplete-Listen finden Sie im Abschnitt Skript-basierte Autocomplete-Listen.

Annotieren von Ressourcenfeldern

Die Eigenschaften eines Ressourcenfeldes werden mithilfe von Annotationen definiert, z. B. die Sichtbarkeit, die Position im Web Client, ob das Feld indiziert werden soll, ob es reportbar sein soll usw. Sie kennen dieses Prinzip wahrscheinlich von der Arbeit mit Annotationen von Ticketfeldern.

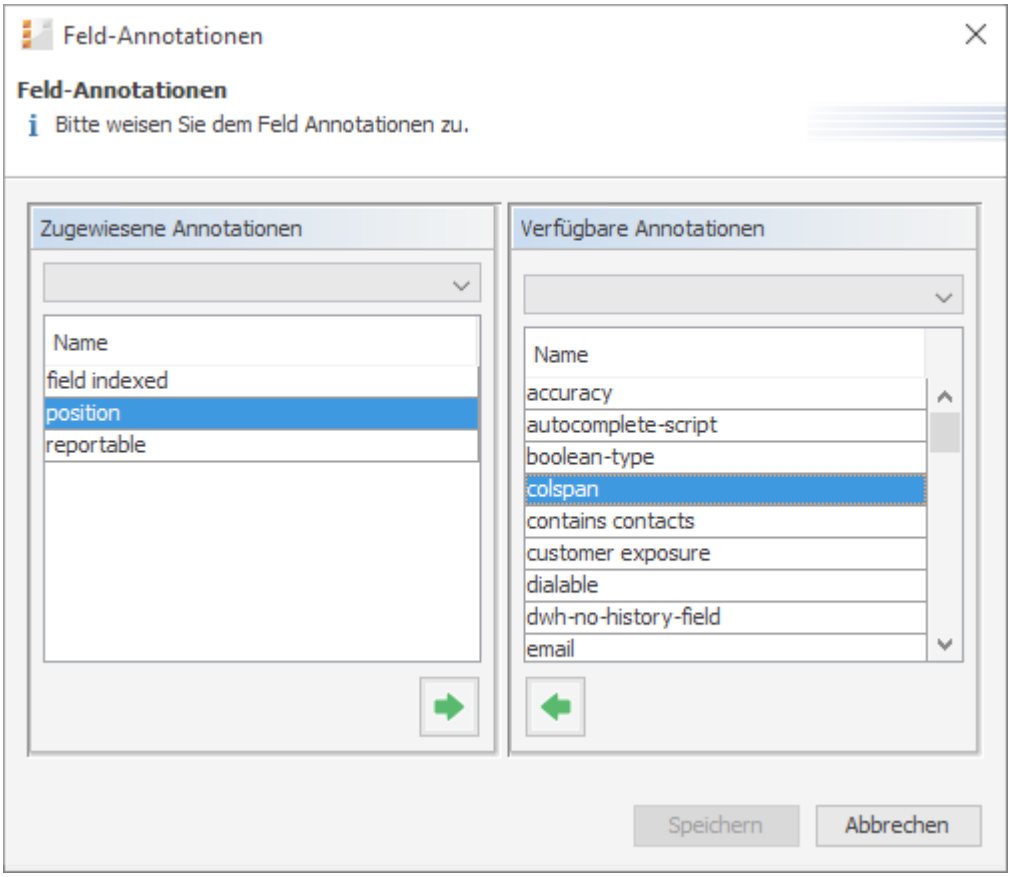


Abbildung 626: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Definieren von Ressourcenfeld-Annotationen*

Es sind die gleichen Annotationen verfügbar wie für Ticketfelder. Sie sind detailliert in Annotationen erklärt.

Die Annotation `position` bestimmt die Position des Feldes an folgenden Stellen:

- Das Formular zur Erstellung einer neuen Ressource, siehe Abschnitt Kurze Einführung in CM/Resource Pool-Funktionen im Web Client, Formular zum Erstellen von Ressourcen.
- Die Ressourcenseite, die im Abschnitt Kurze Einführung in CM/Resource Pool-Funktionen im Web Client, Ressourcendetailseite erklärt ist.

Wie beim Ticketlayout können Sie auch Ressourcenfelder nach dem Matrix-Prinzip auf der GUI anordnen und ihre Position folgendermaßen festlegen:

0;0	0;1	0;2
1;0	1;1	1;2
2;0	2;1	2;2
n;0	n;1	n;2

Abbildung 627: *Beispielmatrix für die Annotation position*

G.1.4.3 Kopieren von Ressourcenfeldern

Es ist nicht nötig, dass Sie alle Datenfelder über das Pop-up-Fenster erstellen. Sie können stattdessen auch ein vorhandenes Datenfeld kopieren. Das kann ein Ticketfeld, ein Kundenfeld oder ein Ressourcenfeld sein. Wählen Sie das entsprechende Datenfeld in der Datenfeldgruppe und klicken Sie auf den Button *Kopieren*. Navigieren Sie dann in die Datenfeldgruppe, in die Sie das kopierte Feld einfügen möchten. Das kann eine Ticketfeldgruppe, eine Kundenfeldgruppe oder eine Ressourcenfeldgruppe sein. Sie können ein Feld von einem Objekttyp (z. B. Ticket) in einen anderen (z. B. Kunde) kopieren. Klicken Sie in die Liste der Datenfelder der Zielfeldgruppe und klicken Sie auf den Button *Einfügen*.

Es kann jeder Feldtyp kopiert werden, von einfachen String-Feldern über Sortierte Listen bis hin zu ganzen Strukturen (STRUCTS). Um ein STRUCT zu kopieren, markieren und kopieren Sie nur die Liste. Die untergeordneten Objekte (z. B. STRUCT mit Feldern) werden automatisch kopiert.

Folgendes wird kopiert:

- fast alle Annotationen mit ihren Werten
- alle lokalisierten Werte

Folgendes wird nicht kopiert:

- die folgenden Annotationen
  - *ldapid*: Diese Annotation kann in einem Datenmodell nur einmal gesetzt werden. Eine Duplizierung würde zu einem ungültigen Modell führen.
  - *username*: Diese Annotation kann in einem Datenmodell nur einmal gesetzt werden. Eine Duplizierung würde zu einem ungültigen Modell führen.
  - *password*: Diese Annotation kann in einem Datenmodell nur einmal gesetzt werden. Eine Duplizierung würde zu einem ungültigen Modell führen.
  - *position*: Der Wert dieser Annotation muss eindeutig sein. Die Annotation wird also dupliziert, aber der Wert wird aus der Kopie entfernt.
  - *ticket-list-position*: Diese Annotation sollte nur für spezielle Fälle verwendet werden und daher nur für wenige Felder gesetzt werden. Durch die Duplizierung würde sie möglicherweise multipliziert, obwohl sie für die Mehrzahl der Kopien nicht benötigt wird.

Benennung der kopierten Felder:

- Kopie in eine andere Feldgruppe: es wird der Feldname verwendet
- Kopie in die gleiche Feldgruppe: es wird <Feldname>_copy verwendet

Neben den Buttons zum Kopieren und Einfügen können Sie auch die Tastenkombinationen verwenden:

- Kopieren: STRG-C
- Einfügen: STRG-V

G.1.4.4 Verwenden von skriptbasierter Feldvisualisierung für Ressourcenfelder

Mit der skriptbasierten Feldvisualisierung können Sie die Anzeige von Daten in Ressourcenfeldern verbessern. Details dazu finden Sie im Abschnitt Skripte des Typs Feld-Visualisierung.

G.1.4.5 Erstellen von weiteren Ressourcentypen innerhalb einer Ressourcenkategorie

Um innerhalb einer Ressourcenkategorie einen neuen Ressourcentypen hinzuzufügen, markieren Sie einen anderen Ressourcentypen der gewünschten Ressourcenkategorie in der Liste und klicken Sie auf den Button *Hinzufügen*. Geben Sie danach alle erforderlichen Daten so ein, wie im Abschnitt Editieren des Ressourcentyps beschrieben.

G.1.4.6 Erstellen von Ressourcenfeldgruppen und Ressourcenfeldern

Um eine neue Ressourcenfeldgruppe innerhalb eines Ressourcentyps zu erstellen, wählen Sie eine andere Ressourcenfeldgruppe des gewünschten Ressourcentyps in der Liste aus und klicken Sie auf den Button *Hinzufügen*. Geben Sie danach alle erforderlichen Daten so ein, wie im Abschnitt Editieren der Ressourcenfeldgruppe beschrieben.

G.1.4.7 Erstellen weiterer Ressourcenkategorien

Um eine neue Ressourcenkategorie zu erstellen (in unserem Beispiel *Office_Software)*, wählen Sie eine andere Ressourcenkategorie in der Liste aus und klicken Sie auf den Button *Hinzufügen*.

Geben Sie dann alle erforderlichen Daten so ein, wie im Abschnitt Editieren der Ressourcenkategorie beschrieben.

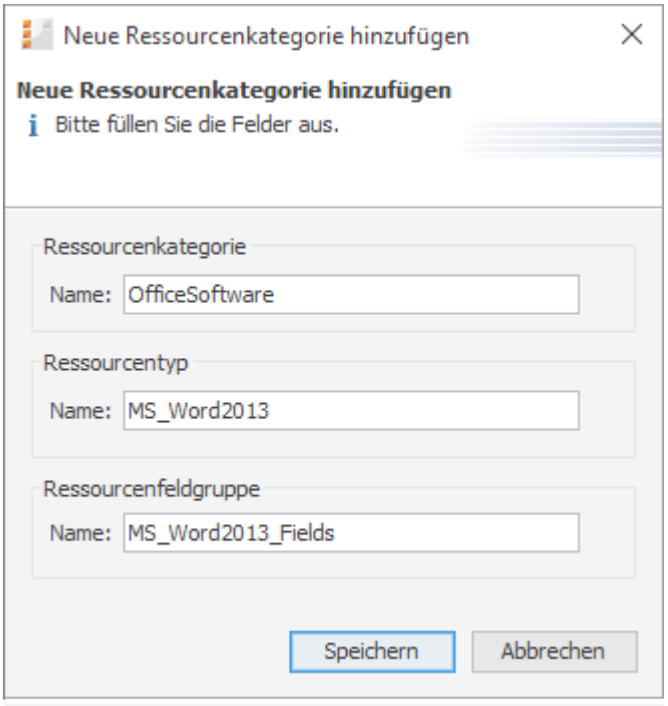


Abbildung 628: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Hinzufügen einer neuen Ressourcenkategorie*

Um eine neue Ressourcenkategorie mit Ressourcentypen und Ressourcenfeldern zu füllen, gehen Sie so vor, wie in den vorherigen Abschnitten beschrieben.

G.1.4.8 Beispiel für ein komplettes Ressourcenmodell

In unserem Beispiel haben wir das folgende Ressourcenmodell erstellt. Es wird in diesem Handbuch weiter verwendet.

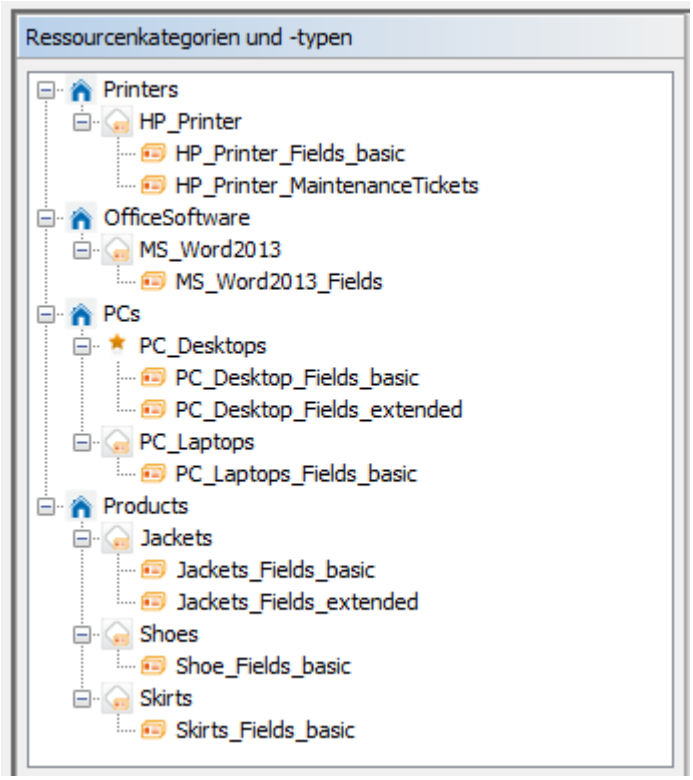


Abbildung 629: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Beispiel für Ressourcenmodell*

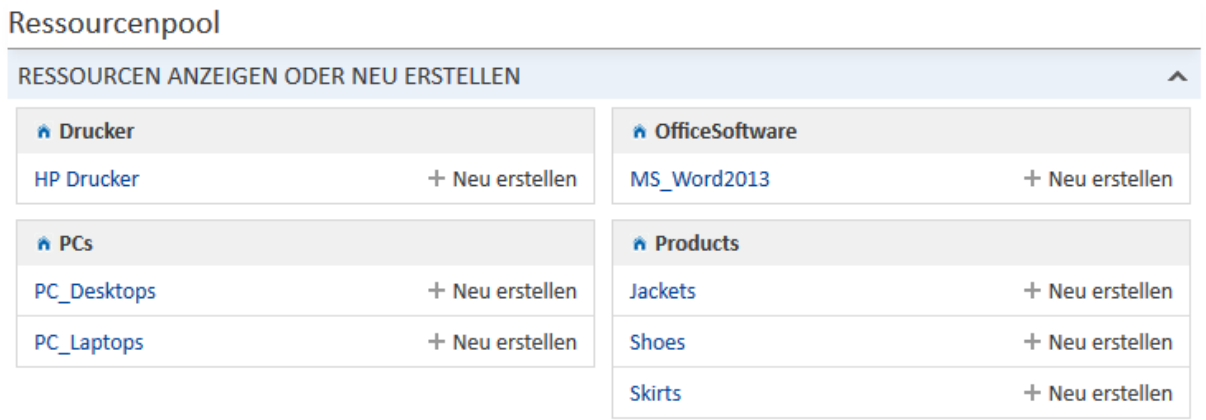


Abbildung 630: *ConSol CM Web Client - Beispiel für ein Ressourcenmodell (im Ressourcenpool-Dashboard)*

G.1.4.9 Definieren von Ressourcenrelationen

Dies wird im Abschnitt CM/Resource Pool - Ressourcenrelationen näher erklärt.

G.1.4.10 Definieren von Ressourcenaktionen

Dies wird im Abschnitt CM/Resource Pool - Ressourcenaktionen näher erklärt.

G.1.4.11 Zuweisen von Zugangsberechtigungen für Ressourcen zu Bearbeitern mittels Rollen

Dies wird im Abschnitt CM/Resource Pool - Zuweisen von Berechtigungen für Ressourcen näher erklärt.

G.1.5 CM/Resource Pool - Templates für Ressourcendaten

G.1.5.1 Einführung in die Verwendung von Templates für die Anzeige von Ressourcendaten

Im ConSol CM Web Client werden Ressourcendatensätze an mehreren Stellen in der auf den Templates basierenden Kurzform angezeigt. Im Ticketprotokoll werden möglicherweise der Ressourcenname, die Inventarnummer und der Standort benötigt, wohingegen in der Schnellsuche nur der Name und die Inventarnummer angezeigt werden sollen. In diesem Abschnitt wird erklärt, wo die Kurzformen verwendet werden, und wie Sie die zugehörigen Templates mit dem Admin Tool konfigurieren können.

Templates werden immer für den Ressourcentyp definiert. In unserem Beispiel ist *HP_Printer* der Ressourcentyp, d. h. die Templates gelten für alle realen HP-Drucker, die im CM-System gespeichert sind.

Die Konfiguration basiert auf folgendem Prinzip (sehr ähnlich wie das Prinzip für die Anzeige von Kundendaten):

- Ein Template für eine bestimmte Stelle im Web Client wird in der Definition des Ressourcendatenmodells einem Ressourcentyp zugewiesen (Navigationsgruppe *Ressourcen*, Navigationselement *Datenmodell*).
- Das referenzierte Template muss der Name des Templates sein, das im Admin Tool im Navigationselement *Skripte und Templates* der Navigationsgruppe *System* gespeichert ist. Sie als Administrator können dem Template einen beliebigen Namen geben. Sie müssen nur sicherstellen, dass der Name des in der Ressourcentypdefinition referenzierten Templates mit dem Namen des Templates im Abschnitt *Skripte und Templates* identisch ist.

G.1.5.2 Erstellen und Editieren von Templates für das Format von Ressourcendaten

Um Templates für das Format von Ressourcendaten zu erstellen und zu editieren, editieren Sie zuerst den gewünschten Ressourcentyp im Admin Tool und geben Sie dort die Namen der Templates ein. Wählen Sie dazu beliebige Namen oder halten Sie sich an die Konventionen Ihres Unternehmens bzgl. der Benennung.

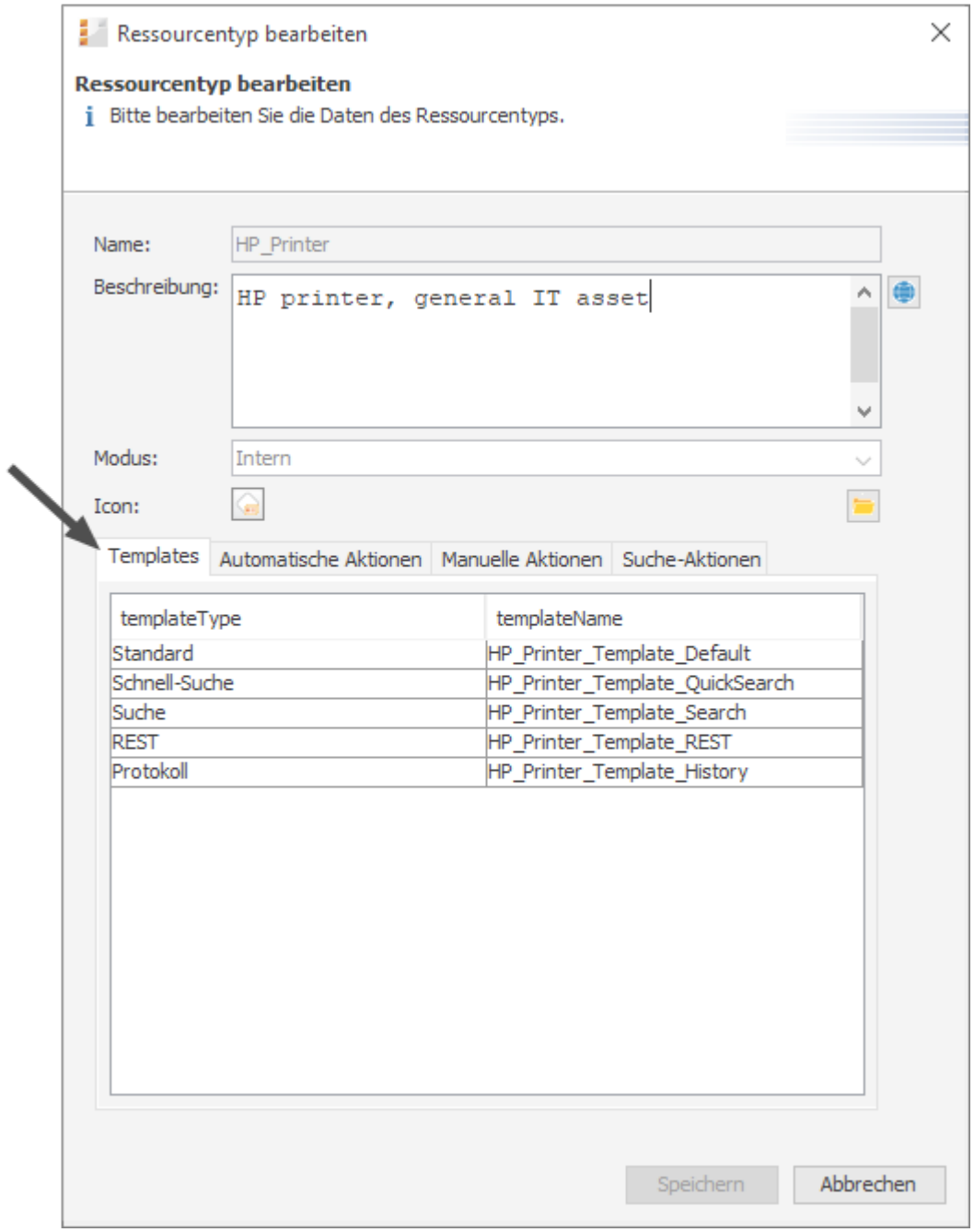


Abbildung 631: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Definieren eines Templates für das Ressourcendatenformat eines Ressourcentyps*

Im nächsten Schritt müssen Sie die entsprechenden Templates erstellen und im Bereich *Skripte und Templates* speichern. Eine detaillierte Beschreibung finden Sie im folgenden Abschnitt.

G.1.5.3 Coding von Templates für das Format von Ressourcendaten

Allgemeines Prinzip

Die Templates werden in der ***FreeMarker-Notation*** geschrieben. Detaillierte Informationen dazu finden Sie auf der FreeMarker-Website.

Innerhalb der Templates arbeiten Sie mit drei Objekttypen:

1. `resource`: Dies ist das aktuelle Ressourcenobjekt.
2. Der technische Name der Ressourcenfeldgruppe.
3. Der technische Name des Ressourcenfelds.

Beachten Sie, dass es innerhalb eines Ressourcentyps mehr als eine Ressourcenfeldgruppe geben kann!

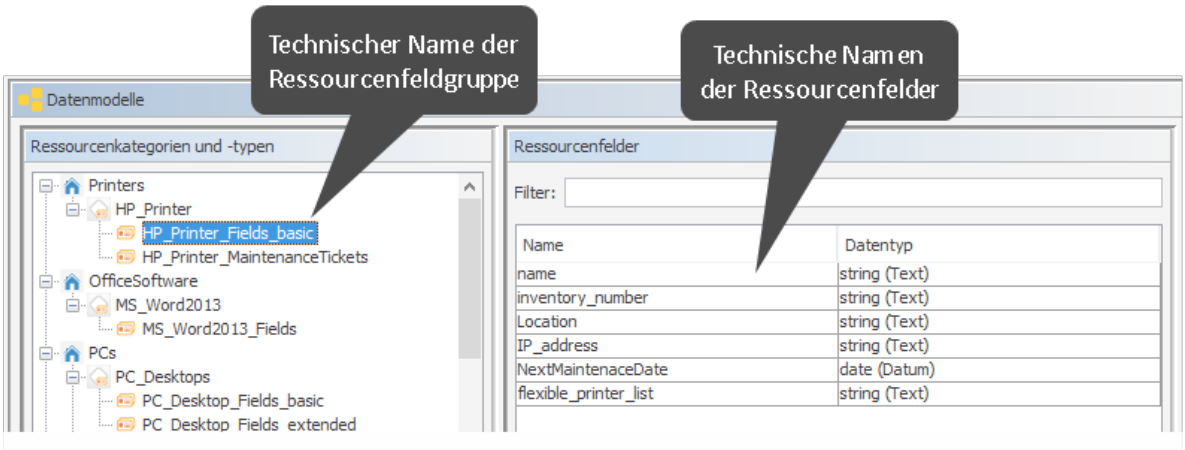


Abbildung 632: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Beim Definieren der Templates für Ressourcendaten verwendete Objekte*

⚠ Die Ressourcen-Templates dürfen nur eine Zeile haben! Zeilenumbrüche sind nicht zulässig!

Beispiel für Templates

```
// Example 1
// Resource Field group is named vehicleProperties, Resource Fields are named
 vehicleIdentification and vehicleLicense
${resource.get("vehicleProperties", "vehicleIdentification") + ' (' + resource.get
 ("vehicleProperties", "vehicleLicense") + ')'}
```

```
// Example 2
// // Resource Field group is named publicTransportProperties, Resource Fields is
 named passName
${resource.get("publicTransportProperties", "passName")}
```

Lokalisieren von Werten von Sortierten Listen in Ressourcen-Templates

Ab ConSol CM-Version 6.10.5.4 können Werte von Sortierten Listen lokalisiert werden, d. h. Sie können den lokalisierten Wert im Web Client anzeigen. Die Sortierte Liste *SLA_country* enthält zum Beispiel eine Liste der Länder, in denen eine SLA gültig ist. Im Admin Tool werden technische Werte verwendet und es muss für jede gewünschte Sprache ein lokalisierter Wert definiert werden. Im Web Client soll der richtige Ländername in der im Browser eingestellten Sprache angezeigt werden. Wenn im Browser keine Sprache eingestellt ist, wird die Standardsprache von CM verwendet.

Das folgende Beispiel funktioniert mit der Sortierten Liste *SLA_country*.

```
SLA: ${resource.getFieldValue("SLA_Fields_basic","SLA_Name")!} (${localize
 (resource.getFieldValue("SLA_Fields_basic","SLA_country"))!})
```

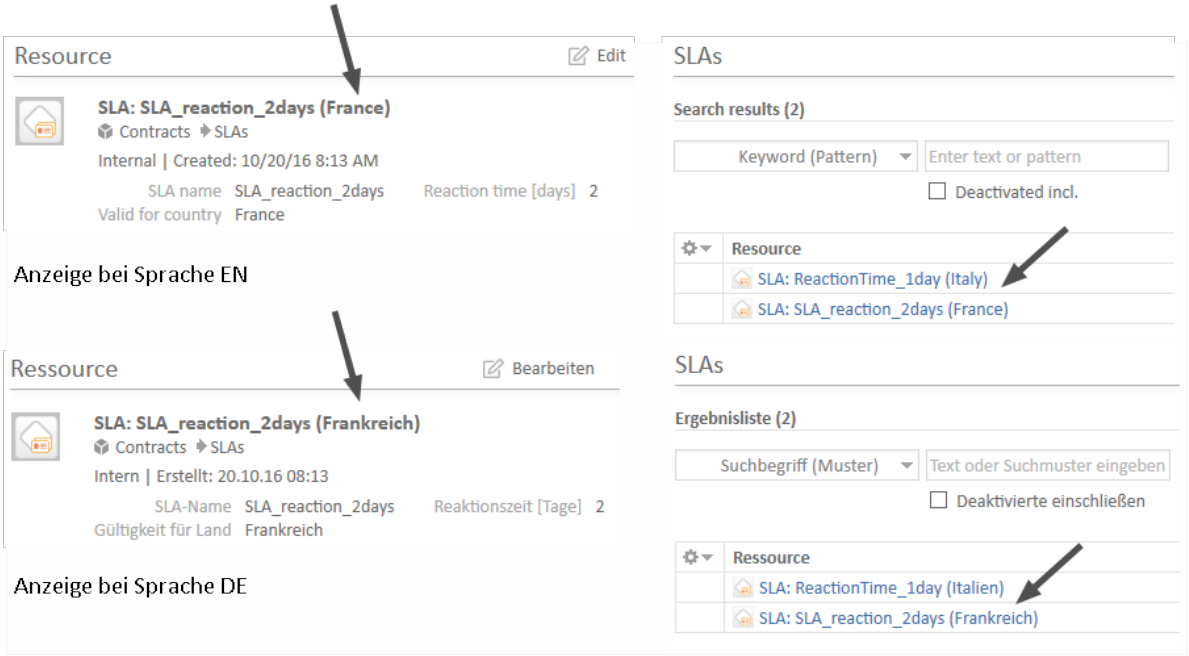


Abbildung 633: *ConSol CM Web Client: Anzeige des Wertes einer Sortierten Liste mittels Ressourcen-Template*

G.1.5.4 Template-Typen

Es können folgende Template-Typen definiert werden:

- Standard
- Schnell-Suche
- Suche
- REST
- Protokoll

Standard

Dieses Template muss immer definiert werden. Wenn es nicht definiert ist, tritt im Web Client ein Fehler auf (statt dem Namen der Ressourcen dieses Typs wird *unknown* angezeigt). Wenn keine anderen Templates definiert wurden, wird das Template an allen Stellen im Web Client und, sofern es kein spezielles Template des Typs REST gibt, auch als REST-Template für CM/Track verwendet. Die in den folgenden Abschnitten beschriebenen Templates überschreiben das Standard-Template für bestimmte Stellen im Web Client.

```
HP-Drucker: ${resource.getFieldValue("HP_Printer_Fields_basic","inventory_
number")!}
```

Das Template wird verwendet für

- Titel der Ressourcenseite
- Gezogene Ressource bei Drag-and-Drop-Vorgängen
- Favoriten

Schnell-Suche

Dieses Template definiert das Format der Ressourcendaten in den Ergebnissen der Schnellsuche. Ein Beispiel finden Sie im Abschnitt Schnellsuche.

Suche

Dieses Template definiert das Format der Ressourcendaten im Ergebnis (den Listen mit Vorschlägen) von Suchen nach Ressourcen, wenn eine Ressource mit einem anderen Objekt, z. B. einem Ticket verknüpft ist. Ein Beispiel finden Sie im Abschnitt Suche.

REST

Dieses Template definiert das Format von Ressourcendaten als Ergebnis der REST API. In der Standardkonfiguration werden keine Ressourcendaten im ConSol CM-Portal CM/Track angezeigt, das auf der REST-API basiert. Dieses Template gilt nur dann, wenn die REST-API direkt angesprochen wird, z. B. bei der Programmierung von ConSol CM-Schnittstellen. Ein Beispiel finden Sie im Abschnitt REST.

Protokoll

Dieses Template definiert das Format der Ressourcendaten im Ticket-, Kunden- und Ressourcenprotokoll, z. B. wenn eine Relation zu oder von einer Ressource hinzugefügt wurde. Ein Beispiel finden Sie im Abschnitt Protokollbereiche der Ressourcenseite, der Kundenseite und des Tickets.

G.1.5.5 Stellen im Web Client für die Verwendung von Templates

Templates werden an folgenden Stellen verwendet:

- Titel der Ressourcenseite
- Gezogene Ressource bei Drag-and-Drop-Vorgängen
- Favoriten
- Schnellsuche
- Suche
- REST
- Protokollbereiche der Ressourcenseite, der Kundenseite und des Tickets

Titel der Ressourcenseite

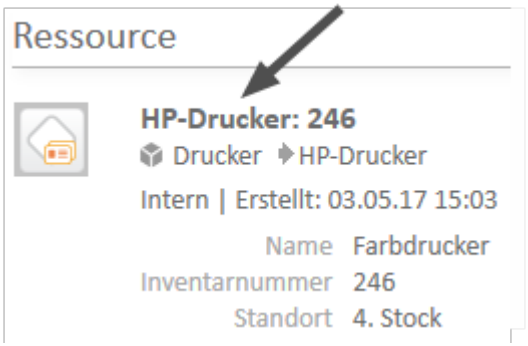
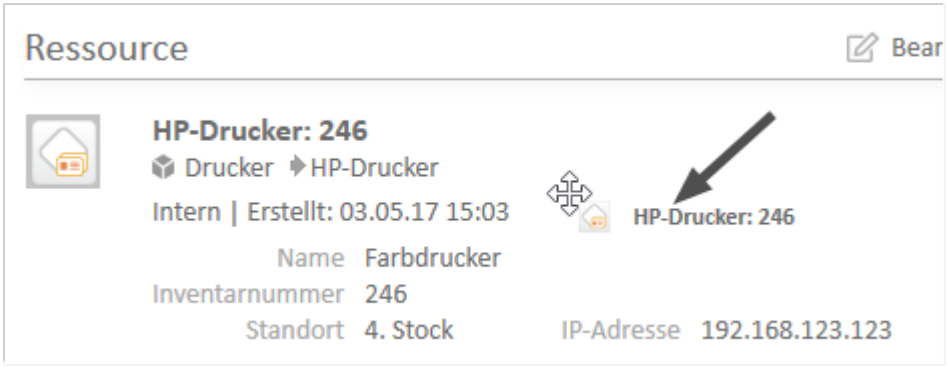


Abbildung 634: *ConSol CM Web Client - Titel der Ressourcenseite*

Verwendetes Template: *Standard* (siehe obiges Beispiel)

Gezogene Ressource bei Drag-and-Drop-Vorgängen



ConSol CM Web Client - Gezogene Ressource bei Drag-and-Drop-Vorgängen

Verwendetes Template: *Standard* (siehe obiges Beispiel)

Favoriten



Abbildung 635: *ConSol CM Web Client - Ressourcenname in den Favoriten*

Verwendetes Template: *Standard* (siehe obiges Beispiel)

Schnellsuche

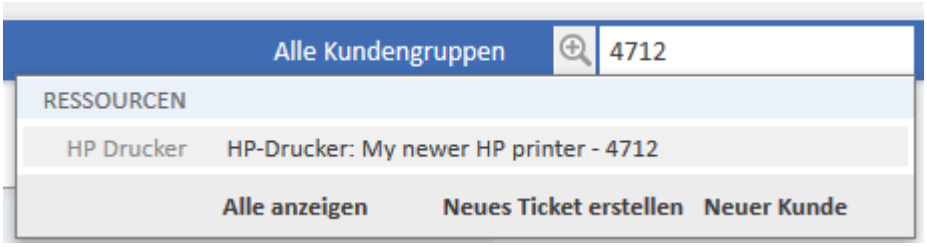


Abbildung 636: *ConSol CM Web Client - Ergebnisse der Schnellsuche (Ressource)*

Verwendetes Template: *Schnell-Suche*, sofern definiert. Andernfalls *Standard*.

```
HP-Drucker: ${resource.getFieldValue("HP_Printer_Fields_basic","name")!} -
${resource.getFieldValue("HP_Printer_Fields_basic","inventory_number")!}
```

Suche

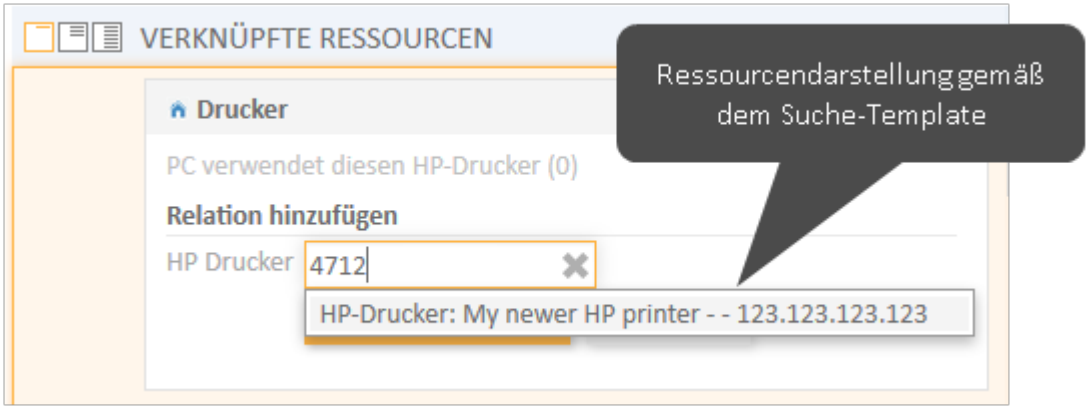


Abbildung 637: *ConSol CM Web Client - Ressource in der Liste der Vorschläge, Format basiert auf dem Suche-Template*

Verwendetes Template: *Suche*, sofern definiert. Andernfalls *Standard*.

```
HP-Drucker: ${resource.getFieldValue("HP_Printer_Fields_basic","name")!} - -
${resource.getFieldValue("HP_Printer_Fields_basic","IP_address")!}
```

Protokollbereiche der Ressourcenseite, der Kundenseite und des Tickets

Das Protokoll-Template einer Ressource wird zum Beispiel dann angezeigt, wenn im Ticketprotokoll für die erweiterte Ansicht *Anzeigen alle Einträge* ausgewählt ist.

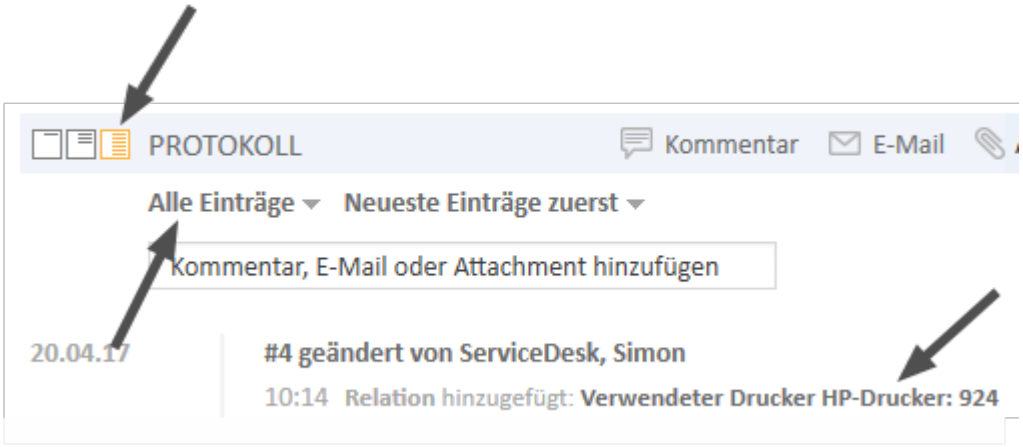


Abbildung 638: *ConSol CM Web Client - Protokoll-Template für Ressourcendaten*

Verwendetes Template: *Protokoll*, sofern definiert. Andernfalls *Standard*.

REST

Das folgende Beispiel zeigt eine Abfrage von Ressourcendaten durch einen REST Client und die entsprechende Antwort des ConSol CM-Servers über die REST API. Der Wert im Tag `<mark>` der XML-Ausgabe ist die Ressourceninformation, die mit dem REST-Template formatiert wurde. In diesem Beispiel umfasst das Template den Ressourcennamen (My new HP Printer) und die Inventarnummer (4712).

```
http://cm6doku-cm1.int.consol.de:8480/restapi/resources
```

Code-Beispiel 105: *GET-Abfrage zum Abrufen von Ressourcen im System*




```
{
    "lastPageNumber": "0",
    "pageNumber": "0",
    "pageSize": "14",
    "totalNumberOfElements": "14",
    "resource":
    [
        {
            "@uri": "http://cm6doku-cm1.int.consol.de:8480/restapi/resources/13",
            "@id": "13"
        },
        {
            "@uri": "http://cm6doku-cm1.int.consol.de:8480/restapi/resources/6",
            "@id": "6"
        },
        {
            "@uri": "http://cm6doku-cm1.int.consol.de:8480/restapi/resources/4",
            "@id": "4"
        },
        {
            "@uri": "http://cm6doku-cm1.int.consol.de:8480/restapi/resources/9",
            "@id": "9"
        },
```


Abbildung 639: *REST API - Liste aller Ressourcen*

```
http://cm6doku-cm1.int.consol.de:8480/restapi/resources/2.xml
```

Code-Beispiel 106: *GET-Abfrage zum Abrufen der Ressource mit der ID 2*




```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<resource id="2" uri="http://cm6doku-cm1.int.consol.de:8480/restapi/resources/2">
  <mark>HP Printer: My newer HP printer - 4712</mark>
  <resourceType uri="http://cm6doku-cm1.int.consol.de:8480/restapi/resourcetypes/HP_Printer?v=ITyTXos9hx4du4xOqfwzSg%3D%3D"/>
  <accessModeDate>1436523378885</accessModeDate>
  <modificationDate>1490263215984</modificationDate>
  <groups>
    <group name="HP_Printer_MaintenanceTickets">
      <definition uri="http://cm6doku-cm1.int.consol.de:8480/restapi/definitions/groups/HP_Printer_MaintenanceTickets?v=hcfixibBva3Bcb4ibFiPfA%3D%3D"/>
      <fields>
        <field xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="listFieldElement" class="LIST" name="MaintenanceTicketsList" group="HP_Printer_MaintenanceTickets">
          <name>Maintenance tickets</name>
          <uuid>56f9e101-df16-11e6-8068-b93703e2234d</uuid>
          <value/>
        </field>
      </fields>
    </group>
```


Abbildung 640: *REST API - Details einer ausgewählten Ressource (ID 2)*

```
HP Printer: ${resource.getFieldValue("HP_Printer_Fields_basic","name")!} -
${resource.getFieldValue("HP_Printer_Fields_basic","inventory_number")!}
```

Verwendetes Template: *REST*, sofern definiert. Andernfalls *Standard*.

G.1.6 CM/Resource Pool - Ressourcenrelationen

G.1.6.1 Einleitung

Eine Ressource kann Relationen zu anderen ConSol CM-Objekten haben. Im Gegensatz zu Kundenrelationen wird dabei keine Hierarchie definiert. Alle Relationen sind einfache Referenzen. Eine Ressource kann null oder mehr Relationen zu folgenden Arten von ConSol CM-Objekten haben:

- **Ticket**
  z. B. ein Vorfallticket hat eine Ticket-Ressource-Relation zu einem Drucker, der ein Problem hervorgerufen hat.
- **Firma**
  z. B. eine Firma hat eine Firma-Ressource-Relation zu einer bestimmten SLA.
- **Kontakt**
  z. B. ein Kontakt hat eine Kontakt-Ressource-Relation zu den Zugangsberechtigungen zum Terminalserver.
- **Kunde** (d. h. Firma **oder** Kontakt einer Kundengruppe)
  z. B. eine Ressource hat eine Ressource-Kunde-Relation zu einer Kundengruppe, um widerzuspiegeln, dass die Kunden in dieser Kundengruppe potentielle Empfänger von einem Newsletter sind.
- **Ressource**
  z. B. ein Laptop hat eine Ressource-Ressource-Relation zu allen angeschlossenen/konfigurierten Druckern.

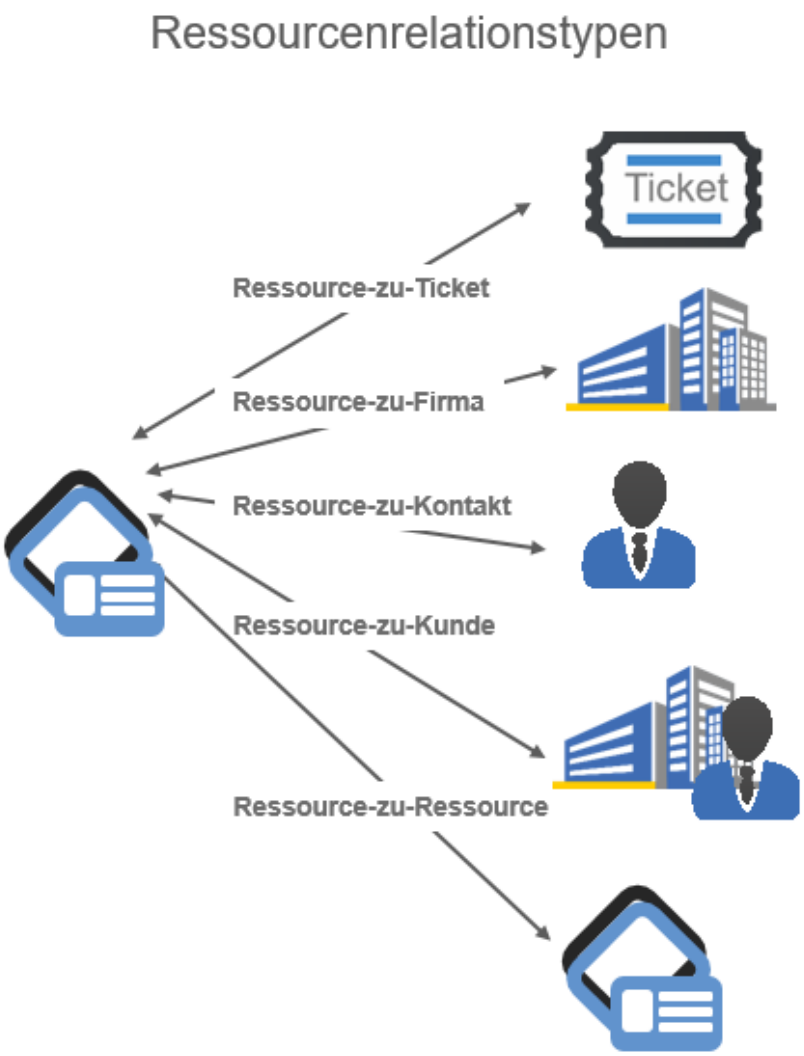


Abbildung 641: *ConSol CM - Arten von Ressourcenrelationen*

Für die Ressourcenrelation muss die Wertigkeit (Kardinalität) definiert werden, d. h. für jede Relation ist definiert, für wie viele Objekte des ausgewählten Objekttyps eine Relation erstellt werden kann. Folgende Einstellungen sind bei der Wertigkeit (Kardinalität) der Ressourcenrelation möglich:

- **1 zu 1**
  Jede Ressource kann nur eine Relation zu einem einzigen Zielobjekt haben.
- **1 zu m**
  Jede Ressource kann Relationen zu vielen unterschiedlichen Zielobjekten haben (z. B. kann ein Laptop eine Ressource-Ressource-Relation zu einem, zwei oder mehr Druckern haben).
- **m zu 1**
  Viele Ressourcen können eine Relation zu einem einzigen Zielobjekt haben.

- **m zu n**
  Viele Ressourcen können viele Relationen zu unterschiedlichen Zielobjekten haben.

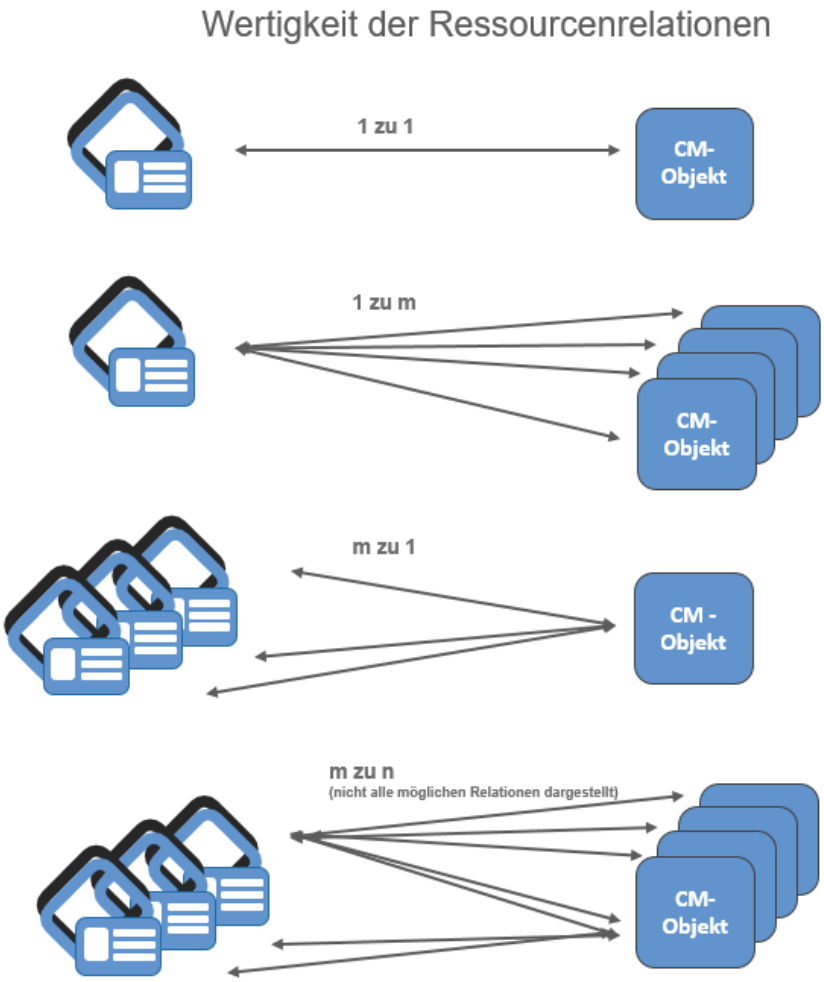


Abbildung 642: *ConSol CM - Kardinalität von Ressourcenrelationen*

G.1.6.2 Erstellen von Ressourcenrelationen im Datenmodell mit dem Admin Tool

Ressourcenrelationen werden immer für einen Ressourcentyp definiert, z. B. für den Ressourcentyp *HP_Printers*.

Um eine neue Ressourcenrelation zu erstellen oder zu editieren, öffnen Sie den Ressourcentyp im Admin Tool (Navigationsgruppe *Ressourcen*, Navigationselement *Datenmodelle*).

> Beachten Sie, dass Sie Ressourcenrelationen nicht im Navigationselement *Überblick Relationen* erstellen oder editieren können. In diesem Abschnitt werden die Relationen nur angezeigt und Sie können die Reihenfolge festlegen, in der die Relationen im Web Client angezeigt werden!

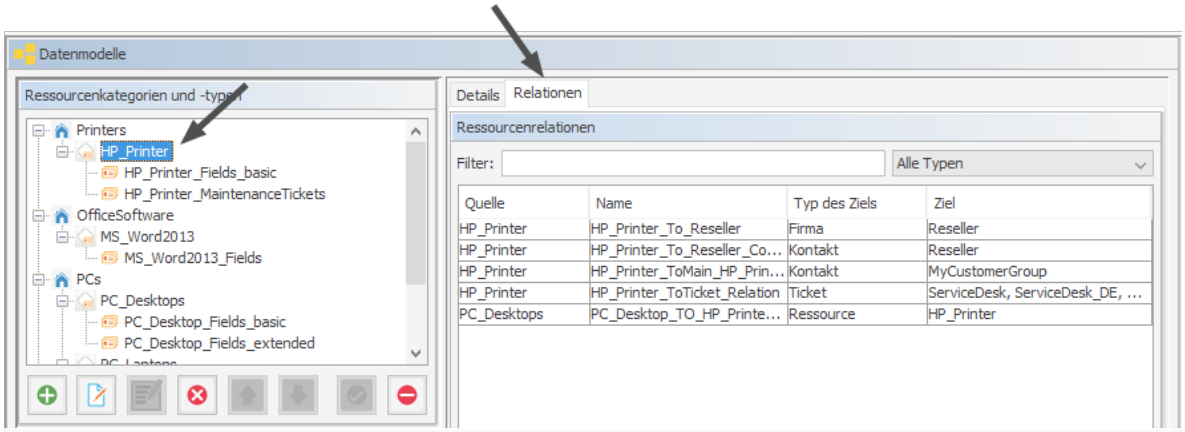


Abbildung 643: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Erstellen und Editieren von Ressourcenrelationen*

Um eine neue Ressourcenrelation zu definieren, klicken Sie auf den Button *Hinzufügen* und füllen Sie im Pop-up-Fenster die erforderlichen Felder aus.

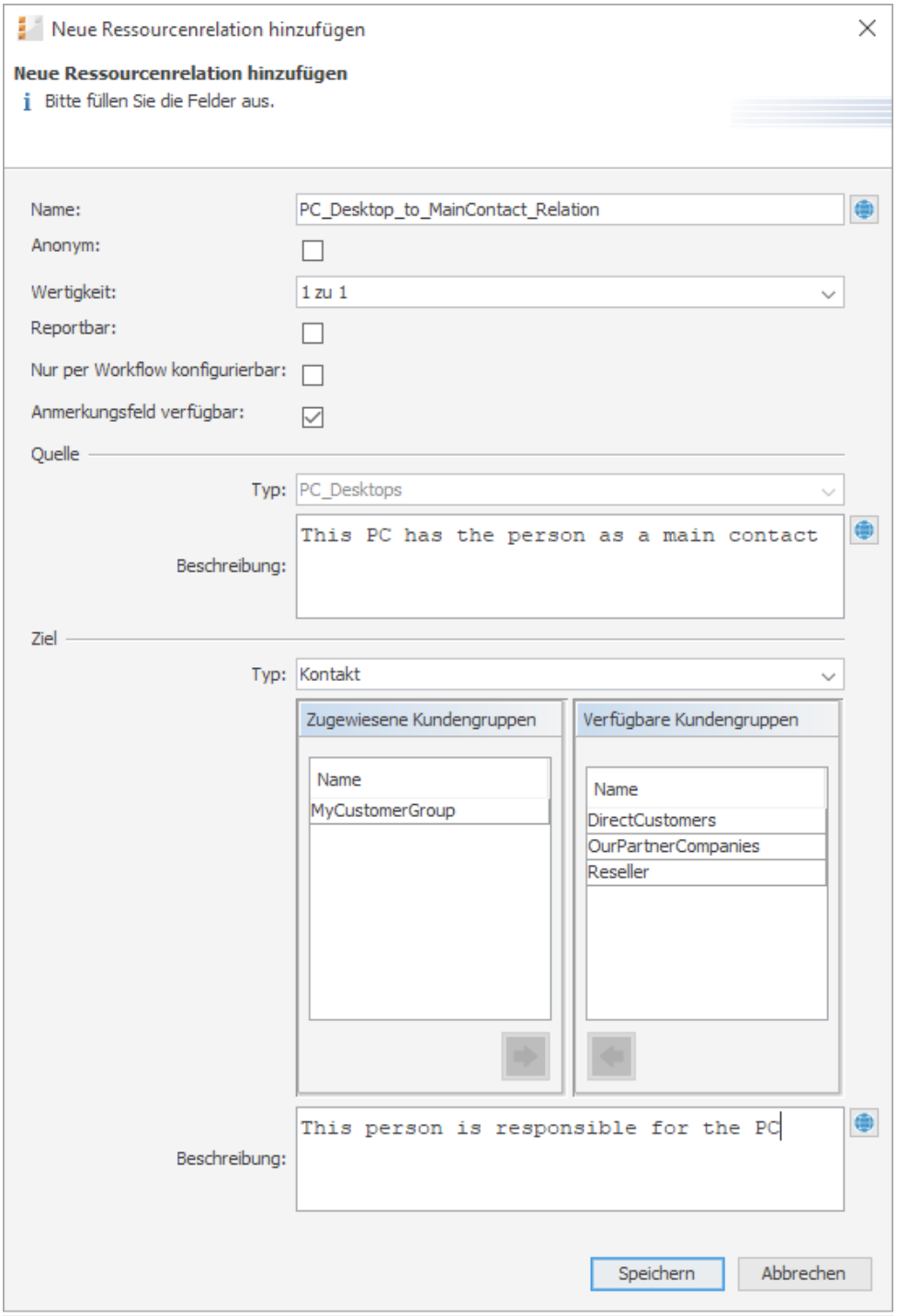


Abbildung 644: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Erstellen einer neuen Ressourcenrelation*

- **Name**
  Der technische Name der Ressourcenrelation. Sie können den Namen über den Button

Lokalisieren lokalisieren. Dieser Name wird im Web Client angezeigt, wenn eine Ressourcenrelation dieses Typs erstellt wurde. Eine detaillierte Erklärung des Lokalisierungsmechanismus finden Sie im Abschnitt Lokalisierung von Objekten allgemein, Typ 1.

- **Anonym**
  Wenn diese Checkbox markiert ist, wird im Web Client als Beschreibung für die Relation *related* angezeigt. Verwenden Sie diese Checkbox, wenn keine Beschreibung angezeigt werden soll. (Da Sie das Namensfeld nicht leer lassen können, würde sonst der technische Name der Relation angezeigt.)
- **Wertigkeit**
  Wählen Sie die Wertigkeit (Kardinalität) der Relation (siehe Erklärung zur Kardinalität oben).
- **Reportbar**
  Wenn diese Checkbox markiert ist, werden die Relationen dieses Typs an das DWH übertragen.
- **Nur per Workflow konfigurierbar**
  Wenn diese Checkbox markiert ist, kann die Relation nur über Workflow-Skripte gesetzt bzw. entfernt werden. Die Relation wird nicht in den entsprechenden Menüs des Web Clients angeboten.
- **Anmerkungsfeld verfügbar**
  Wenn diese Checkbox markiert ist, wird für jede Ressource ein Kommentarfeld angezeigt, wenn eine neue Ressourcenrelation erstellt wird (siehe die folgenden beiden Abbildungen).

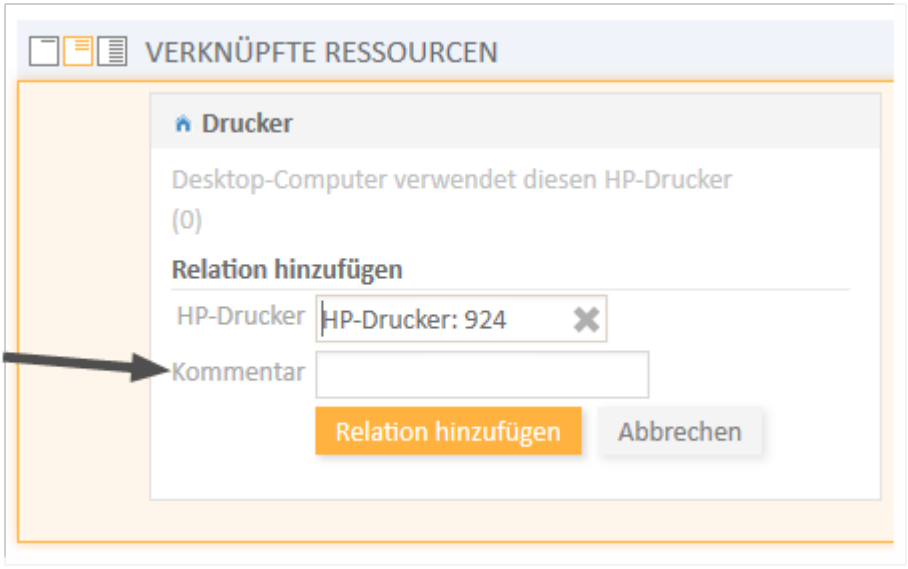


Abbildung 645: *ConSol CM Web Client - Erstellen einer Ressource-Ressource-Relation mit Anmerkungsfeld*

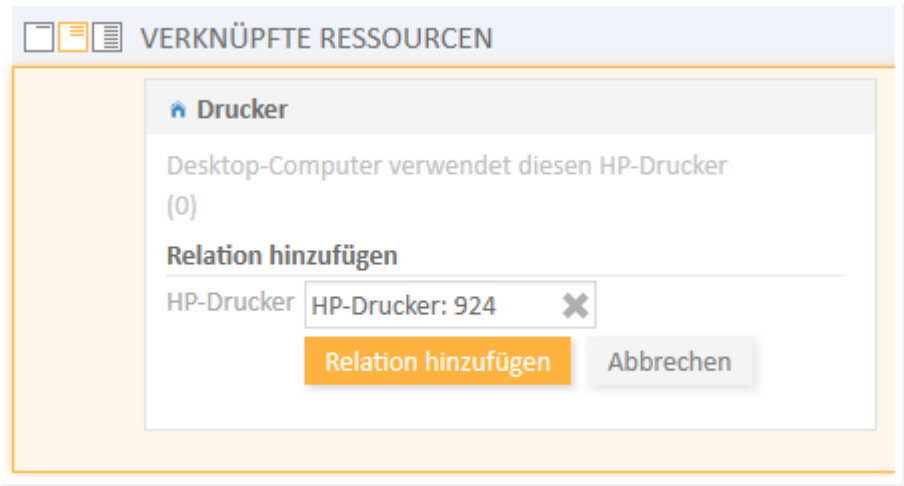


Abbildung 646: *ConSol CM Web Client - Erstellen einer Ressource-Ressource-Relation ohne Anmerkungsfeld*

- **Quelle**
  - **Typ**
    Die aktuelle Ressource. Der Wert ist bereits ausgewählt und kann nicht verändert werden.
  - **Beschreibung**
    Die Beschreibung der Relation. Wird im Web Client auf der Quellseite als Titel aller Relationen dieses Typs angezeigt. Die Beschreibung kann mit dem Button *Lokalisieren* lokalisiert werden. Eine detaillierte Erklärung des Lokalisierungsmechanismus finden Sie im Abschnitt Lokalisierung von Objekten allgemein, Typ 1.
- **Ziel**
  - **Typ**
    Der Objekttyp des Ziels der Relation. Eine Erklärung dazu finden Sie im Abschnitt über Relationstypen.
  - **Zuweisungstabelle**
    Hier können Sie der Relation Objekte des ausgewählten Typs zuweisen, d. h. die Ressourcenrelation ist nur für diese Objekte verfügbar. Abhängig vom ausgewählten Zielobjekttyp können verschiedene Objekte in der Zuweisungstabelle ausgewählt werden:
    - Zieltyp **Ressource**: zugewiesene/verfügbare Ressourcentypen
    - Zieltyp **Kunde**: zugewiesene/verfügbare Kundengruppen
    - Zieltyp **Kontakt**: zugewiesene/verfügbare Kundengruppen
    - Zieltyp **Firma**: zugewiesene/verfügbare Kundengruppen
    - Zieltyp **Ticket**: zugewiesene/verfügbare Queues

- **Beschreibung**
  Die Beschreibung der Relation. Wird im Web Client auf der Zielseite als Titel aller Relationen dieses Typs angezeigt. Die Beschreibung kann mit dem Button Lokalisieren lokalisiert werden.

G.1.6.3 Überblick Relationen

Das Navigationselement *Überblick Relationen* liefert eine Liste aller definierten Relationen und bietet die Möglichkeit, die Reihenfolge der Anzeige im Web Client zu definieren.

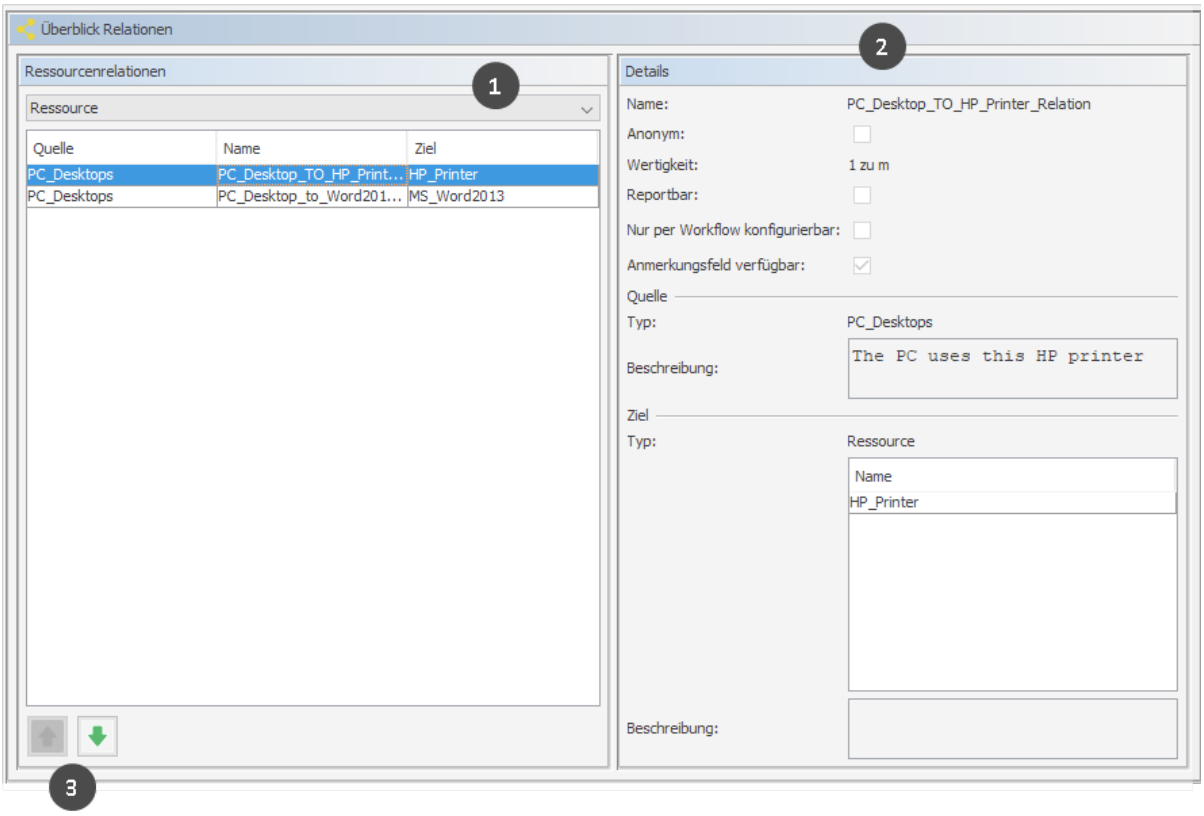


Abbildung 647: *ConSol CM Admin Tool - Ressourcen, Überblick Relationen*

Wählen Sie den Objekttyp des Ziels mit dem Filter (Drop-down-Menü) in der obersten Zeile (1). In der Liste werden nur die Relationen angezeigt, die diesen Objekttyp als Ziel haben. Details zu den möglichen Zielobjekttypen finden Sie im Abschnitt über Relationstypen.

Beachten Sie den Filter über der Liste. Sie müssen immer einen Ressourcenrelationstyp auswählen:

- Ressource
- Kunde
- Kontakt
- Firma
- Ticket

Es werden nur Relationen dieses Typs in der Liste angezeigt.

Der *Überblick Relationen* bietet zwei Funktionen:

- **Anzeigen der Relationsdetails**
  Wählen Sie eine Relation aus. Auf der rechten Seite werden die Relationsdetails im schreibgeschützten Modus angezeigt (2).
- **Ändern der Anzeigereihenfolge**
  Wählen Sie eine Relation und verschieben Sie sie mit den Pfeilen *Nach oben* und *Nach unten* innerhalb der Liste. Damit wird die Position der Relation in der Relationsliste definiert, falls mehr als eine Relation im Web Client zur Auswahl angeboten wird (3).

G.1.6.4 Erstellen von Ressourcenrelationen im Web Client

Ressourcenrelationen können je nach Quellobjekt auf mehreren Seiten des Web Clients erstellt werden.

- Ticket-zu-X-Relationen können auf der Ticketseite erstellt werden.
- Ressource-zu-X-Relationen können auf der Ressourcenseite erstellt werden.
- Kontakt-zu-X-Relationen können auf der Kontaktseite erstellt werden.
- Firma-zu-X-Relationen können auf der Firmenseite erstellt werden.

Eine kurze Beschreibung der Arbeit mit Ressourcenrelationen finden Sie im Abschnitt Kurze Einführung in die Funktionen von CM/Resource Pool im Web Client. Eine detaillierte Beschreibung aller Web-Client-Funktionen im Zusammenhang mit den Ressourcen finden Sie im *ConSol CM Benutzerhandbuch*.

G.1.6.5 Konfiguration von Ressourcenrelationen im Web Client

Einige Parameter im Zusammenhang mit dem Anzeigemodus der Ressourcenrelationen können in der Seitenanpassung konfiguriert werden. Dies sind:

- Das Standardlimit für die Anzahl an Ressourcenrelationen, damit die Steuerelemente des Tabellenfilters angezeigt werden können.
- Die Sortierstrategie der Ressourcenrelationen.

Details zu den Parametern, die sich auf Kundenseiten (Unit-Seiten) auswirken, finden Sie im Abschnitt UnitResourceRelation des Kapitels Seitenanpassung.

Details zu den Parametern, die sich auf Ressourcenseiten auswirken, finden Sie im Abschnitt TicketRelation des Kapitels Seitenanpassung.

Details zu den Parametern, die sich auf die Ticketseite auswirken, finden Sie im Abschnitt resourceRelations des Kapitels Seitenanpassung.

G.1.7 CM/Resource Pool - Ressourcenaktionen

G.1.7.1 Einleitung

Ressourcenaktionen sind Teil des ConSol CM Action Frameworks. Ressourcenaktionen sind Aktionen, die für eine Ressource ausgeführt werden, d. h. für ein im Ressourcenpool gespeichertes Objekt. Die Aktionen können automatisch vom System durchgeführt werden oder manuell von einem Bearbeiter mit den entsprechenden Berechtigungen angestoßen werden. Mögliche Anwendungsfälle für Ressourcenaktionen sind beispielsweise:

- Erstellen eines Wartungstickets für einen Drucker
- Finden aller Firmen, die eine bestimmte SLA nutzen

Sie können folgende Typen von Ressourcenaktionen verwenden:

- **Automatische** Aktionen, die vom System nach einer der folgenden Ressourcenoperationen durchgeführt werden:
  - Erzeugen
  - Aktualisieren
  - Löschen
  - Relation
  - Suche
- **Manuelle** Aktionen, die von einem Bearbeiter mithilfe der Aktivitätslinks auf der Ressourcenseite des Web Clients durchgeführt werden (ähnlich wie *Workflow-Aktivitäten* für Tickets). Manuelle Aktionen werden nur für die angezeigte Ressource ausgeführt.

> ⚠ Beachten Sie, dass nur Bearbeiter, die mindestens eine Rolle mit den folgenden Zugangsberechtigungen für den entsprechenden Ressourcentyp haben, die Ressourcenaktionen verwenden dürfen, d. h. nur bei ihnen werden die Aktivitäten im Web Client angezeigt!
>
> - Ausführen

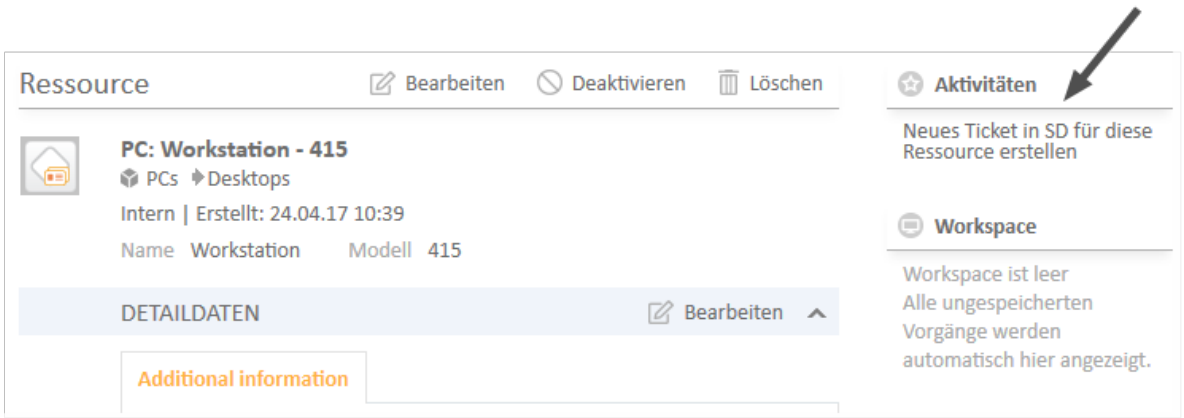


Abbildung 648: *ConSol CM Web Client - Ressourcenaktion (auf der Ressourcenseite)*

Ressourcenaktionen werden als Groovy-Skripte definiert, die im Abschnitt *Skripte und Templates* des Admin Tools gespeichert werden.

Die Ausführung von Ressourcenaktionen kann mit Bedingungsskripten gesteuert werden, d. h. Sie können ein Bedingungsskript implementieren, das vor der Ressourcenaktion selber ausgeführt wird. Das Ausführungsscript wird nur ausgeführt, wenn das Bedingungsskript „true“ zurückgibt.

Es gibt also zwei Arten von Skripten, mit denen Sie im Zusammenhang mit ConSol CM-Ressourcenaktionen arbeiten:

- **Ressourcen-Aktionsskripte**
  Definiert die Aktion, die ausgeführt werden soll.
- **Ressourcen-Bedingungsskripte**
  Definiert eine oder mehrere Bedingungen für die Ausführung des Aktionsskripts. Muss „true“ oder „false“ zurückgeben. Wenn „false“ zurückgegeben wird, wird das Aktionsskript nicht ausgeführt. Wenn es sich um eine manuelle Aktion handelt, wird diese nicht als *Aktivität* im Web Client angezeigt.

Führen Sie zur Implementierung einer Ressourcenaktion folgende drei Schritte durch:

1. Erstellen Sie ein Skript für die Ressourcenaktion (entweder nur ein Aktionsskript oder ein Aktionsskript und ein Bedingungsskript).
2. Erstellen Sie die Ressourcenaktionen, die die Skripte verwenden.
3. Weisen Sie die Ressourcenaktionen den Ressourcentypen zu, in denen sie verfügbar sein sollen.

In den folgenden Abschnitten sind diese drei Schritte detailliert beschrieben.

G.1.7.2 Erstellen von Ressourcenaktionen mit dem Admin Tool

Schritt 1: Schreiben des Ressourcenaktionsskripts

Erstellen Sie ein neues Admin-Tool-Skript des Typs *Ressourcen-Aktion*. Falls erforderlich, erstellen Sie ein anderes Skript des Typs *Ressourcen-Bedingung*.

Eine detaillierte Beschreibung der Admin-Tool-Skripte finden Sie im Abschnitt Admin-Tool-Skripte.

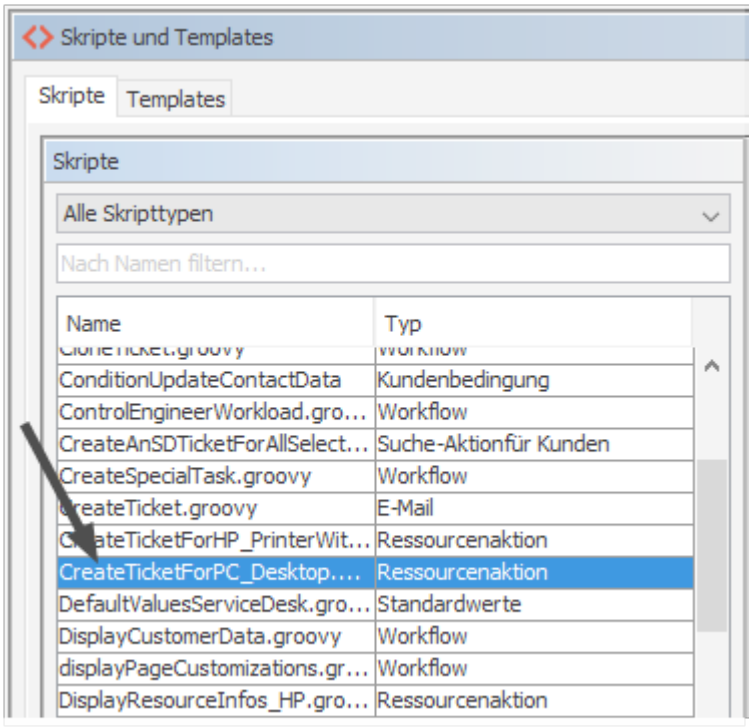


Abbildung 649: *ConSol CM Admin Tool - System, Skripte und Templates: Ressourcenaktionsskript*

```
// this script creates a new ticket for the resource from which the activity is
 executed, i.e., creates new ticket and links it to resource
// resource - ticket relation must be configured beforehand!

import com.consol.cmas.common.model.ticket.Ticket
import com.consol.cmas.common.model.customfield.Unit
import com.consol.cmas.common.model.resource.*
import com.consol.cmas.common.service.resource.*
import com.consol.cmas.common.model.ticket.Queue
import com.consol.cmas.common.model.resource.meta.*

println 'CreateTicketForResource.groovy started ...'

Ticket newtic = new Ticket()

Queue qu = queueService.getByName("ServiceDesk")
newtic.setQueue(qu)

newtic.setSubject("New Ticket for Resource: " + resource.getId())
newtic.set("helpdesk_standard.priority","low")

// use main contact person of the resource as main contact for the ticket
Unit maincont = new Unit()
def crit = new ResourceRelationWithTargetUnitCriteria()
crit.setResource(resource)
```

```
List<ResourceRelationWithTargetUnit> cont_list =
 resourceRelationService.getByCriteria(crit)
if (cont_list.size() == 0) {
   //cmweb.rp.resource.action.no_contact_set has to be configured as label in the
    AT
   return actionScriptResultFactory.getPostAction(PostActionType.FAILURE,
    "cmweb.rp.resource.action.no_contact_set")
} else {
   def cont_rel = cont_list[0]
   maincont = cont_rel.getTargetUnit()
}

ticketService.createWithUnit(newtic,maincont)

println 'New Ticket created for resource with ID' + resource.getId()

// link ticket to resource

def resRelationDefCriteria = new ResourceRelationDefinitionCriteria()
resRelationDefCriteria.addDefinitionName("PC_Desktop_to_Ticket_Relation")
def s_res_type = resource.getResourceType()
resRelationDefCriteria.addSourceResourceType(s_res_type)
resRelationDefCriteria.addTargetQueue(qu)

//log.info "resRelationDefCriteria = " + resRelationDefCriteria
//log.info "resRelationDefCriteria.definitionName = " +
 resRelationDefCriteria.getDefinitionsNames()

def resRelationDef = resourceRelationDefinitionService.getByCriteriaUniqueResult
 (resRelationDefCriteria)
def resRelation = new ResourceTicketRelation(resRelationDef, resource, newtic)
// log.info "resRelation" + resRelation
resourceRelationService.create(resRelation)
```

Code-Beispiel 107: *Ressourcenaktionsskript*

Schritt 2: Erstellen der Ressourcenaktionen, die das Skript verwenden

Öffnen Sie zum Erstellen, Editieren oder Löschen von Ressourcenaktionen das Navigationselement *Aktionen* der Navigationsgruppe *Ressourcen* im Admin Tool.

Um eine neue Aktion zu erstellen oder hinzuzufügen, klicken Sie auf den Button *Hinzufügen* und geben Sie die erforderlichen Daten im Pop-up-Fenster ein. Es wird das gleiche Pop-up-Fenster für das Hinzufügen und das Editieren von einer Ressourcenaktion angezeigt.

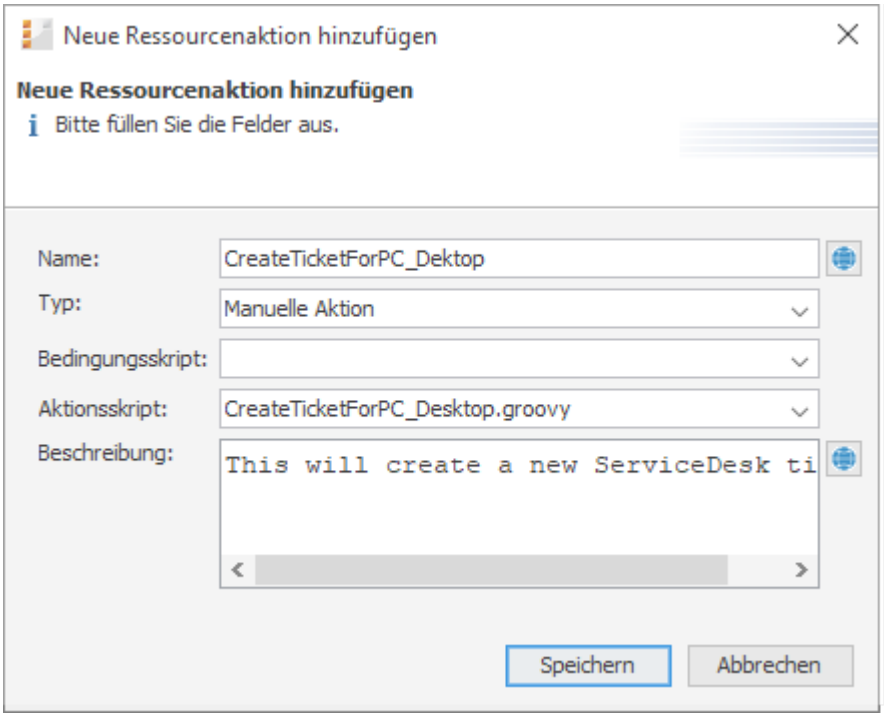


Abbildung 650: *ConSol CM Admin Tool - Ressourcen, Aktionen: Erstellen einer Ressourcenaktion*

- **Name**
  Der technische Name der Ressourcenaktion. Sie können den Wert über den Button *Lokalisieren* lokalisieren. Der lokalisierte Name wird im Web Client angezeigt. Eine detaillierte Erklärung des Lokalisierungsmechanismus finden Sie im Abschnitt Lokalisierung von Objekten allgemein, Typ 1.
- **Typ**
  Der Typ der Ressourcenaktion. Wählen Sie einen der folgenden Typen aus. Ein einmal definierter Typ kann später nicht mehr geändert werden (wenn Sie eine vorhandene Ressourcenaktion editieren).
  - **Erzeugen**
    Diese Aktion wird automatisch ausgeführt, wenn die Ressource erstellt wird.
  - **Aktualisieren**
    Diese Aktion wird automatisch ausgeführt, wenn die Ressource aktualisiert wird, d. h. wenn die Daten (entweder manuell oder automatisch) verändert und gespeichert wurden.
  - **Löschen**
    Diese Aktion wird automatisch ausgeführt, wenn die Ressource gelöscht wird.
  - **Relation**
    Dieses Skript wird automatisch ausgeführt, wenn eine Relation zu oder von einer Ressource dieses Typs

- erstellt
- gelöscht

wird. (Das Skript wird nicht ausgeführt, wenn der Kommentar einer Relation geändert wird.)

- **Manuell**
  Die Ressourcenaktion wird als *Aktivität* im Web Client angezeigt und kann nur manuell ausgeführt werden. Wenn ein Ressourcen-Bedingungsskript implementiert wurde, wird die Aktivität im Web Client nur angezeigt, wenn das Bedingungsskript „true“ zurückgegeben hat.
- **Suche**
  Die Ressourcenaktion ist eine Suche-Aktion. Aktionen dieses Typs sind im Abschnitt Action Framework - Suche-Aktionen erklärt.

- **Bedingungsskript**
  Wenn vor dem Aktionsskript ein Bedingungsskript ausgeführt werden soll, muss der Name des Bedingungsskripts hier eingetragen werden. Über das Drop-down-Menü können Sie das gewünschte Skript aus einer Liste mit allen Skripten des Typs *Ressourcen-Bedingung* auswählen, die im Skriptbereich des Admin Tools gespeichert sind. Das Ausführungsscript wird nur ausgeführt, wenn das Bedingungsskript „true“ zurückgibt. Lassen Sie das Feld leer, wenn es keine Bedingung gibt.
- **Aktionsskript**
  Der Name des Aktionsskripts, das ausgeführt werden soll. Über das Drop-down-Menü können Sie das gewünschte Skript aus einer Liste mit allen Skripten des Typs *Ressourcen-Aktion* auswählen, die im Skriptbereich des Admin Tools gespeichert sind.
- **Beschreibung**
  Geben Sie die Beschreibung ein, die im Web Client als Mouseover angezeigt werden soll (nur für manuelle Aktionen).

Speichern Sie die Aktion. Danach können Sie sie Ressourcentypen zuweisen. Siehe folgenden Schritt.

Schritte 3: Zuweisen von Ressourcenaktionen zu Ressourcentypen

Um bereits definierte Ressourcen-Ausführungs- und/oder Ressourcen-Bedingungsskripte den Ressourcentypen zuzuweisen, müssen die entsprechenden manuellen oder automatischen Aktionen einem Ressourcentyp zugewiesen werden. Öffnen Sie das Navigationselement *Datenmodelle* der Navigationsgruppe *Ressourcen* im Admin Tool. Wählen Sie den Ressourcentyp, den Sie editieren möchten, und klicken Sie auf den Button *Bearbeiten*, um das Pop-up-Fenster zu öffnen, in dem Sie die Ressourcenaktionen zuweisen können. Eine Aktion kann nur ein Ressourcenaktionsskript enthalten oder sowohl ein Ressourcenaktionsskript als auch ein Ressourcen-Bedingungsskript.

Im folgenden Beispiel (nächste Abbildung) wird eine manuelle Ressourcenaktion dem Ressourcentyp *PC_Desktops* zugewiesen.

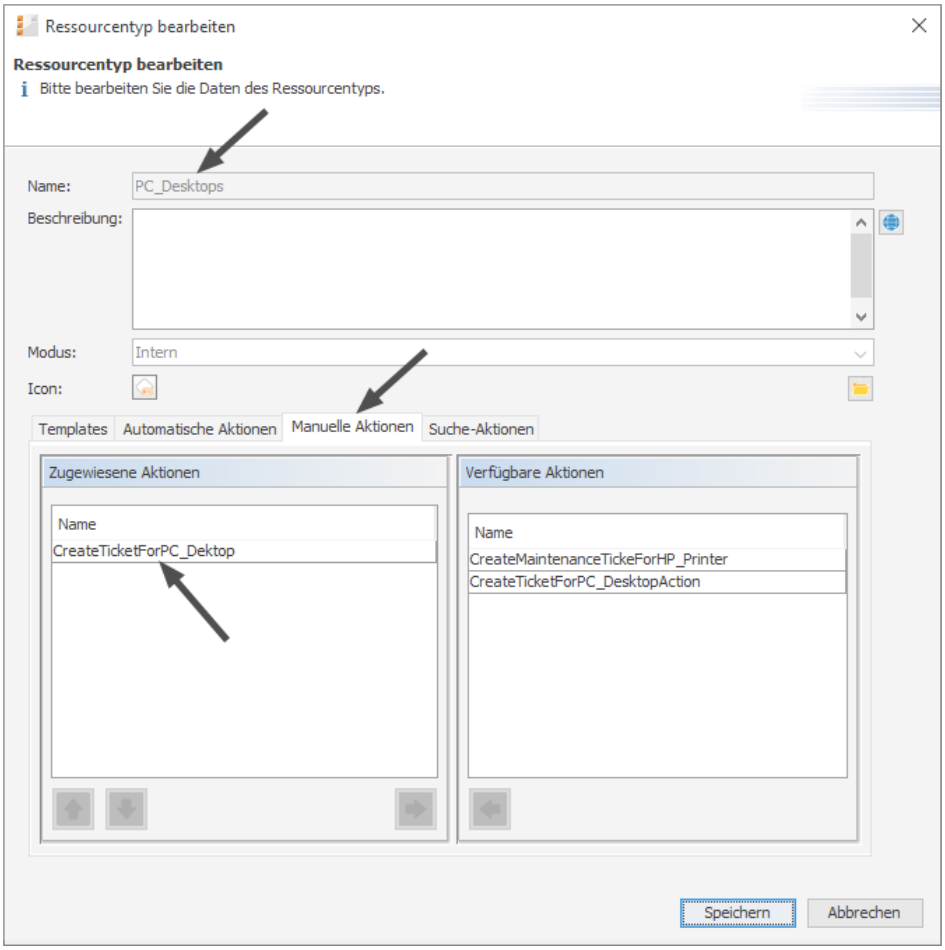


Abbildung 651: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Zuweisen von manuellen Ressourcenaktionen zu einem Ressourcentyp*

Sie können Ressourcenaktionen folgender Aktionstypen zuweisen:

- **Automatische Aktionen**
  Diese Aktionen werden automatisch ausgeführt, wenn das entsprechende Ereignis (Erstellen, Aktualisieren oder Löschen) eintritt. Wählen Sie eine Aktion für jeden gewünschten Typ. In den jeweiligen Drop-down-Menüs werden nur die Aktionen des richtigen Typs angeboten (die zuvor als Ressourcenaktionen definiert wurden, siehe Schritt 2). Zum Beispiel werden für eine automatische Aktion des Typs *Erzeugen* nur (zuvor im Navigationselement *Aktionen* definierte) Ressourcenaktionen des Typs *Erzeugen* angeboten.
- **Manuelle Aktionen**
  Diese Aktionen werden im Web Client als Aktivitäten auf der Ressourcenseite angezeigt und müssen manuell ausgeführt werden. Eine Aktivität wird nur angezeigt, wenn entweder kein Ressourcen-Bedingungsskript vorhanden ist, oder wenn das entsprechende Ressourcen-Bedingungsskript „true“ zurückgegeben hat.

- **Suche-Aktionen**
  Siehe Abschnitt Action Framework - Suche-Aktionen.

G.1.7.3 Verwendung von Ressourcenaktionen im Web Client (als Bearbeiter)

Als Bearbeiter (Benutzer) sind die beiden Ressourcenaktionstypen relevant für Sie, die als Aktivitäten im Web Client angezeigt werden:

- **Manuell**
  Manuelle Aktionen werden im Web Client angeboten, ähnlich wie Workflow-Aktivitäten für Tickets. Siehe *Beispiel 1* im nächsten Abschnitt.
- **Suche**
  Siehe Abschnitt Action Framework - Suche-Aktionen.

Die Aktionen *Löschen*, *Aktualisieren* und *Erzeugen* laufen im Hintergrund.

G.1.7.4 Programmieren von Ressourcen-Ausführungs- und Bedingungsskripten

Eine allgemeine Einführung in die Grundprinzipien, Klassen und Methoden für Ausführungs- und Bedingungsskripte finden Sie im Abschnitt Skripte für das Action Framework.

G.1.7.5 Beispiele für Ressourcenaktionen

Beispiel 1: Einfache manuelle Aktion

Anwendungsfall: Der Bearbeiter soll in der Lage sein, ein neues ServiceDesk-Ticket direkt auf der Ressourcenseite eines PCs zu erstellen. Das neue Ticket soll mit der Ressource (PC) verknüpft sein. Der Hauptkontakt des neuen ServiceDesk-Tickets soll die für den PC verantwortliche Person sein. Dies wird als *Ressource-Kontakt-Relation* im Ressourcentyp *PC_Desktop* implementiert. Um die Ressourcenaktion zu implementieren, führen Sie folgende Schritte durch.

Schreiben Sie das Ressourcenaktionsskript:

```
// this script creates a new ticket for the resource from which the activity is
 executed, i.e., creates new ticket and links it to resource
// resource - ticket relation must be configured beforehand!

import com.consol.cmas.common.model.ticket.Ticket
import com.consol.cmas.common.model.customfield.Unit
import com.consol.cmas.common.model.resource.*
import com.consol.cmas.common.service.resource.*
import com.consol.cmas.common.model.ticket.Queue
import com.consol.cmas.common.model.resource.meta.*
import com.consol.cmas.core.server.service.action.*

println 'CreateTicketForResource.groovy started ...'

Ticket newtic = new Ticket()

Queue qu = queueService.getByName("ServiceDesk")
newtic.setQueue(qu)
```

```
def subj = resource.get("PC_Desktop_Fields_basic.name")

// newtic.setSubject("New Ticket for Resource: " + resource.getId())
newtic.setSubject("New Ticket for Resource: " + subj)

newtic.set("helpdesk_standard.priority","low")

// use main contact person of the resource as main contact for the ticket
Unit maincont = new Unit()
def crit = new ResourceRelationWithTargetUnitCriteria()
crit.setResource(resource)
List<ResourceRelationWithTargetUnit> cont_list =
 resourceRelationService.getByCriteria(crit)
if (cont_list.size() == 0) {
   workflowApi.addValidationError("ERRROR","No contact set!")
} else {
   def cont_rel = cont_list[0]
   maincont = cont_rel.getTargetUnit()
}

ticketService.createWithUnit(newtic,maincont)

println 'New Ticket created for resource with ID' + resource.getId()

// link ticket to resource

def resRelationDefCriteria = new ResourceRelationDefinitionCriteria()
resRelationDefCriteria.addDefinitionName("PC_Desktop_to_Ticket_Relation")
def s_res_type = resource.getResourceType()
resRelationDefCriteria.addSourceResourceType(s_res_type)
resRelationDefCriteria.addTargetQueue(qu)

log.info "resRelationDefCriteria = " + resRelationDefCriteria
log.info "resRelationDefCriteria.definitionName = " +
 resRelationDefCriteria.getDefinitionsNames()

def resRelationDef = resourceRelationDefinitionService.getByCriteriaUniqueResult
 (resRelationDefCriteria)
def resRelation = new ResourceTicketRelation(resRelationDef, resource, newtic)
log.info "resRelation" + resRelation
resourceRelationService.create(resRelation)

// go to new ticket
return actionScriptResultFactory.getPostAction(PostActionType.GOTO_TICKET, newtic)
```

Code-Beispiel 108: *Ressourcenaktionsskript für PC_Desktops zur Erstellung eines neuen ServiceDesk-Tickets für den Kontakt, der für den PC zuständig ist*

Erstellen Sie eine Ressourcenaktion, die auf dem Skript basiert:

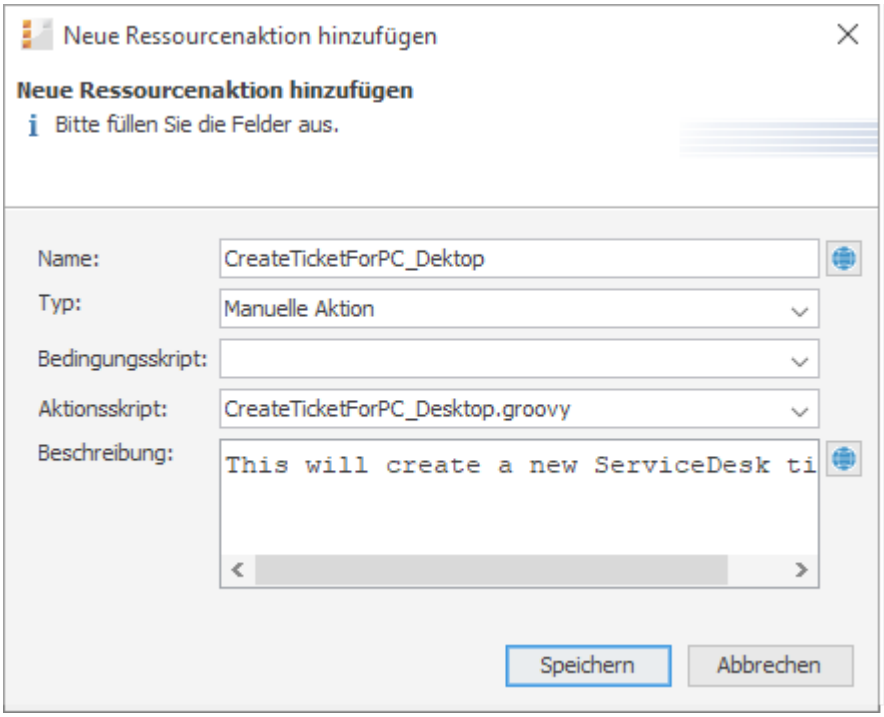


Abbildung 652: *ConSol CM Admin Tool - Ressourcen, Aktionen: Erstellen einer neuen Ressourcenaktion*

Weisen Sie die Aktion dem richtigen Ressourcentyp zu:

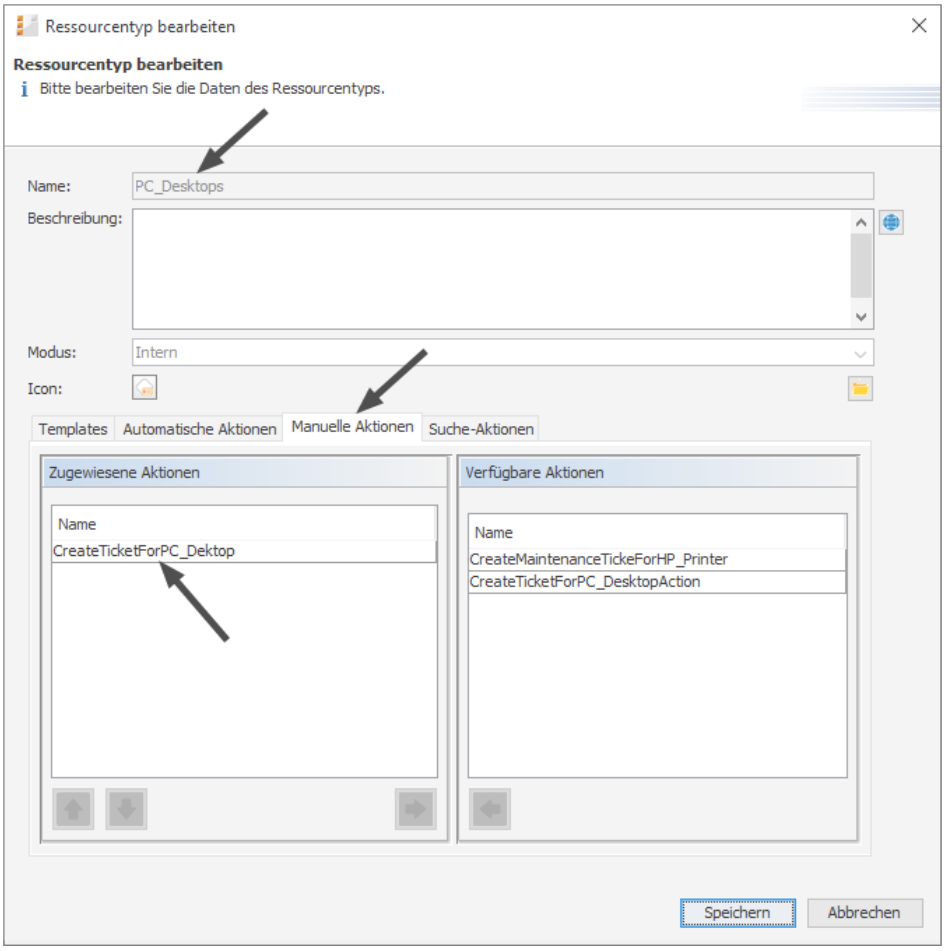


Abbildung 653: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Zuweisen der Ressourcenaktion zum richtigen Ressourcentyp*

Überprüfen Sie die Funktion mit dem Web Client:



Abbildung 654: *ConSol CM Web Client - Ressourcenseite mit der Ressourcenaktion*

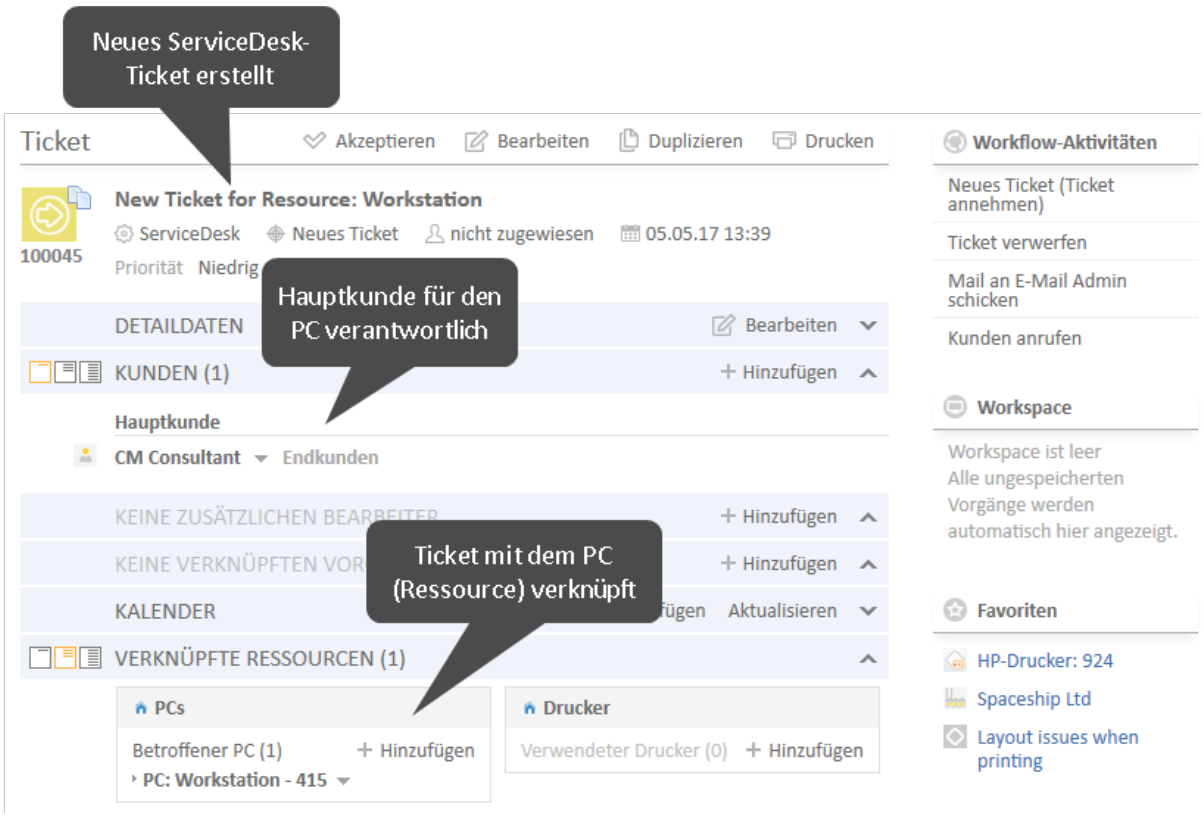


Abbildung 655: *ConSol CM Web Client - Mit der Ressourcenaktion erstelltes, neues ServiceDesk-Ticket*

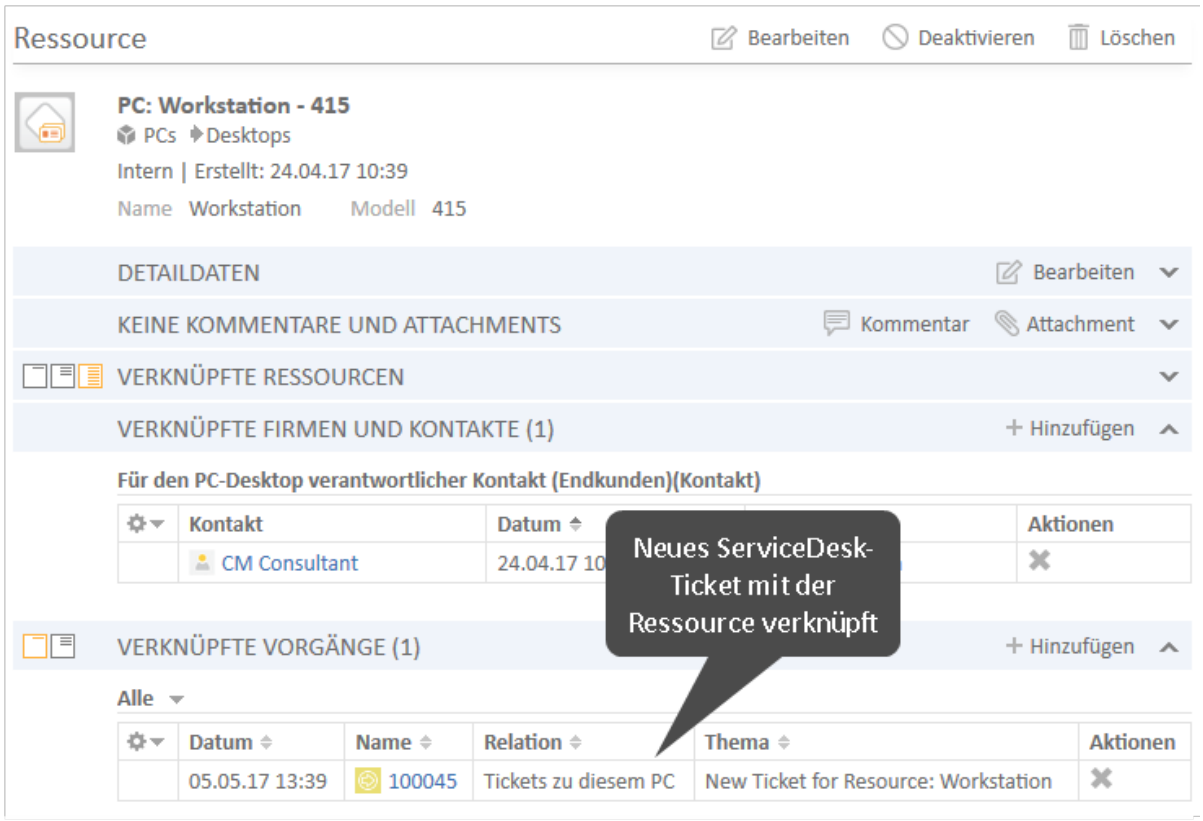


Abbildung 656: *ConSol CM Web Client - Ressourcenseite mit einer oder mehreren Ticketrelationen (neues ServiceDesk-Ticket)*

Beispiel 2: Öffnen eines Wartungstickets aus einer Ressource, mit ACF

Das Action Framework bietet die Möglichkeit, ein ACF (Activity Control Form) zu öffnen, wenn ein Ticket erstellt wird. Mit dem ACF werden Daten für die folgende Workflow-Aktivität gesammelt, d. h. das Ticket kann erstellt und einfach durch den ersten Workflow-Schritt bewegt werden. Diese Funktion wird im folgenden Beispiel dargestellt. Für die Ressource *HP-Drucker* soll ein Wartungsticket erstellt werden. Während dieses Vorgangs soll ein ACF geöffnet werden, um nach Daten zu fragen, die danach in der nächsten Workflow-Aktivität verwendet werden.

Um die Ressourcenaktion zu implementieren, führen Sie folgende Schritte durch.

Schreiben Sie das Ressourcenaktionsskripts:

```
// this script creates a new ticket for the resource from which the activity is
 executed, i.e., creates new ticket and links it to resource
// resource - ticket relation must be configured beforehand!
// CreateTicketForHP_PrinterWithACF.groovy

import com.consol.cmas.common.model.ticket.Ticket
import com.consol.cmas.common.model.customfield.Unit
import com.consol.cmas.common.model.resource.*
```

```
import com.consol.cmas.common.service.resource.*
import com.consol.cmas.common.model.ticket.Queue
import com.consol.cmas.common.model.resource.meta.*
import com.consol.cmas.core.server.service.action.*
import com.consol.cmas.common.service.*

println 'CreateTicketForHP_PrinterWithACF.groovy started ...'

Ticket newtic = new Ticket()

Queue qu = queueService.getByName("SpecialTasks")
newtic.setQueue(qu)

newtic.setSubject("New Ticket for HP Printer: " + resource.getId())
newtic.set("SpecialTasks_Fields.SpecialTasksPrio","normal")

// use main contact person of the resource as main contact for the ticket
Unit maincont = new Unit()
def crit = new ResourceRelationWithTargetUnitCriteria()
crit.setResource(resource)
List<ResourceRelationWithTargetUnit> cont_list =
 resourceRelationService.getByCriteria(crit)
if (cont_list.size() == 0) {
   log.info("ERRROR in script CreateTicketForHP_PrinterWithACF -- No contact set!")
} else {
   def cont_rel = cont_list[0]
   maincont = cont_rel.getTargetUnit()
}

ticketService.createWithUnit(newtic,maincont)

println 'New Ticket created for resource with ID' + resource.getId()

// link ticket to resource

def resRelationDefCriteria = new ResourceRelationDefinitionCriteria()
resRelationDefCriteria.addDefinitionName("HP_Printer_ToTicket_Relation")
def s_res_type = resource.getResourceType()
resRelationDefCriteria.addSourceResourceType(s_res_type)
resRelationDefCriteria.addTargetQueue(qu)

log.info "resRelationDefCriteria = " + resRelationDefCriteria
log.info "resRelationDefCriteria.definitionName = " +
 resRelationDefCriteria.getDefinitionsNames()

def resRelationDef = resourceRelationDefinitionService.getByCriteriaUniqueResult
 (resRelationDefCriteria)
def resRelation = new ResourceTicketRelation(resRelationDef, resource, newtic)
log.info "resRelation" + resRelation
resourceRelationService.create(resRelation)

// go to new ticket, but fill ACF before
```

```
def executionContext = activityFormDefinitionService.getExecutionContext(newtic,
 "defaultScope/TaskInProgress/Aufgabe_annehmen")

if (!executionContext) {
   return actionScriptResultFactory.getPostAction(PostActionType.FAILURE,
    "action.fail.wrong.activity")
}

// Modify entities from the execution context - not the original ones
// - since the user may still press cancel.
executionContext.ticket.add("SpecialTasks_Fields","Deadline", new Date());

return actionScriptResultFactory.getPostAction(PostActionType.GOTO_TICKET, newtic,
 executionContext);
```

Code-Beispiel 109: *Ressourcenaktionsskript, das ein Ticket mit ACF öffnet*

Erstellen Sie eine Ressourcenaktion, die auf dem Skript basiert.

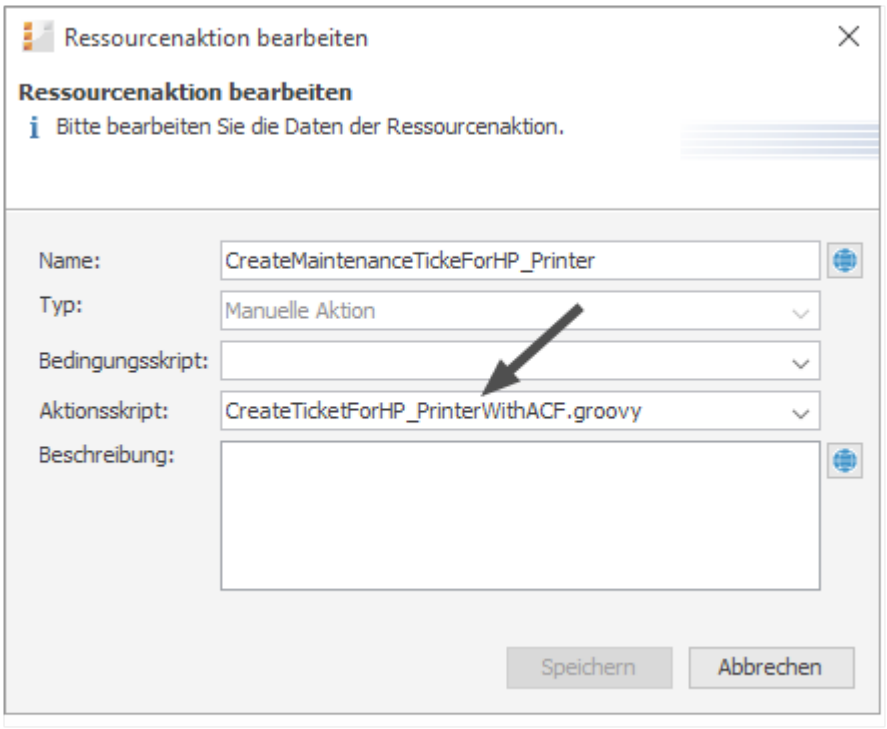


Abbildung 657: *ConSol CM Admin Tool - Ressourcen, Aktionen: Erstellen der Ressourcenaktion für das HP-Drucker-Wartungsticket*

Weisen Sie die Aktion dem richtigen Ressourcentyp zu:

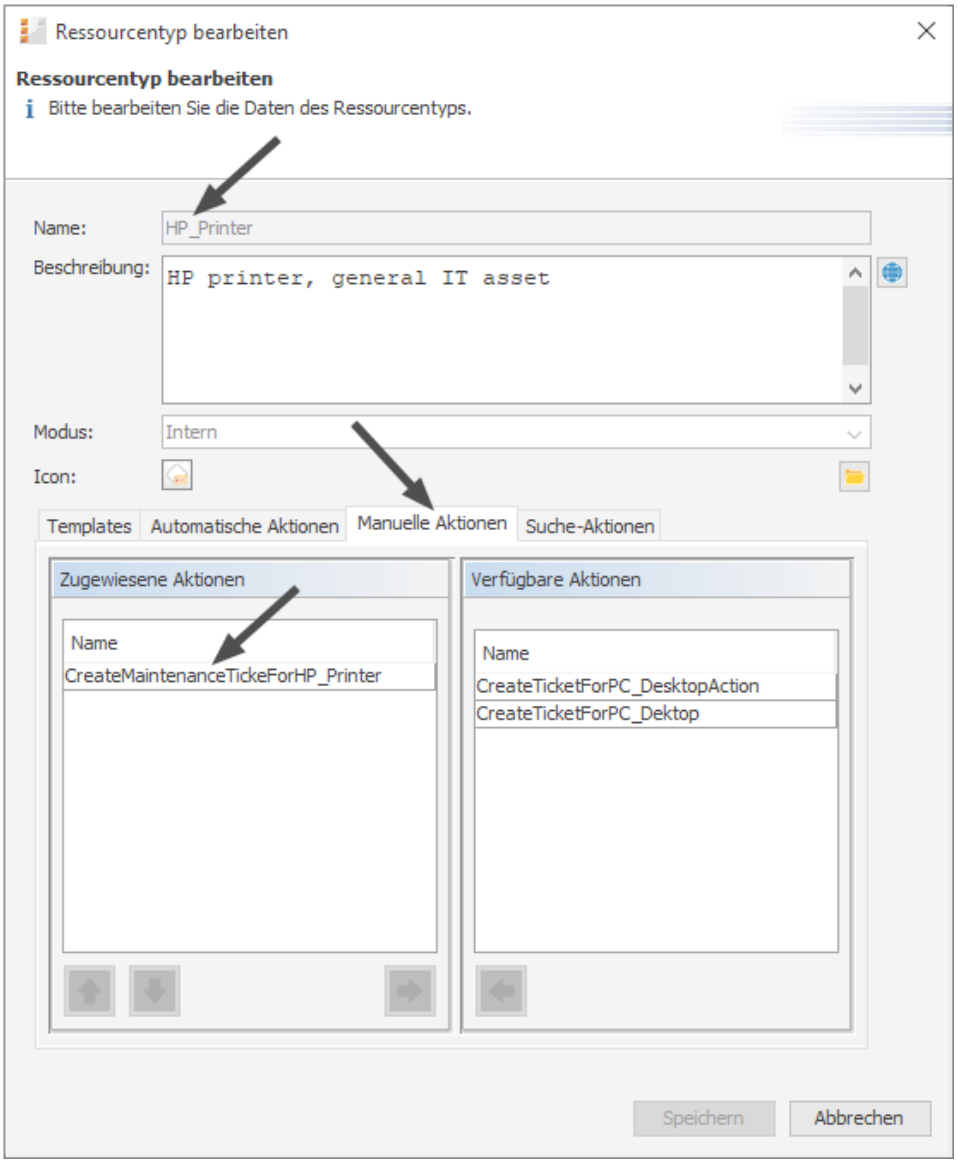


Abbildung 658: *ConSol CM Admin Tool - Ressourcen, Datenmodelle: Zuweisen der Ressourcenaktion zum richtigen Ressourcentyp (HP-Drucker)*

Überprüfen Sie die Funktion mit dem Web Client:



Abbildung 659: *ConSol CM Web Client - Ressourcenaktion für HP-Drucker*

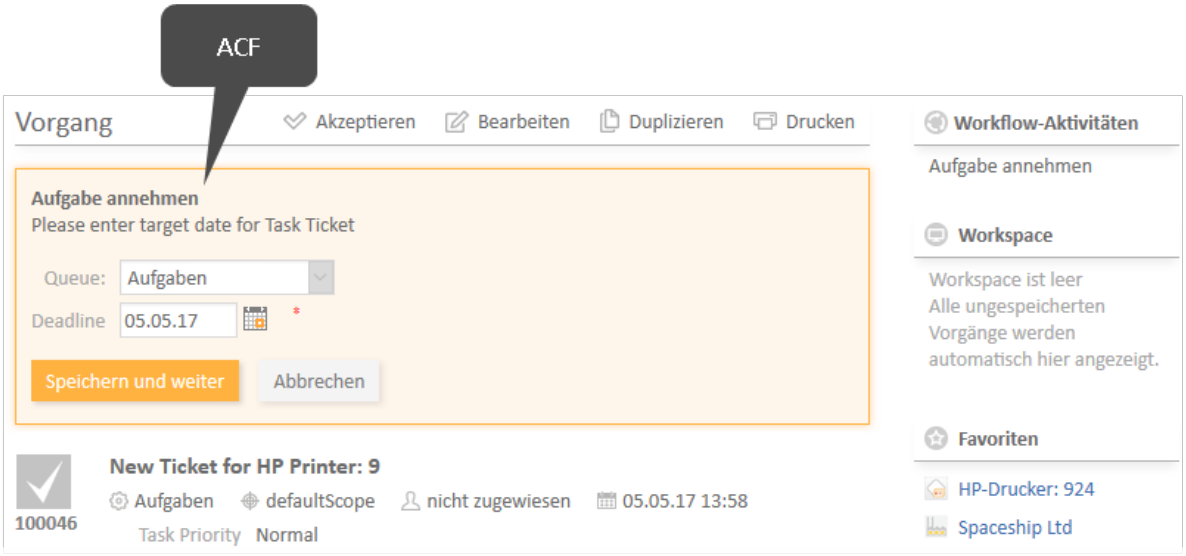


Abbildung 660: *ConSol CM Web Client - Neues Wartungsticket für Ressource (HP-Drucker), ACF*

Arbeiten mit dem Changes-Objekt in Ressourcenaktionen des Typs Aktualisieren

Ab CM-Version 10.5.4 ist es möglich, die Änderungen zu überwachen, die während einer Aktion zur Ressourcenaktualisierung durchgeführt wurden. (Dasselbe gilt für Aktionen zur Unit-Aktualisierung, siehe Arbeiten mit dem Changes-Objekt in Kundenaktionen des Typs Aktualisieren).

Mit dem Objekt der Klasse `ResourceChanges` in Ressourcenaktionen können Sie herausfinden, welche Änderungen erfolgt sind.

Denken Sie daran, dass das Skript *Aktualisieren* in folgenden Fällen ausgeführt wird:

- bei einer expliziten Aktion des Typs *Aktualisieren*
- wenn Kommentare und/oder Attachments hinzugefügt werden
- wenn Kommentare und/oder Attachments entfernt werden

Es gibt zwei Methoden des ResourceChanges-Objekts, die Informationen über die geänderten Daten bereitstellen:

- `getCustomFieldChangeInfo()`
  liefert Informationen über Änderungen an Ressourcendaten (in Ressourcenfeldern)
- `getContentChangeInfo()`
  liefert Informationen über Änderungen im Ressourcenprotokoll (Kommentare, Attachments)

Da die Rückgabeparameter der Methode ziemlich komplexe Bestandteile haben, empfehlen wir, die API Doc der Klasse `ResourceChanges` zu lesen. Das folgende Codebeispiel enthält ein Skript, in dem das Objekt `ResourceChanges` verwendet wird.

```
/**
* Available script variables:
* Manual action:
* resource - Resource for which action is executed
*
* Create, Delete Action:
* resource - Resource for which action is executed
*
* Update Action:
* resource - Resource for which action is executed
* changes - ResourceChanges object containing information about changes done for
 resource entity
*
* Relation action:
* resource - Resource for which action is executed (resource-resource)
* relation - Relation object
* resourceExternalId - External resource id (resource, unit, ticket - ext resource)
*/

// Update Action Script displayPC_DesktpChangesInLog.groovy for resources in PC_
 Desktops

import com.consol.cmas.common.model.content.unit.UnitCommentEntry
import com.consol.cmas.common.model.content.unit.UnitAttachmentEntry

log.info 'Resource (PC_Desktop) data have been UPDATEd!'
// Are there any changes?
if (changes) {
   log.info 'Yes, changes have been made to unit'
   log.info 'Changes object is a ' + changes.class
}

// Have Custom Fields been changed? If yes - which?
if (changes.customFieldChangeInfo) {
   log.info 'Yes, changes have been made to Custom Fields (Resource Fields)'
   log.info changes.customFieldChangeInfo
   log.info changes.customFieldChangeInfo.each { k, v ->
   log.info "Changed field: ${k.groupName}/ ${k.fieldName}"
   log.info "New value: ${v.value.value}"
   log.info "Old value: ${v.previousValue.value}"
}
```

```
} else {
   log.info 'No changes to Custom Fields'
}

// Have comments or attachmenst been changed? If yes - which?

log.info changes.contentChangeInfo

if (changes.contentChangeInfo){
   log.info 'Yes, changes have been made in detail section'
   if (changes.contentChangeInfo.value){
      log.info changes?.contentChangeInfo.each { ctEntry ->
         if (ctEntry?.value[0] instanceof UnitCommentEntry){
            log.info 'A comment has been added.'
            log.info 'Old value: ' + ctEntry?.previousValue
            log.info 'New value: ' + ctEntry.value[0]?.text
            log.info 'Made by the engineer ' + ctEntry.value[0]?.engineer?.name
            log.info 'Creation date of the comment: ' + ctEntry.value
              [0]?.creationDate
         } else if (ctEntry?.value[0] instanceof UnitAttachmentEntry){
            log.info 'An attachment has been added.'
            log.info 'Old value: ' + ctEntry?.previousValue
            log.info 'New value text: ' + ctEntry.value[0]?.text
            log.info 'New value file name: ' + ctEntry.value[0]?.filename
         }
      }
   } else {
      log.info 'Entry has been deleted.'
   }
}
```

Code-Beispiel 110: *Ressourcen-Aktualisierungsskript, in dem Änderungen überwacht und in die Datei server.log geschrieben werden*

Wenn bei einer Ressource des Typs *PC_Desktops* der Inhalt der Ressourcenfelder *modell* und *name* geändert wird, wird folgender Text in die Datei `server.log` geschrieben.

```
PC_DesktpChangesInLog.groovy] [Susan-] Resource (PDC_Desktop) data have been
 UPDATEd!
PC_DesktpChangesInLog.groovy] [Susan-] Yes, changes have been made to unit
PC_DesktpChangesInLog.groovy] [Susan-] Changes object is a class
 com.consol.cmas.common.model.resource.history.ResourceChanges
PC_DesktpChangesInLog.groovy] [Susan-] Yes, changes have been made to Custom Fields
 (Resource Fields)
PC_DesktpChangesInLog.groovy] [Susan-] {(modell,PC_Desktop_Fields_
 basic)=Modification{value=AbstractField{key=(modell,PC_Desktop_Fields_basic),
 value=Computer-0815-01}, previousValue=AbstractField{key=(modell,PC_Desktop_
 Fields_basic), value=Computer-0815}}, (name,PC_Desktop_Fields_basic)=Modification
 {value=AbstractField{key=(name,PC_Desktop_Fields_basic), value=My cool PC11},
 previousValue=AbstractField{key=(name,PC_Desktop_Fields_basic), value=My cool
 PC}}}
PC_DesktpChangesInLog.groovy] [Susan-] Changed field: PC_Desktop_Fields_basic/
 modell
PC_DesktpChangesInLog.groovy] [Susan-] New value: Computer-0815-01
PC_DesktpChangesInLog.groovy] [Susan-] Old value: Computer-0815
PC_DesktpChangesInLog.groovy] [Susan-] Changed field: PC_Desktop_Fields_basic/ name
PC_DesktpChangesInLog.groovy] [Susan-] New value: My cool PC11
PC_DesktpChangesInLog.groovy] [Susan-] Old value: My cool PC
```

Code-Beispiel 111: *Log-Ausgabe für das obige Skript*

G.1.8 CM/Resource Pool - Zuweisen von Berechtigungen für Ressourcen

G.1.8.1 Einführung in die Berechtigungen für Ressourcen

Gemäß dem Grundprinzip von ConSol CM haben Bearbeiter nur dann Zugriff auf Ressourcen, wenn sie über die erforderlichen Berechtigungen verfügen, d. h. wenn sie mindestens eine Rolle mit den erforderlichen Berechtigungen haben. Die Ressourcenberechtigungen werden für die Ressourcentypen vergeben. So können z. B. Mitglieder mit der Rolle *ResourceManager_IT* auf die Ressourcentypen *HP_Printer*, *MS_Word2013*, *PC_Desktops* und *PC_Laptops* zugreifen.

G.1.8.2 Zuweisen von Ressourcenberechtigungen mit dem Admin Tool

Öffnen Sie zum Verwalten der Ressourcenberechtigungen den Tab *Ressourcentyp-Berechtigungen* (Navigationsgruppe *Zugriff und Rollen*, Navigationselement *Rollen*) im Admin Tool.

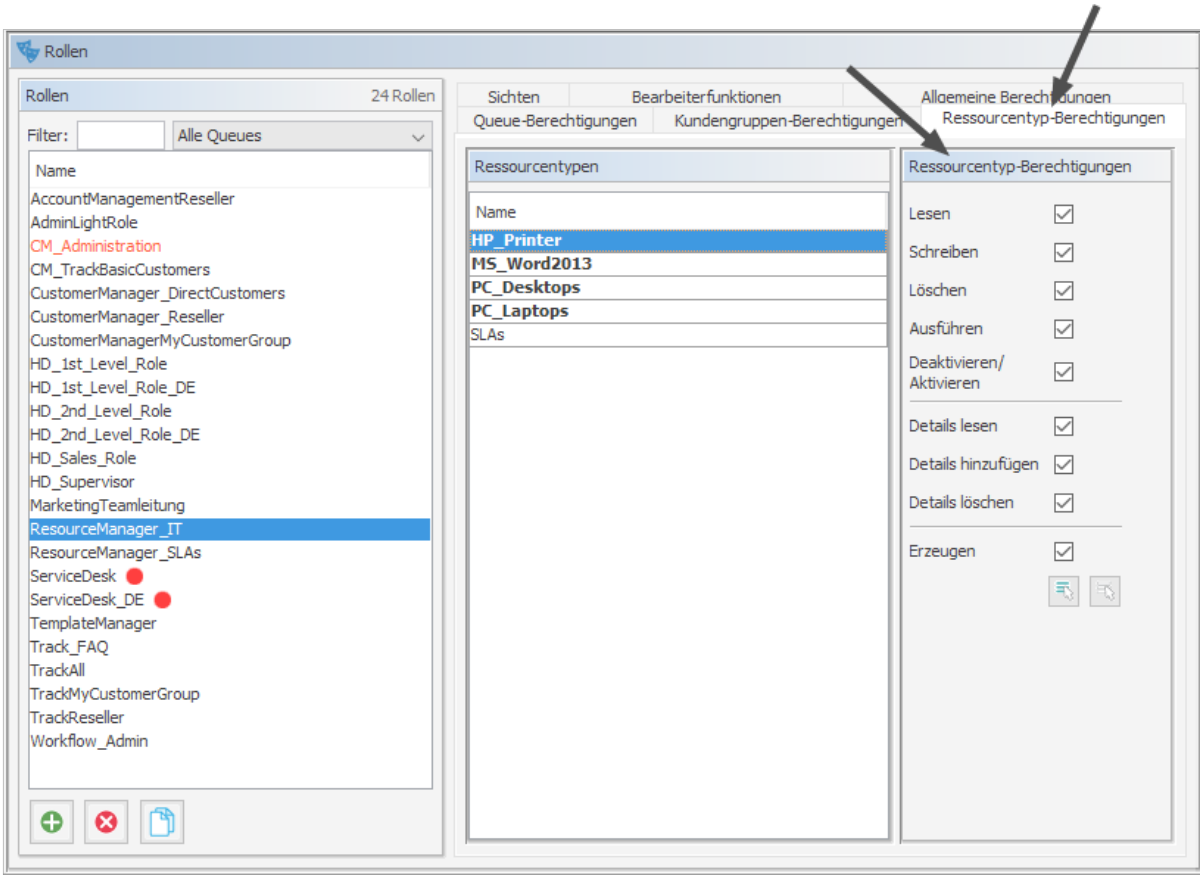


Abbildung 661: *ConSol CM Admin Tool - Zugriff und Rollen, Rollen: Zuweisen von Ressourcenberechtigungen*

Eine detaillierte Beschreibung der Berechtigungen finden Sie im Abschnitt Tab Ressourcentyp-Berechtigungen.

G.1.9 CM/Resource Pool - Das Ressourcenpool-Dashboard

G.1.9.1 Einleitung

Das Ressourcenpool-Dashboard bietet eine Übersicht über alle Objekte des Ressourcenpools. Gemäß dem Grundprinzip von ConSol CM werden nur die Objekte angezeigt, für die der Bearbeiter, der sich anmeldet und das Dashboard öffnet, Zugangsberechtigungen hat.

Das Ressourcenpool-Dashboard enthält immer die Übersicht über alle Ressourcen (Abschnitt *Ressourcen anzeigen oder neu erstellen*, siehe folgende Abbildung). Es kann zusätzlich Diagramme und/oder Tabellen enthalten, die Zahlen bzw. Reports über Ressourcen zeigen.

Öffnen Sie das Ressourcenpool-Dashboard, indem Sie im Hauptmenü auf *Ressourcenpool* (bzw. die in Ihrem CM-System konfigurierte Bezeichnung, siehe Abschnitt Bezeichnungen) klicken:

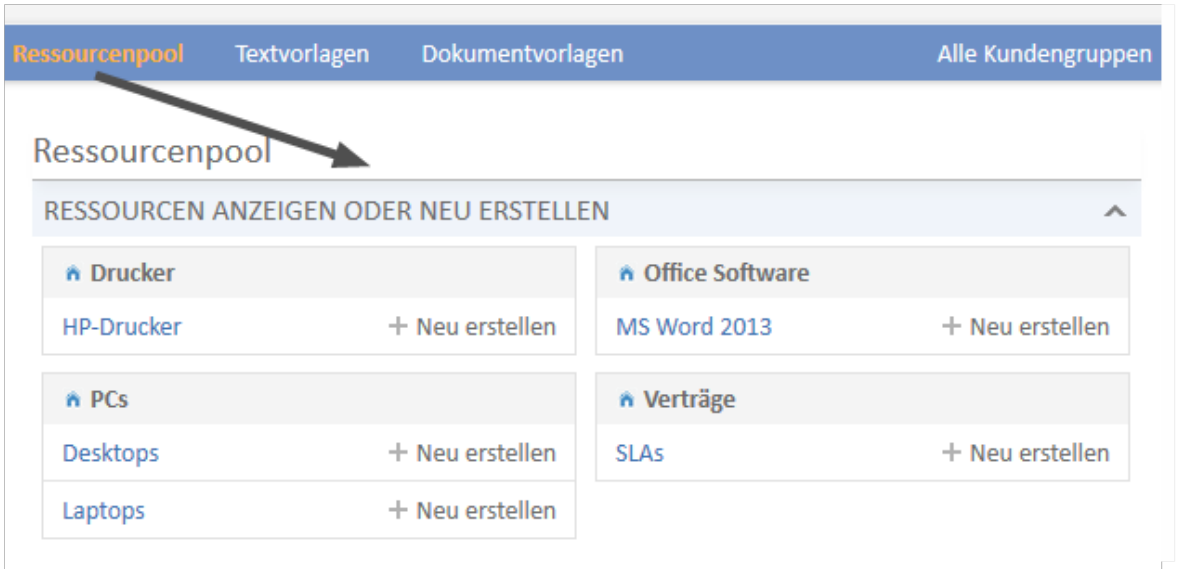


Abbildung 662: *ConSol CM Web Client - Ressourcenpool-Dashboard ohne Reports*

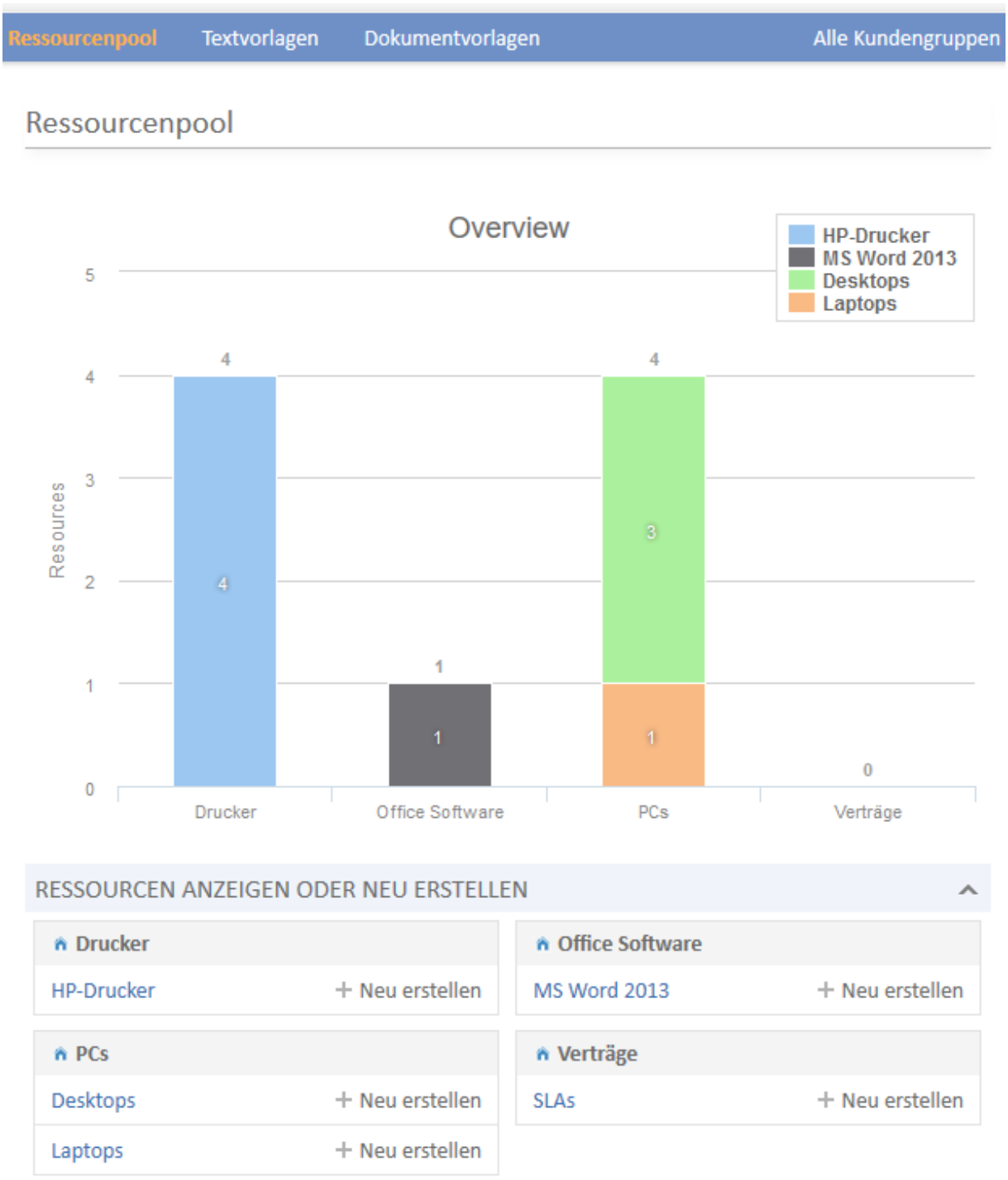


Abbildung 663: *ConSol CM Web Client - Ressourcenpool-Dashboard mit Reports*

Der Ressourcenpool, mit dem Ressourcenpool-Dashboard, ist eine neue Funktion, die mit CM-Version 6.10 eingeführt wurde. Nach einem Systemupdate von einer Version vor 6.10 wird allerdings nur der

Bereich *Ressourcen anzeigen oder neu erstellen* angezeigt (sofern die Lizenz für CM/Resource Pool vorhanden ist). Alle Reports des Ressourcenpool-Dashboards (Chart- und/oder Tabellen-Widgets) müssen manuell konfiguriert werden.

G.1.9.2 Konfigurieren von Reports für das Ressourcenpool-Dashboard

Die Konfiguration von Reports mit Grafiken und Tabellen (oder beidem) für das Ressourcenpool-Dashboard funktioniert nach dem gleichen Prinzip wie die Konfiguration des Web Client Dashboards (siehe Abschnitt Seitenanpassung für das Web Client Dashboard).

Das **Ressourcenpool-Dashboard** wird in der *Seitenanpassung* konfiguriert.

Melden Sie sich als *admin* an, öffnen Sie die Seite *Ressourcenpool* und wählen Sie im Hauptmenü *Seitenanpassung aktivieren*. Neben anderen Attributen, die für das Dashboard nicht relevant sind und im Abschnitt Seitenanpassung erklärt werden, können die folgenden, für das Ressourcen-Dashboard relevanten, Elemente konfiguriert werden (d. h. die Attribute können gesetzt werden). Jedes der drei Elemente wird durch eine Unterstruktur in der Baumstruktur der Seitenanpassung dargestellt.

1. **widgetsGrid / resourceDashboard**
   Hier wird das Ressourcenpool-Dashboard ein- oder ausgeschaltet. Wenn ein gültiger Wert im Layout-Feld eingegeben ist, wird das Dashboard angezeigt. Wenn das Feld leer ist, wird kein Dashboard angezeigt.
   Hier können folgenden Konfigurationen vorgenommen werden:
   a. Das Layout des Dashboards, d. h. das Layout des Rasters, auf dem das Dashboard basiert (siehe Abschnitt Definition des übergreifenden Dashboard-Layouts). Dies beinhaltet:
      i. Die Widgets, die angezeigt werden sollen.
      ii. Die Reihenfolge und Anordnung dieser Widgets auf der Dashboard-Seite.
2. **chartWidget / resourceDashboard** (nur verfügbar, wenn es Chart-Widgets gibt)
   a. Die Definition bzw. das Layout für alle Chart-Widgets unter *chartWidget* in der Baumstruktur.
   b. Jedes Widget ist ein Knoten, der den Namen des Widgets trägt, z. B. *chartWidget / resourceDashboard / ResourceDashboardOverview1*.
   c. Ein neues Chart-Widget kann hinzugefügt werden, indem der Name im Attribut *layout* hinzugefügt wird.
   d. Attribute können für alle Chart-Widgets gemeinsam auf der Ebene *chartWidget* oder *chartWidget / resourceDashboard* definiert werden. Alternativ können sie für jedes einzelne Chart-Widget individuell über die Werte der Attribute des Chart-Widgets konfiguriert werden, z. B. *chartWidget / resourceDashboard / ResourceDashboardOverview1*.
3. **tableWidget / resourceDashboard** (nur verfügbar, wenn es Tabellen-Widgets gibt)
   a. Die Definition aller Tabellen-Widgets unter *tableWidget* in der Baumstruktur.
   b. Jedes Widget ist ein Knoten, der den Namen des Widgets trägt, z. B. *tableWidget / resourceDashboard / ResourceDashboardOverview2_table*.

c. Ein neues Tabellen-Widget kann hinzugefügt werden, indem der Name im Attribut *layout* hinzugefügt wird.

d. Attribute können für alle Tabellen-Widgets gemeinsam auf der Ebene *tableWidget* oder *tableWidget / resourceDashboard* definiert werden. Alternativ können sie für jedes einzelne Tabellen-Widget individuell über die Werte der Attribute des Tabellen-Widgets konfiguriert werden, z. B. *tableWidget / resourceDashboard / ResourceDashboardOverview2_table*.

Die folgende Abbildung zeigt ein Beispiel für eine Baumstruktur der Seitenanpassung mit den entsprechenden Unterstrukturen für das Ressourcenpool-Dashboard. Eine detaillierte Beschreibung finden Sie in den folgenden Abschnitten.

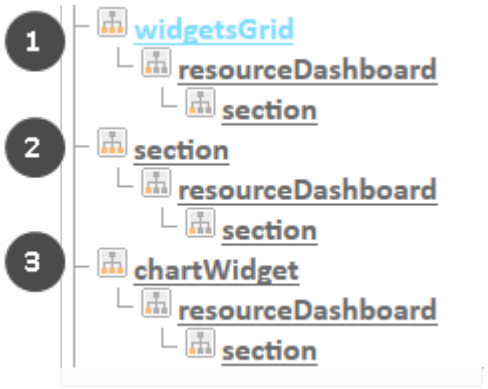


Abbildung 664: *ConSol CM Web Client - Unterstruktur der Seitenanpassung für die Konfiguration des Ressourcenpool-Dashboards*

- Gesamtlayout des Ressourcenpool-Dashboards (1)
- Abschnitt mit der Dashboard-Übersicht mit allen Ressourcen (Standard) (2)
- Definition des Layouts für alle Chart-Widgets im Ressourcenpool-Dashboard (3)

G.1.9.3 Definition des übergreifenden Dashboard-Layouts

Das übergreifende Layout des gesamtes Ressourcenpool-Dashboards wird mit dem Seitenanpassungsattribut *widgetsGrid / resourceDashboard* definiert.

Attribute:

- **layout**
  Damit wird das Layout des gesamten Dashboards auf der Seite des Ressourcenpool-Dashboards basierend auf folgenden Prinzipien definiert:
  - Jede Zeile des Dashboard-Rasters stellt eine Reihe Elemente dar: [x,y,z]. Ein neues Widget-Objekt wird automatisch zur Baumstruktur der Seitenanpassung hinzugefügt, wenn es als Wert zum Attribut *layout* hinzugefügt wird, z. B. wenn der Wert *vorher* [*ResourceDashboardOverview1:Chart*] war und *jetzt* [*ResourceDashboardOverview1:Chart*, *ResourceDashboardOverview1*, *ResourceDashboardOverview2_table:Table*] ist, wird ein neues Tabellen-Widget mit dem Namen *ResourceDashboardOverview2_table* in der Baumstruktur der Seitenanpassung angezeigt (siehe obige Abbildung). Auf diese Weise

können Widgets auch aus dem Dashboard gelöscht werden. Entfernen Sie einfach den Namen und Typ des Widgets aus dem Attribut *layout*. Nachdem Sie die Seite gespeichert und neu geladen haben, sind alle Layout-Änderungen für die weitere Konfiguration in der Baumstruktur verfügbar.

- Ein Widget wird durch seinen Namen und Typ beschrieben. Diese sind durch einen Doppelpunkt getrennt, d. h. *ResourceDashboardOverview1:Chart*. Der Name eines Widgets muss eindeutig sein.
- Das Raster beginnt in der oberen linken Ecke (0,0) und wird Zeile um Zeile aufgebaut. Ein Layout-Wert mit zwei Paaren von eckigen Klammern [ ] beschreibt zwei Zeilen wie in der Abbildung und im Code unten gezeigt.
- *null* ist das reservierte Schlüsselwort für eine leere Zelle.
- Der Typ eines Widgets ist *Chart* oder *Table*. Der Typ muss nur bei der ersten Nennung des Widget-Namens angegeben werden. Danach kann er weggelassen werden, z. B. [*ResourceDashboardOverview1:Chart*, *ResourceDashboardOverview1*, *ResourceDashboardOverview1*] bei einem Chart mit drei Spalten.
- Das Widget kann mehrere nebeneinander liegende Zeilen und Spalten belegen.
- Das ganze Dashboard kann deaktiviert werden, indem der Wert im Attribut *layout* entfernt wird.

Die folgende Abbildung zeigt die Organisation eines Beispielrasters und seine Darstellung in der Seitenanpassung. Dies ist ein einfaches Beispiel mit nur einem Chart. Kompliziertere Beispiele werden für das Web-Client-Dashboard gezeigt, siehe Abschnitt Seitenanpassung für das Web Client Dashboard.



Abbildung 665: *Organisation eines Beispielrasters (nur 1 Zeile, 1 Spalte) für die Seitenanpassung des Ressourcenpool-Dashboards*

Der Wert des entsprechenden Attributs *layout* wäre in diesem Fall:

```
[ResourceDashboardOverview1:Chart]
```

Wenn Sie mit nur einem Chart oder einer Tabelle arbeiten, ist es nicht erforderlich, mehr als eine Spalte anzugeben, da die komplette Breite der Seite mit dem Widget gefüllt wird.

G.1.9.4 Konfiguration der Widgets

Konfigurationsskript für Widgets

Jedes Chart-Widget und jedes Tabellen-Widget hat ein Konfigurationsskript, das die Daten für den Chart bzw. die Tabelle bereitstellt (z. B. Herstellen einer Verbindung zur Datenbank, Ausführen der SQL-Statements, mit denen die Daten abgerufen werden, und korrektes Anzeigen der Daten im nächsten Schritt). Dabei handelt es sich um ein Groovy-Skript, das im Admin-Tool-Bereich *Skripte und Templates* hinterlegt ist und durch seinen Namen referenziert wird. Das Skript muss den Typ *Seitenanpassung* haben. Wählen Sie das Widget im Definitionsbereich der Seitenanpassung aus und tragen Sie den Namen des Skripts ein:

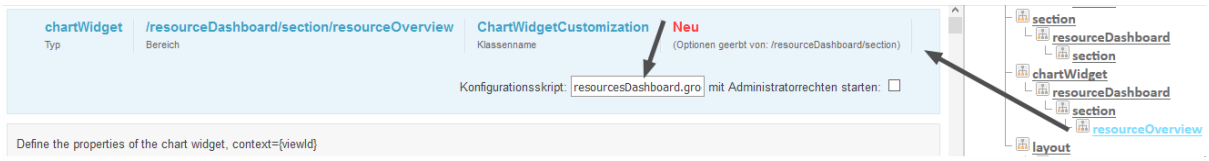


Abbildung 666: *ConSol CM Web Client - Definition des Skripts für ein Chart-Widget*

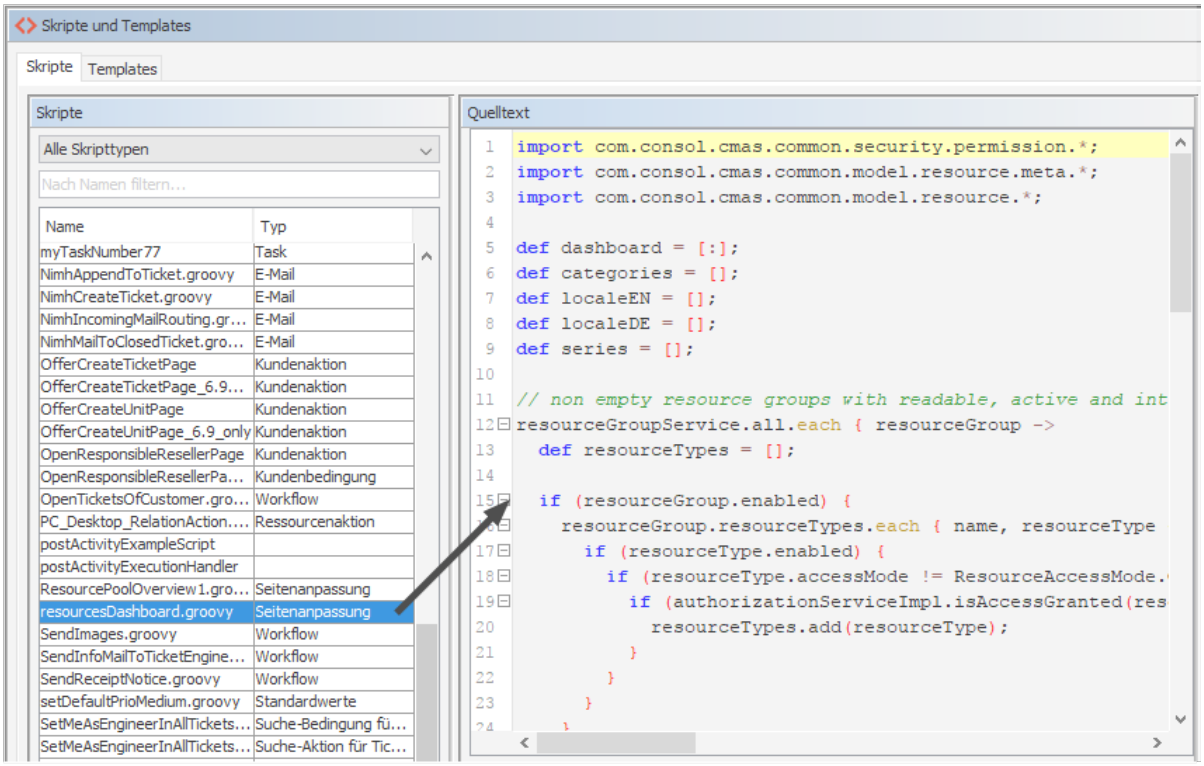


Abbildung 667: *ConSol CM Admin Tool - System, Skripte und Templates: Admin-Tool-Skript für ein Widget des Ressourcenpool-Dashboards*

Im Konfigurationsskript des Widgets werden die Statements, mit denen die erforderlichen Daten aus dem ConSol CM-System abgerufen werden, und das Layout des Widgets definiert. Die Ausführung dieses Groovy-Skripts ist ein Kernstück der Anpassung. Das Skript muss eine Map mit Variablen zurückgeben, die zu den definierten Widget-Eigenschaften passen.

> ⚠ Das Skript überschreibt die in der Seitenanpassung angegebenen Konfigurationsdaten. Die Werte werden nicht zusammengeführt!
>
> Das Skript überschreibt alle in der Seitenanpassung gesetzten Werte der Widget-Attribute. Stellen Sie daher sicher, dass das gewünschte Attribut nicht mittels Skript gesetzt wird, wenn Sie die Attribute in der Seitenanpassung definieren möchten.

Ein **Skript**, das mit einem Widget verknüpft ist, wird normalerweise mit den Berechtigungen des Benutzers (= Bearbeiters) ausgeführt. Manchmal müssen allerdings Werte verwendet werden, die im Kontext des Bearbeiters nicht verfügbar sind, z. B. eskalierte Tickets (aller Bearbeiter) in einer bestimmten Queue. Um das Skript mit Administratorberechtigungen auszuführen, markieren Sie die Checkbox *mit Administratorrechten starten*. Denken Sie daran, dass die Ergebnisse der Java- oder Groovy-Methoden, mit denen die Daten abgerufen werden, je nach Kontext unterschiedlich sein können. Die Methode `ticketService.getAll()` gibt zum Beispiel nur Tickets zurück, für die der aktuelle Bearbeiter mindestens Leseberechtigungen hat. Wenn die gleiche Methode mit Administratorberechtigungen ausgeführt wird, gibt sie aber alle im System vorhandenen Tickets zurück.

Die **Chart-Darstellung** des Ressourcenpool-Dashboards basiert auf der Highcharts-Bibliothek. Für Chart-Widgets muss das Admin-Tool-Skript die zu setzenden Attribute als HashMap zurückgeben (siehe Return-Statement im nachfolgenden Code-Beispiel).

Die **Tabellendarstellung** im Ressourcenpool-Dashboard basiert auf der Datatables-Bibliothek. Für Tabellen-Widgets muss das Admin-Tool-Skript die zu setzenden Attribute daher als HashMap zurückgeben.

> ⚠ Bitte beachten Sie, dass sehr komplexe Skripte die Systemleistung beeinträchtigen können!

Das folgende Beispiel zeigt das Skript `ResourceDashboardOverview1.groovy`.

```
import com.consol.cmas.common.security.permission.*;
import com.consol.cmas.common.model.resource.meta.*;
import com.consol.cmas.common.model.resource.*;

def dashboard = [:];
def categories = [];
def localeEN = [];
def localeDE = [];
def series = [];

// non empty resource groups with readable, active and internal resource types
resourceGroupService.all.each { resourceGroup ->
   def resourceTypes = [];
```

```
   if (resourceGroup.enabled) {
      resourceGroup.resourceTypes.each { name, resourceType ->
         if (resourceType.enabled) {
            if (resourceType.accessMode != ResourceAccessMode.ON_THE_FLY) {
               if (authorizationServiceImpl.isAccessGranted(resourceType,
                [ResourceTypePermissionType.READ] as Set)) {
                  resourceTypes.add(resourceType);
               }
            }
         }
      }
   };

   if (resourceTypes) {
      Collections.sort(resourceTypes);
      dashboard.put(resourceGroup, resourceTypes);
      categories.add('_(\'' + resourceGroup.name + '\')');
   }
};

// add localization
dashboard.each { resourceGroup, resourceTypes ->
   def resourceGroupEN = localizationService.getLocalizedProperty
    (ResourceGroup.class, "name", resourceGroup.getId(), Locale.ENGLISH);
   def resourceGroupDE = localizationService.getLocalizedProperty
    (ResourceGroup.class, "name", resourceGroup.getId(), Locale.GERMAN);
   if (resourceGroupEN) {
      localeEN.add("${resourceGroup.name}:${resourceGroupEN}" as String);
   }
   if (resourceGroupDE) {
      localeDE.add("${resourceGroup.name}:${resourceGroupDE}" as String);
   }
   resourceTypes.each { resourceType ->
      def resourceTypeEN = localizationService.getLocalizedProperty
       (ResourceType.class, "name", resourceType.getId(), Locale.ENGLISH);
      def resourceTypeDE = localizationService.getLocalizedProperty
       (ResourceType.class, "name", resourceType.getId(), Locale.GERMAN);
      if (resourceTypeEN) {
         localeEN.add("${resourceType.name}:${resourceTypeEN}" as String);
      }
      if (resourceTypeDE) {
         localeDE.add("${resourceType.name}:${resourceTypeDE}" as String);
      }
   };
};

// prepare data
Map<Long, Long> counters =
 resourceService.getResourceTypesIdsToActiveResourcesCountersMapping();

dashboard.eachWithIndex { resourceGroup, resourceTypes, index ->
   resourceTypes.each { resourceType ->
      def count = counters.get(resourceType.id);
```

```
      if (count != null && count > 0) {
         def data = [null] * categories.size;
         data[index] = count;
         series.add("{ name: _('${resourceType.name}'), data:[${data.join(',')}] }"
          as String);
      }
   };
};

return [
   visible: "true",
   chart: "{ type: 'column' }",
   title: "{ text: 'Overview' }",
   legend: "{ layout: 'vertical', align: 'right', verticalAlign: 'top', floating:
    true, maxHeight: 200, backgroundColor: 'white', borderColor: '#CCC',
    borderWidth: 1, shadow: false, navigation: { animation: true } }",
   xAxis: "{ categories: [${categories.join(',')}] }" as String,
   yAxis: "{ allowDecimals: false, min: 0, title: { text: 'Resources' },
    stackLabels: { enabled: true, style: { fontWeight: 'bold', color: 'gray' } }
    }",
   tooltip: "{ headerFormat: '<b>{point.key}: </b>', pointFormat: '
    {point.stackTotal}<br><span style=\"color:{series.color}\">?</span>
    {series.name}: {point.y}'",
   plotOptions: "{ column: { stacking: 'normal', dataLabels: { enabled: true,
    zIndex: 5, color: 'white', style: { textShadow: '0 0 3px black' } } } }",
   series: "[ ${series.join(',')} ]" as String,
   localization: "{ de: {${localeDE.join(',')}}, en: {${localeEN.join(',')}}}" as
    String
];
```

Code-Beispiel 112: *ResourceDashboardOverview1.groovy*

Das folgende Chart wird mit dem obigen Skript definiert.

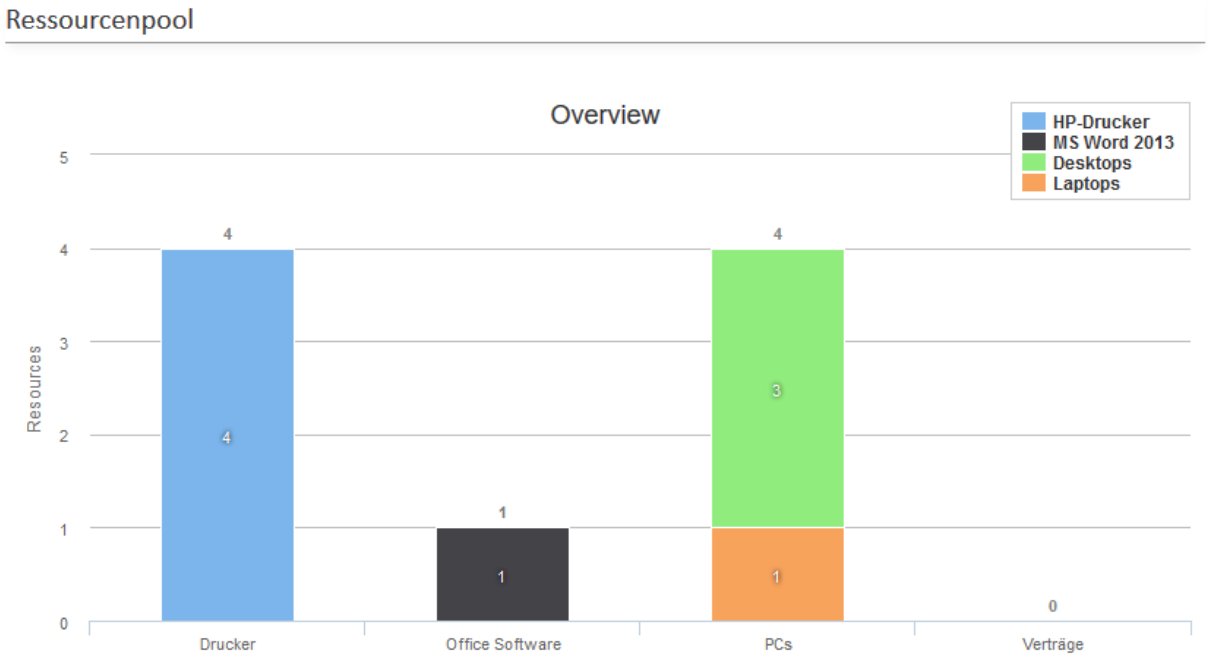


Abbildung 668: *Beispiel für ein Chart-Widget*

Konfigurationsattribute für Widgets

Es gibt die gleichen Konfigurationsattribute wie für das Web Client Dashboard. Siehe Abschnitt Konfigurationsattribute für Widgets.

G.2 CM/Doc

CM/Doc ist ein ConSol CM-Add-on, mit dem Sie Dokumentvorlagen erstellen können. Diese Vorlagen können dann verwendet werden, um direkt aus dem Geschäftsprozess Dokumente zu erstellen. CM/Doc unterstützt Microsoft Word-Dokumente und (in CM-Versionen 6.10.1 und höher) OpenOffice-Dokumente.

Eine detaillierte Beschreibung dieses Add-ons finden Sie im Abschnitt CM/Doc.

G.3 Das Kundenportal: CM/Track

G.3.1 Einleitung

Das Kundenportal ***CM/Track*** ermöglicht es den Kunden, sich im ConSol CM-System anzumelden. Wie der ConSol CM Web Client ist auch CM/Track eine webbasierte Applikation, d. h. der Kunde braucht nur einen normalen Webbrowser, um auf das Portal zuzugreifen.

Technisch gesehen werden die Daten für CM/Track über die REST API (Representational State Transfer, siehe Glossar) abgerufen.

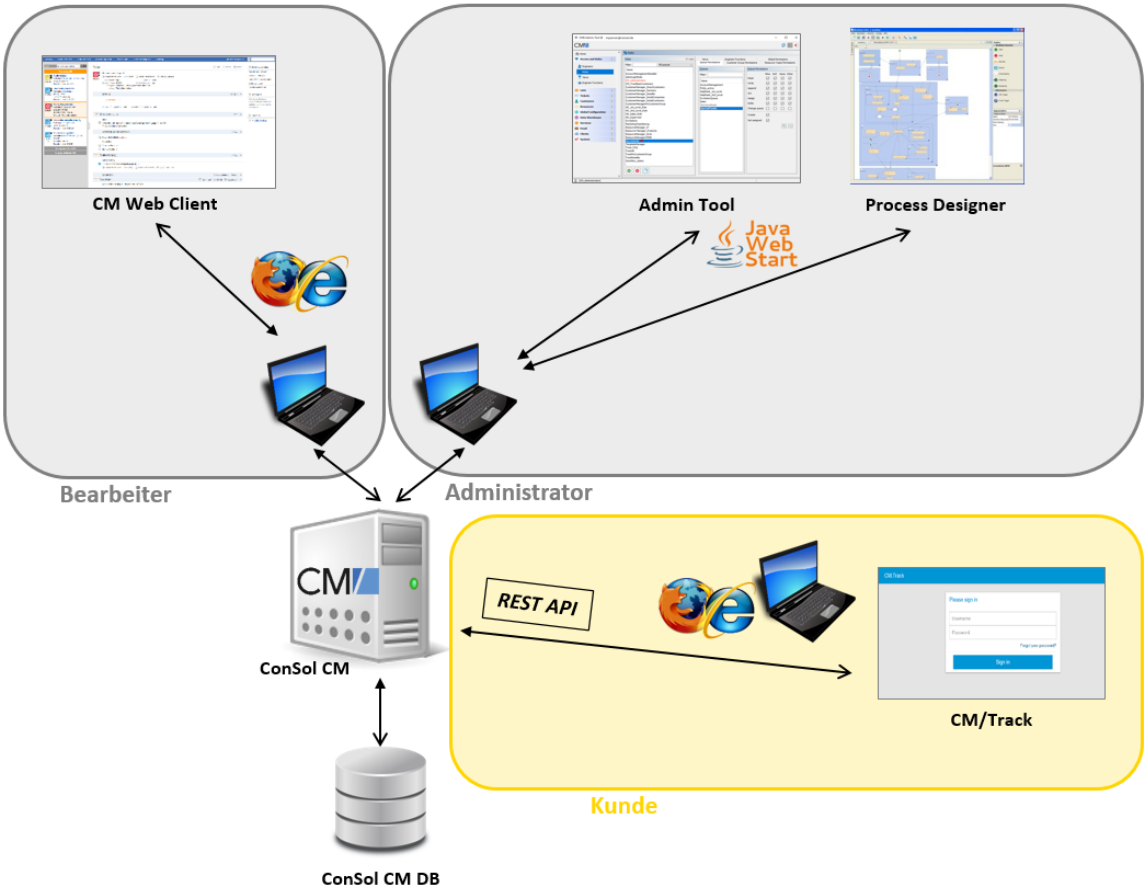


Abbildung 669: *ConSol CM-Systemarchitektur mit CM/Track*

In einer Standardumgebung kann der Kunde über CM/Track folgende Vorgänge durchführen:

- Anzeigen einer Liste mit seinen Tickets
- Anzeigen einer Liste aller Tickets seiner Firma (sofern konfiguriert)
- Hinzufügen eines Kommentars und/oder Attachments zu einem Ticket
- Öffnen/Erstellen eines neuen Tickets
- Durchsuchen der FAQ nach Lösungen

G.3.2 CM/Track-Versionen

Es gibt zwei Versionen von CM/Track:

- CM/Track V1
  Verfügbar als Portallösung in ConSol CM-Versionen 6.10.4 und niedriger.
- CM/Track V2
  Verfügbar als (neue) Portallösung in ConSol CM-Versionen 6.10.5 und höher.

Die Funktionen und die Benutzeroberfläche sind in beiden Versionen sehr ähnlich. Um Missverständnisse zu vermeiden und Ihnen einen schnellen Zugriff auf die Version Ihres CM-Systems zu bieten, finden Sie im Administratorhandbuch zwei getrennte Kapitel.



Wenn Sie CM/Track V1 haben und ein Update auf CM-Version 6.10.5 (oder höher) ausführen, können Sie die V1 weiter betreiben. Sie können natürlich auch zur V2 migrieren, was bedeutet, dass Sie die V2 in der gleichen Weise anpassen wie die V1, sofern Sie nicht die Standardinstallation von CM/Track verwenden. Beachten Sie, dass es **kein automatisches Update** von V1 auf V2 gibt, da es sich um zwei separate Web-Applikationen handelt, die auf dem Application Server installiert werden.

Wir empfehlen Ihnen die Verwendung von (und sofern erforderlich, Migration auf) V2, da diese Version erweiterte Sicherheitsfunktionen enthält und Teil von zukünftigen ConSol CM-Versionen ist.

G.3.3 Authentifizierungsmethoden für Kunden in CM/Track

G.3.3.1 Verfügbare Authentifizierungsmethoden

Es gibt drei mögliche Authentifizierungsmethoden:

- Gegen die ConSol CM-Datenbank
  Diese Methode heißt *Datenbank-Modus*, siehe Datenbank-Authentifizierung für Kunden.
- Gegen einen LDAP-Server
  Diese Methode heißt *LDAP-Modus*, siehe LDAP-Authentifizierung für Kunden in CM/Track.
- Gegen einen LDAP-Server und die ConSol CM-Datenbank
  Die Reihenfolge kann konfiguriert werden. Diese Methode heißt *Gemischter Modus*, siehe Gemischte Authentifizierungsmethode

G.3.3.2 Definieren der Authentifizierungsmethode

Die Authentifizierungsmethode wird durch die System-Property cmas-core-security, contact.authentication.method bestimmt. Eine Änderung dieser Property erfordert keinen Neustart des Servers und wird für alle Cluster-Nodes übernommen.

Die möglichen Werte (siehe auch Abschnitt System-Properties) und ihr entsprechendes Systemverhalten sind:

- **DATABASE**
  Zuerst wird die Anmeldung über die Datenbank versucht, sofern der Kunde ein Datenbankpasswort hat. d. h. wenn Login und Passwort in der ConSol CM-Datenbank gespeichert sind und damit von den ConSol CM-Bearbeitern, oder indirekt von den Kunden selber verwaltet werden, wenn diese ihr Passwort zurücksetzen, siehe Abschnitt Template zum Zurücksetzen des Passworts durch Kunden in CM/Track V2.
- **LDAP**
  Es wird eine Anmeldung über die verfügbaren LDAP-Server versucht, wenn eine LDAP-ID angegeben wurde, d. h. das Passwort ist im LDAP-Verzeichnis gespeichert und kann nicht über ConSol CM geändert werden, weder vom Kunden noch von einem Bearbeiter.
- **LDAP,DATABASE**
  Beim ersten Authentifizierungsversuch werden die verfügbaren LDAP-Server verwendet, sofern eine LDAP-ID angegeben ist. Sollte dieser fehlschlagen, wird eine Anmeldung über die Datenbank versucht, sofern der Kunde ein Datenbankpasswort hat.
- **DATABASE,LDAP**
  Zuerst wird die Anmeldung über die Datenbank versucht, sofern der Kunde ein Datenbankpasswort hat. Wenn diese fehlschlägt, wird die Authentifizierung über einen verfügbaren LDAP-Server versucht, sofern eine LDAP-ID angegeben wurde.

Groß- und Kleinschreibung, Kommas und Leerzeichen werden bei den Werten ignoriert.

G.3.3.3 Datenbank-Authentifizierung für Kunden

Die Datenbank-Authentifzierung durch das Setzen der System-Property cmas-core-security, contact.authentication.method auf „DATABASE“ (Standardwert) aktiviert.

Es gibt zwei Schritte, die durchgeführt werden müssen, um die Datenbank-Authentifizierung für Kunden mit CM/Track einzurichten:

- Erstellen Sie Kundenfelder für den Benutzernamen (Login) und das Passwort im Admin Tool (siehe Definieren der Kundenfelder für das CM/Track-Login und -Passwort)
- Geben Sie den Benutzernamen und das Passwort für die jeweiligen Kunden im Web Client ein (siehe Gewähren von Zugang zu CM/Track für Kunden)

Wenn die Datenbank-Authentifizierung verwendet wird, können Sie es Ihren Kunden ermöglichen, ihre eigenen Passwörter zu ändern, siehe Konfigurieren von CM/Track für das Zurücksetzen des Passworts durch den Kunden.

Definieren der Kundenfelder für das CM/Track-Login und -Passwort

Die Felder für das Login und Passwort für einen Kunden sind normale Kundenfelder auf der Kontaktstufe. Eine Einführung in die Verwaltung von Kundenfeldern und die GUI-Konfiguration für Kundendaten finden Sie im Abschnitt Einrichten des Kundendatenmodells.

Editieren Sie die Felder, die Kundendaten enthalten (wenn es zwei Stufen gibt: **nicht** auf der Firmenstufe sondern auf der Kontaktstufe!) wie im folgenden Beispiel gezeigt. Sie gelangen auf die folgende Seite, indem Sie in der Navigationsgruppe *Kunden* das Navigationselement *Datenmodelle* öffnen.

- Erstellen Sie ein Feld für das **Login** mit der Annotation `username` = „true“.

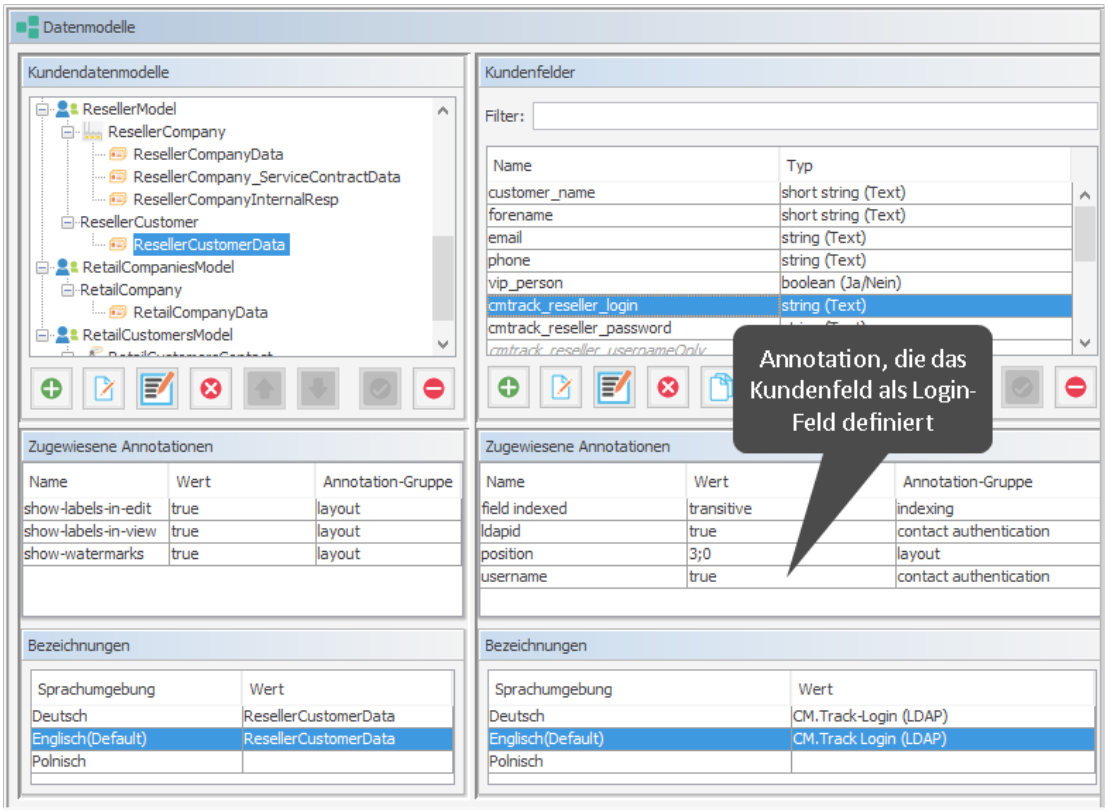


Abbildung 670: *ConSol CM Admin Tool - Kunden, Datenmodelle, CM/Track: Annotation „username“ für Login*

> ⚠ Die Zuweisung der Annotation `username` zu einem Kundenfeld ist nur möglich, wenn diese Annotation noch nicht zugewiesen wurde. Andernfalls ist sie nicht erlaubt. Beim Zuweisen der Annotation muss zuerst eine Warnung bestätigt werden, bevor der Vorgang ausgeführt wird, da dieser länger dauern kann. Das Entfernen der Annotation muss ebenfalls bestätigt werden, da es nicht rückgängig gemacht werden kann: Beim Entfernen werden die Werte für die Benutzernamen unwiderruflich aus dem internen Speicher gelöscht.

- Erstellen Sie ein Feld für das Passwort mit der Annotation `password` = „true“. Die Annotation `text-type` = „password“ sorgt dafür, dass nur Sternchen/Punkte im Web Client angezeigt werden und das Passwort nicht im Klartext angezeigt wird.

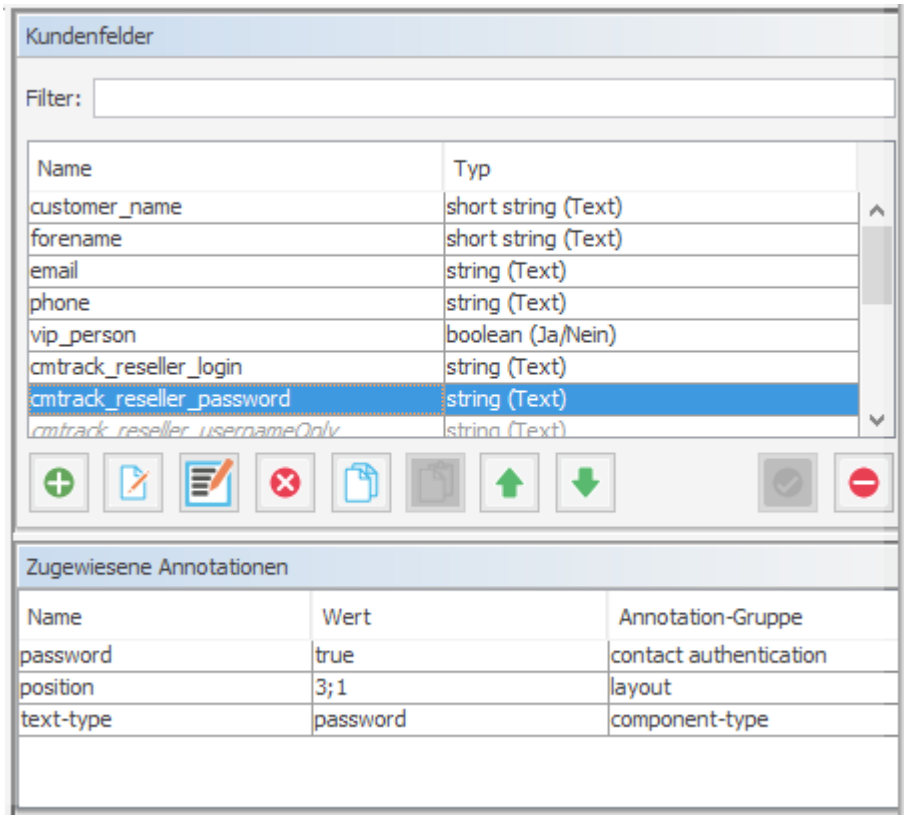


Abbildung 671: *ConSol CM Admin Tool - Kunden, Datenmodelle, CM/Track: Annotation für Passwort*

> ⚠ Die Annotation `password` muss vor der Zuweisung bestätigt werden.
>
> Bei einer Aktualisierung von einer CM-Version unter 6.11 auf 6.11 und höher: Wenn diese Annotation gesetzt wird, liest das System die Passwörter im Klartext aus den ursprünglichen Feldwerten, verschlüsselt sie und speichert sie verschlüsselt im internen Speicher. Die ursprünglichen Werte werden gelöscht, sodass die Werte im Klartext nicht mehr wiederhergestellt werden können.
>
> Beim Versuch, die Annotation `password` zu entfernen, muss der Vorgang ebenfalls bestätigt werden, da die verschlüsselten Passwörter aus dem internen Speicher gelöscht werden. Nach dem Entfernen der Annotation sind die Passwort-Informationen nicht mehr vorhanden und können nicht mehr wiederhergestellt werden.

Wenn ein Szenario aus einer CM-Version unter 6.11 in ein System mit CM 6.11 oder höher importiert wird, wird die Transformation der Benutzernamen und Passwörter automatisch ausgeführt. Dies ist detailliert in Transformation von Benutzernamen- und Passwortfeldern bei einem Import nach CM 6.11 beschrieben.

Gewähren von Zugang zu CM/Track für Kunden

Der Bearbeiter, der mit dem Web Client arbeitet, kann jedem Kunden, der Zugang zum Portal CM/Track haben soll, einen Benutzernamen, ein initiales Passwort und ein CM/Track-Benutzerprofil zuweisen. Der Benutzername muss eindeutig sein. Dies wird vom System überprüft. Sie können einen Benutzernamen nicht ein zweites Mal eingeben, wenn dieser bereits für einen anderen Kunden verwendet wird. Das Passwort wird verschlüsselt in der CM-Datenbank gespeichert. Das bedeutet, dass ein Bearbeiter ein neues Passwort setzen kann, z. B. wenn ein Kunde anruft und darum bittet. Es ist allerdings nie möglich, das Passwort aus dem System auszulesen.

> Als Administrator können Sie festlegen, ob bei den CM/Track-Benutzernamen zwischen Groß- und Kleinbuchstaben unterschieden werden soll. Verwenden Sie dazu die CM-System-Property cmas-core-security, policy.track.username.case.sensitive. Dabei handelt es sich um eine Boolean-Variable. Wenn sie auf „true“ gesetzt ist, wird bei den CM/Track-Benutzernamen zwischen Groß- und Kleinschreibung unterschieden. Stellen Sie sicher, dass die verwendete Datenbank-Collation Strings mit Groß- und Kleinschreibung unterstützt.

Das folgende Beispiel zeigt die Kundendaten eines Beispielkontaktes im ConSol CM Web Client. Sie gelangen auf diese Seite, indem Sie einen Kontaktdatensatz im Bearbeitungsmodus öffnen.

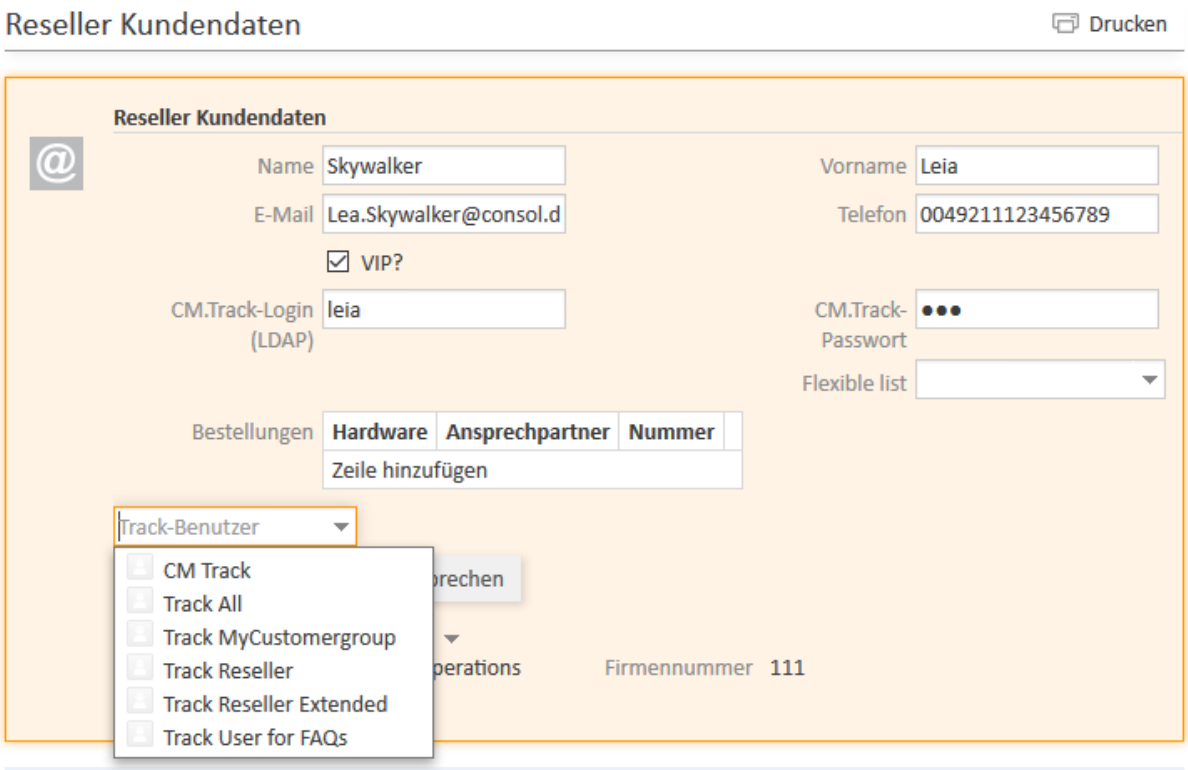


Abbildung 672: *ConSol CM Web Client, Kontaktseite - CM/Track-Benutzerdaten*

Konfigurieren von CM/Track für das Zurücksetzen des Passworts durch den Kunden

CM/Track kann so konfiguriert werden, dass den Kunden ein Link angezeigt wird, mit dem sie ihr Passwort zurücksetzen können. Diese Funktion basiert auf dem Template `track-password-reset-template`. Eine detaillierte Erklärung dazu finden Sie im Abschnitt Template zum Zurücksetzen des Passworts durch Kunden in CM/Track V1 / Template zum Zurücksetzen des Passworts durch Kunden in CM/Track V2.Das Zurücksetzen des Passworts in CM/Track ist nur möglich, wenn der Datenbank-Modus verwendet wird. Bei LDAP-Authentifizierung ist dies nicht möglich. Eine Beschreibung aller verfügbaren Authentifizierungsmethoden finden Sie im Abschnitt Authentifizierungsmethoden für Kunden in CM/Track.

Beachten Sie, dass die From-Adresse der E-Mail, die an einen Kunden gesendet wird, der ein neues Passwort angefordert hat, mit der CM-System-Property cmas-core-security, password.reset.mail.from gesetzt werden kann.

G.3.3.4 LDAP-Authentifizierung für Kunden in CM/Track

Einführung in die LDAP-Authentifizierung in ConSol CM

ConSol CM bietet LDAP-Authentifizierung für CM/Track als Standardfunktion an, d. h. die Passwörter der ConSol CM-Kunden werden nicht in der ConSol CM-Datenbank verwaltet, sondern von einem LDAP-Server (wie z. B. ein ***Microsoft Active Directory***-Server) abgerufen.

Wenn sich die Kunden in CM/Track anmelden möchten, geben sie ihren Benutzernamen und ihr Passwort ein und drücken die Eingabetaste. Im Hintergrund sendet der ConSol CM-Server eine Anfrage mit dem Benutzernamen und dem Passwort an den LDAP-Server und dieser überprüft, ob die Anmeldeinformationen gültig sind.

Wenn die Anmeldeinformationen gültig sind, wird die Genehmigung zurück an den ConSol CM-Server gesendet und der Kunde wird in CM/Track angemeldet.

> Beachten Sie, dass die LDAP-Verbindung nur zur Authentifizierung des Kunden (Bestätigung seiner Identität) verwendet wird. Die Autorisierung (d. h. die Zuweisung von Zugangsberechtigungen im System) erfolgt über Zuweisung eines CM/Track-Benutzerprofils im Web Client. Die CM/Track-Benutzerprofile werden in der Bearbeiter- und Rollenverwaltung im Admin Tool verwaltet.

Der CM-Authentifizierungsprozess mittels LDAP ist in der folgenden Abbildung erklärt.

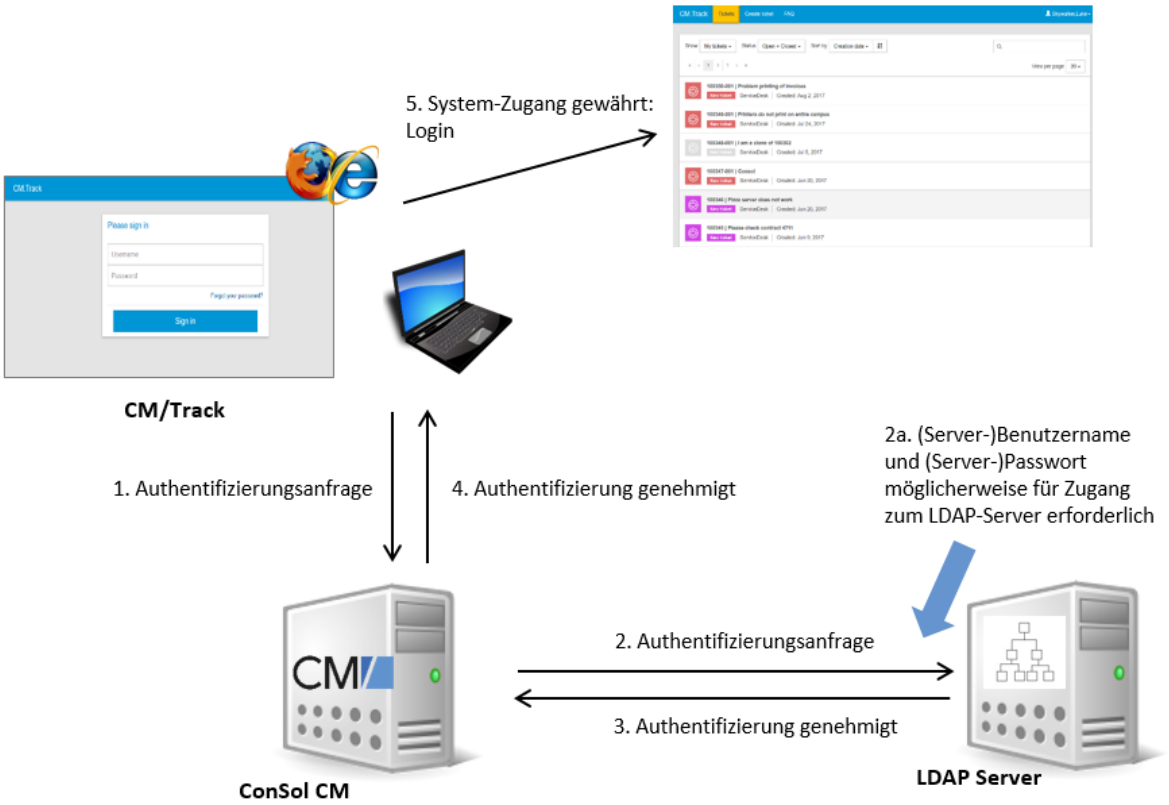


Abbildung 673: *ConSol CM - LDAP-Authentifizierungsprozess für CM/Track*

Konfigurieren von LDAP-Authentifizierung

Die LDAP-Authentifizierung wird durch Setzen der System-Property cmas-core-security, contact.authentication.method auf „LDAP“ aktiviert.

Als nächstes müssen Sie die erforderlichen Werte in den System-Properties setzen (Navigationsgruppe *System*, Navigationselement *System-Properties*, siehe System-Properties).

Die LDAP-Server können mit folgenden System-Properties aus dem Modul `cmas-core-security` definiert werden.

{name} ist ein String, den Sie frei wählen können, um die LDAP-Server zu unterscheiden. Er muss immer angegeben werden, auch wenn nur ein LDAP-Server konfiguriert wird. Sie sollten einen einfachen String für {name} verwenden, der keine Schlüsselwörter wie *internal* oder *external* und keine Sonderzeichen enthält.

- contact.authentication.method
  LDAP
- ldap.contact.{name}.providerurl
  Der Wert der Property ist die Adresse des LDAP-Servers in der Form *ldap[s]://host:port*.

- ldap.contact.{name}.userdn
  Der Wert ist der Benutzer-DN, der verwendet wird, um den Kontakt-DN über die LDAP-ID nachzuschlagen. Wenn der Wert nicht gesetzt ist, wird ein anonymes Konto verwendet.
- ldap.contact.{name}.password
  Die Property enthält das Passwort, mit dem der Kontakt-DN über die LDAP-ID nachgeschlagen wird. Wenn der Wert nicht gesetzt ist, wird ein anonymes Konto verwendet.
- ldap.contact.{name}.basedn
  Dies ist der Stammpfad, in dem der Kontakt-DN über die LDAP-ID gesucht wird, z. B. *ou=accounts,dc=mycompany,dc=de*.
- ldap.contact.{name}.searchattr
  Der Wert dieser Property steht für das Attribut, mit dem der Kontakt-DN über die LDAP-ID gesucht wird, z. B. *uid*.

Diese System-Properties sind möglicherweise initial nicht in Ihrem CM-System vorhanden. Fügen Sie sie einfach manuell hinzu. Nach einer Änderung einer der obigen System-Properties ist kein Neustart des Servers erforderlich. Die Änderung wird für alle Cluster-Nodes übernommen. Der Platzhalter {name} ermöglicht die Definition von Konfigurationen für mehrere unterschiedliche LDAP-Server.

- ldap.initialcontextfactory
  Dies ist eine vordefinierte globale Property. Wenn sie nicht gesetzt ist, wird *com.sun.jndi.ldap.LdapCtxFactory* als Wert verwendet.

Authentifizierungsversuche gegen LDAP-Server finden bis zum ersten erfolgreichen Versuch statt, wobei die Reihenfolge der Server durch die Werte von {name} bestimmt wird (aufsteigende alphabetische Reihenfolge der Werte).

Gemischte Authentifizierungsmethode

Setzen Sie die System-Property cmas-core-security, contact.authentication.method entsprechend der gewünschten Reihenfolge der Authentifizierungsinstanzen:

- **LDAP,DATABASE**
  Beim ersten Authentifizierungsversuch werden die verfügbaren LDAP-Server verwendet, sofern eine LDAP-ID angegeben ist. Sollte dieser fehlschlagen, wird eine Anmeldung über die Datenbank versucht, sofern der Kunde ein Datenbankpasswort hat
- **DATABASE,LDAP**
  Zuerst wird die Anmeldung über die Datenbank versucht, sofern der Kunde ein Datenbankpasswort hat. Wenn diese fehlschlägt, wird die Authentifizierung über einen verfügbaren LDAP-Server versucht, sofern eine LDAP-ID angegeben wurde.

Das CM-System kontaktiert zuerst die Instanz, die an erster Stelle steht, und danach die zweite Instanz. Wenn die Authentifizierungsmethode für Kontakte zum Beispiel auf „LDAP,DATABASE“ gesetzt ist, und der Kunde (Kontakt) ein Passwort verwendet, das nur in der Datenbank gültig ist, ist die Anmeldung erfolgreich.

Im `server.log` wird dann folgende Meldung angezeigt:

```
LDAP login failed: [LDAP: error code 49 - Invalid Credentials]; nested exception is
 javax.naming.AuthenticationException: [LDAP: error code 49 - Invalid Credentials]
```

Logging aller LDAP-Anmeldeversuche in CM/Track V2

Alle aufgetretenen LDAP-Fehler werden ohne Stack Trace von Loggern mit folgendem Präfix protokolliert:

- `com.consol.cmas.core.security.contact`

Der Stack Trace von LDAP-Fehlern wird nicht protokolliert, da die Log-Dateien ansonsten mit Details zu fehlgeschlagenen Anmeldeversuchen auf dem ersten LDAP-Server vollgeschrieben würden, obwohl die darauf folgende Anmeldung beim zweiten LDAP-Server erfolgreich war.

Verwendung von LDAPS (LDAP über SSL)

Einleitung

Standardmäßig werden die Daten im Klartext übertragen, wenn ein LDAP-Client auf einen LDAP-Server zugreift. Wenn Sie möchten, dass der Benutzername und das Passwort verschlüsselt an den LDAP-Server übertragen werden, müssen Sie die LDAP-Authentifizierung mit LDAPS aufsetzen.

Vorbereitungen

Sie müssen den Rechner des CM-Servers (Java) so konfigurieren, dass er Zertifikate verwenden kann. Eine Methode dafür ist im folgenden Abschnitt für einen Linux-Rechner beschrieben.

1. Rufen Sie das Zertifikat ab:
   `openssl s_client -connect dc2.mydomain.com:ldaps`
2. Die Antwort enthält einen Abschnitt, der mit "---BEGIN CERTIFICATE " beginnt und mit "END CERTIFICATE ---" endet.
   Kopieren Sie diesen Abschnitt in eine Datei, z. B. /tmp/certificate2_dc2_mydomain_com.txt
3. Importieren Sie das Zertifikat in den Truststore des Rechners, z. B. /home/mydirectory/mytruststore
   `$JAVA_HOME/bin/keytool -import -alias <arbitrary> -trustcacerts -keystore /home/mydirectory/mytruststore -file`/tmp/certificate2_dc2_mydomain_com.txt
   Sie müssen ein Passwort eingeben (setzen).
4. Fügen Sie den Truststore in die ConSol CM-Konfigurationsdatei unter JAVA_OPTS ein:
   `-Djavax.net.ssl.trustStore=/home/mydirectory/mytruststore -Djavax.net.ssl.trustStorePassword=<see above>`

LDAPS-Konfiguration mit dem ConSol CM Admin Tool (System-Properties)

Konfigurieren Sie den ConSol CM-Server so, wie im folgenden Beispiel gezeigt:

- cmas-core-security, ldap.authentication = simple
- cmas-core-security, ldap.contact.name.basedn = OU=myOU,DC=myDC
- cmas-core-security, ldap.initialcontextfactory = com.sun.jndi.ldap.LdapCtxFactory

- cmas-core-security, ldap.contact.name.password = myLDAPpw
- cmas-core-security, ldap.contact.name.searchattr = sAMAccountName
- cmas-core-security, ldap.contact.name.userdn = myLDAP_UserDN

Je nach Konfiguration des LDAP-Servers, verwenden Sie einen der folgenden Werte für die Server-URL:

- **Standard LDAPs port**
  cmas-core-security, ldap.contact.name.providerurl = ldaps://dc2.mydomain.com:636
- **LDAPs port Global Catalogue**
  cmas-core-security, ldap.contact.name.providerurl = ldaps://dc2.mydomain.com:3269

Einrichten von Kundenkonten für LDAP

Sie müssen zwei Schritte ausführen, um LDAP-Authentifizierung für Kunden in CM/Track einzurichten:

- Setzen Sie die erforderliche Annotation für das Kundenfeld, in dem die LDAP-ID im Admin Tool gespeichert werden soll.
- Geben Sie die LDAP-IDs für die jeweiligen Kunden im Web Client ein.

Wenn der LDAP-Modus genutzt wird, muss das Kundenfeld, das für den CM/Track-Benutzernamen (Login) verwendet wird, zwei Annnotationen haben:

- `username` = true
- `ldapid` = true

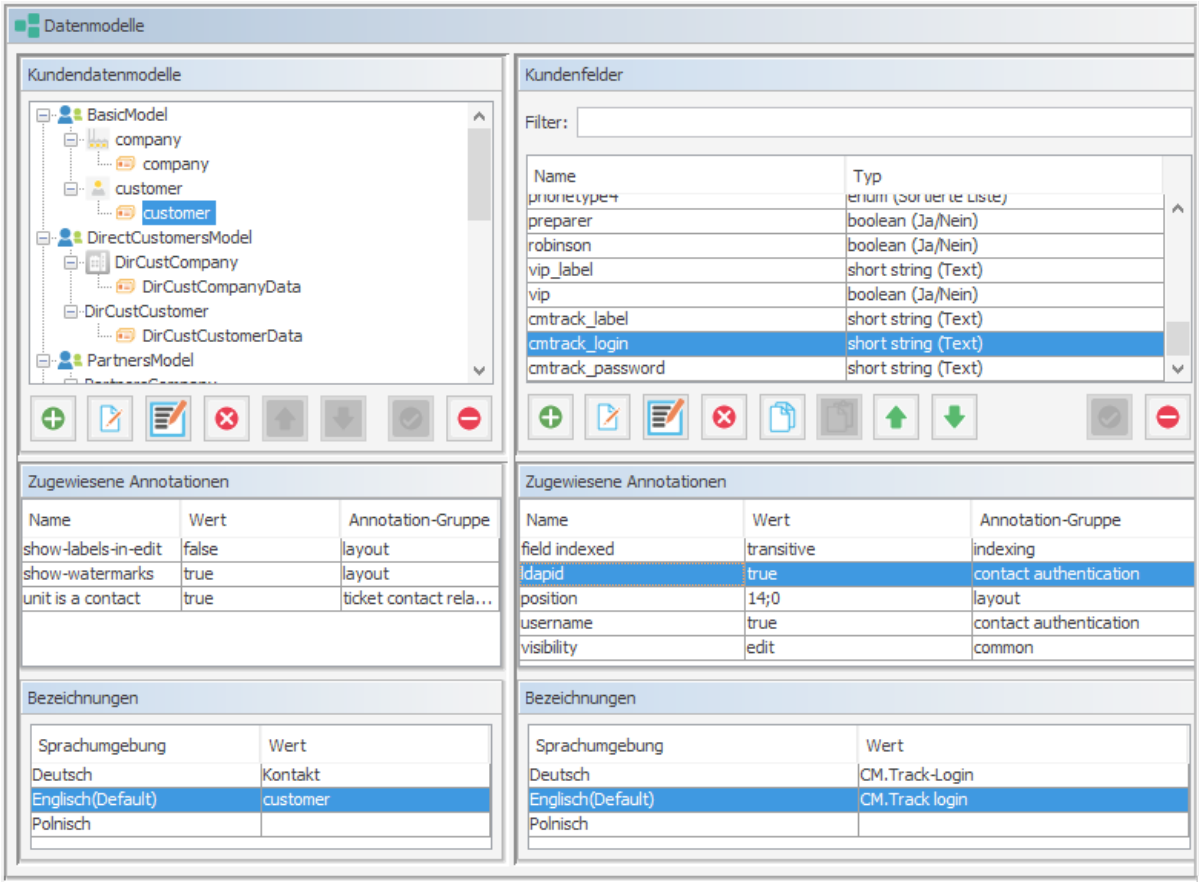


Abbildung 674: *ConSol CM Admin Tool - Kunden, Datenmodelle: Kundenfeld für die LDAP-Authentifizierung von CM/Track-Benutzern*

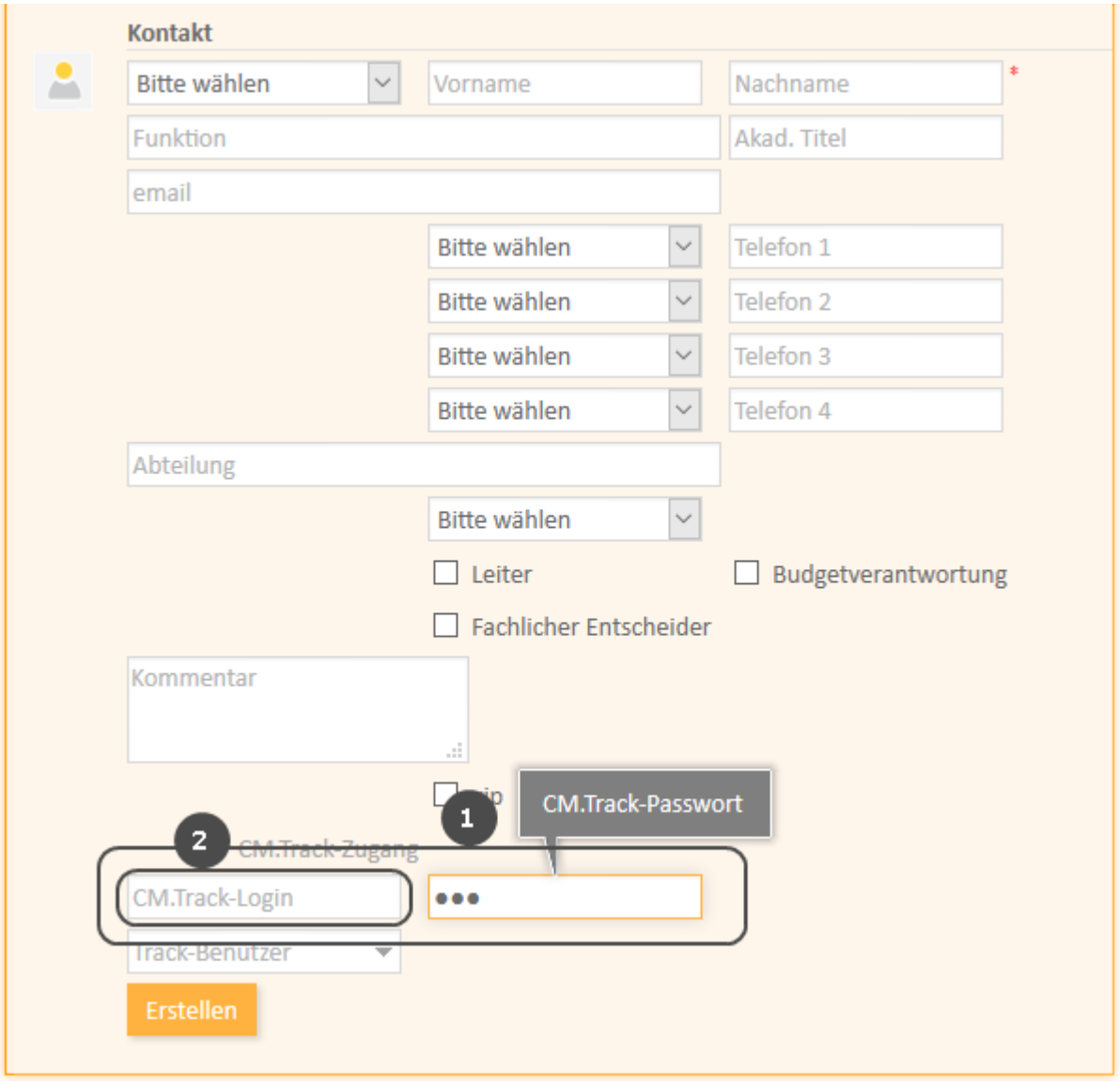


Abbildung 675: *ConSol CM Web Client - Feld für die LDAP-ID in den Kontaktdaten*

- Für Datenbankauthentifizierung im gemischten Modus verwendet (1)
- Für LDAP-Authentifizierung verwendet (2)

G.3.4 CM/Track V1

In den folgenden Abschnitten werden Themen im Zusammenhang mit CM/Track V1 behandelt.

- Authentifizierungsmethoden für Kunden in CM/Track (sowohl V1 als auch V2):
  Siehe Abschnitt Authentifizierungsmethoden für Kunden in CM/Track.
- Allgemeiner Systemzugang zu CM/Track für Kunden:
  Siehe Abschnitt CM/Track V1: Systemzugang für CM/Track-Benutzer (Kunden).
- Verwendung des Portals für FAQs:
  Siehe Abschnitt CM/Track V1: FAQs in CM/Track

G.3.4.1 CM/Track V1: Systemzugang für CM/Track-Benutzer (Kunden)

Im folgenden Abschnitt finden Sie detaillierte Informationen darüber, wie Sie Ihr ConSol CM-System konfigurieren, um Ihren Kunden Zugang zu CM/Track (dem Portal von ConSol CM) zu gewähren.

> CM/Track V1 ist ein Add-on von ConSol CM, das separat erworben werden muss.
>
> Bedenken Sie, dass für jeden CM/Track-Benutzer (d. h. für jedes Benutzerprofil) eine eigene ConSol CM-Lizenz benötigt wird. Da sich mehrere Kunden mit einem Benutzerprofil in CM/Track anmelden können, brauchen Sie nicht für jeden Kunden eine ConSol CM-Lizenz. Details dazu finden Sie im Abschnitt Lizenzverwaltung.

Voraussetzung

CM/Track V1 im Standardlieferumfang der ConSol CM-Versionen 6.10.4 und niedriger enthalten, das heißt, es müssen keine neuen Dateien installiert werden. Der Standardfunktionsumfang von CM/Track beinhaltet grundlegende Funktionen (z. B. Anzeigen der Ticketliste, Erstellen eines neuen Tickets, Anzeigen der Ticketdetails) und die Seiten haben das CM-Standardlayout. Um CM/Track als leistungsstarkes Portal für den Zugang Ihrer Kunden zum System einsetzen zu können, sollten Sie das Layout an das Corporate Design Ihres Unternehmens anpassen. Dieser Vorgang heißt *Skinning*. Die Formulare und Listen, die dem Kunden angezeigt werden, können verändert und/oder erweitert werden. Wenden Sie sich an unser Consulting-Team oder Ihren Account Manager, wenn Sie CM/Track optimal auf Ihr Unternehmen zuschneiden möchten.

Technischer Hintergrund von CM/Track

Das Portal CM/Track basiert auf der ***REST API*** von ConSol CM. Details dazu finden Sie im Dokument *ConSol CM REST API Documentation*.

Allgemeines Prinzip des Systemzugangs über CM/Track

Ein Kunde, der über das Portal CM/Track Zugang zu Ihrem ConSol CM-System haben möchte oder erhalten soll, braucht ein Login und ein Passwort. Diese können entweder vom Bearbeiter, der die Kundendaten im CM Web Client editiert, dort eingegeben werden, oder die Werte können automatisch in die Datenbank importiert werden.

Die Felder für das Login und das Passwort der Kunden sind Kundenfelder, die wie alle anderen Kundenfelder definiert werden und eine besondere Annotation haben. Wenn Sie mit Kundenfeldern nicht vertraut sind, lesen Sie bitte Abschnitt Verwaltung von Kundenfeldern und GUI-Design für Kundendaten.

Die Zugangsberechtigungen der Kunden werden definiert, indem Sie dem Konto des Kunden ein Benutzerprofil zuweisen. Die Benutzerprofile werden vom ConSol CM-Administrator mit dem Admin Tool verwaltet.

Definieren der Benutzerprofile / Zugangsberechtigungen für CM/Track

Im ersten Schritt müssen Sie die Benutzerprofile definieren, d. h. die Profile mit den Zugangsberechtigungen für CM/Track. Ein CM/Track-Benutzerprofil wird wie ein normaler Bearbeiter definiert (siehe Abschnitt Bearbeiterverwaltung), aber als *Track* markiert.

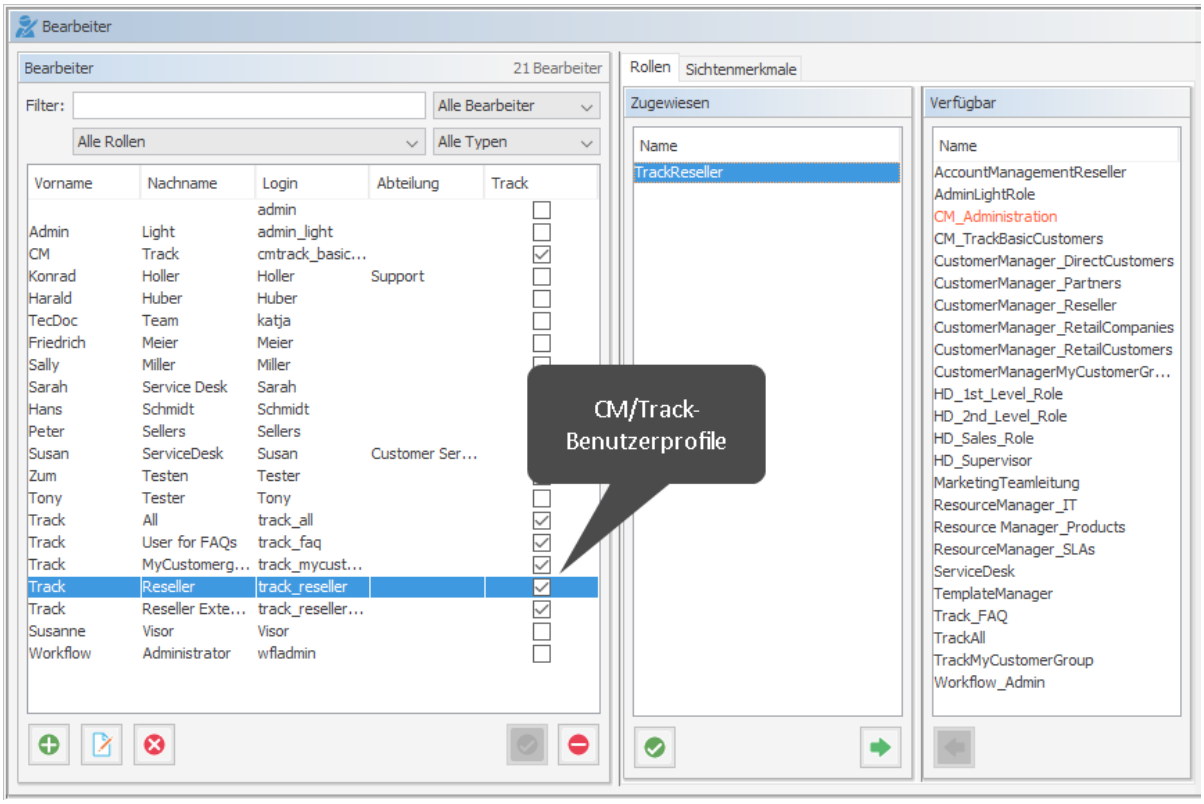


Abbildung 676: *ConSol CM Admin Tool - Zugriff und Rollen, Bearbeiter: Name des Benutzerprofils für CM/Track*

Dem Benutzerprofil werden eine oder mehrere Rollen zugewiesen, um die Zugangsberechtigungen zu Queues und Kundengruppen zu definieren. Sie können zum Beispiel ein Benutzerprofil (Bearbeiter) *cmtrack_basiccustomers* erstellen, das die Rolle *TrackAll* hat. Diese Rolle hat Berechtigungen zum Lesen/Schreiben/Hinzufügen in den Queues *Helpdesk_2nd_Level* und *ServiceDesk* und Kundengruppen-Berechtigungen für drei Kundengruppen. Bitte beachten Sie Folgendes:

- Es müssen Queue- **und** Kundengruppen-Berechtigungen erteilt werden, damit die Kunden über CM/Track auf Tickets zugreifen können.
- Sie müssen passende Queue- und Kundengruppenberechtigungen zuweisen, d. h. Queues zuweisen, denen die entsprechenden Kundengruppen zugewiesen wurden (siehe Abschnitt Queue-Verwaltung).
- In den meisten Fällen (Standardfällen) reichen Leseberechtigungen für die Kundengruppen aus, da es nicht möglich ist, über das Portal Kundendaten zu editieren.

In unserem Beispiel hat die Rolle *TrackAll* Lesezugriff auf alle Kundengruppen. In Ihrem System ist es möglicherweise erforderlich, unterschiedliche CM/Track-Rollen mit Zugriff auf unterschiedliche Kundengruppen zu erstellen. Eine detaillierte Einführung in die Rollenverwaltung finden Sie im Abschnitt Rollenverwaltung.

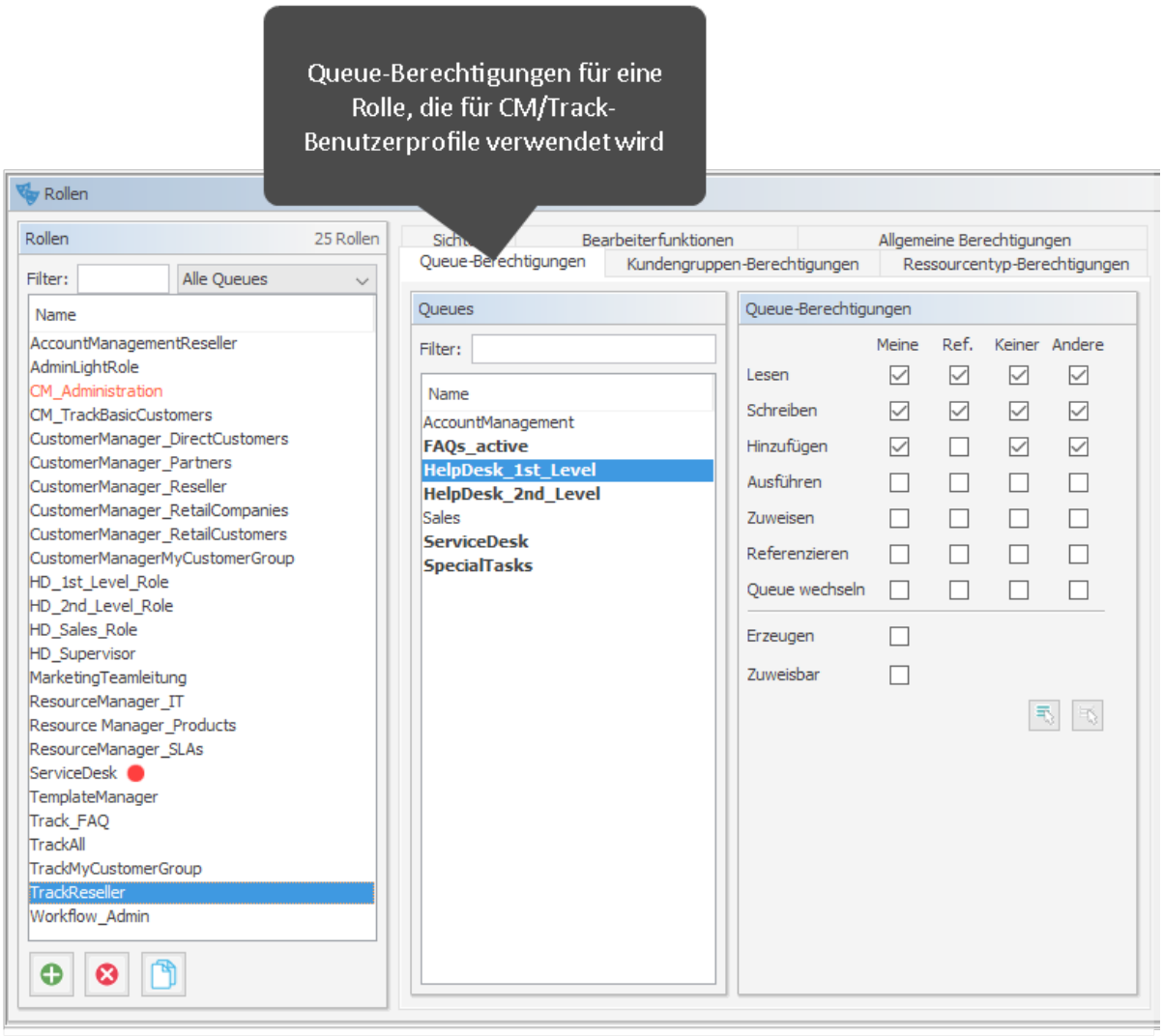


Abbildung 677: *ConSol CM Admin Tool - Zugriff und Rollen, Rollen: CM/Track-Benutzerprofil, Queue-Berechtigungen*

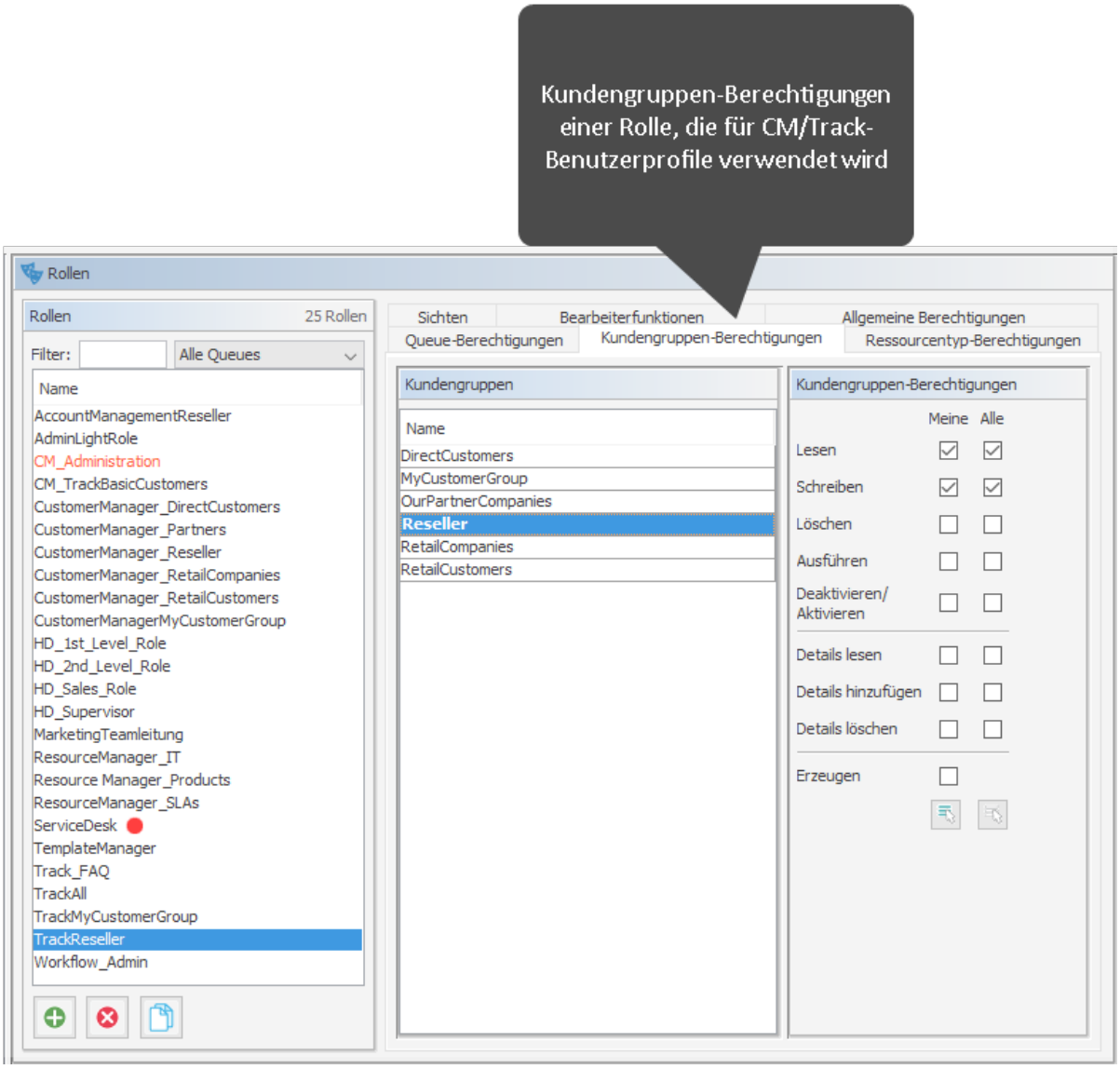


Abbildung 678: *ConSol CM Admin Tool - Zugriff und Rollen, Rollen: CM/Track-Benutzerprofil, Kundengruppen-Berechtigungen*

Auf diese Weise kann ein Kunde mit dem CM/Track-Benutzerprofil *cmtrack_basiccustomers* nur Tickets in diesen Queues sehen und dort Kommentare hinzufügen. Ein anderes Benutzerprofil könnte Zugang zu den Tickets in der Queue *Sales* und/oder zur Queue *FAQ* haben.

Definieren der Kundenfelder für das CM/Track-Login und -Passwort

Die Felder für das Login und Passwort für einen Kunden sind normale Kundenfelder auf der Kontaktstufe. Eine Einführung in die Verwaltung von Kundenfeldern und die GUI-Konfiguration für Kundendaten finden Sie im Abschnitt Einrichten des Kundendatenmodells.

Editieren Sie die Felder, die Kundendaten enthalten (wenn es zwei Stufen gibt: **nicht** auf der Firmenstufe sondern auf der Kontaktstufe!):

- Es muss ein Feld für das Login erstellt werden, Annotation `username` = „true“.

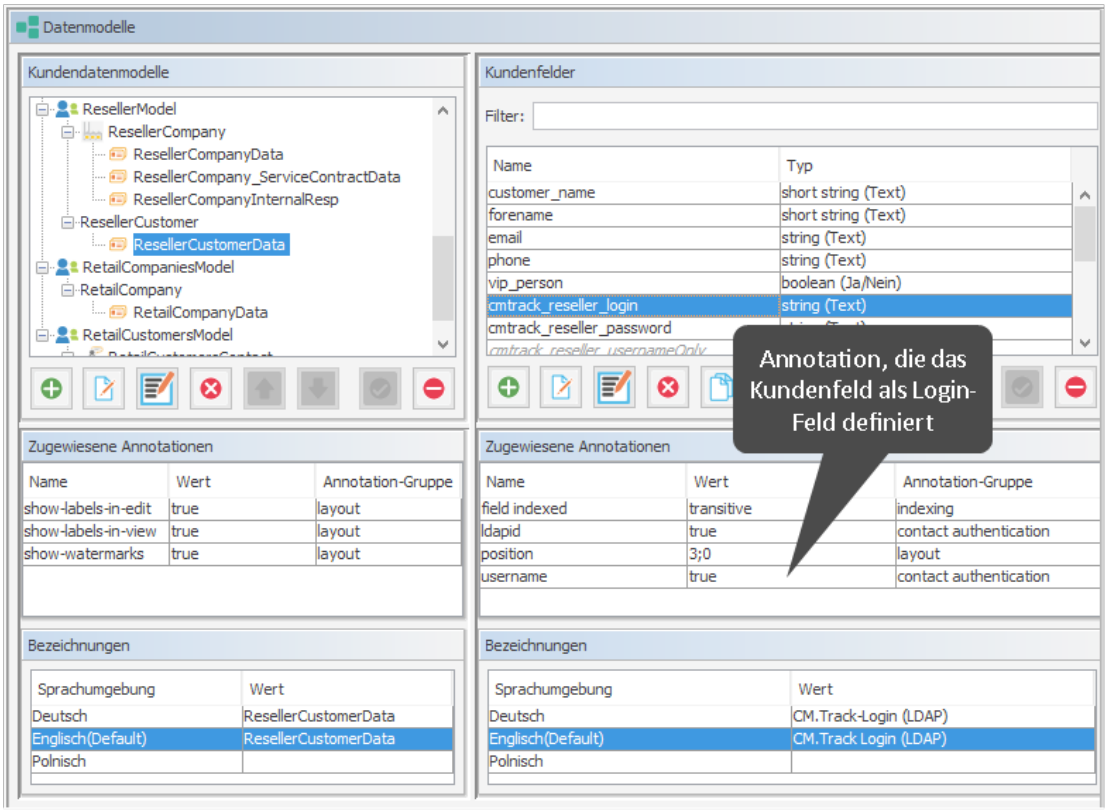


Abbildung 679: *ConSol CM Admin Tool - Kunden, Datenmodelle: Annotation für Login für CM/Track*

- Es muss ein Feld für das Passwort erstellt werden, Annotation `password` = „true“. Die Annotation `text-type` = „password“ sorgt dafür, dass nur Sternchen/Punkte im Web Client angezeigt werden und das Passwort nicht im Klartext angezeigt wird.

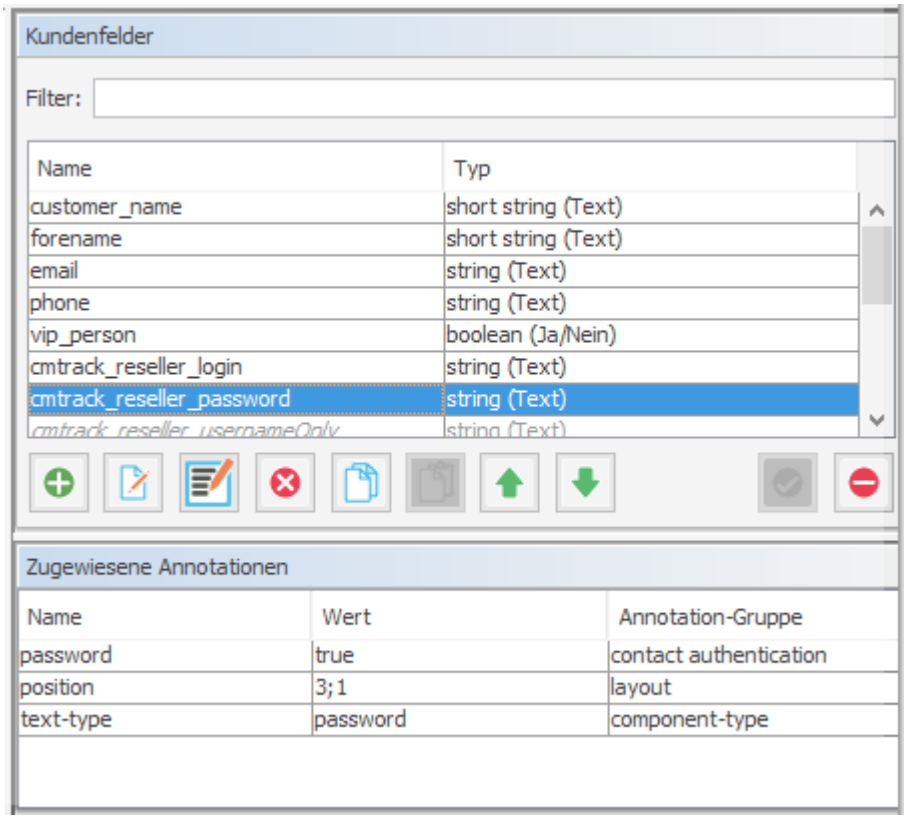


Abbildung 680: *ConSol CM Admin Tool - Kunden, Datenmodelle: Annotation für Passwort für CM/Track*

Gewähren von Zugang zu CM/Track für Kunden

Der Bearbeiter, der mit dem Web Client arbeitet, kann jedem Kunden, der Zugang zum Portal CM/Track haben soll, einen Benutzernamen, ein initiales Passwort und ein CM/Track-Benutzerprofil zuweisen.

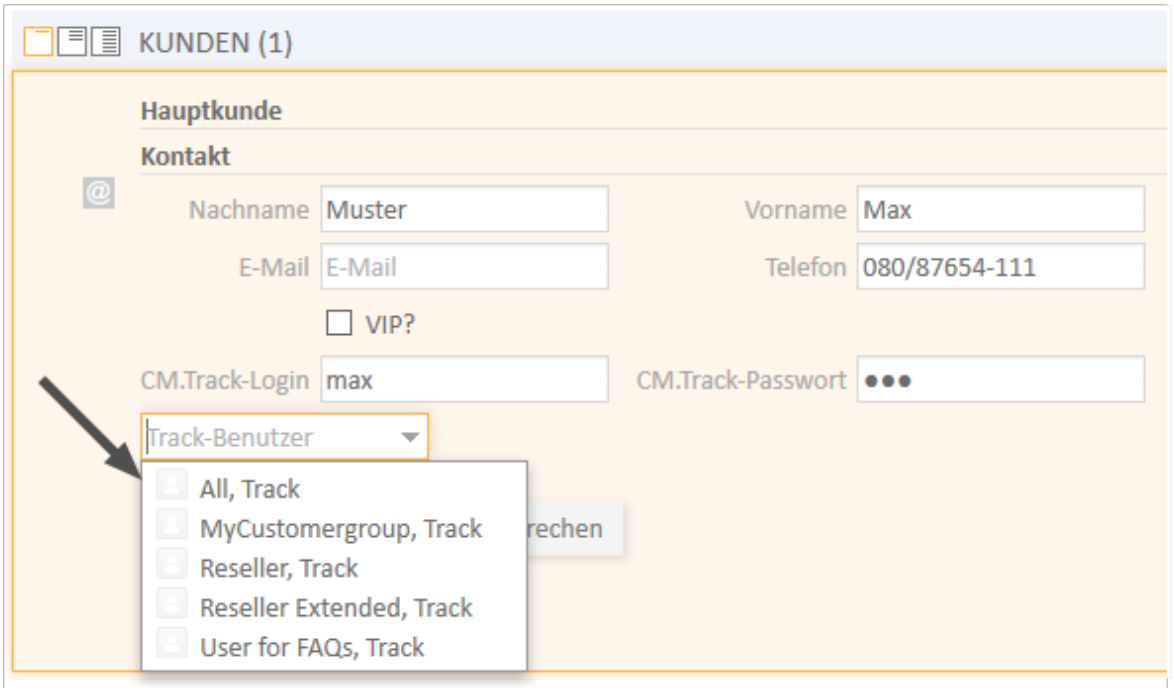


Abbildung 681: *ConSol CM Web Client - CM/Track-Benutzer*

Anmeldung der Kunden im System

Die Kunden können sich dann im System anmelden und ihre Tickets sehen. Eine detaillierte Erklärung darüber, wie Sie als Kunde mit ConSol CM arbeiten, finden Sie im Abschnitt *CM/Track* des *ConSol CM Benutzerhandbuchs*.

Es gibt zwei Wege für die Benutzerauthentifizierung:

- einfache Authentifizierung
- LDAP-Authentifizierung

Details dazu finden Sie im Abschnitt Authentifizierungsmethoden für Kunden in CM/Track.

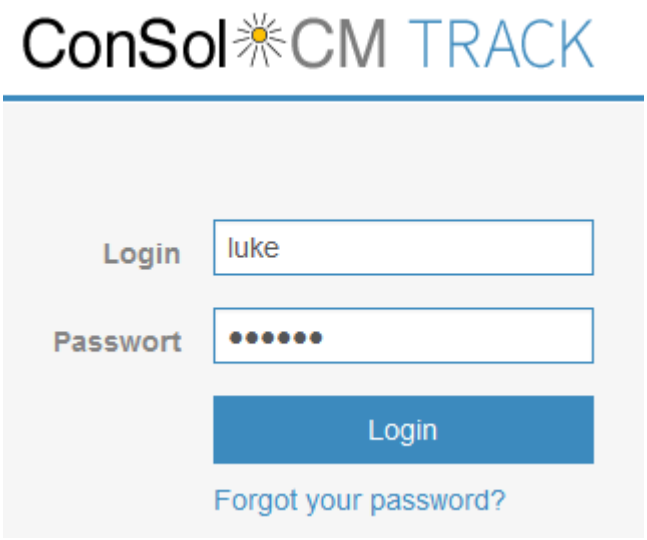


Abbildung 682: *ConSol CM/Track - Kundenanmeldung*

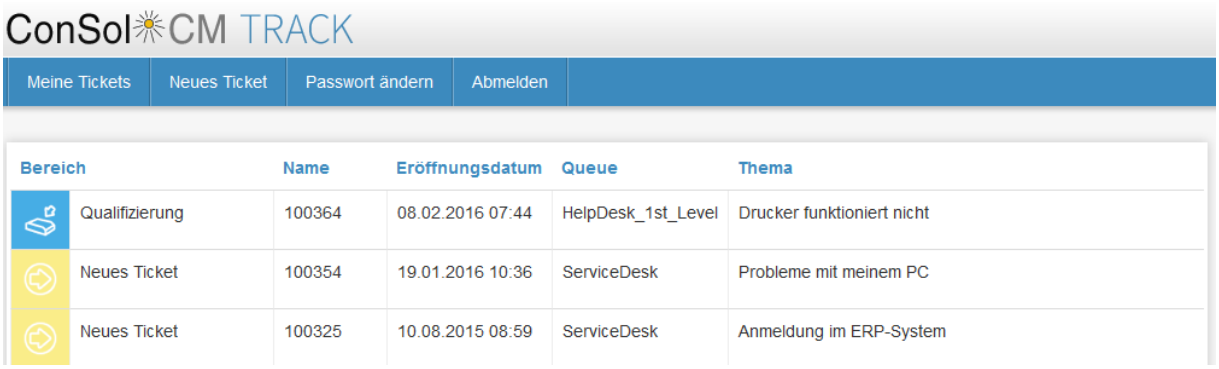


Abbildung 683: *ConSol CM/Track - Ticketliste*

Erweiterte Kundenberechtigungen zum Anzeigen von Tickets der Firma

In einigen Fällen ist es möglicherweise erforderlich, dass Kunden sich im ConSol CM-Portal CM/Track anmelden, und nicht nur auf ihre persönlichen Tickets Zugriff haben, sondern auf alle Tickets ihrer Firma. In diesem Fall muss der Rolle des CM/Track-Benutzers (Benutzerprofil) unter *Track-Benutzer-Berechtigungen* die Berechtigung *Auf Tickets der eigenen Firma zugreifen* zugewiesen werden. Eine detaillierte Erklärung dazu finden Sie im Abschnitt Rollenverwaltung.

Konfigurieren von CM/Track für das Zurücksetzen des Passworts durch den Kunden

Ab ConSol CM-Version 6.10.0 kann CM/Track so konfiguriert werden, dass den Kunden ein Link angezeigt wird, mit dem sie ihr Passwort zurücksetzen können. Diese Funktion basiert auf dem Template `track-password-reset-template`. Eine detaillierte Erklärung dazu finden Sie im Abschnitt Template zum Zurücksetzen des Passworts durch Kunden in CM/Track V1.

Das Zurücksetzen des Passworts in CM/Track ist nur möglich, wenn der **Datenbank-Modus** verwendet wird. Bei LDAP-Authentifizierung ist dies nicht möglich. Eine Beschreibung aller verfügbaren Authentifizierungsmethoden finden Sie im Abschnitt Authentifizierungsmethoden für Kunden in CM/Track.

G.3.4.2 CM/Track V1: FAQs in CM/Track

Einführung in FAQs in CM/Track

Wenn Sie CM/Track als Portal nutzen, in dem Ihre Kunden auf ihre Tickets oder die Tickets ihrer Firma zugreifen können, möchten Sie Ihren Kunden vielleicht eine Suche der FAQ (Frequently Asked Questions) anbieten. Dies hat sich in Helpdesk- und Servicedesk-Umgebungen bewährt, da Kunden nachschauen können, ob ihr Problem schon einmal aufgetreten ist und es bereits eine Lösung gibt. Nur wenn sie keine Lösung finden, wenden sie sich an den Servicedesk und/oder öffnen ein neues Ticket. Auf diese Weise sparen sowohl die Kunden als auch das Servicedesk-Team Zeit. Sie können eine FAQ-Funktion aber auch in anderen Umgebungen anbieten.

In ConSol CM wird jede FAQ wie ein Ticket behandelt. Die Queues, die als FAQ-Queues über CM/Track verfügbar sein sollen, müssen als spezielle FAQ-Queues definiert werden, da die Kunden normalerweise nur ihre eigenen Tickets oder die Tickets ihrer Firma sehen dürfen, FAQ-Tickets aber keinem Kunden gehören. Alle Kunden mit einem Benutzerprofil, das Zugriff auf die FAQ-Queues hat, können auf die entsprechenden FAQ-Tickets zugreifen. Dafür muss nur Lesezugriff gewährt werden.

Konfigurieren des ConSol CM-Systems für die FAQ-Suche in CM/Track

Der erste Schritt ist die Erstellung eines FAQ-Workflows (siehe *ConSol CM Process Designer Handbuch*) und die Erstellung einer FAQ-Queue, die als Queue für Frequently Asked Questions (Checkbox *FAQ*) gekennzeichnet ist. Außerdem müssen Sie der Queue mindestens eine Textklasse zuweisen, bei der die Option *Sichtbar für Kunden* (siehe Textklassen) ausgewählt ist. Die Kommentare, die für die Kunden in CM/Track sichtbar sein sollen, müssen im Web Client mit dieser Textklasse gekennzeichnet werden.

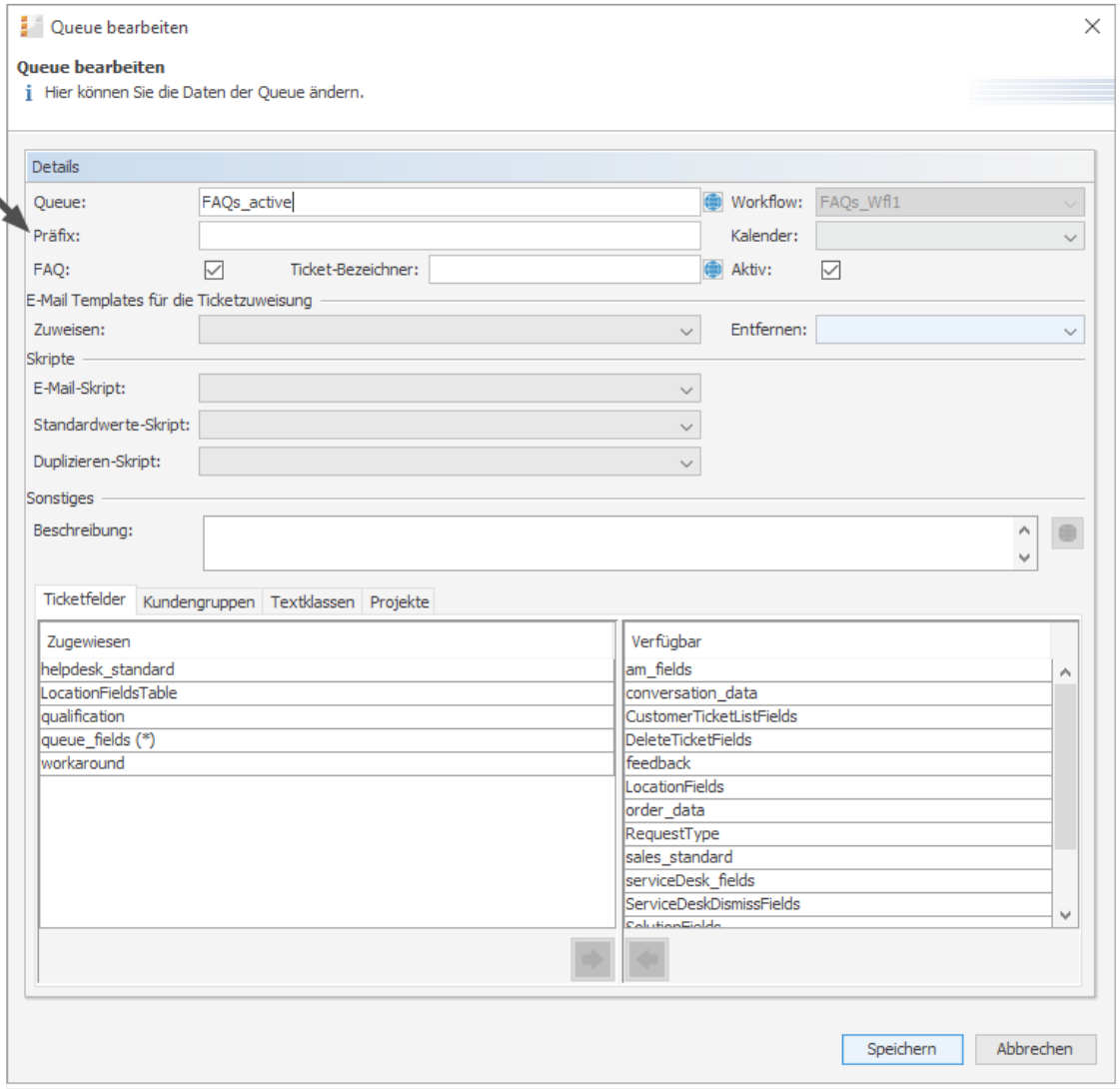


Abbildung 684: *ConSol CM Admin Tool - Globale Konfiguration, Queues*

Danach muss eine Rolle definiert werden, die im schreibgeschützten Modus auf die FAQ-Queue zugreifen kann. Denken Sie daran, dass diese Rolle auch Zugriff auf die Kundengruppe braucht, die in den Tickets der FAQ-Queue verwendet wird.

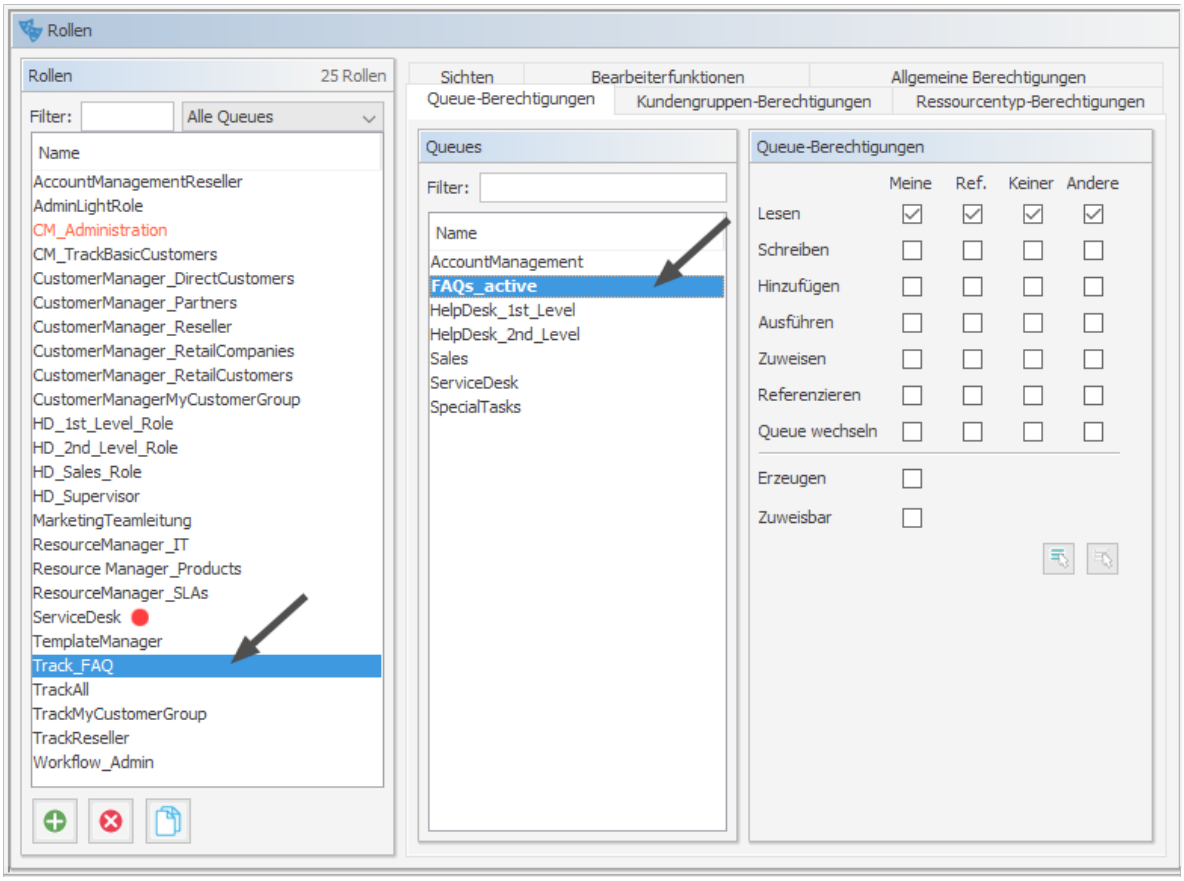


Abbildung 685: *ConSol CM Admin Tool - Zugriff und Rollen, Rollen: Rolle für FAQ*

Diese neue Rolle muss dann dem Benutzer (Profil) zugewiesen werden, der als Benutzer für den Zugang zu CM/Track verwendet wird (siehe Abschnitt CM/Track V1: Systemzugang für CM/Track-Benutzer (Kunden).

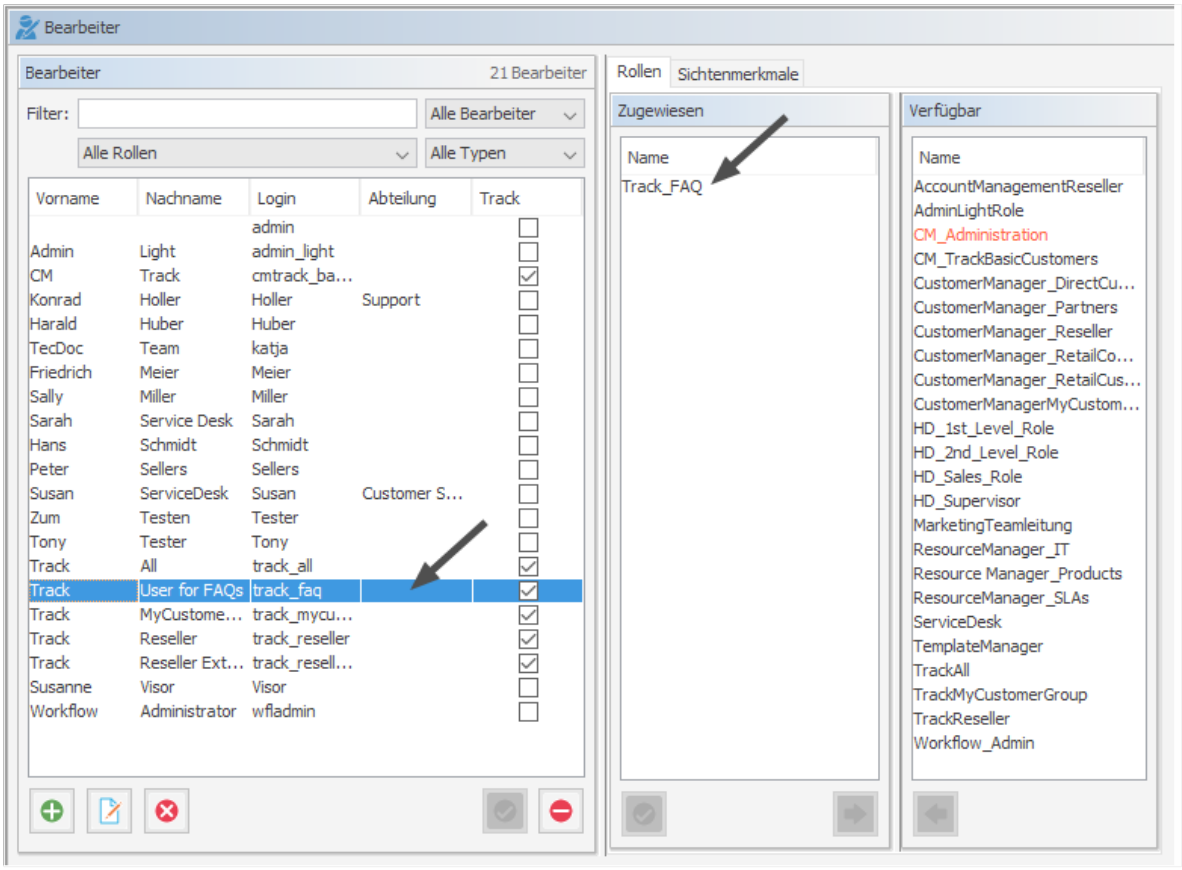


Abbildung 686: *ConSol CM Admin Tool - Zugriff und Rollen, Bearbeiter: Bearbeiter für FAQ*

FAQ-Suche in CM/Track aus der Sicht des Kunden

Ein Kunde kann die FAQ-Queue nach einem Muster durchsuchen. Es wird eine Liste mit Suchergebnissen angezeigt. Durch Klicken auf ein Ticket aus der Liste werden die Felder des Tickets angezeigt. Die Ergebnisse können eine Lösung enthalten, wie im folgenden Beispiel, oder andere Serviceinformationen.

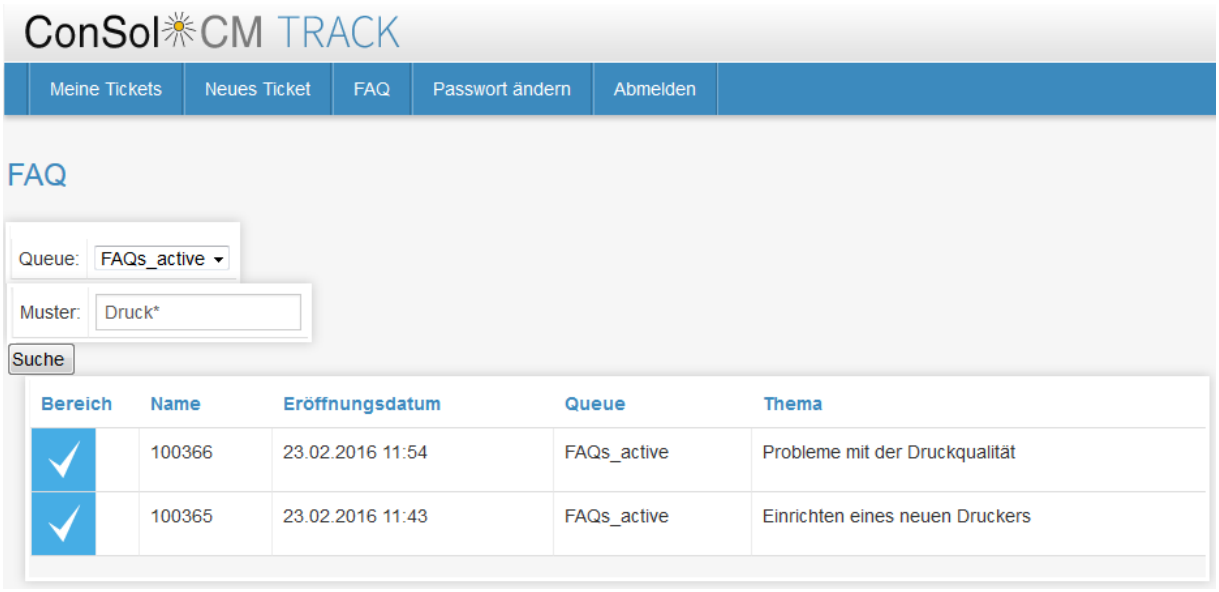


Abbildung 687: *ConSol CM/Track - Beispiel für die FAQ-Suche (1)*

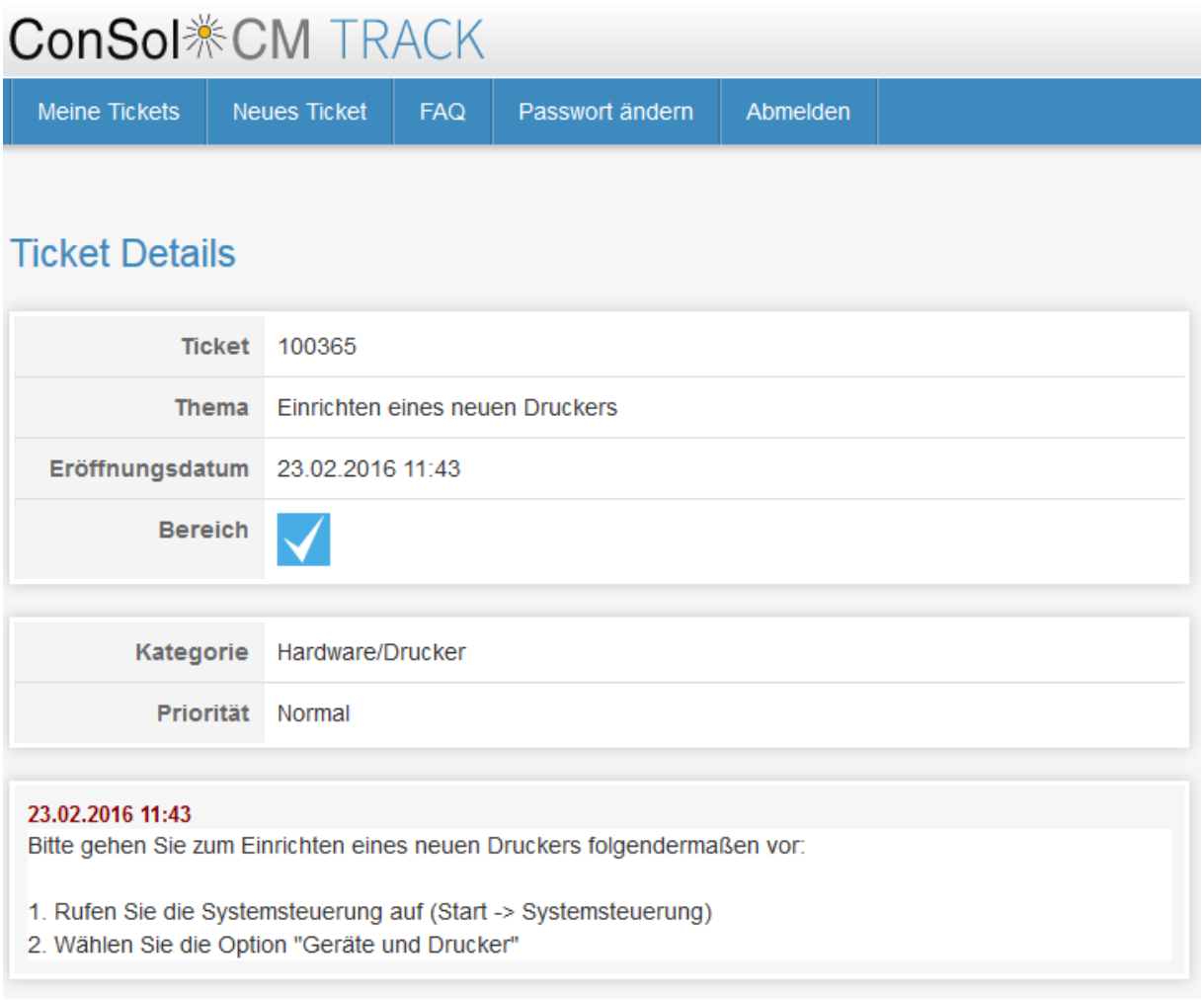


Abbildung 688: *ConSol CM/Track - Beispiel für die FAQ-Suche (2)*

Komplexere Lösungen für die Verwaltung von FAQs

Verwendung von zwei FAQ-Queues: FAQ-Verwaltung und aktive FAQs

Anstatt nur eine FAQ-Queue zu verwenden, können auch zwei Queues definiert werden. Eine kann zur Verwaltung der FAQ dienen. Dort können manuell Tickets erstellt werden, oder bestehende Tickets können aus der Helpdesk- oder Servicedesk-Queue in die FAQ-Verwaltungs-Queue verschoben werden. Ein FAQ-Verantwortlicher überprüft diese und editiert die Tickets falls erforderlich. Dann wird das Ticket in die Queue für aktive FAQs verschoben. Hier können die Kunden darauf zugreifen. Nach einem bestimmten Zeitraum oder wenn der FAQ-Verantwortliche beschließt, dass die FAQ nicht mehr verfügbar sein soll, wird sie zurück in die FAQ-Verwaltungs-Queue geschoben. Dort kann sie wieder aktiviert oder geschlossen werden.

Einrichten von zwei (oder mehr) parallelen FAQ-Umgebungen mit Track-Benutzern

Sie können FAQs für unterschiedliche Kundengruppe bereitstellen, indem Sie mehr als eine FAQ-Queue und die entsprechenden CM/Track-Benutzerprofile erstellen. So können zum Beispiel einer Kundengruppe Fragen und Antworten zum technischen Helpdesk angeboten werden, während die

andere Kundengruppe Support- und Update-Informationen erhält. Natürlich kann es auch ein CM/Track-Benutzerprofil geben, das Zugang zu beiden FAQ-Umgebungen hat.

G.3.5 CM/Track V2

In den folgenden Abschnitten werden Themen im Zusammenhang mit CM/Track V2 behandelt.

- Konfiguration der Startseite für CM/Track V2:
  Siehe Abschnitt Konfiguration der Startseite
- Authentifizierungsmethoden für Kunden in CM/Track (sowohl V1 als auch V2):
  Siehe Abschnitt Authentifizierungsmethoden für Kunden in CM/Track.
- Allgemeiner Systemzugang zu CM/Track V2 für Kunden:
  Siehe Abschnitt CM/Track V2: Systemzugang für CM/Track-Benutzer (Kunden).
- Verwendung des Portals für FAQs:
  Siehe Abschnitt CM/Track V2: FAQs in CM/Track
- Verwaltung der Verfügbarkeit von Benutzerdefinierten Feldern in CM/Track V2:
  Siehe Abschnitt CM/Track V2: Datenverfügbarkeit für Kunden

G.3.5.1 CM/Track V2: Systemzugang für CM/Track-Benutzer (Kunden)

Im folgenden Abschnitt finden Sie detaillierte Informationen darüber, wie Sie Ihr ConSol CM-System konfigurieren, um Ihren Kunden Zugang zu CM/Track V2 (dem Portal von ConSol CM) zu gewähren.

> CM/Track V2 ist ein Add-on von ConSol CM, das separat erworben werden muss.
>
> Bedenken Sie, dass für jeden CM/Track-Benutzer (d. h. für jedes Benutzerprofil) eine eigene ConSol CM-Lizenz benötigt wird. Da sich mehrere Kunden mit einem Benutzerprofil in CM/Track anmelden können, brauchen Sie nicht für jeden Kunden eine ConSol CM-Lizenz. Details dazu finden Sie im Abschnitt Lizenzverwaltung.

Voraussetzung

CM/Track V2 ist nicht im Standardlieferumfang von ConSol CM enthalten. Wenn Sie (Ihre Firma) eine Lizenz erworben haben, erhalten Sie eine .war-Datei, die auf dem Application Server installiert werden muss. Um CM/Track V2 zu betreiben, müssen einige kleinere Änderungen an den CM-Konfigurationsdateien auf dem Application Server vorgenommen werden. Diese sind im *ConSol CM Setup Manual* erklärt.

Der Standardfunktionsumfang von CM/Track beinhaltet grundlegende Funktionen (z. B. Anzeigen der Ticketliste, Erstellen eines neuen Tickets, Anzeigen der Ticketdetails) und die Seiten haben das CM-Standardlayout. Um CM/Track als leistungsstarkes Portal für den Zugang Ihrer Kunden zum System einsetzen zu können, sollten Sie das Layout an das Corporate Design Ihres Unternehmens anpassen. Dieser Vorgang heißt *Skinning*. Die Formulare und Listen, die dem Kunden angezeigt werden, können verändert und/oder erweitert werden. Wenden Sie sich an unser Consulting-Team oder Ihren Account Manager, wenn Sie CM/Track optimal auf Ihr Unternehmen zuschneiden möchten.

Technischer Hintergrund von CM/Track V2

Das Portal CM/Track basiert auf der ***REST API*** von ConSol CM. Details dazu finden Sie im Dokument *ConSol CM REST API Documentation*.

Allgemeines Prinzip des Systemzugangs über CM/Track V2

Ein Kunde, der über das Portal CM/Track Zugang zu Ihrem ConSol CM-System haben möchte oder erhalten soll, braucht ein Login und ein Passwort. Diese können entweder vom Bearbeiter, der die Kundendaten im CM Web Client editiert, dort eingegeben werden, oder die Werte können automatisch in die Datenbank importiert werden.

Die Felder für das Login und das Passwort der Kunden sind Kundenfelder, die wie alle anderen Kundenfelder definiert werden und eine besondere Annotation haben. Wenn Sie mit Kundenfeldern nicht vertraut sind, lesen Sie bitte Abschnitt Verwaltung von Kundenfeldern und GUI-Design für Kundendaten.

Die Zugangsberechtigungen der Kunden werden definiert, indem Sie dem Konto des Kunden ein CM/Track-Benutzerprofil zuweisen. Die CM/Track-Benutzerprofile werden vom ConSol CM-Administrator mit dem Admin Tool verwaltet.

Definieren der Benutzerprofile / Zugangsberechtigungen für CM/Track V2

Im ersten Schritt müssen Sie die CM/Track-Benutzerprofile definieren, d. h. die Profile mit den Zugangsberechtigungen für CM/Track. Ein CM/Track-Benutzerprofil wird wie ein normaler Bearbeiter definiert (siehe Abschnitt Bearbeiterverwaltung), aber als *Track* markiert.

Das folgende Beispiel zeigt einige CM/Track-Benutzerprofile im Admin Tool. Sie gelangen auf diese Seite, indem Sie in der Navigationsgruppe *Zugriff und Rollen* das Navigationselement *Bearbeiter* öffnen.

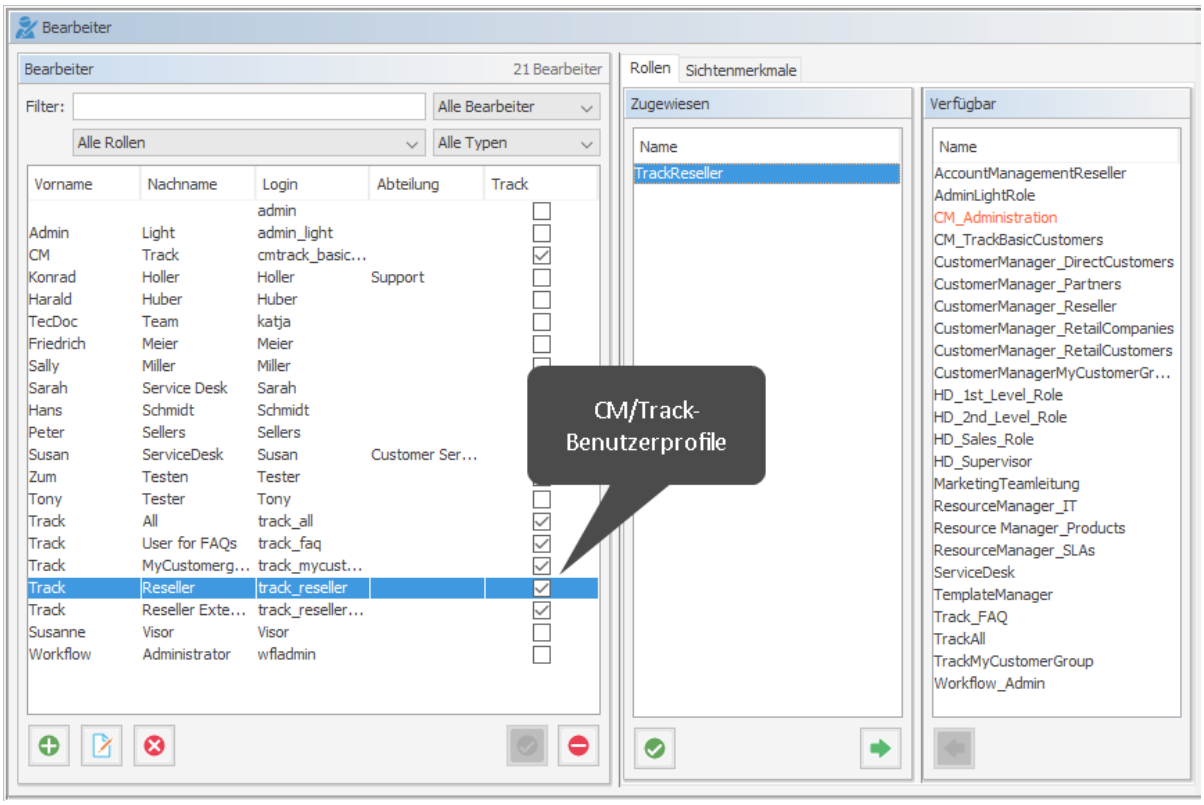


Abbildung 689: *ConSol CM Admin Tool - Zugriff und Rollen, Bearbeiter: Name des Benutzerprofils für CM/Track*

Dem Benutzerprofil werden eine oder mehrere Rollen zugewiesen, um die Zugangsberechtigungen zu Queues und Kundengruppen zu definieren. Sie können zum Beispiel ein Benutzerprofil (Bearbeiter) *track_reseller* erstellen, das die Rolle *TrackReseller* hat. Diese Rolle hat Berechtigungen zum Lesen/Schreiben/Hinzufügen in den Queues *FAQ_active*, *Helpdesk_1st_Level Helpdesk_2nd_Level*, *ServiceDesk* und *SpecialTasks* und Kundengruppen-Berechtigungen für eine Kundengruppe. Bitte beachten Sie Folgendes:

- Es müssen Queue- **und** Kundengruppen-Berechtigungen erteilt werden, damit die Kunden über CM/Track auf Tickets zugreifen können.
- Sie müssen passende Queue- und Kundengruppenberechtigungen zuweisen, d. h. Queues zuweisen, denen die entsprechenden Kundengruppen zugewiesen wurden (siehe Abschnitt Queue-Verwaltung).
- In den meisten Fällen (Standardfällen) reichen Leseberechtigungen für die Kundengruppen aus, da es nicht möglich ist, über das Portal Kundendaten zu editieren.

In unserem Beispiel hat die Rolle *TrackReseller* Lese- und Schreibzugriff auf die Kundengruppe *Reseller*. Sie gelangen auf die folgenden Seiten, indem Sie in der Navigationsgruppe *Zugriff und Rollen* das Navigationselement *Rollen* öffnen. In Ihrem System ist es möglicherweise erforderlich, unterschiedliche CM/Track-Rollen mit Zugriff auf unterschiedliche Kundengruppen zu erstellen. Eine detaillierte Einführung in die Rollenverwaltung finden Sie im Abschnitt Rollenverwaltung.

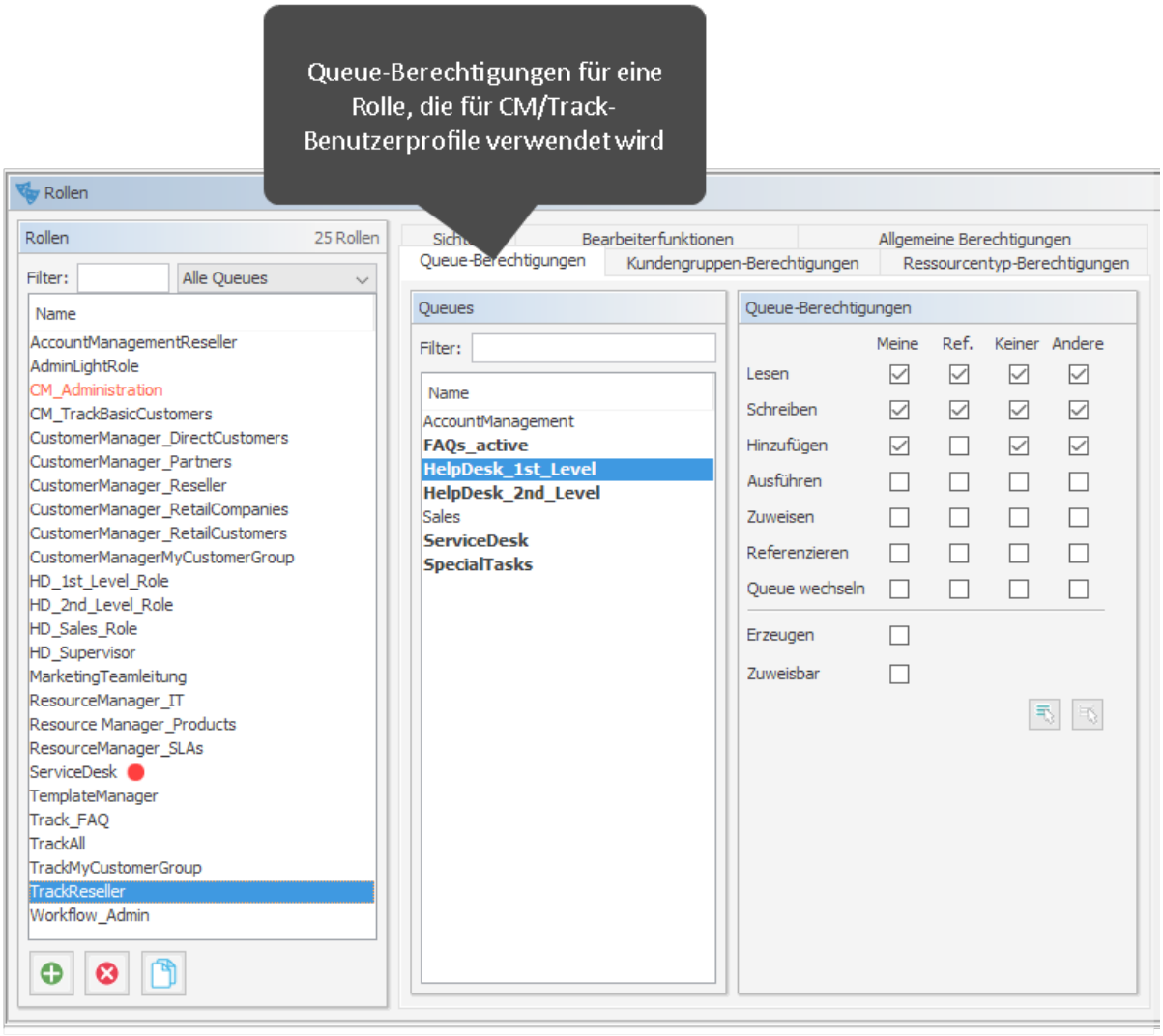


Abbildung 690: *ConSol CM Admin Tool - Zugriff und Rollen, Rollen: CM/Track-Benutzerprofil, Queue-Berechtigungen*

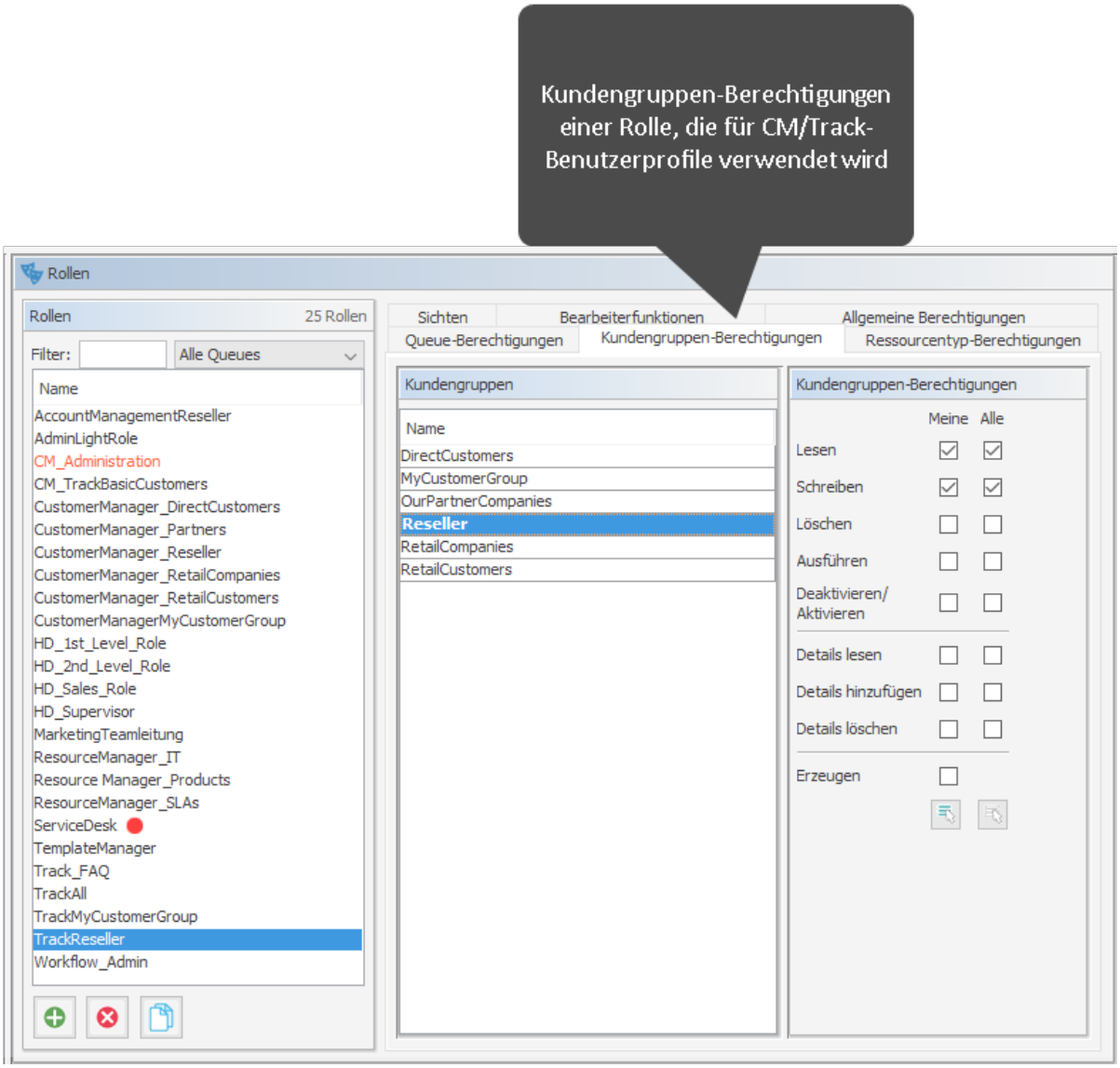


Abbildung 691: *ConSol CM Admin Tool - Zugriff und Rollen, Rollen: CM/Track-Benutzerprofil, Kundengruppen-Berechtigungen*

Auf diese Weise kann ein Kunde mit dem CM/Track-Benutzerprofil *cmtrack_basiccustomers* nur Tickets in diesen Queues sehen und dort Kommentare hinzufügen. Ein anderes Benutzerprofil könnte Zugang zu den Tickets in der Queue *Sales* und/oder zur Queue *FAQ* haben.

Definieren des Modus der Benutzerzuweisung

Der Modus der Benutzerzuweisung wird für jede Kundengruppe definiert und bestimmt das Systemverhalten, wie das CM/Track-Benutzerprofil einem Kontakt zugewiesen wird. Öffnen Sie das Fenster zum Editieren der Kundengruppe. Wählen Sie dazu in der Navigationsgruppe *Kunden* im Navigationselement *Kundengruppen* eine Gruppe aus und klicken Sie auf *Bearbeiten*. In der letzten Zeile des Fensters kann die Zuweisung des CM/Track-Benutzers eingestellt werden.

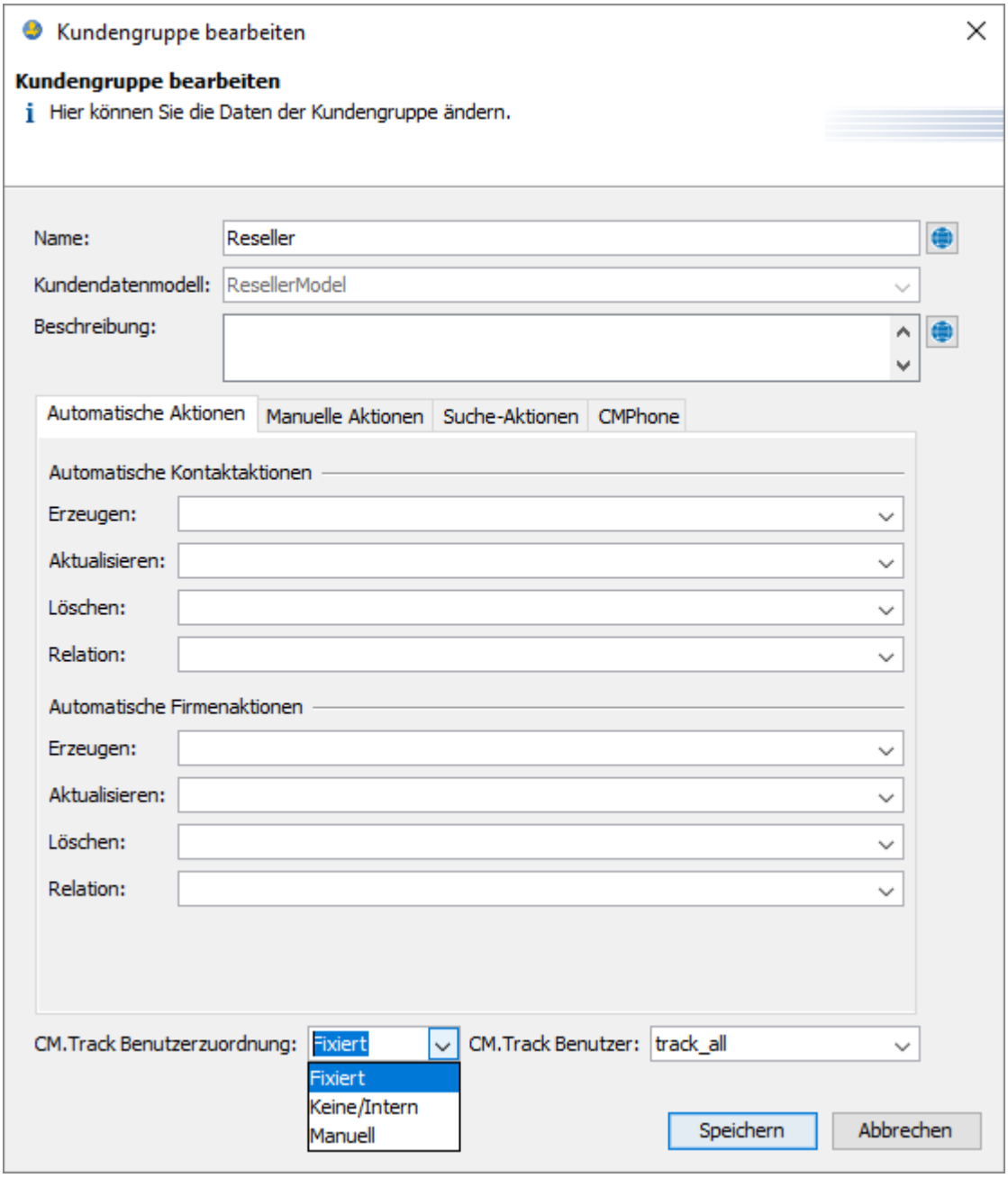


Abbildung 692: *ConSol CM Admin Tool - Kunden, Kundengruppen: Definieren des Modus für das CM/Track-Benutzerprofil für eine Kundengruppe*

Es sind drei Modi für die Zuweisung des CM/Track-Benutzerprofils verfügbar:

- **Fixiert**
  Ein CM/Track-Benutzerprofil (d. h. ein ConSol CM-Bearbeiterobjekt, das als CM/Track-Benutzerprofil definiert wurde, siehe Abschnitt Definieren der Benutzerprofile / Zugangsberechtigungen für CM/Track V2) wird im Drop-down-Menü *CM/Track Benutzer* ausgewählt. Dieses CM/Track-Benutzerprofil wird für alle Kontakte dieser Kundengruppe verwendet. Die

Datenfelder, die im Web Client angezeigt werden können, um dem Kontakt ein CM/Track-Benutzerprofil zuzuweisen (siehe Abschnitt Gewähren von Zugang zu CM/Track V2 für Kunden), werden dann im Web Client nicht angezeigt.

> ⚠ Beachten Sie, dass der Zuweisungsmodus „Fixiert“ nicht mehr geändert werden kann, sobald die Kundengruppe Kontakte hat!

- **Manuell**
  Standardwert. Die Zuweisung des CM/Track-Benutzerprofils erfolgt manuell durch einen Bearbeiter im Web Client, wie im Abschnitt Gewähren von Zugang zu CM/Track V2 für Kunden beschrieben. Das Benutzerprofil kann auch über die REST-API gesetzt werden.
- **Keine/Intern**
  Die Zuweisung eines CM/Track-Benutzerprofils ist in keinem Client möglich und kann bei Bedarf nur über ein Skript erfolgen. Es steht keine Auswahl im Web Client zur Verfügung und ein Versuch, das Benutzerprofil über die REST-API zu ändern, gibt eine Statusmeldung METHOD NOT ALLOWED zurück.

Definieren der Kundenfelder für das CM/Track V2-Login und -Passwort

Die Felder für das Login und Passwort für einen Kunden sind normale Kundenfelder auf der Kontaktstufe. Eine Einführung in die Verwaltung von Kundenfeldern und die GUI-Konfiguration für Kundendaten finden Sie im Abschnitt Einrichten des Kundendatenmodells.

Editieren Sie die Felder, die Kundendaten enthalten (wenn es zwei Stufen gibt: **nicht** auf der Firmenstufe sondern auf der Kontaktstufe!) wie im folgenden Beispiel gezeigt. Sie gelangen auf die folgende Seite, indem Sie in der Navigationsgruppe *Kunden* das Navigationselement *Datenmodelle* öffnen.

- Es muss ein Feld für das Login erstellt werden, Annotation `username` = „true“.

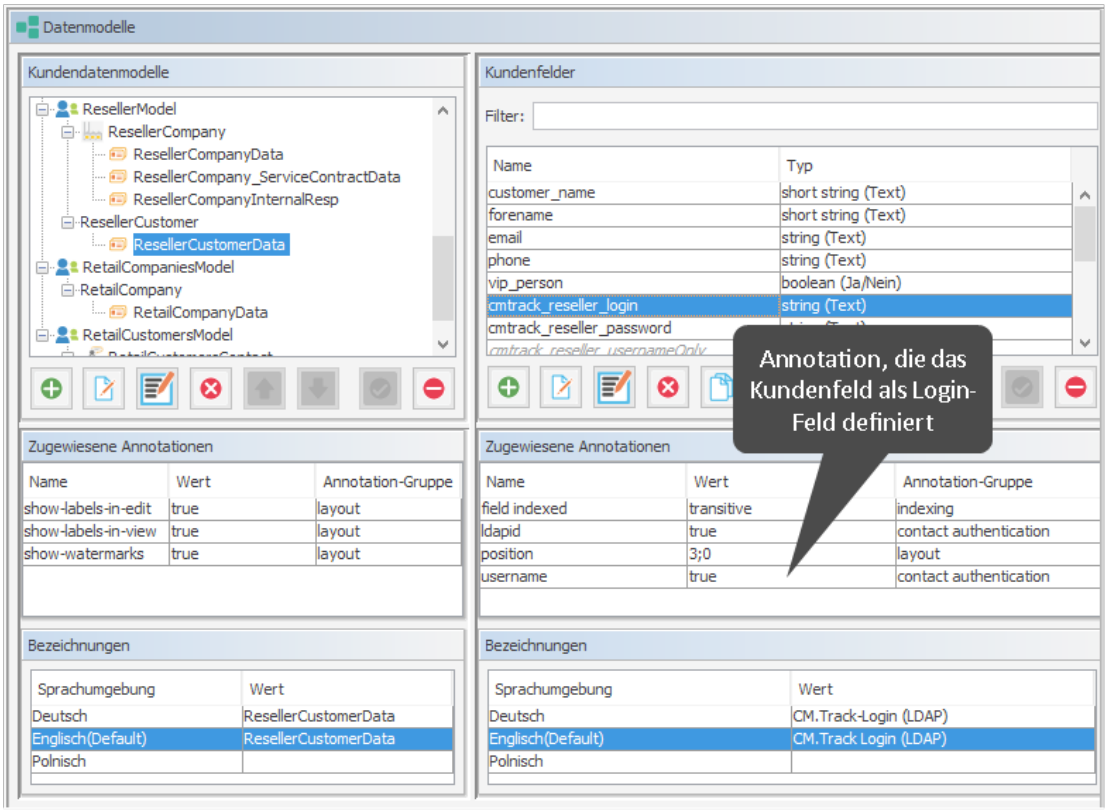


Abbildung 693: *ConSol CM Admin Tool - CM/Track: Annotation für Login*

> ⚠ Die Zuweisung der Annotation `username` zu einem Kundenfeld ist nur möglich, wenn diese Annotation noch nicht zugewiesen wurde. Andernfalls ist sie nicht erlaubt. Beim Zuweisen der Annotation muss zuerst eine Warnung bestätigt werden, bevor der Vorgang ausgeführt wird, da dieser länger dauern kann. Das Entfernen der Annotation muss ebenfalls bestätigt werden, da es nicht rückgängig gemacht werden kann: Beim Entfernen werden die Werte für die Benutzernamen unwiderruflich aus dem internen Speicher gelöscht.

- Es muss ein Feld für das Passwort erstellt werden, Annotation `password` = „true“. Die Annotation `text-type` = „password“ sorgt dafür, dass nur Sternchen/Punkte im Web Client angezeigt werden und das Passwort nicht im Klartext angezeigt wird.

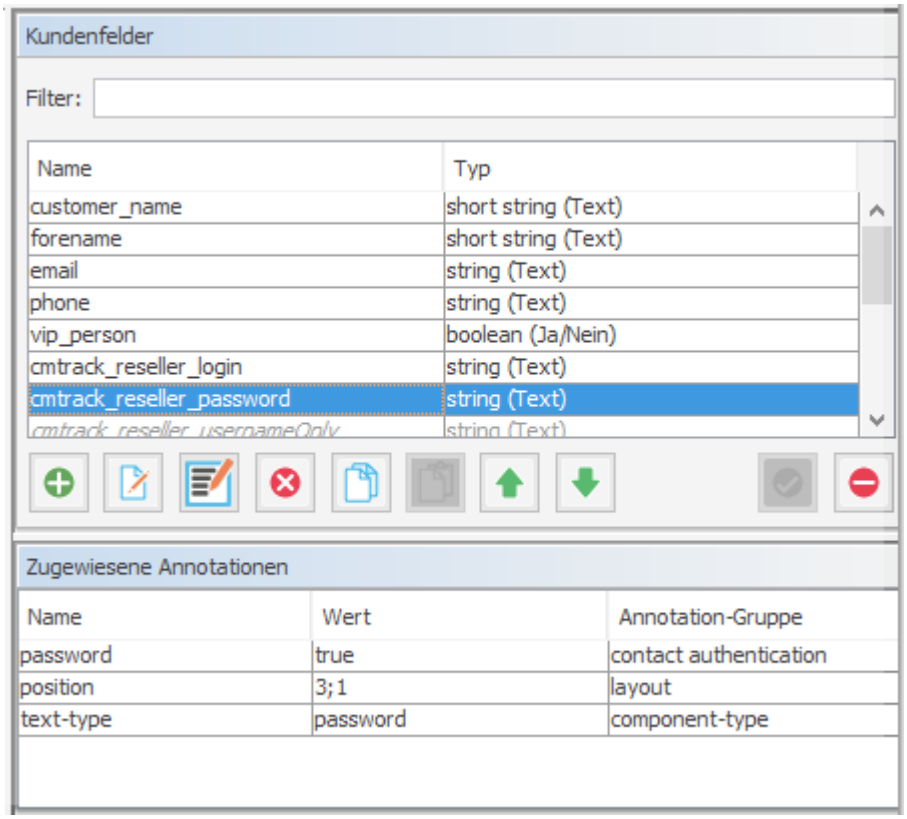


Abbildung 694: *ConSol CM Admin Tool - CM/Track: Annotation für Passwort*

> ⚠ Die Annotation `password` muss vor der Zuweisung bestätigt werden.
>
> Bei einer Aktualisierung von einer CM-Version unter 6.11 auf 6.11 und höher: Wenn diese Annotation gesetzt wird, liest das System die Passwörter im Klartext aus den ursprünglichen Feldwerten, verschlüsselt sie und speichert sie verschlüsselt im internen Speicher. Die ursprünglichen Werte werden gelöscht, sodass die Werte im Klartext nicht mehr wiederhergestellt werden können.
>
> Beim Versuch, die Annotation `password` zu entfernen, muss der Vorgang ebenfalls bestätigt werden, da die verschlüsselten Passwörter aus dem internen Speicher gelöscht werden. Nach dem Entfernen der Annotation sind die Passwort-Informationen nicht mehr vorhanden und können nicht mehr wiederhergestellt werden.

Wenn ein Szenario aus einer CM-Version unter 6.11 in ein System mit CM 6.11 oder höher importiert wird, wird die Transformation der Benutzernamen und Passwörter automatisch ausgeführt. Dies ist detailliert in Transformation von Benutzernamen- und Passwortfeldern bei einem Import nach CM 6.11 beschrieben.

Gewähren von Zugang zu CM/Track V2 für Kunden

Für alle Kundengruppen, bei denen der Modus zur Zuweisung des CM/Track-Benutzerprofils auf „Manuell“ gesetzt ist, kann der Bearbeiter, der mit dem Web Client arbeitet, jedem Kunden, der Zugang zum Portal CM/Track haben soll, einen Benutzernamen, ein initiales Passwort und ein CM/Track-Benutzerprofil zuweisen. Der Benutzername muss eindeutig sein. Dies wird vom System überprüft. Sie können einen Benutzernamen nicht ein zweites Mal eingeben, wenn dieser bereits für einen anderen Kunden verwendet wird. Das Passwort wird verschlüsselt in der CM-Datenbank gespeichert. Das bedeutet, dass ein Bearbeiter ein neues Passwort setzen kann, z. B. wenn ein Kunde anruft und darum bittet. Es ist allerdings nie möglich, das Passwort aus dem System auszulesen.

Als Administrator können Sie festlegen, ob bei den CM/Track-Benutzernamen zwischen Groß- und Kleinbuchstaben unterschieden werden soll. Verwenden Sie dazu die CM-System-Property cmas-core-security, policy.track.username.case.sensitive. Dabei handelt es sich um eine Boolean-Variable. Wenn sie auf „true“ gesetzt ist, wird bei den CM/Track-Benutzernamen zwischen Groß- und Kleinschreibung unterschieden. Stellen Sie sicher, dass die verwendete Datenbank-Collation Strings mit Groß- und Kleinschreibung unterstützt.

Das folgende Beispiel zeigt die Kundendaten eines Beispielkontaktes im ConSol CM Web Client. Sie gelangen auf diese Seite, indem Sie einen Kontaktdatensatz im Bearbeitungsmodus öffnen.

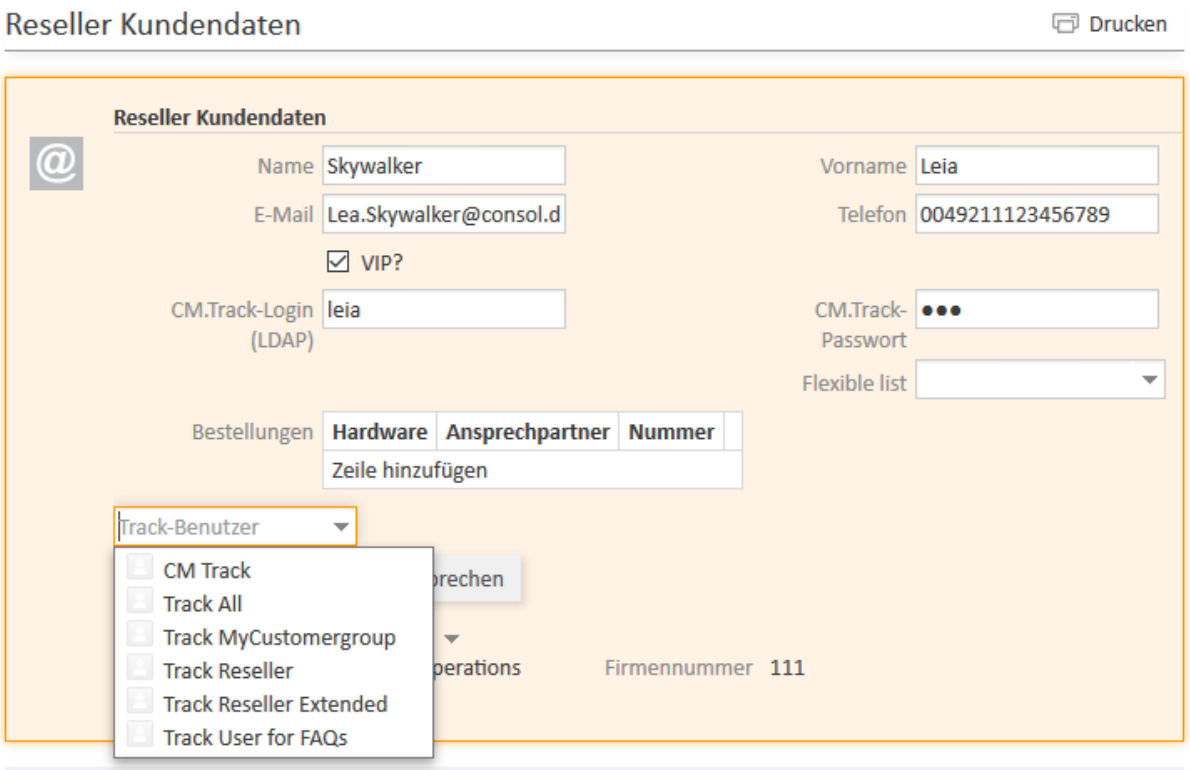


Abbildung 695: *ConSol CM Web Client - Kontaktseite: CM/Track-Benutzerdaten*

Anmeldung der Kunden im System

Die Kunden können sich dann im System anmelden und ihre Tickets sehen. Eine detaillierte Erklärung darüber, wie Sie als Kunde mit ConSol CM arbeiten, finden Sie im Abschnitt *CM/Track* des *ConSol CM Benutzerhandbuchs*.

Es gibt zwei Wege für die Benutzerauthentifizierung:

- einfache Authentifizierung
- LDAP-Authentifizierung

Details dazu finden Sie im Abschnitt Authentifizierungsmethoden für Kunden in CM/Track.

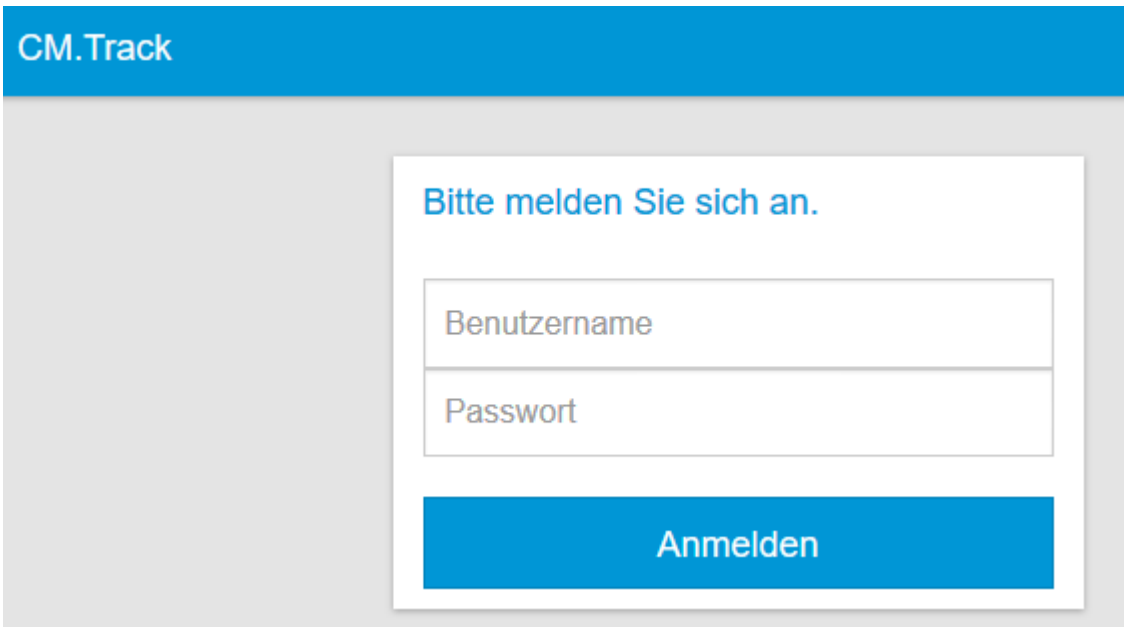


Abbildung 696: *ConSol CM/Track - Kundenanmeldung*

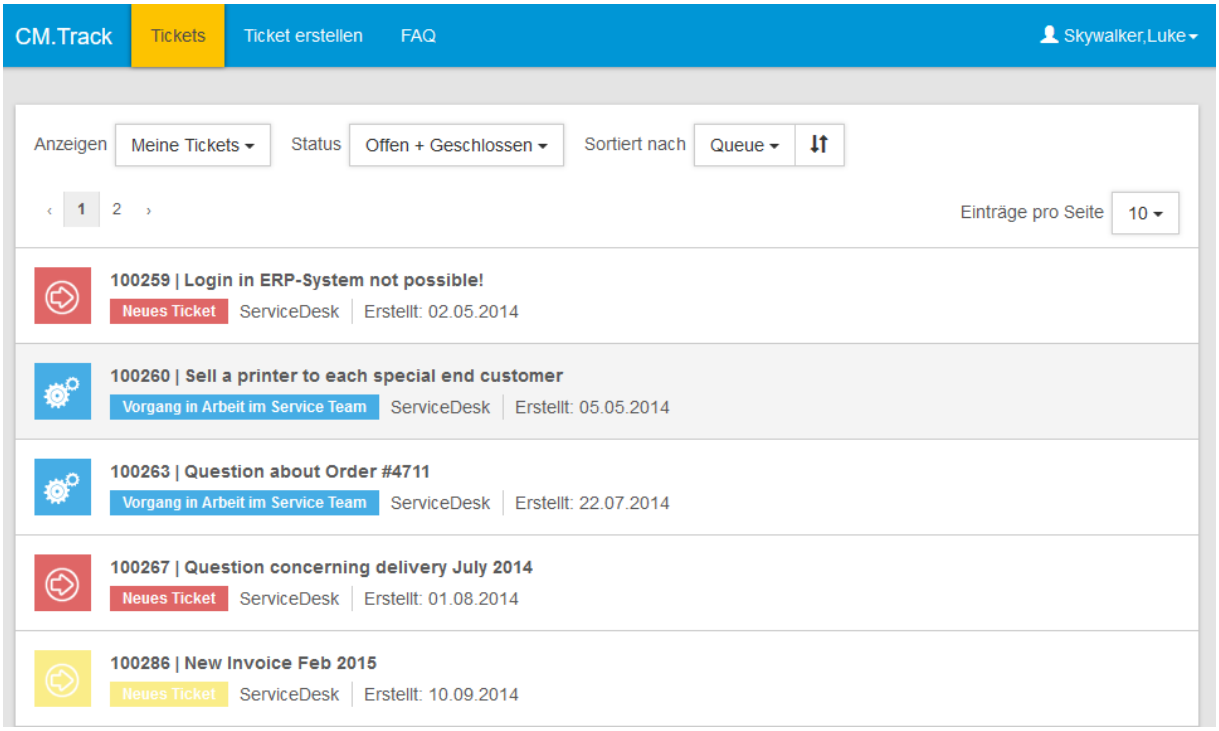


Abbildung 697: *ConSol CM/Track - Ticketliste*

Erweiterte Kundenberechtigungen zum Anzeigen von Tickets der Firma

In einigen Fällen ist es möglicherweise erforderlich, dass Kunden sich im ConSol CM-Portal CM/Track anmelden, und nicht nur auf ihre persönlichen Tickets Zugriff haben, sondern auf alle Tickets ihrer Firma. In diesem Fall muss der Rolle des CM/Track-Benutzers (Benutzerprofil) unter *Track-Benutzer-Berechtigungen* die Berechtigung *Auf Tickets der eigenen Firma zugreifen* zugewiesen werden. Eine detaillierte Erklärung dazu finden Sie im Abschnitt Rollenverwaltung.

Konfigurieren von CM/Track V2 für das Zurücksetzen des Passworts durch den Kunden

Ab ConSol CM-Version 6.10.5.4 kann CM/Track so konfiguriert werden, dass den Kunden ein Link angezeigt wird, mit dem sie ihr Passwort zurücksetzen können. Diese Funktion basiert auf dem Template `track-password-reset-template`. Eine detaillierte Erklärung dazu finden Sie im Abschnitt Template zum Zurücksetzen des Passworts durch Kunden in CM/Track V2.

Das Zurücksetzen des Passworts in CM/Track ist nur möglich, wenn der **Datenbank-Modus** verwendet wird. Bei LDAP-Authentifizierung ist dies nicht möglich. Eine Beschreibung aller verfügbaren Authentifizierungsmethoden finden Sie im Abschnitt Authentifizierungsmethoden für Kunden in CM/Track.

Beachten Sie, dass die From-Adresse der E-Mail, die an einen Kunden gesendet wird, der ein neues Passwort angefordert hat, mit der CM-System-Property cmas-core-security, password.reset.mail.from gesetzt werden kann.

G.3.5.2 CM/Track V2: FAQs in CM/Track

Einführung in FAQs in CM/Track V2

Wenn Sie CM/Track als Portal nutzen, in dem Ihre Kunden auf ihre Tickets oder die Tickets ihrer Firma zugreifen können, möchten Sie Ihren Kunden vielleicht eine Suche der FAQ (Frequently Asked Questions) anbieten. Dies hat sich in Helpdesk- und Servicedesk-Umgebungen bewährt, da Kunden nachschauen können, ob ihr Problem schon einmal aufgetreten ist und es bereits eine Lösung gibt. Nur wenn sie keine Lösung finden, wenden sie sich an den Servicedesk und/oder öffnen ein neues Ticket. Auf diese Weise sparen sowohl die Kunden als auch das Servicedesk-Team Zeit. Sie können eine FAQ-Funktion aber auch in anderen Umgebungen anbieten.

In ConSol CM wird jede FAQ wie ein Ticket behandelt. Die Queues, die als FAQ-Queues über CM/Track verfügbar sein sollen, müssen als spezielle FAQ-Queues definiert werden, da die Kunden normalerweise nur ihre eigenen Tickets oder die Tickets ihrer Firma sehen dürfen, FAQ-Tickets aber keinem Kunden gehören. Alle Kunden mit einem Benutzerprofil, das Zugriff auf die FAQ-Queues hat, können auf die entsprechenden FAQ-Tickets zugreifen. Dafür muss nur Lesezugriff gewährt werden.

Konfigurieren des ConSol CM-Systems für die FAQ-Suche in CM/Track V2

Der erste Schritt ist die Erstellung eines FAQ-Workflows (siehe *ConSol CM Process Designer Handbuch*) und die Erstellung einer FAQ-Queue, die als Queue für Frequently Asked Questions (Checkbox *FAQ*) gekennzeichnet ist. Außerdem müssen Sie der Queue mindestens eine Textklasse zuweisen, bei der die Option *Sichtbar für Kunden* (siehe Textklassen) ausgewählt ist. Die Kommentare, die für die Kunden in CM/Track sichtbar sein sollen, müssen im Web Client mit dieser Textklasse gekennzeichnet werden.

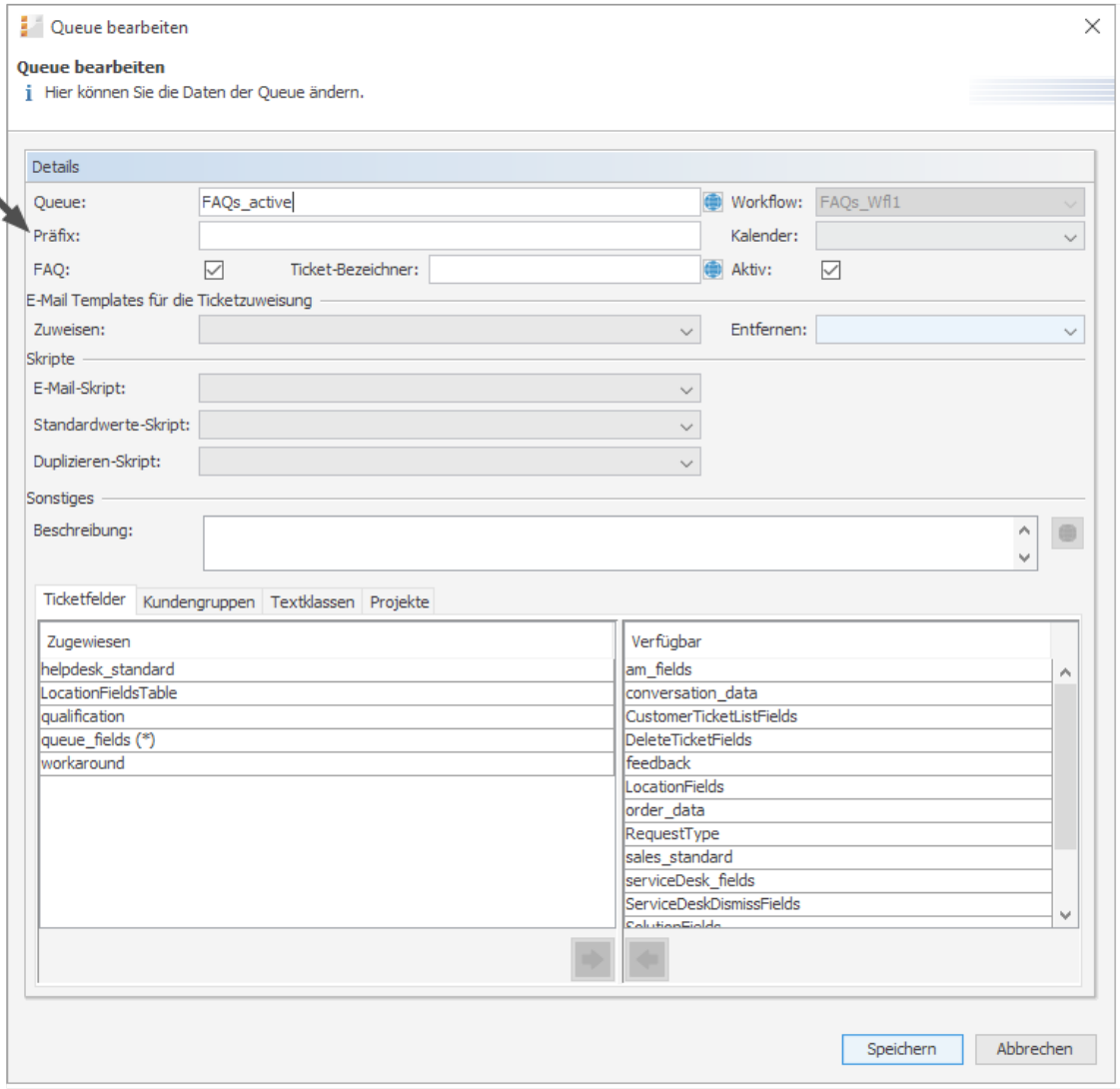


Abbildung 698: *ConSol CM Admin Tool - Globale Konfiguration, Queues*

Danach muss eine Rolle definiert werden, die im schreibgeschützten Modus auf die FAQ-Queue zugreifen kann. Denken Sie daran, dass diese Rolle auch Zugriff auf die Kundengruppe braucht, die in den Tickets der FAQ-Queue verwendet wird.

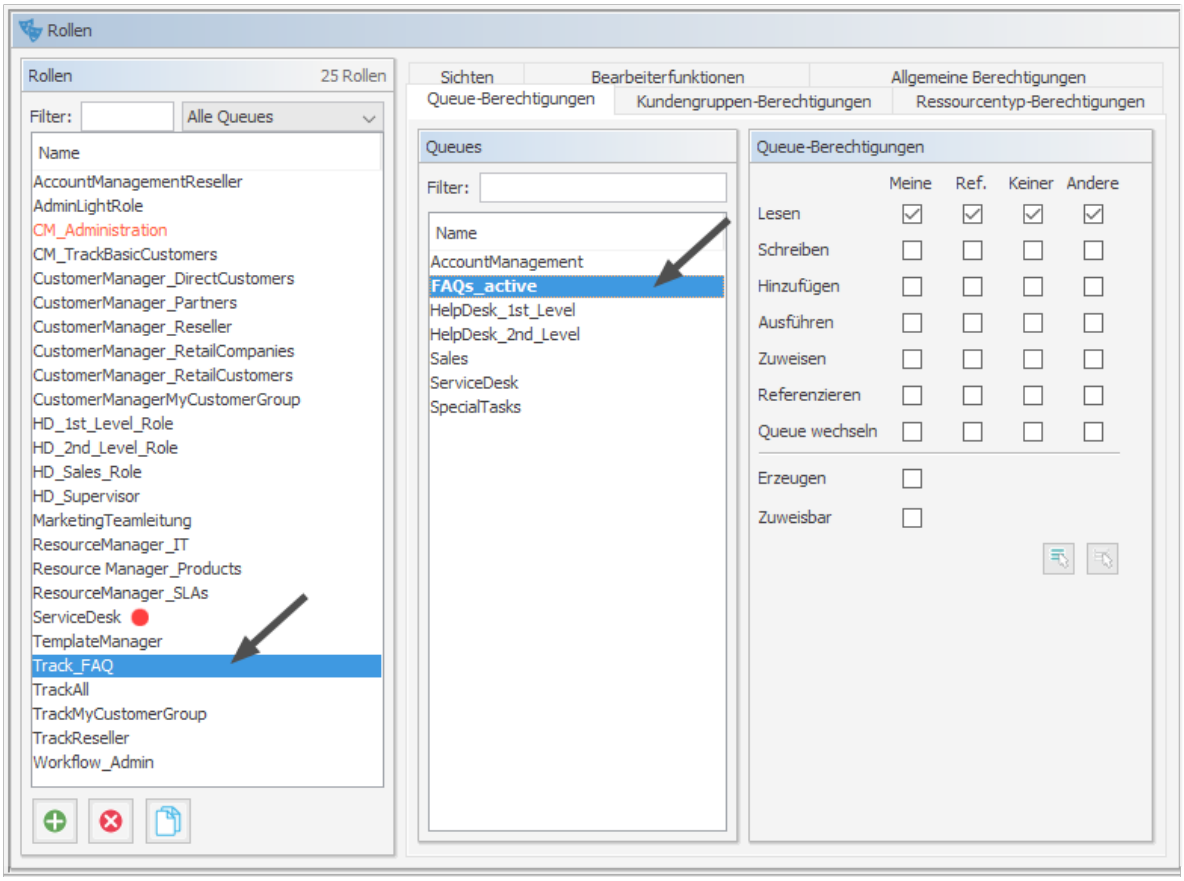


Abbildung 699: *ConSol CM Admin Tool - Zugriff und Rollen, Rollen: Rolle für FAQ*

Diese neue Rolle muss dann dem Benutzer (Profil) zugewiesen werden, der als Benutzer für den Zugang zu CM/Track verwendet wird (siehe Abschnitt CM/Track V2: Systemzugang für CM/Track-Benutzer (Kunden).

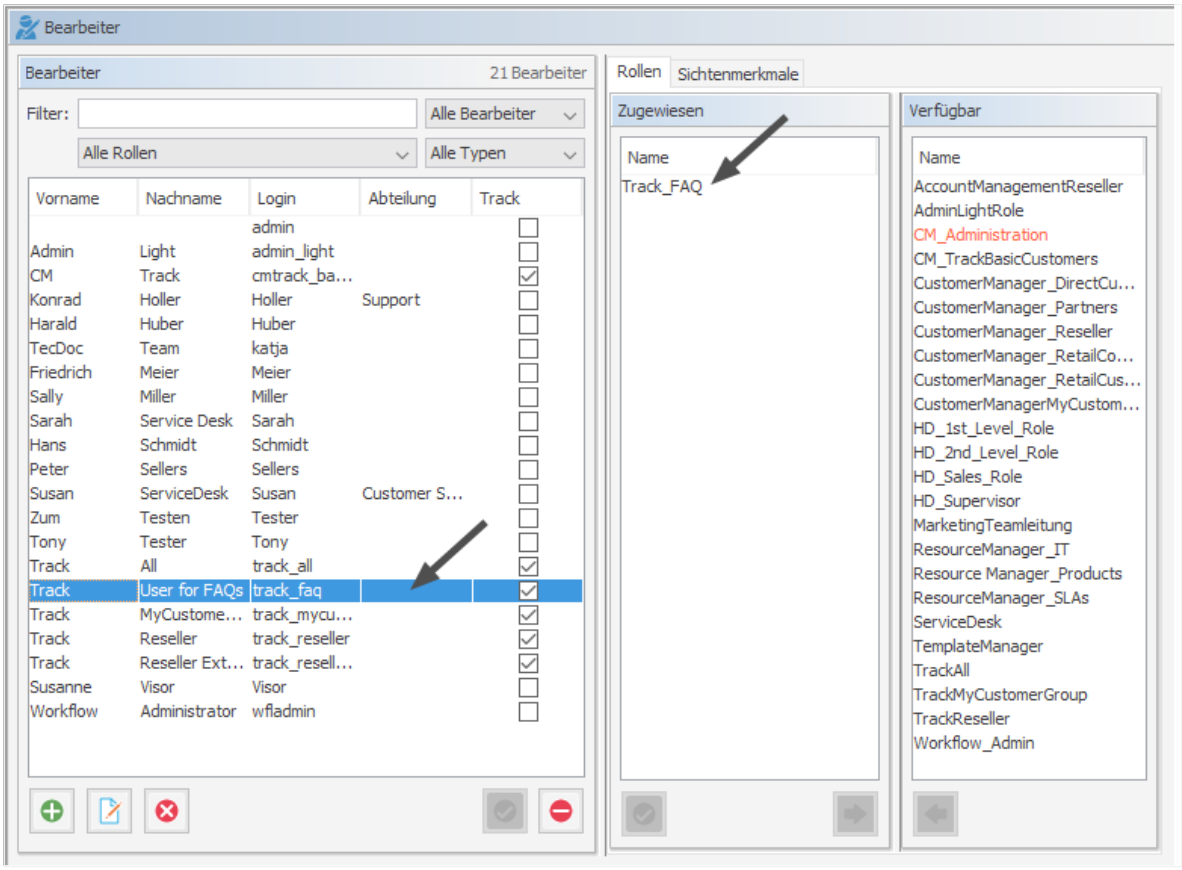


Abbildung 700: *ConSol CM Admin Tool - Zugriff und Rollen, Bearbeiter: Bearbeiter für FAQ*

FAQ-Suche in CM/Track V2 aus der Sicht des Kunden

Alle als *FAQ* gekennzeichneten Queues werden automatisch als FAQ-Queues im Portal angeboten (siehe folgende Abbildung). Es ist keine weitere Zugangskonfiguration erforderlich.

Die Queue-Auswahl ist zuerst auf *Alle FAQ Listen* eingestellt, sodass alle Tickets aus allen FAQ-Queues angezeigt werden. Ein Kunde kann die FAQ-Ticketliste einschränken, indem er entweder die Text-filterfunktion verwendet, oder in der Queue-Auswahl eine bestimmte Queue wählt. Durch Klicken auf ein Ticket aus der Liste werden die Felder des Tickets angezeigt.

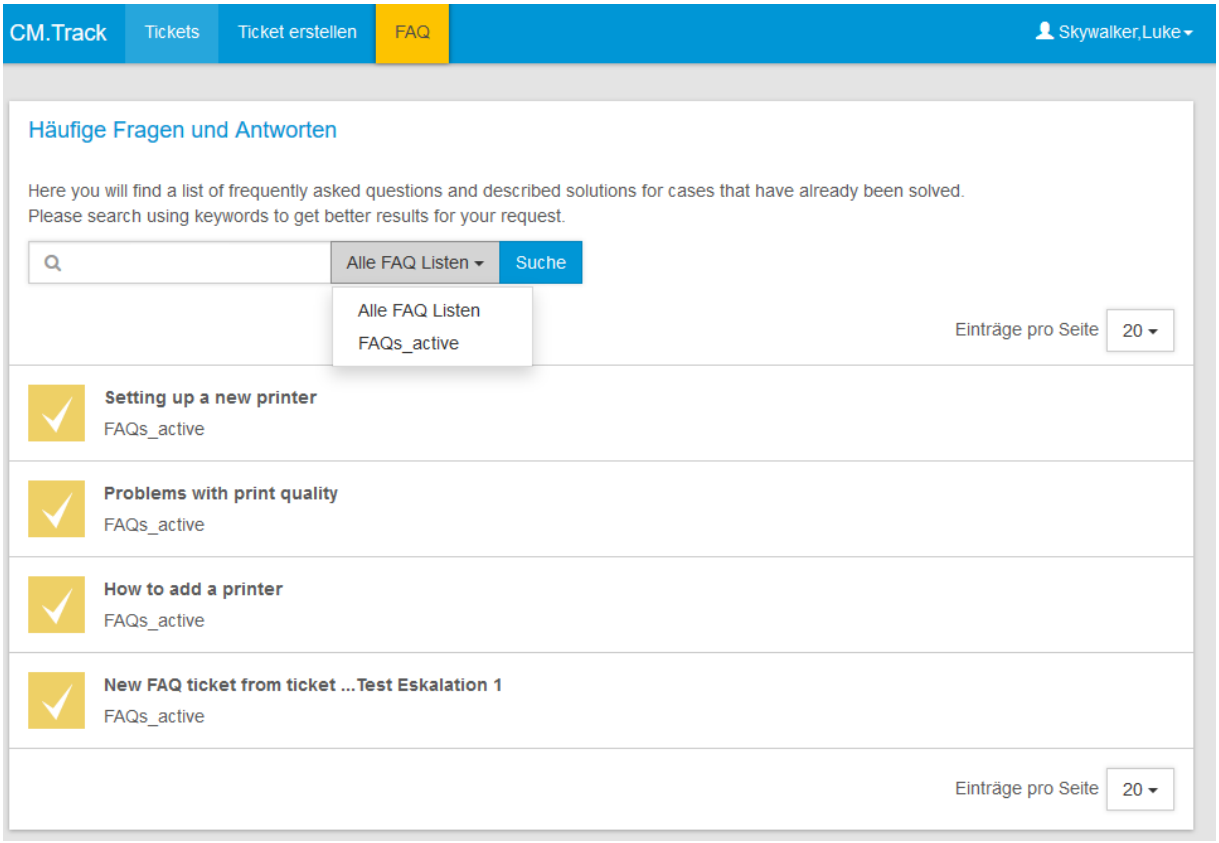


Abbildung 701: *ConSol CM/Track - Beispiel für die FAQ-Suche (1)*

Komplexere Lösungen für die Verwaltung von FAQs

Verwendung von zwei FAQ-Queues: FAQ-Verwaltung und aktive FAQs

Anstatt nur eine FAQ-Queue zu verwenden, können auch zwei Queues definiert werden. Eine kann zur Verwaltung der FAQ dienen. Dort können manuell Tickets erstellt werden, oder bestehende Tickets können aus der Helpdesk- oder Servicedesk-Queue in die FAQ-Verwaltungs-Queue verschoben werden. Ein FAQ-Verantwortlicher überprüft diese und editiert die Tickets falls erforderlich. Dann wird das Ticket in die Queue für aktive FAQs verschoben. Nur diese Queue ist in der Queue-Verwaltung als FAQ gekennzeichnet, sodass die Kunden nur auf Tickets in dieser Queue zugreifen können. Nach einem bestimmten Zeitraum oder wenn der FAQ-Verantwortliche beschließt, dass die FAQ nicht mehr verfügbar sein soll, wird sie zurück in die FAQ-Verwaltungs-Queue geschoben. Dort kann sie wieder aktiviert oder geschlossen werden.

Einrichten von zwei (oder mehr) parallelen FAQ-Umgebungen mit Track-Benutzern

Sie können FAQs für unterschiedliche Kundengruppe bereitstellen, indem Sie mehr als eine FAQ-Queue und die entsprechenden CM/Track-Benutzerprofile erstellen. So können zum Beispiel einer Kundengruppe Fragen und Antworten zum technischen Helpdesk angeboten werden, während die andere Kundengruppe Support- und Update-Informationen erhält. Natürlich kann es auch ein CM/Track-Benutzerprofil geben, das Zugang zu beiden FAQ-Umgebungen hat.

G.3.5.3 CM/Track V2: Datenverfügbarkeit für Kunden

Einleitung

Wenn Ihre Firma mit CM/Track ein Portal für ihre Kunden anbietet, muss eine Strategie bezüglich der Sichtbarkeit der Ticketdaten definiert werden. Sie möchten wahrscheinlich, dass die internen Informationen intern bleiben, Ihre Kunden aber gleichzeitig so detailliert wie möglich über ihre Tickets oder Service-Anfragen informieren.

In diesem Kontext müssen Sie über zwei Themen nachdenken:

- Welche Kommentare (Texteinträge, E-Mails, Attachments) sollen für den Kunden verfügbar sein?
- Welche Datenfelder (Ticketfelder) sollen für den Kunden sichtbar sein?

Beide Themen werden in den folgenden Abschnitten behandelt.

Welche Kommentare sollen für den Kunden verfügbar sein?

Die Sichtbarkeit der Kommentare, d. h.

- Texteinträge
- Attachments
- E-Mails

wird über Textklassen konfiguriert. Alle Kommentare, die mit einer Textklasse markiert sind, die *sichtbar für Kunden* ist, sind für den Kunden sichtbar, wenn die Details eines Tickets angezeigt werden. Details zur Konfiguration von Textklassen finden Sie im Abschnitt Textklassen.

Welche Ticketfelder sollen für den Kunden sichtbar sein

Sichtbarkeit von Ticketfeldern im CM vor 6.10.5.4

In CM-Versionen vor 6.10.5.4 sind alle Ticketfelder (mit Inhalt, leere Felder nicht) des Tickets in CM/Track sichtbar.

Sichtbarkeit von Ticketfeldern im CM in Version 6.10.5.4 und höher

Ab CM-Version 6.10.5.4 kann die Sichtbarkeit von Ticketfeldern in CM/Track für Ticketfeldgruppen sowie für einzelne Ticketfelder konfiguriert werden.

Der Mechanismus, mit dem die Sichtbarkeit von Ticketfeldern in CM/Track gesteuert wird, wird durch die CM-System-Property cmas-rest-api, security.fields.customer.exposure.check.enabled, Boolean mit Standardwert „true“, ein- bzw. ausgeschaltet. Die Werte haben folgende Bedeutung:

- **true**
  Der Sicherheitsmechanismus für die Sichtbarkeit von Ticketfeldern ist eingeschaltet. Die Kunden können in CM/Track nur die Felder sehen, die explizit als sichtbare Felder annotiert sind.
- **false**
  Der Sicherheitsmechanismus für die Sichtbarkeit von Ticketfeldern ist ausgeschaltet. Die Kunden können alle Ticketfelder in CM/Track sehen. Dies führt zu dem gleichen Verhalten wie das Standardverhalten des Systems in CM-Versionen unter 6.10.5.4.

Wenn der Sicherheitsmechanismus für die Sichtbarkeit von Ticketfeldern eingeschaltet ist (d.h. wenn die CM-System-Property cmas-rest-api, security.fields.customer.exposure.check.enabled auf „true“ gesetzt ist), wird die Sichtbarkeit der Daten in Ticketfeldern in CM/Track über folgende Annotationen gesteuert:

- Annotation der Ticketfeldgruppe
  - **customer exposure group**, Werte:
    - **full**
      die Feldgruppe ist verfügbar für
      - Lesen (Anzeige)
      - Schreiben (Speichern)
    - **read**
      die Feldgruppe ist verfügbar für
      - Lesen (Anzeige)
- Annotation des Ticketfeldes
  - **customer exposure**, Werte:
    - **full**
      das Feld ist verfügbar für
      - Lesen (Anzeige)
      - Schreiben (Speichern)
    - **read**
      das Feld ist verfügbar für
      - Lesen (Anzeige)
    - **none**
      das Feld ist nicht verfügbar / sichtbar

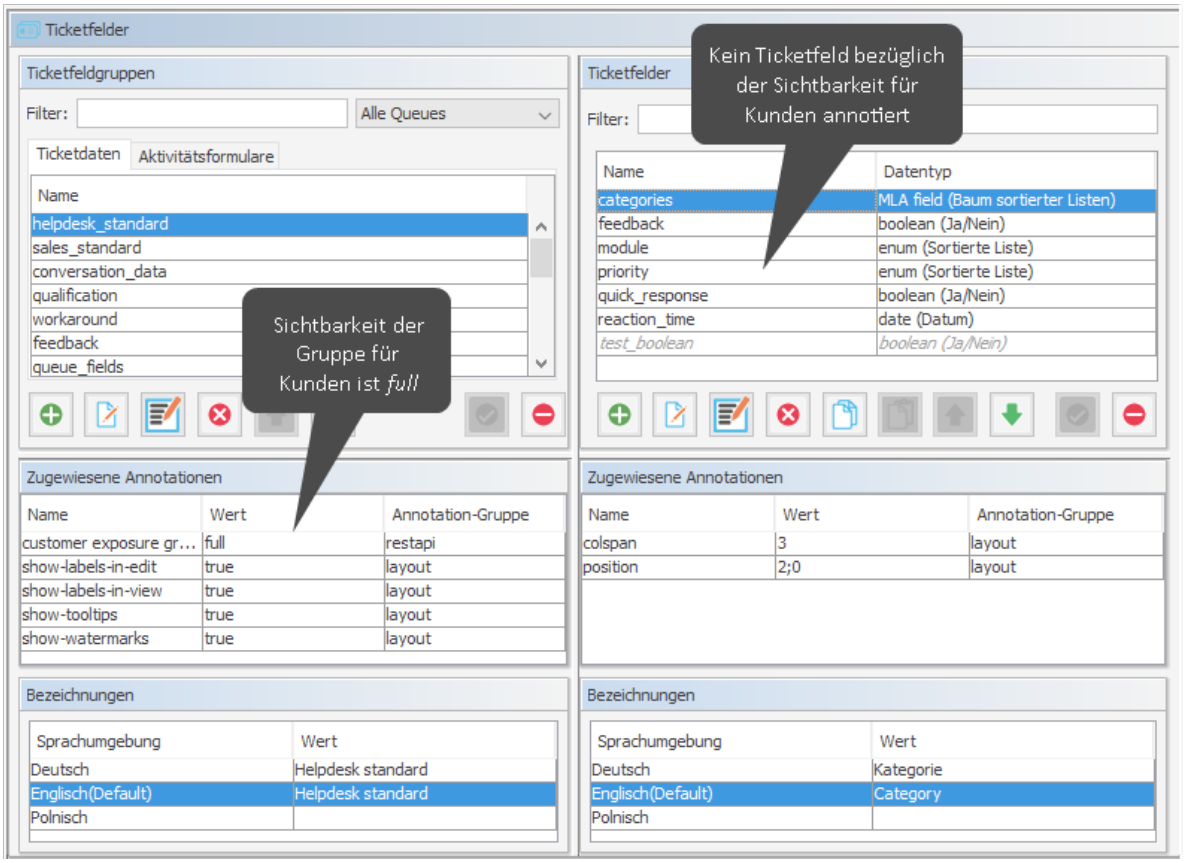
Der Wert der Annotation auf Feldebene überschreibt den Wert der Gruppenannotation. Auf diese Weise können Sie die Sichtbarkeit jedes einzelnen Feldes einstellen, ohne jedes einzelne Feld annotieren zu müssen, sofern ganze Gruppen (nicht) verfügbar sein sollen.

Die Annotationen `group-visiblity` (für Ticketfeldgruppen) und `visibility` (für einzelne Ticketfelder) funktionieren trotzdem und sind stärker als die Annotationen `customer exposure group` und `customer exposure`, d.h. wenn eine Gruppe oder ein Feld mit `(group-)visibility` = „none“ annotiert ist, wird es nicht angezeigt, unabhängig davon, ob eine Annotation `customer exposure (group)` gesetzt ist.

Dies wird in den folgenden Beispielen verdeutlicht.

Beispiel 1: Sicherheitsprüfung aktiviert, Ticketfeldgruppe sichtbar gemacht

`security.fields.customer.exposure.check.enabled` = „true“



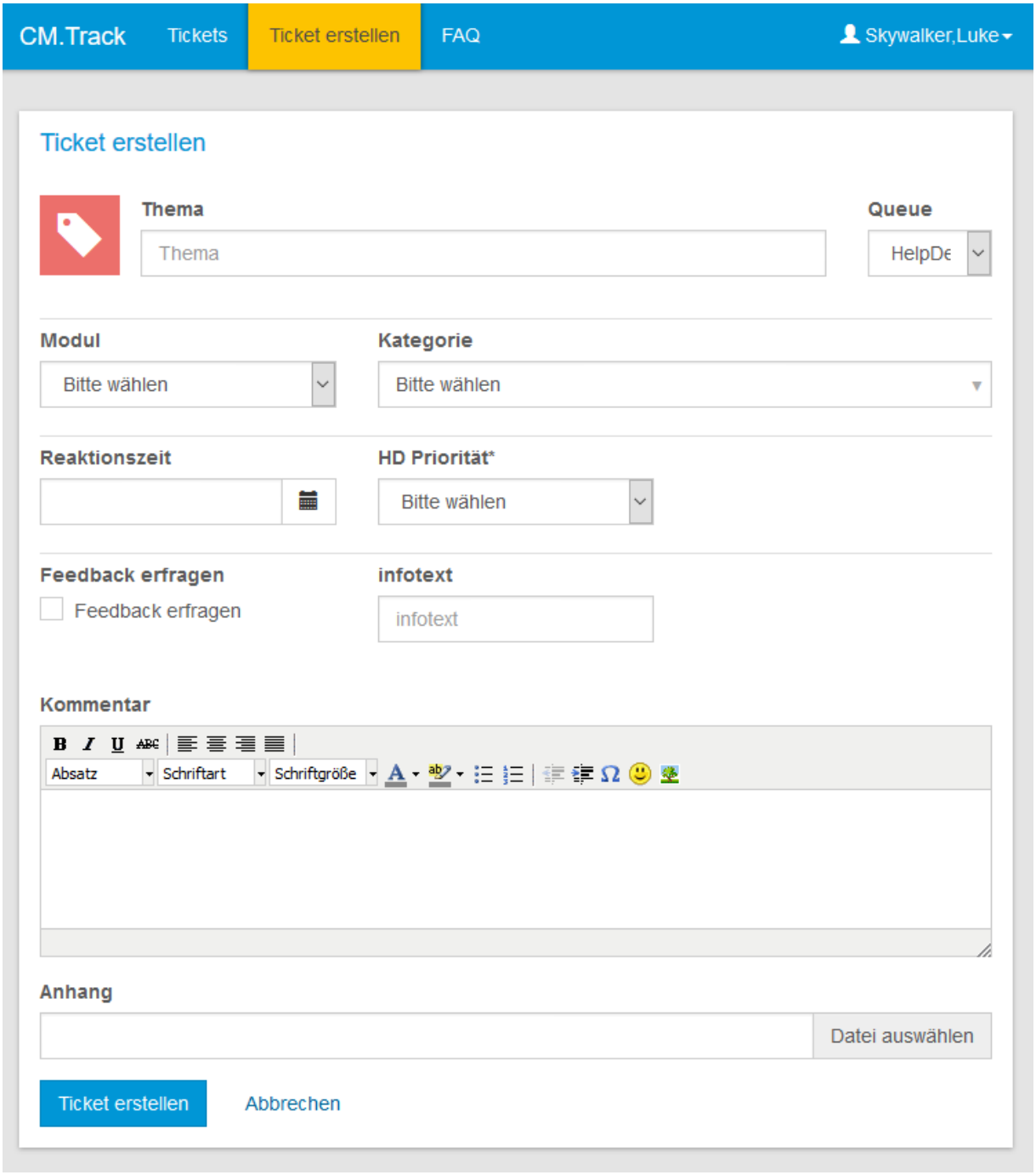
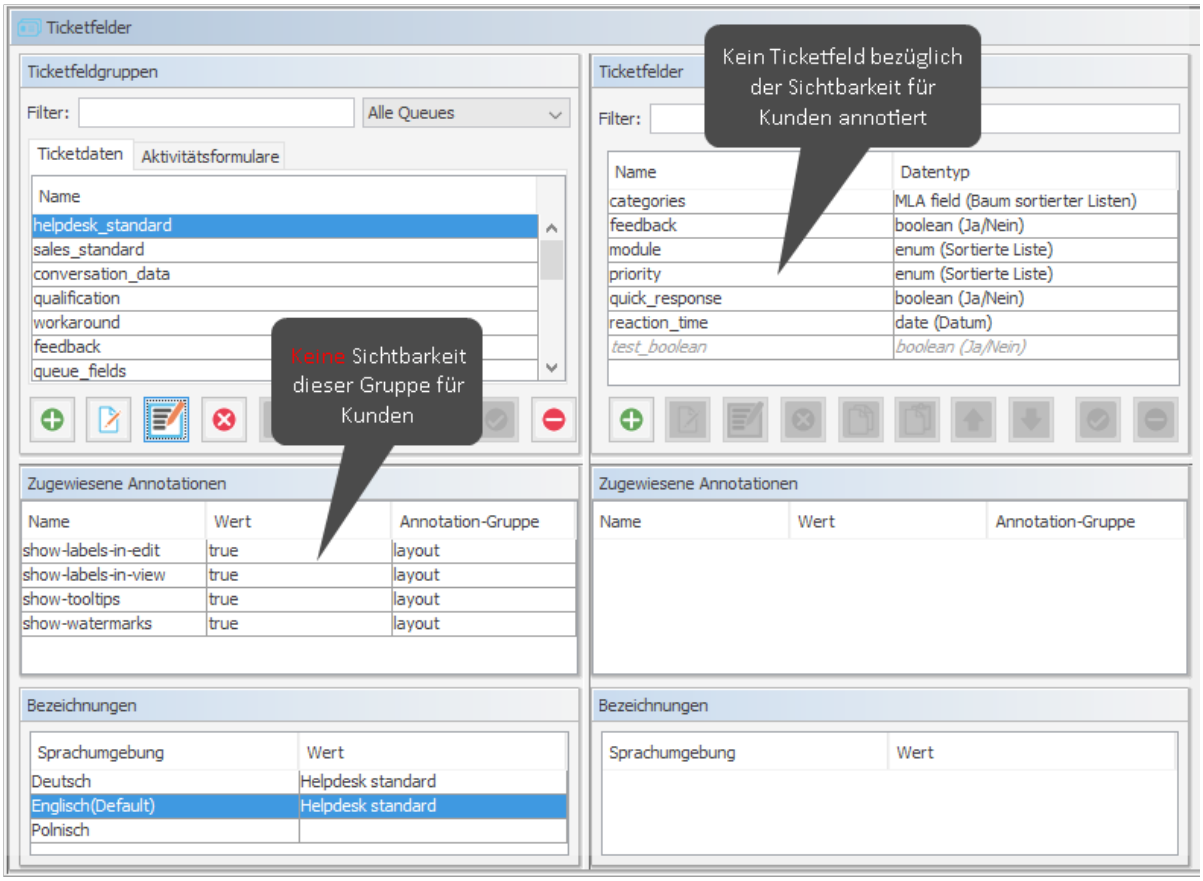


Abbildung 702: *Anzeige von Daten in CM/Track mit eingeschaltetem Sicherheitsmechanismus für die Anzeige von Ticketfeldern. Sichtbarkeit hängt von Annotation ab, hier: Gruppenannotation vollständige Sichtbarkeit.*

Beispiel 2: Sicherheitsprüfung aktiviert, Ticketfeldgruppe nicht sichtbar gemacht

`security.fields.customer.exposure.check.enabled` = „true“



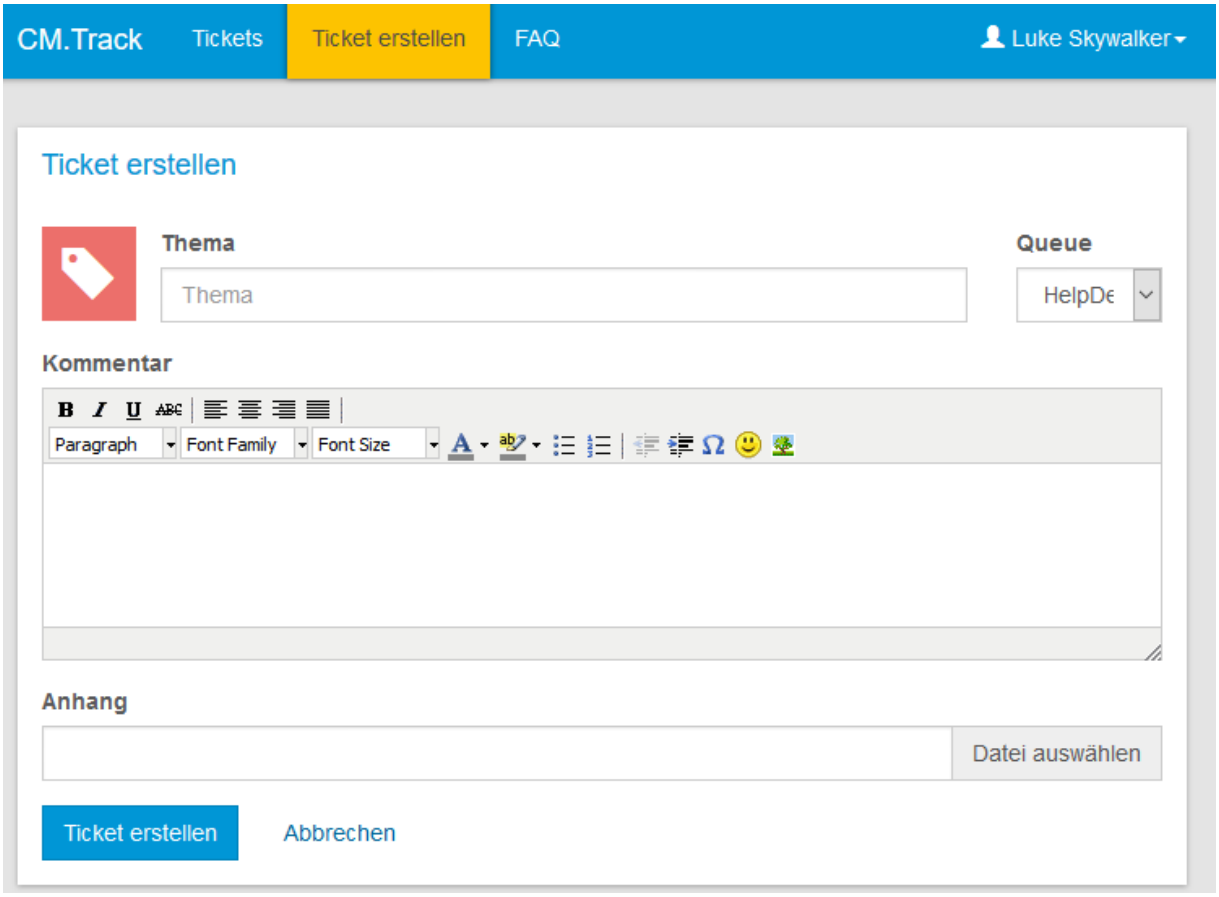
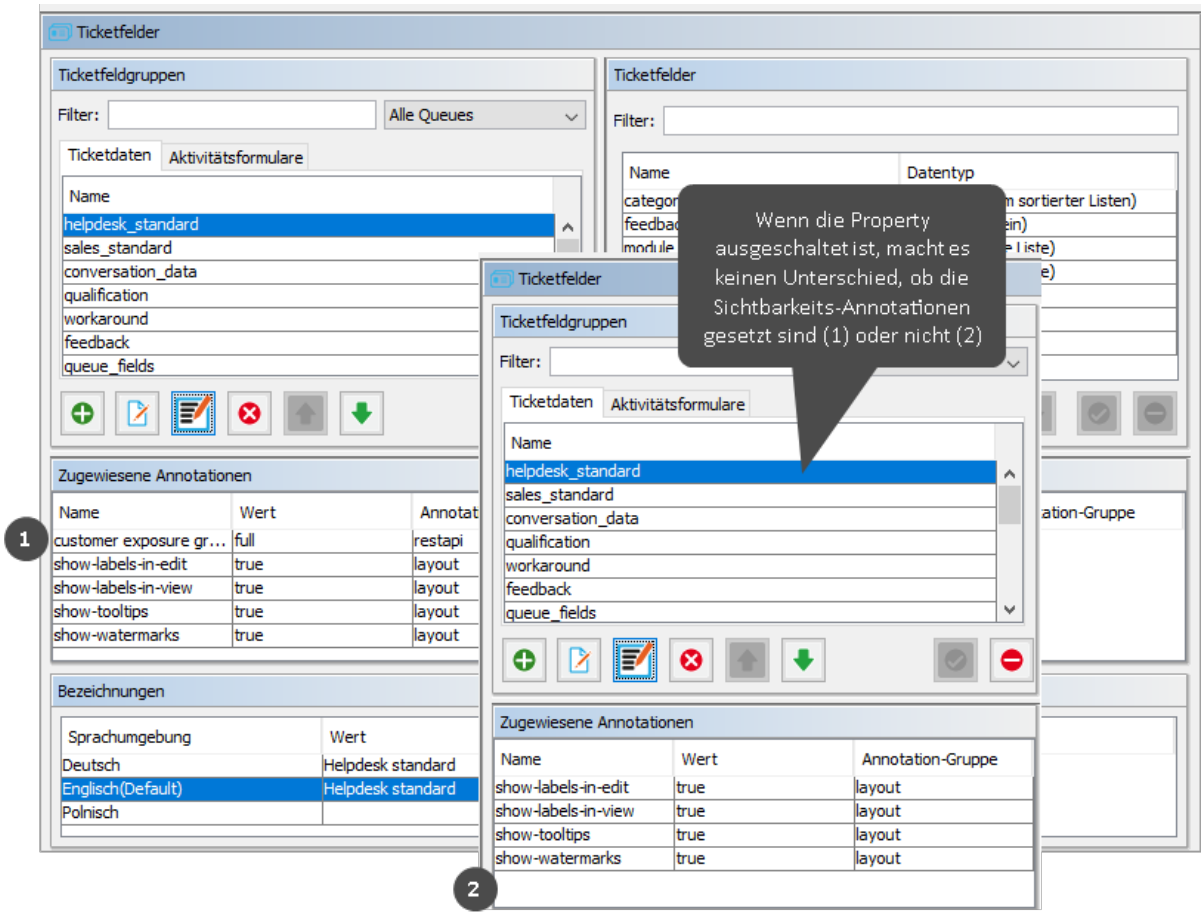


Abbildung 703: *Anzeige von Daten in CM/Track mit eingeschaltetem Sicherheitsmechanismus für die Anzeige von Ticketfeldern. Sichtbarkeit hängt von Annotation an, hier: keine Annotation für Gruppe oder Feld gesetzt.*

Beispiel 3: Sicherheitsprüfung deaktiviert, alle Ticketfelder sichtbar

`security.fields.customer.exposure.check.enabled` = „false“



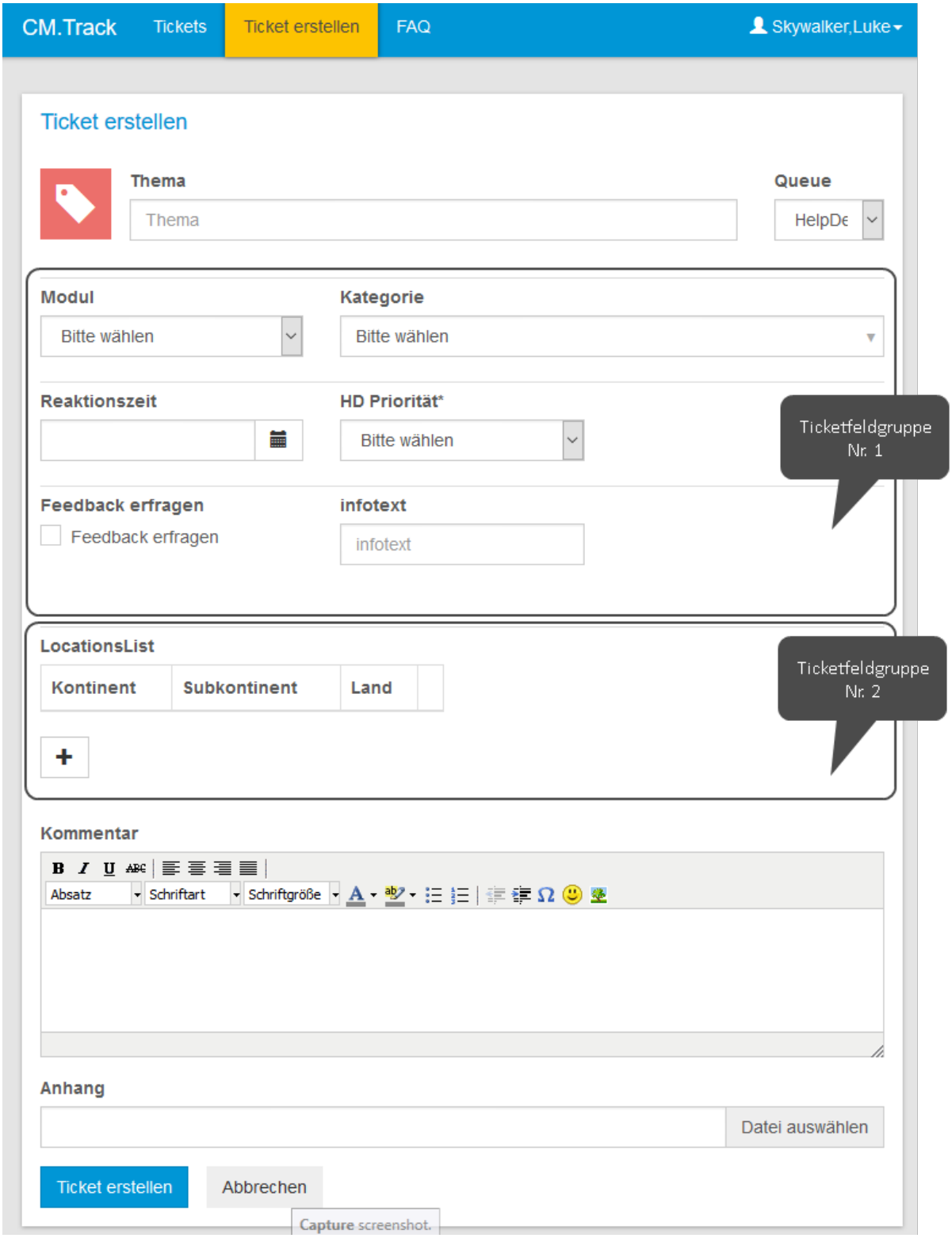


Abbildung 704: *Anzeige von Daten in CM/Track mit ausgeschaltetem Sicherheitsmechanismus für die Anzeige von Ticketfeldern. Die Annotationen customer exposure group und customer exposure haben keine Auswirkungen auf die Sichtbarkeit von Ticketfeldern in CM/Track. Es sind alle*

Ticketfelder der Queue sichtbar.

G.4 CM/Phone: CTI mit ConSol CM

In diesem Kapitel werden folgende Themen behandelt:

G.4.1 Einführung in CM/Phone 1101

G.4.2 Setup von CM/Phone 1103

G.4.3 Konfigurieren von CM/Phone im Admin Tool 1104

G.4.1 Einführung in CM/Phone

CM/Phone ist ein eigenes ConSol CM-Modul, das zusätzlich zum ConSol CM-Kernsystem lizenziert werden muss. Informationen über die Lizenzierung finden Sie im Abschnitt Lizenzverwaltung, REST.

CM/Phone ist eine Windows-Client-Applikation für die Integration von Telefonsystemen über das Protokoll ***TAPI 3***. TAPI ist Teil jedes Windows-Betriebssystems und stellt allgemeine Telefonfunktionen bereit. Der CM/Phone-Client muss auf jedem Windows-Rechner installiert werden, auf dem die CTI-Funktion (Computer Telephony Integration) mit ConSol CM verwendet werden soll.

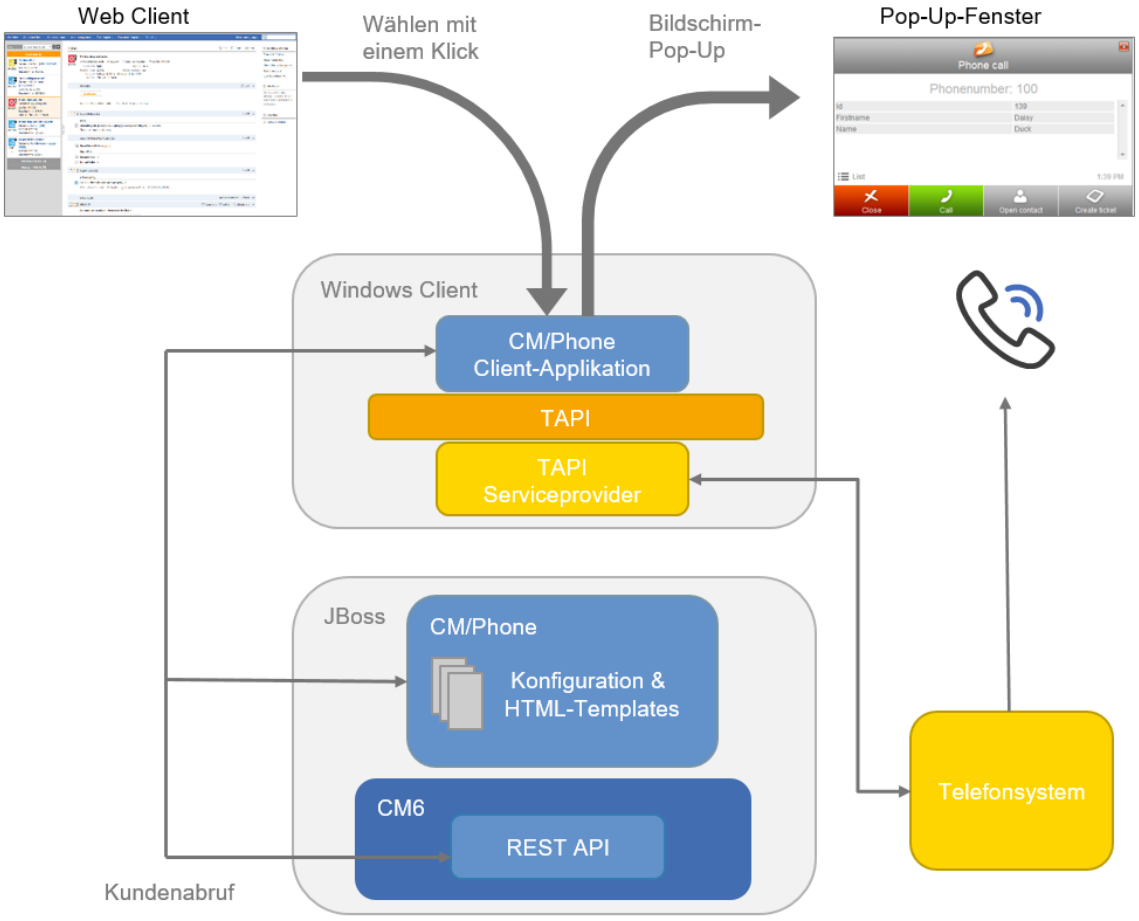


Abbildung 705: *ConSol CM/Phone - Grundprinzip*

G.4.1.1 Eingehende Anrufe

Der CM/Phone-Client überwacht das Telefon (d. h. das ausgewählte TAPI-Gerät, `address` oder `line`), ob ein Anruf eingeht. Wenn ein eingehender Anruf registriert wurde, wird ein Pop-up-Fenster geöffnet, in dem die Telefonnummer des Anrufers angezeigt wird. Die ConSol CM-Kundendatenbank wird nach passenden Kunden durchsucht. Wenn ein oder mehrere Treffer gefunden wurden, wird eine Kundenliste für die Auswahl des Kunden angeboten. Die Bearbeiter können entscheiden, ob sie ein Ticket für den Kunden erstellen möchten, oder ob die Kundenseite angezeigt werden soll. Wenn die Telefonnummer zu keinem vorhandenen Kunden passt, wird die Nummer des Anrufers angezeigt und die Option *Neuer Kunde* angeboten.

> ⚠ Beachten Sie, dass ein Benutzer im Pop-up-Fenster von CM/Phone nur die Kundendaten sehen kann, auf die er entsprechend seinen Berechtigungen Zugriff hat. Die anderen Kunden sind herausgefiltert und daher nicht sichtbar.

Das Pop-up-Fenster basiert auf **HTML-Template-Dateien**, die im CM/Phone-Ordner auf dem ConSol CM-Server gespeichert sind. Diese Templates werden von der CM/Phone-Client-Applikation beim Start geladen. Die im Pop-up-Fenster angezeigten Informationen (Kundenfelder aus dem Kundendatenmodell) können durch Editieren der Template-Dateien angepasst werden (siehe *ConSol CM Setup Manual*).

Die folgenden Optionen können im Pop-up-Fenster ausgewählt werden, wenn genau ein passender Kunde in der CM-Datenbank gefunden wird:

- **Kunden öffnen**
  Öffnet die Kundenseite (Kontakt/Firma) im Web Client (alternativ wird *Neuer Kunde* angeboten, wenn der Anrufer ConSol CM nicht bekannt ist).
- **Ticket erstellen**
  Öffnet die Seite *Neues Ticket* für den gefundenen (oder neuen) Kunden im Web Client.
- **Anrufen**
  Ist bei verpassten Anrufen verfügbar.
- **Schließen**
  Schließt das Pop-up-Fenster von CM/Phone.

Wenn der Kunde noch nicht im ConSol CM-System vorhanden ist, wird die Telefonnummer des Anrufers in das Telefonnummernfeld in den Kundendaten (Kundenfeld mit der Annotation `dialable`) eingefügt. Dies geschieht für neue Kunden und neu erstellte Tickets. Wenn mehrere Felder mit `dialable` annotiert sind, wird das erste von ihnen ausgefüllt. Wenn der Benutzer Zugang zu mehreren Kundengruppen hat, werden die entsprechenden Telefonnummernfelder mit `dialable` für jede Kundengruppe vorausgefüllt.

G.4.1.2 Ausgehende Anrufe

Der Bearbeiter kann einen ausgehenden Anruf direkt beginnen, indem er (z. B. in den Kundendaten) in einem Kundenfeld, das mit `dialable` annotiert wurde, auf eine Telefonnummer klickt. Die CM/Phone-Applikation wird automatisch vom Browser gestartet, und die Telefonnummer wird als Kommandozeilenparameter an das Telefonsystem übergeben. Die CM/Phone-Applikation erzeugt über TAPI einen ausgehenden Anruf und wird sofort danach geschlossen.

G.4.2 Setup von CM/Phone

Eine detaillierte Erklärung der Installation von CM/Phone auf dem CM-Server und den Clients finden Sie im *ConSol CM Setup Manual*. Hier ist nur die Konfiguration von CM/Phone im Admin Tool beschrieben.

G.4.3 Konfigurieren von CM/Phone im Admin Tool

Im Admin Tool müssen Sie folgende Schritte zur Konfiguration von CM/Phone durchführen:

- Setzen der Annotationen für Kundenfelder, die Telefonnummern enthalten.
- Konfigurieren der Admin-Tool-Templates für Kundendaten für jede Kundengruppe.
- Konfigurieren des Telefonnummernformats für jede einzelne Kundengruppe.
- Setzen der System-Properties.
- Optional: Ändern des Wahlpräfixes für ausgehende Anrufe.

G.4.3.1 Setzen der Annotationen für Kundenfelder mit Telefonnummern

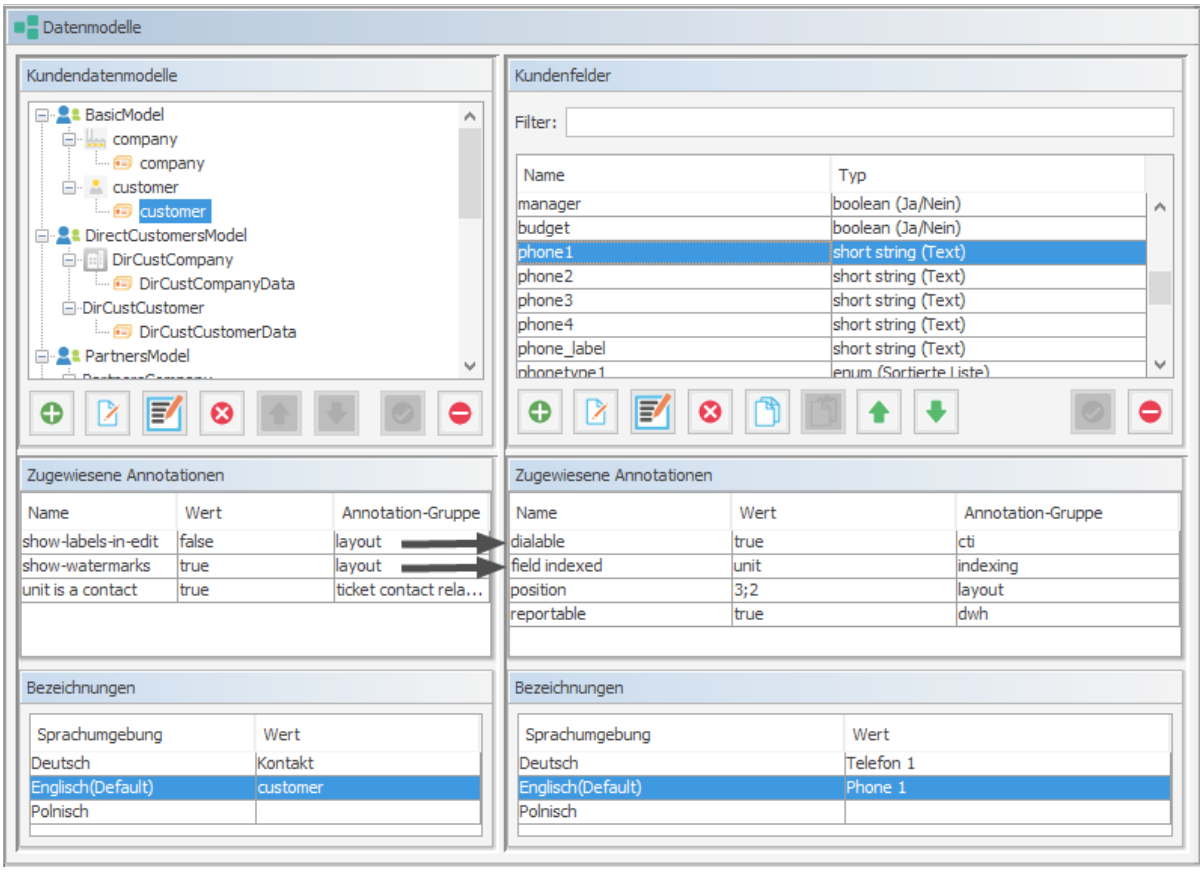


Abbildung 706: *ConSol CM Admin Tool - Kunden, Datenmodelle: Annotationen für Kundenfelder mit Telefonnummern*

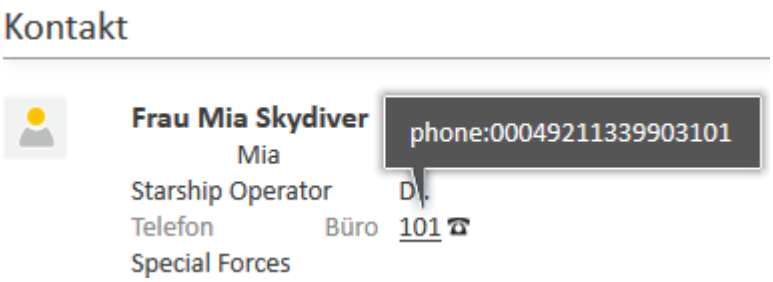


Abbildung 707: *ConSol CM Web Client - Wählbare Nummer bei Verwendung von CM/Phone*

Es sind zwei Annotationen für Kundenfelder, die Telefonnummern enthalten, erforderlich.

- **dialable = „true“**
  Damit werden die Telefonnummern als wählbare Links im Web Client konfiguriert. Diese Einstellung bezieht sich auf ausgehende Anrufe.
- **field-indexed = „local“**
  Damit sind die Felder durchsuchbar, was für die Kundensuche wichtig ist. Diese Einstellung bezieht sich auf eingehende Anrufe.

G.4.3.2 Konfigurieren der Admin-Tool-Templates für Kundendaten für jede Kundengruppe (für eingehende Anrufe)

Die Konfiguration des Kundendatenmodells umfasst zwei CM/Phone-spezifische Templates:

- **CMPhone Kundendetails**
  Für die Anzeige der Kundendaten für einen einzelnen Kunden (Kontakt oder Firma, die Definition kann auf der Kontakt- und/oder Firmenstufe vorgenommen werden).
- **CMPhone Kundenliste**
  Für die Anzeige einer Liste von Kunden (Kontakt oder Firma, die Definition kann auf der Kontakt- und/oder Firmenstufe vorgenommen werden).

Diese werden verwendet, um zu definieren, wie CM/Phone Informationen zu eingehenden Anrufen darstellen soll. Das erste Template wird verwendet, wenn die Telefonnummer bei genau einem Kunden übereinstimmt. Das zweite Template wird verwendet, wenn es mehrere Treffer gibt, sodass der Bearbeiter den gewünschten Kunden auswählen kann.

Sie müssen zwei Schritte durchführen:

1. Schreiben Sie die Templates und speichern Sie sie im Admin Tool unter *Skripte und Templates*.
2. Weisen Sie die Templates den Kundendatenmodellen zu (Navigationsgruppe *Kunden*, Navigationselement *Datenmodelle*).

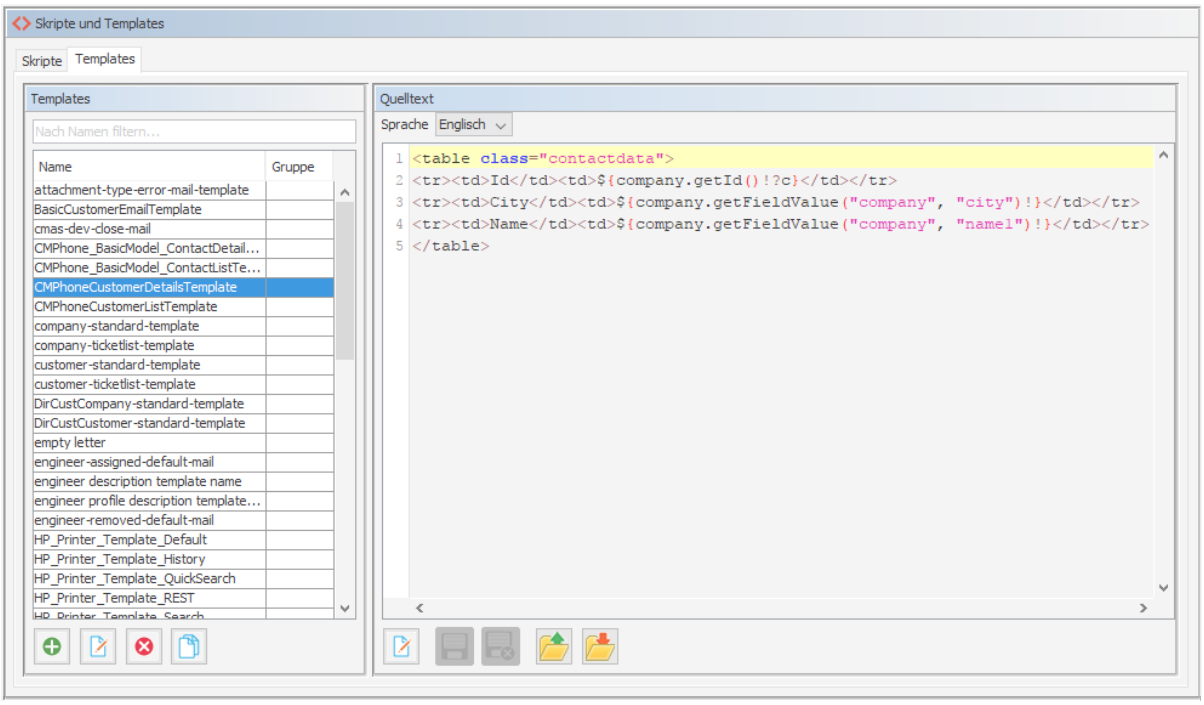


Abbildung 708: *ConSol CM Admin Tool - System, Skripte und Templates: Beispiel-Template für die Darstellung von Kundendaten in CM/Phone*

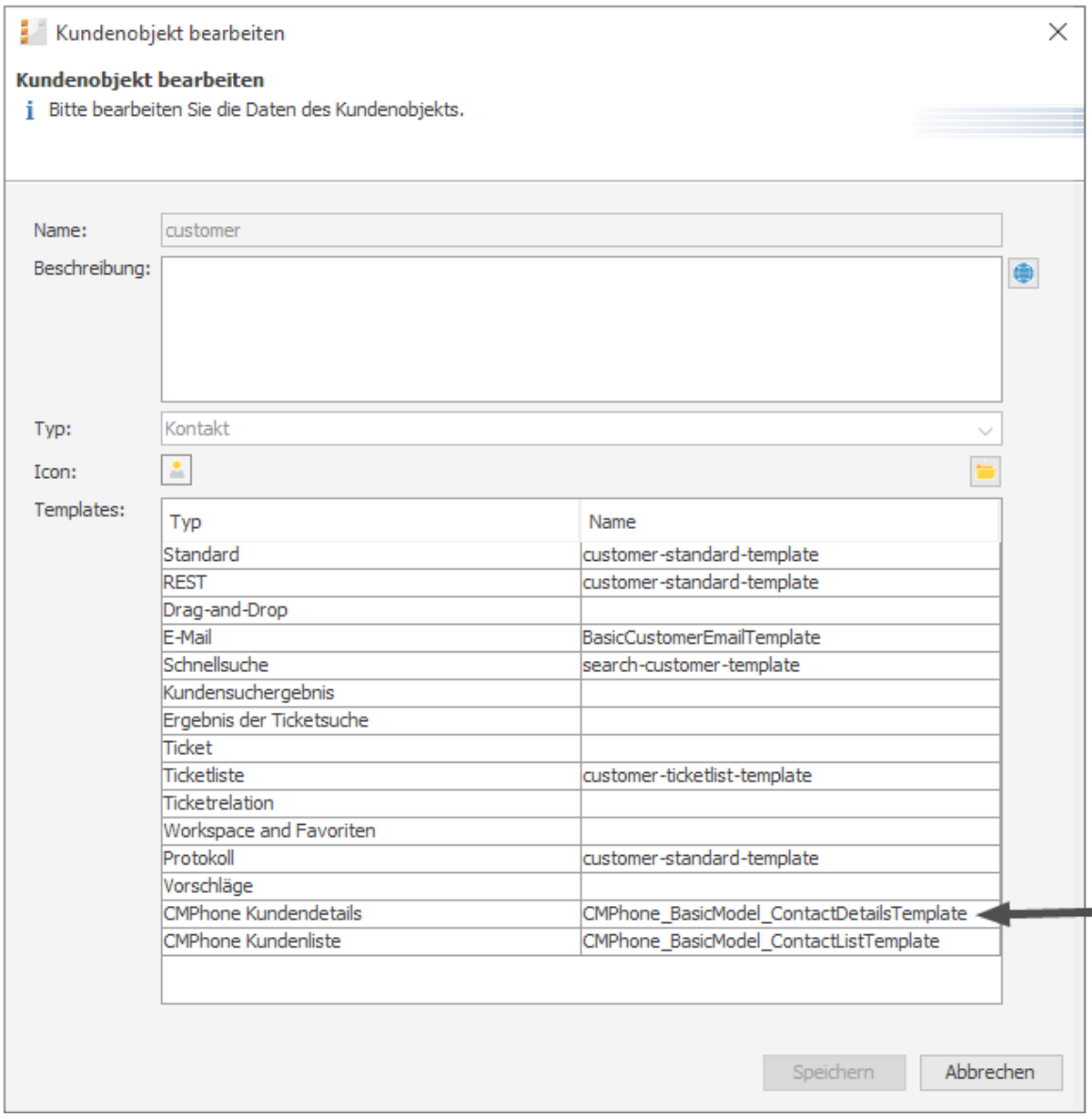


Abbildung 709: *ConSol CM Admin Tool - Kunden, Datenmodelle: Zuweisen von CM/Phone-Templates für Kundendaten zu Kundengruppen*

G.4.3.3 Konfigurieren des Telefonnummernformats für jede einzelne Kundengruppe (für eingehende und ausgehende Anrufe)

Das hier definierte Format wird zur Umwandlung der Telefonnummern (aus der entsprechenden Kundengruppe) in eine gemeinsame kanonische Form verwendet. Der Bearbeiter kann eine Telefonnummer in einem beliebigen Format mit oder ohne Vorwahl eingeben, z. B. als interne Nummer innerhalb der Firma. Um Probleme mit der Interpretation solcher Nummern zu vermeiden, gibt es für jede Kundengruppe eine eigene Konfiguration, die verwendet wird, wenn ein Benutzer eine Tele-

fonnummer für einen bestimmten Kunden eingibt. Die Muster/Elemente für die unterschiedlichen Formate, die als Telefonnummer in den mit `dialable` annotierten Feldern interpretiert werden können, können im Admin Tool genau definiert werden.

Rufen Sie für diese Konfiguration die Navigationsgruppe *Kunden*, Navigationselement *Kundengruppen* auf, nachdem Sie sich im Admin Tool angemeldet haben. Im Navigationselement *Kundengruppen* muss die gewünschte Kundengruppe zum Editieren ausgewählt sein. Klicken Sie unter der Liste der Kundengruppen auf den Button *Bearbeiten*, um das Editierfenster zu öffnen. Es hat den neuen Tab *CMPhone*.

Der Tab *CMPhone* des Fensters *Kundengruppe bearbeiten* enthält Felder, in die Sie die Vorwahlen für unterschiedliche Bereiche und die Muster für unterschiedliche Telefonnummerntypen eingeben können.

Die Felder für die Konfigurationswerte sind:

- **Ländervorwahl**
  Die internationale Ländervorwahl zur Erweiterung nationaler Telefonnummern, ohne Präfixe wie "0" oder "+". Diese Präfixe dürfen an dieser Stelle nicht angegeben werden!
  Die Ländervorwahl ist erforderlich, um zu überprüfen, ob ein ausgehender Anruf innerhalb des gleichen Landes stattfindet oder nicht. Einige Telefonanbieter verarbeiten kanonische Nummern für Inlandsanrufe nicht (obwohl die Nummern theoretisch korrekt sind). Die Ländervorwahl muss daher in diesen Fällen von der Nummer abgeschnitten werden.
- **Ortsvorwahl**
  Die lokale Orts-/Gebietsvorwahl zur Erweiterung von lokalen Telefonnummern. Beachten Sie, dass diese ebenfalls keine allgemeinen Präfixe wie „0“ oder „1“ enthält. Der Eintrag für München in Deutschland wäre also „89“, nicht „089“!
- **Firmenanschluss**
  Die Telefonnummer des Unternehmens, so wie sie in (lokalen) Anrufen ohne Durchwahl verwendet wird. Wenn hier eine Durchwahl hinzugefügt wird, kann ein lokaler Anruf von außerhalb des Unternehmens direkt an diese Nebenstelle getätigt werden.
- **Muster Teilnehmeranschluss**
  Dieser reguläre Ausdruck (RegEx) beschreibt ein Nummernmuster, mit dem ermittelt wird, ob die angegebene Nummer ein vollständiger Teilnehmeranschluss (möglicherweise mit Durchwahl) ist, was lokale Anrufe ermöglicht.
- **Muster interne Nummern**
  Der reguläre Ausdruck (RegEx) in diesem Feld definiert das Muster für Durchwahlen, sofern eine Telefondurchwahl eingegeben wird.
- **Muster Mobilnummern**
  Mit diesem regulären Ausdruck (RegEx) wird ermittelt, ob eine eingegebene Nummer eine inländische Mobilfunknummer ist, was Inlandsanrufe an eine Mobilfunknummer ermöglicht.

Beispiel: Für alle Nummern (12, 33990312, 21133990312) aus den obigen Punkten sollte das Ergebnis immer die vollständige kanonische Telefonnummer 4921133990312 sein. Für Mobilfunknummern wird zusätzlich die Ländervorwahl hinzugefügt, sodass das Ergebnis 49600289906 ist. Wenn der Bearbeiter eine vollständige Nummer eingibt, die mit „+“ oder „0“ beginnt, wird die Konfiguration übersprungen, da ConSol CM annimmt, dass keine Nummernumwandlung erforderlich ist.

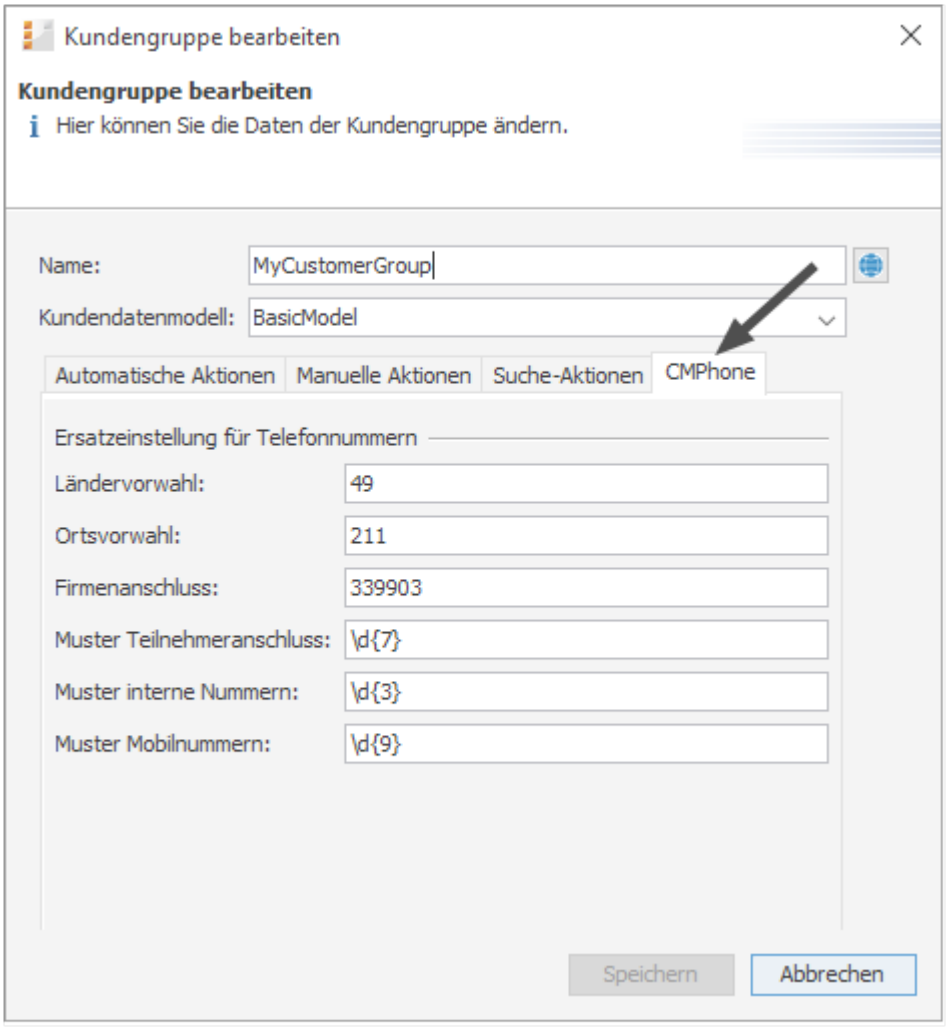


Abbildung 710: *ConSol CM Admin Tool - Kunden, Kundengruppen: Konfiguration der Telefonnummernformate für eine Kundengruppe*

Diese Vorwahlwerte sind Standardwerte für die Erweiterung von nicht vollständig qualifizierten Telefonnummern. Sie können immer durch die Eingabe einer vollständig qualifizierten Telefonnummer außer Kraft gesetzt werden.

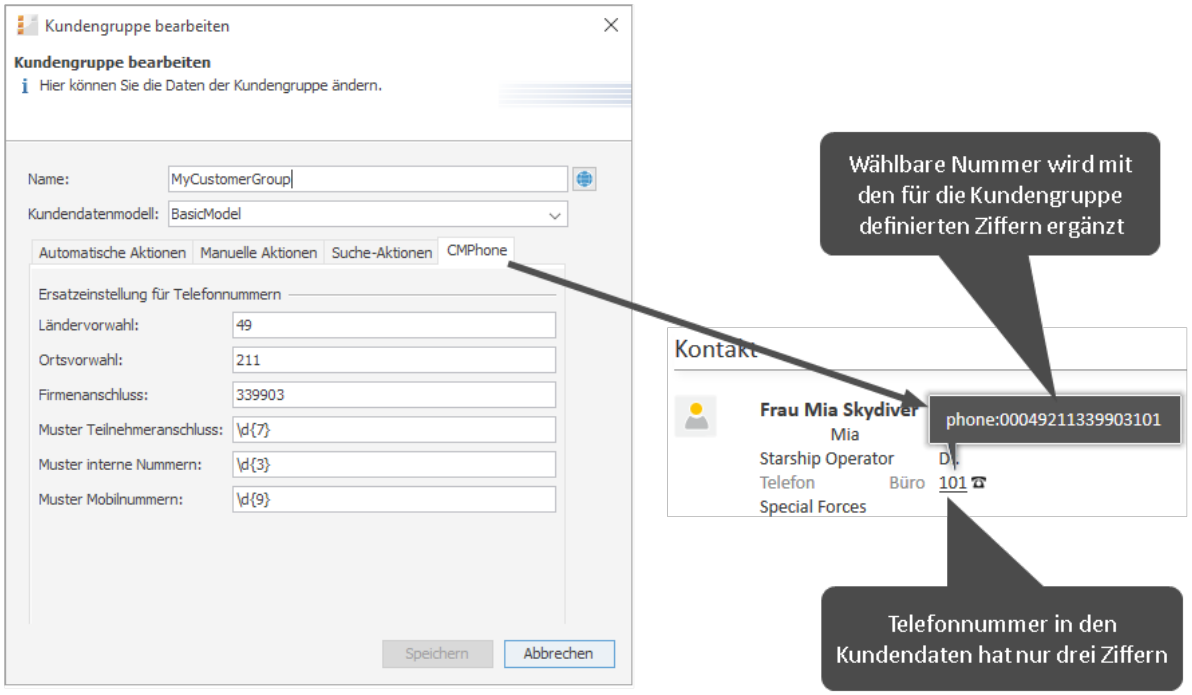


Abbildung 711: *CM/Phone - Verwendung einer kundengruppenspezifischen Konfiguration für die wählbaren Nummern (ausgehende Anrufe)*

Die Muster werden verwendet, um den Typ einer nicht vollständig qualifizierten Telefonnummer zu ermitteln. Der ermittelte Typ bestimmt die Verwendung und die Erweiterungen, die erforderlich sind, um eine Verbindung herzustellen. Deshalb wird nach dem Entfernen der nicht erforderlichen Zeichen zuerst überprüft, ob die Nummer bereits vollständig qualifiziert ist. Andernfalls wird die Nummer gegen diese Muster abgeglichen. Wenn es genau einen Treffer gibt, wird eine gültige Nummer gebildet und verwendet. Wenn es zwei Treffer gibt (Ortsvorwahl und Mobilfunknummer) werden diese mit der Ländervorwahl kombiniert, um eine gültige Nummer erhalten, die dann gewählt wird. In allen anderen Fällen kann die ursprünglich angegebene Nummer nicht zum Aufbau einer Verbindung verwendet werden.

G.4.3.4 Setzen der System-Properties

Es gibt drei neue Properties, die für CM/Phone in ConSol CM relevant sind und im Admin Tool gesetzt werde müssen. Die korrekte Konfiguration dieser Properties ist entscheidend, damit die Telefonnummern zum Herstellen von Verbindungen richtig verwendet werden. Die Properties gehören zum Modul `cmas-core-server`:

- **local.country.prefix**
  String. Dies ist der lokale Ländercode. Der Wert ist ein internationaler Ländercode wie z. B. „49“ für Deutschland. Der Standardwert ist „49“.
- **internal.line.access.prefix**
  Dies ist die Vorwahl, die das Telefonsystem des Unternehmens für externe Anrufe verwendet,

sofern erforderlich. Wenn z. B. eine „0“ oder „9“ gewählt werden muss, um einen Anruf an eine Nummer außerhalb des Unternehmens zu tätigen, muss dieser Wert hier konfiguriert werden. Der Standardwert ist „0“.

- **external.line.access.prefix**
  Die allgemeine Vorwahl, die vor der Ortsvorwahl gewählt wird, um ein Ferngespräch im Land führen zu können. Zum Beispiel muss in Deutschland der Ortsvorwahl eine „0“ vorangestellt werden. Der Standardwert ist „0“.

Alle diese Properties sind optional. Sie müssen also bei Bedarf manuell hinzugefügt werden.

G.4.3.5 Ändern des Präfixes für ausgehende Anrufe

Dieser Schritt ist optional!

Normalerweise wird der Nummer für die Interaktion mit TAPI bei ausgehenden Anrufen das Präfix `phone:` vorangestellt. Wenn ein anderes Präfix (z. B. `tel:`) erforderlich ist, kann dies im ***Windows Registry*** konfiguriert werden. Bitten Sie Ihren ConSol CM-Consultant um Hilfe.

H - Appendix

Dieser Abschnitt enthält mehrere Appendizes:

- Annotationen
- System-Properties
- Administrator-E-Mail-Adressen
- Liste der Code-Beispiele
- Marken
- Glossar
- Index

H.1 Annotationen

Es gibt zwei Arten von Annotationen: Feldannotationen und Gruppenannotationen. Feldannotationen werden auf ein einzelnes Ticket-, Kunden oder Ressourcenfeld angewendet. Gruppenannotationen werden auf eine Ticket-, Kunden oder Ressourcenfeldgruppe angewendet. Siehe:

- Liste der Feldannotationen
- Liste der Gruppenannotationen

H.1.1 Liste der Feldannotationen

personal-data 1116
groupable 1116
sortable 1116
autocomplete-script 1116
leave-trailing-zeros 1117
readonly 1117
visualization 1117
visualize-when-empty 1117
visibility 1118
boolean-type 1118
enum-in-search-type 1118
enum-type 1118
list-type 1119
text-type 1119
ldapid 1120
password 1120
username 1121
dwh-no-history-field 1121
reportable 1121
field indexed 1121
phonetic 1122
colspan 1122
field-group 1122
fieldsize 1122
label-group 1123
label-in-view 1123
order-in-result 1123
position 1123
rowspan 1124
show-label-in-edit 1124
show-label-in-view 1124
show-tooltip 1124

show-watermark 1124
ticket-list-colspan 1125
ticket-list-position 1125
ticket-list-rowspan 1125
no-history-field 1125
dialable 1126
resource-color 1126
customer exposure 1126
contact search result column 1126
contains contacts 1127
enum field with ticket color 1127
accuracy 1127
email 1128
format 1128
matches 1128
maxLength 1128
maxValue 1128
minLength 1129
minValue 1129
required 1129
visibility configuration 1129

H.1.1.1 anonymize (Typ)

personal-data

- **Typ**: anonymize
- **Beschreibung**: Kann Ticket- und Kontaktfeldern zugewiesen werden. Legt fest, dass das Feld persönliche Daten enthält. Kontaktfelder mit dieser Annotation werden gelöscht, wenn der Kontakt anonymisiert wird. Ticketfelder mit dieser Annotation werden gelöscht, wenn der Hauptkunde des Tickets anonymisiert wird.Informationen über das Anonymisieren von Kontakten finden Sie in Beispiel 8: Löschen von Kundendaten.
- **Werte**:
  - *true/false*: Setzen Sie die Annotation auf „true“, wenn das Feld persönliche Daten enthält.

H.1.1.2 cmweb-common (Typ)

groupable

- **Typ**: cmweb-common
- **Beschreibung**: Ermöglicht die Gruppierung nach diesem Feld in der Ticketliste. Eine detaillierte Erklärung dazu finden Sie im Abschnitt Gruppierung.
- **Werte**:
  - *true*: Wird nur für Datenfelder des Typs ENUM verwendet. Entfernen Sie die Annotation, wenn Sie die Gruppierung deaktivieren möchten.

sortable

- **Typ**: cmweb-common
- **Beschreibung**: Ermöglicht die Sortierung nach diesem Feld in der Ticketliste. Eine detaillierte Erklärung dazu finden Sie im Abschnitt Sortierung.
- **Werte**:
  - *true*: Wird für Datenfelder des Typs DATE oder ENUM verwendet. Entfernen Sie die Annotation, wenn Sie die Sortierung deaktivieren möchten.
    Für Felder des Typs ENUM : Funktioniert nur, wenn alle Werte des ENUM-Feldes indiziert werden.

H.1.1.3 common (Typ)

autocomplete-script

- **Typ**: common
- **Beschreibung**: Gibt das Skript (Typ *Text-Autovervollständigung*) an, das für skriptbasierte Autocomplete-Felder verwendet wird.
- **Werte**: <Name des Skripts>

Hinweis:

Diese Annotation gehört zur alten Methode zur Konfiguration von Autocomplete-Feldern, die nur in ConSol CM-Versionen bis 6.11.1.0 verwendet werden sollte. Ab Version 6.11.1.1 ist eine neue Methode zur Implementierung von skriptbasierten Autocomplete-Listen verfügbar, siehe Abschnitt *Skriptbasierte Autocomplete-Listen* im *ConSol CM Administratorhandbuch*.

Vorhandene Felder, die die alte Methode verwenden, bleiben in Version 6.11.1.1 funktionsfähig. Diese Felder werden bei einer Aktualisierung nicht verändert. Die vorher eingesetzte Methode `onEditDisplayEntered` funktioniert für vorhandene Felder weiterhin. Da sie nicht mehr benötigt wird, wird sie in einer zukünftigen Version von ConSol CM entfernt.

leave-trailing-zeros

- **Typ**: common
- **Beschreibung**: Wird für die Anzeige von Festkommazahlen verwendet.
- **Werte**:
  - *true* / *false*: Nullen am Ende der Nachkommastellen werden nicht abgeschnitten, wenn der Wert „true “ist.

readonly

- **Typ**: common
- **Beschreibung**Zeigt an, dass das Ticketfeld nicht verändert werden kann.
- **Werte**:
  - *true* / *false*: Das Feld ist schreibgeschützt, wenn der Wert auf „true“ gesetzt ist. Wenn kein Wert oder ein anderer Wert als „false“ gesetzt ist, wird „true“ angenommen.

visualization

- **Typ**: common
- **Beschreibung**: Gibt das Skript (Typ *Feldvisualisierung*) an, mit dem der Inhalt eines Datenfeldes dargestellt wird. Eine detaillierte Erklärung dazu finden Sie im Abschnitt *Skripte des Typs Feld-Visualisierung* .
- **Werte**: <Name des Skripts>

visualize-when-empty

- **Typ**: common
- **Beschreibung**: Legt fest, ob das Visualisierungsskript eines Datenfeldes mit der Annotation `visualization` auch ausgeführt werden soll, wenn das Feld leer ist. Standardmäßig wird das Visualisierungsskript nur ausgeführt, wenn das Feld einen Wert enthält.

- **Werte**:
  - *true / false*: Setzen Sie die Annotation auf „true“, wenn das Feldvisualisierungsskript ausgeführt werden soll, auch wenn das Feld leer ist.



Diese Annotation wird nur auf die skriptbasierte Feldvisualisierung im Web Client angewendet, nicht in CM/Track V2.

visibility

- **Typ**: common
- **Beschreibung**: Definiert, wann das Feld sichtbar ist.
- **Werte**:
  - *edit*: Das Feld wird im Bearbeitungsmodus angezeigt.
  - *view*: Das Feld wird im Ansichtsmodus angezeigt.
  - *none*: Das Feld ist nicht sichtbar.
  - Wenn ein anderer Wert oder kein Wert gesetzt ist, ist das Feld immer sichtbar.

H.1.1.4 component-type (Typ)

boolean-type

- **Typ**: component-type
- **Beschreibung**: Definition des Layouts eines Boolean-Felds.
- **Werte**:
  - *checkbox* (Standardwert): Das Feld kann markiert werden (standardmäßig auf „false “gesetzt).
  - *radio*: 2 Radio-Buttons (ja/nein) zur Auswahl (es kann nur einer aktiv sein).
  - *select*: Drop-down-Feld mit 2 Werten (ja/nein).

enum-in-search-type

- **Typ**: component-type
- **Beschreibung**: Definiert, ob für ein Feld des Datentyps ENUM bei einer Suche über mehrere Werte gesucht werden kann.
- **Werte**:
  - *single* (Standardwert) / *multiple*: Wenn der Wert „multiple“ gesetzt ist, kann über mehrere Werte gesucht werden.

enum-type

- **Typ**: component-type
- **Beschreibung**: Definition des Layouts der Listenansicht

- **Werte**:
  - *select* (Standardwert): Drop-down-Liste für die Auswahl.
  - *radio*: Liste mit Radio-Buttons für die Auswahl (es kann nur eine Option aktiv sein).
  - *autocomplete*: Drop-down-Liste für die Auswahl. Das Feld ist ein Eingabefeld, mit dem die Liste gefiltert werden kann.

list-type

- **Typ**: component-type
- **Beschreibung**: Deaktiviert die Optionen zum Hinzufügen bzw. Löschen bei Datenfeldern des Typs LIST oder STRUCT.
- **Werte**:
  - *fixed-size*: Es ist nicht möglich, Felder/Zeilen zu löschen oder hinzuzufügen.
  - *non-shrinkable*: Es ist nicht möglich, Felder/Zeilen zu löschen.
  - *non-growable*: Es ist nicht möglich, Felder/Zeilen hinzuzufügen.

text-type

- **Typ**: component-type
- **Beschreibung**: Definiert mögliche Arten eines Felds des Datentyps STRING .
- **Werte**:
  - *text* (Standardwert): Einzeiliges Eingabefeld
  - *textarea*: Mehrzeiliges Eingabefeld
  - *password*: Eingabefeld für Passwörter.
    Das Passwort wird im Ansichtsmodus als ******* angezeigt.
  - *label*: Eingabe wird als Label angezeigt, d.h. das Feld wird nur angezeigt, es ist keine Dateneingabe möglich.
  - *autocomplete*: Das Feld wird als Autocomplete-Liste verwendet. Eine detaillierte Erklärung dazu finden Sie im Abschnitt Skriptbasierte Autocomplete-Listen.
  - *url*: Eingabe wird im Anzeige-Modus als Link dargestellt. Der String muss dafür diesem URL-Muster entsprechen:
    "^((?:mailto\:|(?:(?:ht|f)tps?)\://)1\S+)(?: (?:\|)?(.*))?$"
    Beispiel: "http://consol.de ConSol"
  - *file-url*: Die Eingabe wird als Link auf eine Datei im Dateisystem angezeigt. Der Webbrowser muss solche Links zulassen/unterstützen! Informationen dazu finden Sie im Abschnitt Details über String-Felder: Verwenden von Annotationen zum Anpassen von Strings. Dieser Link wird auch als Tooltip angezeigt.

    Die URL ist korrekt formatiert, wenn folgende Bedingungen erfüllt sind:

    Sie beginnt mit „file:“ gefolgt von normalen Schrägstrichen:

- drei Schrägstriche "///" für Dateien, die sich auf demselben Computer befinden wie der Browser (alternativ "//localhost/") oder
- zwei Schrägstriche gefolgt vom Servernamen und einem weiteren Schrägstrich für Dateien auf Dateiservern, die vom Rechner, auf dem der Webbrowser läuft, erreichbar sind.

Danach folgt der vollständige Dateipfad, der mit dem Dateinamen endet. Auf Microsoft Windows-Systemen wird der Pfad ebenfalls mit normalen Schrägstrichen anstelle von umgekehrten Schrägstrichen geschrieben.

Der Laufwerksbuchstabe eines lokalen Pfads auf Microsoft Windows-Systemen wird wie üblich verwendet, zum Beispiel C:. Pfade mit Leerzeichen und Sonderzeichen wie "{, }, ^, #, ?" müssen auf Microsoft Windows-Systemen mit Prozentzeichen kodiert werden (z. B. mit "%20" für ein Leerzeichen).

Beispiel-URLs:

- file://file-server/path/to/my/file.ext
- file:///linux/local/file.pdf
- file:///C:/Users/myuser/localfile.doc

Siehe auch Erklärung im Abschnitt Details über String-Felder: Verwenden von Annotationen zum Anpassen von Strings.

H.1.1.5 contact authentication (Typ)

ldapid

- **Typ**: contact authentication
- **Beschreibung**: Wird in einer Kundenfeldgruppe des Typs *Kontakt* für das Kundenfeld verwendet, das die LDAP-ID für die Authentifizierung in CM/Track enthält.
- **Werte**: Gibt an, dass das Feld im Authentifizierungsprozess als LDAP-ID verwendet wird. Als Datentyp ist STRING erforderlich.
  Da die Definition auf Kundengruppenebene vorgenommen wird, kann die LDAP-Authentifizierung im gemischten Modus verwendet werden, d. h. es kann für einige Kundengruppen LDAP verwendet werden und für andere Kundengruppen die reguläre Authentifizierung.

password

- **Typ**: contact authentication
- **Beschreibung**: Gibt an, dass das Feld im Authentifizierungsprozess als Passwort verwendet wird.
- **Werte**:
  - *<string>*: Für CM/Track verwendet.

username

- **Typ**: contact authentification
- **Beschreibung**: Gibt an, dass dieses Feld im Authentifizierungsprozess als Anmeldename verwendet wird.
- **Werte**:
  - *true* / *false*: Für CM/Track verwendet.

H.1.1.6 dwh (Typ)

dwh-no-history-field

- **Typ**: dwh
- **Beschreibung**: Die Annotation wird verwendet, um festzulegen, dass das Feld nicht im DWH protokolliert wird.
- **Werte**:
  - *true* / *false*: Seit Version 6.10.2.0.

reportable

- **Typ**: dwh
- **Beschreibung**: Gibt an, dass das Feld für Reports verwendet werden kann und an das DWH übertragen werden soll.
- **Werte**:
  - *true* / *false*: Das Feld kann in Reports verwendet werden, wenn der Wert auf „true“ gesetzt ist.

H.1.1.7 indexing (Typ)

field indexed

- **Typ**: indexing
- **Beschreibung**: Gibt an, dass für das Feld ein Datenbankindex erstellt wird. Wenn es möglich sein soll, Ergebnistabellen (im Web Client) durch Klicken auf die Spaltenüberschrift nach einer Spalte zu sortieren, muss das entsprechende Feld indiziert sein!
- **Werte**:
  - *transitive* (Standardwert): Alle Daten werden angezeigt (Ticketdaten, Kundendaten und Ressourcendaten).
  - *unit*: Für Kundendaten verwendet. Es werden nur die Unit und die Parent Unit (d. h. Firma) als Suchergebnis zurückgegeben, Tickets werden nicht angezeigt.
  - *local*: Für Kundendaten verwendet. Es wird nur die Unit als Suchergebnis zurückgegeben, Firmen und Tickets werden nicht angezeigt.
  - *<Annotation nicht gesetzt>*: Das Feld ist nicht indiziert.

phonetic

- **Typ**: indexing
- **Beschreibung**: Aktiviert die phonetische Suche für dieses Feld. Kann nur für Datenfelder des Typs String verwendet werden (für long oder short string ebenfalls).
- **Werte**: *true*/*false*. Wird automatisch auf „true “gesetzt, wenn die Annotation hinzugefügt wird.

H.1.1.8 layout (Typ)

colspan

- **Typ**: layout
- **Beschreibung**: Legt fest, wie viele Spalten für das Feld im Layout reserviert werden.
- **Werte**:
  - **<Zahl>**: Anzahl der Spalten.

> ⚠ Diese Annotation funktioniert nur, wenn auch die Annotation `position` für das Feld gesetzt ist.

field-group

- **Typ**: layout
- **Beschreibung**: Ermöglicht die Gruppierung der Felder im Ansichtsmodus. Die Annotation wird im Bearbeitungsmodus ignoriert.
- **Werte**:
  - **<string>**: Um Felder zu gruppieren, muss für jedes Feld dieselbe Zeichenfolge als Annotation gesetzt werden. Zwei oder mehr Datenfelder sind verbunden, wenn sie für diese Annotation den gleichen Wert besitzen. Die Gruppe der verbundenen Datenfelder wird nur angezeigt, wenn in allen von ihnen Werte gesetzt sind.
- Entfernt in ConSol CM-Version 6.11.0.1

fieldsize

- **Typ**: layout
- **Beschreibung**: Die angezeigte Feldgröße im Ticketlayout.
- **Werte**:
  - **<Zeilen>;<Spalten>**: Angezeigte Feldgröße.
    Format für Felder des Datentyps STRING und NUMBER: n gibt die Anzahl der Zeichen an. Für String-Felder ist das die Anzahl an Großbuchstaben M in einer Monospace-Schriftart).
    Format für `textarea`: Zeilen;Spalten (entspricht <textarea Zeilen="" Spalten="">).
    Sortierte Listen werden anstatt als Drop-down-Liste als Auswahlfeld mit n Elementen angezeigt. Format für Sortierte Listen: n.
    Hinweis: Diese Annotation dient nur zur Konfiguration des Layouts. Verwenden Sie für

die Validierung `maxlength` vom Typ `group validation`.

- **<Zahl>**: Für Datenfelder des Typs ENUM. Definiert, wie viele Werte im Listenfeld direkt sichtbar sind. Nur zu Layout-Zwecken.

label-group

- **Typ**: layout
- **Beschreibung**: Zeigt eine Gruppe von Feldern im Ansichtsmodus zusammen mit ihrem beschreibenden Label an. Die Annotation wird im Bearbeitungsmodus ignoriert.
- **Werte**:
  - **<string>**: Gibt eine Gruppe Datenfelder zusammen mit ihrem beschreibenden Label an. Die Annotation wird im Ansichtsmodus verwendet und im Bearbeitungsmodus ignoriert. Die Gruppe kann genau ein Label haben (ein Datenfeld des Typs STRING, dem die zusätzliche Annotation `text-type` mit dem Wert „label“ zugewiesen ist). Das Label wird angezeigt, wenn für mindestens ein Datenfeld der Gruppe ein Wert gesetzt ist. Alle Felder mit demselben Label werden gruppiert und unter diesem Label angezeigt. Außerdem muss dem Label die Annotation `label-group` zugewiesen werden.

label-in-view

- **Typ**: layout
- **Beschreibung**: Zeigt den Wert des Datenfelds im Ansichtsmodus als Label an. Die Annotation wird im Bearbeitungsmodus ignoriert.
- **Werte**:
  - *true*: Entfernen Sie die Annotation, wenn das Label im Ansichtsmodus nicht angezeigt werden soll.

order-in-result

- **Typ**: layout
- **Beschreibung**: Zeigt das Feld in der Liste der Suchergebnisse als Spalte an der angegebenen Position an.
- **Werte**:
  - **<Zahl>**: Die Spalten sind aufsteigend sortiert.
    Seit CM-Version 6.0.1. Eine detaillierte Erklärung dazu finden Sie in der Infobox im Abschnitt Konfiguration von Suche und Indexer, order-in-result.

position

- **Typ**: layout
- **Beschreibung**: Definiert die Position eines Feldes im Rasterlayout oder in einer (LIST OF STRUCTS).

- **Werte**:
  - **<Zahl>;<Zahl>**: Die Werte geben die Zeile und Spalte an (Zeile;Spalte), die Nummerierung beginnt bei 0;0. Wenn kein Wert gesetzt ist, wird das Datenfeld in der ersten freien Zelle des Rasters angezeigt.
  - **0;<Zahl>**: Es wird nur der Wert der Spalte verwendet, der Wert der Zeile wird ignoriert.

rowspan

- **Typ**: layout
- **Beschreibung**: Legt fest, wie viele Zeilen das Feld im Layout belegt.
- **Werte**:
  - **<Zahl>**: Anzahl der Zeilen.

⚠ Diese Annotation funktioniert nur, wenn auch die Annotation `position` für das Feld gesetzt ist.

show-label-in-edit

- **Typ**: layout
- **Beschreibung**: Ob das Datenfeld im Bearbeitungsmodus mit Label angezeigt werden soll.
- **Werte**:
  - *true* / *false*: Seit Version 6.9.4.

show-label-in-view

- **Typ**: layout
- **Beschreibung**: Ob das Datenfeld im Ansichtsmodus mit Label angezeigt werden soll.
- **Werte**:
  - *true* / *false*: Seit Version 6.9.4.

show-tooltip

- **Typ**: layout
- **Beschreibung**: Ob das Datenfeld als Tooltip angezeigt werden soll.
- **Werte**:
  - *true* / *false*: Seit Version 6.9.4.

show-watermark

- **Typ**: layout
- **Beschreibung**: Ob das Datenfeld mit Wasserzeichen angezeigt werden soll.
- **Werte**:
  - *true* / *false*: Seit Version 6.9.4.

ticket-list-colspan

- **Typ**: layout
- **Beschreibung**: Definiert wie viele Spalten das Feld im Ticketlistenfeld belegt.
- **Werte**:
  - **<Zahl>**: Anzahl der Spalten.

> ⚠ Diese Annotation funktioniert nur, wenn auch die Annotation `ticket-list-position` für das Feld gesetzt ist.

ticket-list-position

- **Typ**: layout
- **Beschreibung**: Definiert die Position des Feldes im Ticketlistenfeld.
- **Werte**:
  - **<Zahl>;<Zahl>**: Die Werte geben die Zeile und Spalte an (Zeile;Spalte), die Nummerierung beginnt bei 0;0.

ticket-list-rowspan

- **Typ**: layout
- **Beschreibung**: Definiert wie viele Zeilen das Feld im Ticketlistenfeld belegt.
- **Werte**:
  - **<Zahl>**: Anzahl der Zeilen.

> ⚠ Diese Annotation funktioniert nur, wenn auch die Annotation `ticket-list-position` für das Feld gesetzt ist.

H.1.1.9 performance (Typ)

no-history-field

- **Typ**: performance
- **Beschreibung**: Gibt an, dass ein einzelnes Datenfeld nicht protokolliert werden soll. Überschreibt die Gruppenannotation `no-history`.
- **Werte**:
  - *true* / *false*: Die Annotation ist aktiv, wenn der Wert auf „true“ gesetzt ist. Für Felder, die gespeichert aber nicht im Ticketprotokoll angezeigt werden sollen, wird die Annotation `visibility configuration` verwendet.
    In CM-Versionen bis 6.10.2 wird auch die Übertragung der Feldhistorie an das DWH mit dieser Annotation gesteuert.
    Verwenden Sie dazu ab CM-Version 6.10.2 die Annotation `dwh-no-history-field`.

H.1.1.10 phone commander (Typ)

dialable

- **Typ**: phone commander (CM/Phone)
- **Beschreibung**: Definiert ein Feld als Telefonnummer.
- **Werte**:
  - *true*: Für für CM/Phone verwendet. Markiert eine Telefonnummer als automatisch wählbar für ausgehende Anrufe im CTI-System.

H.1.1.11 resource (Typ)

resource-color

- **Typ**: resource
- **Beschreibung**:
- **Werte**:
  - *true* / *false*: Kann einem Feld des Datentyps ENUM mit einer Farbe zugewiesen werden. Die Farbe des ausgewählten Listenwerts wird als Hintergrundfarbe für das Ressourcen-Icon verwendet.

H.1.1.12 restapi (Typ)

customer exposure

- **Typ**: restapi
- **Beschreibung**: Gibt an, ob ein Datenfeld für Kunden über die REST-API verfügbar sein soll, z. B. in CM/Track.
- **Werte**:
  - *full* (Standardwert): Das Datenfeld ist zum Lesen und Schreiben verfügbar.
  - *read*: Das Datenfeld ist nur zum Lesen verfügbar.
  - *none*: Das Datenfeld ist nicht verfügbar. Dieser Wert kann hilfreich sein, wenn zum Beispiel eine ganze Ticketfeldgruppe mit *customer exposure group* = *full* | *read* konfiguriert wurde und einzelne Felder in CM/Track (oder allgemein über die REST-API) nicht verfügbar sein sollen.

H.1.1.13 search result (Typ)

contact search result column

- **Typ**: search result
- **Beschreibung**: Gibt an, ob das Feld standardmäßig in den Suchergebnissen angezeigt werden soll. **Veraltet! Nicht mehr verwenden!** In ConSol CM-Version 6.11 entfernt.

- **Werte**:
  - *true*:
    Entfernen Sie die Annotation, wenn das Feld nicht standardmäßig angezeigt werden soll.

    Seit CM-Version 6.1.3
    (Ersetzt durch `order-in-result`, `contact search result column` ist veraltet!)

H.1.1.14 ticket contact relation type (Typ)

contains contacts

- **Typ**: ticket contact relation type
- **Beschreibung**: Innerhalb eines Datenmodells darf nur eine Instanz dieser Annotation vorhanden sein! Sie wird nur für die Definition von Listenfeldern verwendet und gibt an, dass das Feld Unit-Referenzen zu Kontakten enthalten kann. Ab ConSol CM-Version 6.11 erzeugt das so annotierte Feld keine Einträge im Ticketprotokoll. Auf diese Weise wird die Ladezeit von Tickets, die viele Kunden haben, verringert.
- **Werte**:
  - *true* / *false*: Der Wert ist boolean. Gibt an, ob die Liste innerhalb eines Kontakts („true“) oder eines Tickets („false“) angezeigt wird.

H.1.1.15 ticket display (Typ)

enum field with ticket color

- **Typ**: ticket display
- **Beschreibung**: Legt die Hintergrundfarbe des Ticket-Icons in der Ticketliste und im Ticket fest.
- **Werte**:
  - *true* / *false*: Das Feld muss in den Sortierten Listen vorhanden sein, wo Listen, Werte und Farben definiert werden.

H.1.1.16 validation (Typ)

accuracy

- **Typ**: validation
- **Beschreibung**: Für Ticket-, Kunden und Ressourcenfelder des Typs DATUM, um den Detailgrad der angezeigten Daten zu definieren
- **Werte**:
  - *date* (Standardwert): Datum wird ohne Zeit angezeigt.
  - *date-time*: Datum wird mit Zeit angezeigt.
  - *only-time*: Es wird nur die Zeit nicht aber das Datum angezeigt.

email

- **Typ**: validation
- **Beschreibung**: Verwendet, um bei E-Mail-Adressen zu prüfen, ob das Format korrekt ist, d. h. ob es <name>@<domain> entspricht.
- **Werte**:
  - *true*: Kann für Datenfelder des Typs STRING verwendet werden. Entfernen Sie die Annotation, wenn das Format nicht validiert werden soll.

format

- **Typ**: validation
- **Beschreibung**: Verwendet, um das Format von Datumsfeldern zu validieren.
- **Werte**:
  - **<Datumsformat>**: Das Muster für das Datum basiert auf `SimpleDateFormat`, z. B. dd.MM.yyyy.
    Denken Sie daran, den richtigen Wert für `colspan` einzusetzen, wenn das Format Stunden/Minuten beinhaltet. Eine Formatreferenz finden Sie in http://docs.oracle.com/javase/6/docs/api/java/text/SimpleDateFormat.html.

matches

- **Typ**: validation
- **Beschreibung**: Prüft, ob die Eingabe in Datenfelder des Typs STRING einer angegebenen RegEx entspricht.
- **Werte**:
  - **<string>**: Kann bei Datenfeldern des Typs STRING verwendet werden.

maxLength

- **Typ**: validation
- **Beschreibung**: Definiert die maximale Länge der Eingabe in Datenfelder des Typs STRING.
- **Werte**:
  - **<Zahl>**: Kann für Datenfelder des Typs STRING verwendet werden.

maxValue

- **Typ**: validation
- **Beschreibung**: Definiert den Maximalwert für Datenfelder mit Zahlen.
- **Werte**:
  - **<Zahl>**: Kann für Datenfelder des Typs NUMBER verwendet werden, d. h. NUMBER und FIXED-POINT NUMBER.

minLength

- **Typ**: validation
- **Beschreibung**: Definiert die minimale Länge der Eingabe in Datenfelder des Typs STRING.
- **Werte**:
  - **<Zahl>**: Kann für Datenfelder des Typs STRING verwendet werden.

minValue

- **Typ**: validation
- **Beschreibung**: Definiert den Minimalwert für Datenfelder mit Zahlen.
- **Werte**:
  - **<Zahl>**: Kann für Datenfelder des Typs NUMBER verwendet werden, d. h. NUMBER und FIXED-POINT NUMBER.

required

- **Typ**: validation
- **Beschreibung**: Gibt an, dass es sich um ein Pflichtfeld handelt.
- **Werte**:
  - *true* / *false*: Das Feld ist ein Pflichtfeld, wenn der Wert auf „true“ gesetzt ist. Der Benutzer kann das Ticket nicht speichern, ohne einen Wert in das Pflichtfeld eingegeben zu haben. Im Web Client sind Pflichtfelder mit einem roten Sternchen gekennzeichnet.

H.1.1.17 visibility (Typ)

visibility configuration

- **Typ**: visibility
- **Beschreibung**: Legt die Sichtbarkeit des Felds im Protokoll fest.
- **Werte**:
  - *on every level*: Das Feld wird im Protokoll in jedem Sichtbarkeitslevel angezeigt.
  - *2nd level and 3rd level*: Das Feld wird im 2. und 3. Sichtbarkeitslevel des Protokolls angezeigt.
  - *only 3rd level*: Das Feld wird nur im 3. Sichtbarkeitslevel des Protokolls angezeigt.

H.1.2 Liste der Gruppenannotationen

group-visibility 1131
dwh-no-history 1131
reportable group 1131
auto-open-group 1131
show-contact-in-ticket-list 1132
show-in-group-section 1132
show-labels-in-edit 1132
show-labels-in-view 1132
show-tooltips 1132
show-watermarks 1132
no-history 1133
resource-fields-group-mode 1133
resource-custom-fields-group-mode 1133
customer group exposure 1134
unit is a contact 1134

H.1.2.1 common (Typ)

group-visibility

- **Typ**common
- **Beschreibung**: Definiert die Standardsichtbarkeit einer Datenfeldgruppe.
- **Werte**:
  - *true* / *false*: Die Annotation kann auf Feldebene überschrieben werden.

H.1.2.2 dwh (Typ)

dwh-no-history

- **Typ**: dwh
- **Beschreibung**: Gibt an, dass alle Felder dieser Gruppe nicht im DWH protokolliert werden.
- **Werte**:
  - *true* / *false*: Seit Version 6.10.2.0

reportable group

- **Typ**: dwh
- **Beschreibung**: Gibt an, dass alle Datenfelder, die zu dieser Gruppe gehören, in Reports verwendet werden können und an das CMRF übertragen werden sollen.
- **Werte**:
  - *true* / *false*: Es muss ein Wert gesetzt werden. Die Annotation ist aktiv, wenn der Wert auf „true“ gesetzt wird.

H.1.2.3 layout (Typ)

auto-open-group

- **Typ**: layout
- **Beschreibung**: Die Gruppe ist initial geöffnet. Es kann mehr als ein Wert als durch Kommas oder Semikolons getrennte Liste angegeben werden (kann für die Kundenannotation verwendet werden).
- **Werte**:
  - *ticket:create*: Die Gruppe ist initial geöffnet, wenn ein neues Ticket erstellt wird.
  - *customer:create*: Die Gruppe ist initial geöffnet, wenn ein neuer Kunde erstellt wird.
  - *customer:view*: Die Gruppe ist initial geöffnet, wenn die Kundenseite (Kontakt- oder Firmenseite) geöffnet wird.

show-contact-in-ticket-list

- **Typ**: layout
- **Beschreibung**: Veraltet! Verwenden Sie die Seitenanpassung! accordionTicketList.mainCustomerDescriptionVisible={true, false}
- **Werte**: veraltet

show-in-group-section

- **Typ**: layout
- **Beschreibung**: Legt fest, dass eine Datenfeldgruppe im Bereich für Detaildaten (als Tab) angezeigt wird.
- **Werte**:
  - *true* / *false*: Ohne diese Annotation wird die Gruppe ohne Tab im Ticket-, Kunden- oder Ressourcenbereich angezeigt.

show-labels-in-edit

- **Typ**: layout
- **Beschreibung**: Ob die Datenfelder dieser Gruppe im Bearbeitungsmodus mit Labeln angezeigt werden sollen.
- **Werte**:
  - *true* / false: Seit Version 6.9.4.

show-labels-in-view

- **Typ**: layout
- **Beschreibung**: Ob die Datenfelder dieser Gruppe im Ansichtsmodus mit Labeln angezeigt werden sollen.
- **Werte**:
  - *true* / *false*: Seit Version 6.9.4.

show-tooltips

- **Typ**: layout
- **Beschreibung**: Ob die Datenfelder dieser Gruppe mit Tooltips angezeigt werden sollen.
- **Werte**:
  - *true* / *false*: Seit Version 6.9.4.

show-watermarks

- **Typ**: layout
- **Beschreibung**: Ob die Datenfelder dieser Gruppe mit Wasserzeichen angezeigt werden sollen.
- **Werte**:
  - *true* / *false*: Seit Version 6.9.4.

H.1.2.4 performance (Typ)

no-history

- **Typ**: performance
- **Beschreibung**: Legt fest, dass alle Datenfelder, die zu dieser Gruppe gehören, nicht protokolliert werden.
- **Werte**:
  - *true* / *false*: Legt fest, dass alle Datenfelder, die zu dieser Gruppe gehören, nicht protokolliert werden sollen. Mögliche Werte sind „true“, wenn die Annotation aktiv sein soll, oder „false“, was dem Entfernen der Annotation entspricht. Verwenden Sie diese Annotation, wenn Sie verhindern möchten, dass ein Protokoll für alle/viele Felder der Gruppe gespeichert wird. Wenn Sie nur die Protokollierung einzelner Felder verhindern möchten, können Sie die Annotation `no-history-field` auf Feldebene setzen.
    In CM-Versionen bis 6.10.2 wird auch die Übertragung der Feldhistorie an das DWH mit dieser Annotation gesteuert.
    Verwenden Sie dazu ab CM-Version 6.10.2 die Annotation `dwh-no-history`.

H.1.2.5 resource (Typ)

resource-fields-group-mode

- **Typ**: resource
- **Beschreibung**: Steuert den Modus einer Ressourcenfeldgruppe beim Editieren mit dem Web Client.
- **Werte**:
  - *internal* / *external*: Mögliche Werte: „internal“, „external“. Ein Ressourcenfeld kann nicht im Web Client editiert werden, wenn der Wert „external “ist.
    Seit 6.10.4.0
    Entfernt in 6.10.5.0

resource-custom-fields-group-mode

- **Typ**: resource
- **Beschreibung**: Steuert den Modus einer Ressourcenfeldgruppe beim Editieren mit dem Web Client.
- **Werte**:
  - *internal* / *external*: Mögliche Werte: „internal“, „external“. Ein Ressourcenfeld kann nicht im Web Client editiert werden, wenn der Wert „external “ist.
    Seit 6.10.5.0

H.1.2.6 restapi (Typ)

customer group exposure

- **Typ**: restapi
- **Beschreibung**: Gibt an, ob eine Datenfeldgruppe für Kunden über die REST-API verfügbar sein soll, z. B. in CM/Track.
- **Werte**:
  - *full* (Standardwert): Die Datenfeldgruppe ist zum Lesen und Schreiben verfügbar.
  - *read*: Die Datenfeldgruppe ist nur zum Lesen verfügbar.

H.1.2.7 ticket contact relation (Typ)

unit is a contact

- **Typ**: ticket contact relation
- **Beschreibung**: veraltet
- **Werte**:
  - *true* / *false*: Entfernt in Version 6.9.0.

H.2 System-Properties

Das folgende Kapitel enthält detaillierte Informationen über die in ConSol CM verwendeten System-Properties.

- Alphabetische Liste der System-Properties
- Liste der System-Properties nach Modul
- Liste der System-Properties nach Bereich

H.2.1 Alphabetische Liste der System-Properties

admin.email 1147
admin.login 1147
admin.tool.consumed.licences.check.interval 1147
admin.tool.consumed.licences.pool.name 1148
admin.tool.session.check.interval 1148
attachment.allowed.types 1148
attachment.max.size 1149
attachment.upload.timeout 1149
authentication.method 1149
autocommit.cf.changes 1150
autocomplete.enabled 1150
automatic.booking.enabled 1150
batch-commit-interval 1151
big.task.minimum.size 1151
cache-cluster-name 1151
calendar.csv.dateFormat 1152
calendar.csv.separator 1152
checkUserOnlineIntervalInSeconds 1152
cluster.mode 1153
cluster.unicast 1153
cmas.dropSchemaBeforeSetup 1153
cmoffice.enabled 1153
cmoffice.oo.path.NUMBER 1154
cmoffice.strict.versioning.enabled 1154
comment.authors.disabled 1154
commentRequiredForTicketCreation 1155
communication.channel 1155
config.data.version 1155
config.import.global.transaction.enabled 1156
connection.release.mode 1156
contact.authentication.method 1156
contact.inherit.permissions.only.to.own.customer.group 1157

csrf.domain.white.list 1157
csrf.request.filter.enabled 1157
customizationVersion 1157
dao.log.threshold.milliseconds 1158
dao.log.username 1158
data.directory 1158
data.optimization 1159
database.notification.enabled 1159
database.notification.redelivery.delay.seconds 1159
database.notification.redelivery.max.attempts 1160
defaultAttachmentEntryClassName 1160
defaultCommentClassName 1160
defaultContentEntryClassName 1161
defaultIncommingMailClassName 1161
defaultNumberOfCustomFieldsColumns 1161
defaultOutgoingMailClassName 1161
delete.ticket.enabled 1162
diffTrackingEnabled 1162
diffTrackingEnabledForUnitAndResource 1163
diff.tracking.disabled 1163
disable.admin.task.auto.commit 1163
dwh.administration.refresh.interval.seconds 1163
dwh.mode 1164
engineer.description.cache.enabled 1164
engineer.description.mode 1164
engineer.description.template.name 1165
eviction.event.queue.size 1165
eviction.max.nodes 1165
eviction.wakeup.interval 1166
expert.mode 1166
external.line.access.prefix 1166
favoritesSizeLimit 1167
fetchLock.interval 1167

fetchSize.strategy 1167
fetchSize.strategy.FetchSizeFixedStrategy.value 1167
fetchSize.strategy.FetchSizePageBasedStrategy.limit 1168
fetchSize.strategy.FetchSizeThresholdStrategy.value 1168
filesystem.polling.threads.number 1168
filesystem.polling.threads.shutdown.timeout.seconds 1169
filesystem.polling.threads.watchdog.interval.seconds 1169
filesystem.task.enabled 1169
filesystem.task.interval.seconds 1170
filesystem.task.polling.folder 1170
filesystem.task.timeout.seconds 1170
filesystem.task.transaction.timeout.seconds 1171
forward.mails.to.representatives 1171
globalSearchResultSizeLimit 1171
heartbeat 1172
helpFilePath 1172
hibernate.dialect 1172
hideTicketSubject 1173
ignore-queues 1173
index.attachment 1173
index.history 1174
index.status 1174
index.task.worker.threads 1174
index.version.current 1175
index.version.newest 1175
indexed.assets.per.thread.in.memory 1175
indexed.engineers.per.thread.in.memory 1175
indexed.resources.per.thread.in.memory 1176
indexed.tickets.per.thread.in.memory 1176
indexed.units.per.thread.in.memory 1176
initialized 1177
internal.line.access.prefix 1177
is.cmrf.alive 1177

java.naming.factory.initial 1178
java.naming.factory.url.pkgs 1178
java.naming.provider.url 1178
jobExecutor.adminMail 1179
jobExecutor.idleInterval 1179
jobExecutor.idleInterval.seconds 1179
jobExecutor.jobExecuteRetryNumber 1180
jobExecutor.jobMaxRetries 1180
jobExecutor.jobMaxRetriesReachedSubject 1180
jobExecutor.lockingLimit 1181
jobExecutor.lockTimeout.seconds 1181
jobExecutor.mailFrom 1181
jobExecutor.maxInactivityInterval.minutes 1182
jobExecutor.threads 1182
jobExecutor.timerRetryInterval 1182
jobExecutor.timerRetryInterval.seconds 1183
jobExecutor.txTimeout.seconds 1183
kerberos.v5.enabled 1183
kerberos.v5.username.regex 1184
last.config.change 1184
last.config.change.templates 1184
last.ping.timestamp 1184
ldap.authentication 1185
ldap.basedn 1185
ldap.certificate.basedn 1185
ldap.certificate.content.attribute 1186
ldap.certificate.password 1186
ldap.certificate.providerurl 1186
ldap.certificate.searchattr 1187
ldap.certificate.userdn 1187
ldap.contact.name.basedn 1187
ldap.contact.name.password 1187
ldap.contact.name.providerurl 1188

ldap.contact.name.searchattr 1188
ldap.contact.name.userdn 1188
ldap.initialcontextfactory 1188
ldap.password 1189
ldap.providerurl 1189
ldap.searchattr 1189
ldap.userdn 1190
live.start 1190
local.country.prefix 1190
mail.attachments.validation.info.sender 1191
mail.attachments.validation.info.subject 1191
mail.db.archive 1191
mail.encryption 1192
mail.error.from.address 1192
mail.error.to.address 1192
mail.from 1192
mail.notification.engineerChange 1193
mail.notification.sender 1193
mail.on.error 1193
mail.process.error 1194
mail.redelivery.retry.count 1194
mail.reply.to 1194
mail.sender.address 1195
mail.smtp.email 1195
mail.smtp.envelopesender 1195
mail.smtp.tls.enabled 1196
mail.ticketname.pattern 1196
mailbox.1.connection.host 1196
mailbox.1.connection.password 1196
mailbox.1.connection.port 1196
mailbox.1.connection.protocol 1197
mailbox.1.connection.username 1197
mailbox.2.connection.host 1197

mailbox.2.connection.password 1197
mailbox.2.connection.port 1197
mailbox.2.connection.protocol 1197
mailbox.2.connection.username 1197
mailbox.default.connection.host 1198
mailbox.default.connection.password 1198
mailbox.default.connection.port 1198
mailbox.default.connection.protocol 1198
mailbox.default.connection.username 1199
mailbox.default.session.mail.debug 1199
mailbox.default.session.mail.imap.timeout 1199
mailbox.default.session.mail.mime.address.strict 1200
mailbox.default.session.mail.pop3.timeout 1200
mailbox.default.session.mail.PROTOCOL.fetchsize 1200
mailbox.default.session.mail.PROTOCOL.partialfetch 1201
mailbox.default.task.delete.read.messages 1201
mailbox.default.task.enabled 1201
mailbox.default.task.interval.seconds 1202
mailbox.default.task.max.message.size 1202
mailbox.default.task.max.messages.per.run 1202
mailbox.default.task.timeout.seconds 1203
mailbox.default.task.transaction.timeout.seconds 1203
mailbox.polling.threads.mail.log.enabled 1203
mailbox.polling.threads.number 1204
mailTemplateAboveQuotedText 1204
max.licences.perUser 1204
maxSizePerPagemapInMegaBytes 1205
monitoring.engineer.login 1205
monitoring.unit.login 1205
nimh.enabled 1206
notification.error.description 1206
notification.error.from 1206
notification.error.subject 1207

notification.error.to 1207
notification.finished_successfully.description 1207
notification.finished_successfully.from 1207
notification.finished_successfully.subject 1208
notification.finished_successfully.to 1208
notification.finished_unsuccessfully.description 1208
notification.finished_unsuccessfully.from 1209
notification.finished_unsuccessfully.subject 1209
notification.finished_unsuccessfully.to 1209
notification.host 1209
notification.password 1210
notification.port 1210
notification.protocol 1210
notification.tls.enabled 1211
notification.username 1211
number.of.tasks 1211
outdated.lock.age 1211
pagemapLockDurationInSeconds 1212
password.reset.mail.from 1212
policy.password.age 1212
policy.password.pattern 1213
policy.rotation.ratio 1213
policy.track.username.case.sensitive 1213
policy.username.case.sensitive 1214
postActivityExecutionScriptName 1214
queue.polling.threads.number 1214
queue.polling.threads.shutdown.timeout.seconds 1215
queue.polling.threads.watchdog.interval.seconds 1215
queue.task.error.pause.seconds 1215
queue.task.interval.seconds 1216
queue.task.max.retries 1216
queue.task.timeout.seconds 1216
queue.task.transaction.timeout.seconds 1217

queuesExcludedFromGS 1217
recent.items.cleanup.cluster.node.id 1217
recent.items.cleanup.interval.minutes 1217
recent.items.max.per.engineer 1218
recent.items.persistence.enabled 1218
recoverable.exceptions 1218
refreshTimeInCaseOfConcurrentRememberMeRequests 1219
rememberMeLifetimeInMinutes 1219
request.scope.transaction 1219
resetCode.expiriationPeriod 1220
resource.replace.batchSize 1220
resource.replace.timeout 1220
scene 1221
script.logging.threshold.seconds 1221
script.validation.interval.seconds 1221
searchPageSize 1222
searchPageSizeOptions 1222
security.fields.customer.exposure.check.enabled 1222
security.restrict.unit.access.to.own.data 1222
serial.mods.tracking.enabled 1223
server.session.archive.reaper.interval 1223
server.session.archive.timeout 1224
server.session.reaper.interval 1224
server.session.timeout 1224
serverPoolingInterval 1225
skip-ticket 1225
skip-ticket-history 1226
skip-unit 1226
skip-unit-history 1226
skip.wfl.transfer.cleanup 1227
skip.wfl.transfer.translations.cleanup 1227
split.history 1227
start.groovy.task.enabled 1228

statistics.calendar 1228
statistics.client.group 1228
statistics.contact.role 1228
statistics.content.entry 1229
statistics.content.entry.class 1229
statistics.content.entry.history 1229
statistics.customer.definition 1230
statistics.engineer 1230
statistics.enum.group 1230
statistics.field.definition 1230
statistics.group.definition 1231
statistics.locale 1231
statistics.localized.property 1231
statistics.mla 1232
statistics.project 1232
statistics.queue 1232
statistics.resource 1232
statistics.resource.group 1233
statistics.resource.history 1233
statistics.resource.relation.definition 1233
statistics.resource.type 1234
statistics.ticket 1234
statistics.ticket.function 1234
statistics.ticket.history 1234
statistics.time.booking 1235
statistics.timestamp 1235
statistics.unit 1235
statistics.unit.history 1236
statistics.unit.relation.definition 1236
statistics.workflow 1236
strict.utf.bmp.enabled 1236
supportEmail 1237
synchronize.master.address 1237

synchronize.master.security.token 1238
synchronize.master.security.user 1238
synchronize.master.timeout.minutes 1238
synchronize.megabits.per.second 1239
synchronize.sleep.millis 1239
task.execution.interval.seconds 1239
task.execution.node.id 1239
task.panel.refresh.interval.seconds 1240
themeOverlay 1240
ticket.delete.timeout 1240
ticket.from.incoming.message.accepted.links 1241
ticketListRefreshIntervalInSeconds 1241
ticketListSizeLimit 1242
tickets.delete.size 1242
time.buffer 1242
transaction.timeout.minutes 1242
tx.read.only.mode.enabled 1243
unit.description.mode 1243
unit.replace.batchSize 1244
unit.replace.timeout 1244
unit.transfer.order 1244
unitIndexSearchResultSizeLimit 1245
unused.content.remover.cluster.node.id 1245
unused.content.remover.enabled 1245
unused.content.remover.polling.minutes 1246
unused.content.remover.ttl.minutes 1246
update.6.11.0.0.sleep 1246
update.6.11.0.0.timezone 1247
urlLogoutPath 1247
voCacheEnabled 1247
warmup.executor.enabled 1248
webSessionTimeoutInMinutes 1248
wfl.sticky.transfer.disabled 1249

wicketAjaxRequestHeaderFilterEnabled 1249
X-Frame-Options 1249

admin.email

- **Modul**: cmas-core-security
- **Beschreibung**: Die E-Mail-Adresse des ConSol CM-Administrators. Anfänglich wird hier der Wert genommen, den Sie bei der Systemeinrichtung eingegeben haben.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: myuser@consol.de
- **Seit**: 6.0

admin.login

- **Modul**: cmas-core-security
- **Beschreibung**: Der Name des ConSol CM-Administrators. Anfänglich wird hier der Wert genommen, den Sie bei der Systemeinrichtung eingegeben haben.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: admin
- **Seit**: 6.0

admin.tool.consumed.licences.check.interval

- **Modul**: cmas-app-admin-tool
- **Beschreibung**: Setzt das Intervall (in Sekunden) zum Überwachen der Anzahl der verwendeten Lizenzen. Der Standardwert ist 30.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 30
- **Seit**: 6.11.0.0

admin.tool.consumed.licences.pool.name

- **Modul**: cmas-app-admin-tool
- **Beschreibung**: Setzt den Namen des Lizenzpools zum Überwachen der verwendeten Lizenzen. Der Standardwert ist „CONCURRENT_USERS“.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: CONCURRENT_USERS
- **Seit**: 6.11.0.0

admin.tool.session.check.interval

- **Modul**: cmas-app-admin-tool
- **Beschreibung**: Intervall, in dem inaktive (beendete) Sitzungen im Admin Tool überprüft werden (in Sekunden).
- **Typ**: integer
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 30
- **Seit**: 6.7.5

attachment.allowed.types

- **Modul**: cmas-core-server
- **Beschreibung**: Durch Kommas getrennte Liste der erlaubten Dateinamenserweiterungen (wenn kein Werte definiert ist, sind alle Dateinamenserweiterungen erlaubt).
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: txt,zip,doc
- **Seit**: 6.5.0

attachment.max.size

- **Modul**: cmas-core-server
- **Beschreibung**: Maximale Größe von Attachments in MB. Dies ist eine Validierungs-Property der CM-API. Sie steuert die Größe der Attachments in Tickets, Units und Ressourcen. Außerdem steuert sie die Größe der eingehenden (nicht ausgehenden!) E-Mail-Attachments.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 100
- **Seit**: 6.4.0

attachment.upload.timeout

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Definiert das Transaktions-Timeout in Minuten für das Hinzufügen von Attachments zu einem Ticket, einer Ressource oder einem Kunden. Dabei zählt die Zeit für den Upload aller Attachments einer Transaktion. Wenn das Timeout eintritt, werden alle Dateien, die temporär auf dem Server gespeichert waren, gelöscht. Es wird keine Datei hochgeladen.
- **Typ**: Integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 3
- **Seit**: 6.10.5.3

authentication.method

- **Modul**: cmas-core-security
- **Beschreibung**: Methode der Bearbeiter-Authentifizierung für den Web Client (interne CM-Datenbank oder LDAP-Authentifizierung). Mögliche Werte sind LDAP oder DATABASE.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: DATABASE
- **Seit**: 6.0

autocommit.cf.changes

- **Modul**: cmas-dwh-server
- **Beschreibung**: Definiert, ob DWH-Aufgaben, die aufgrund von Konfigurationsänderungen an Ticketfeldern anfallen, automatisch ohne manuelle Interaktion im Admin Tool ausgeführt werden. Diese Property kann auch im Admin Tool im Navigationselement *DWH* gesetzt werden. Der Standardwert und empfohlene Wert ist *false*.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.7.0

autocomplete.enabled

- **Modul**: cmas-app-admin-tool
- **Beschreibung**: Wenn diese System-Property fehlt oder der Wert „false“ ist, wird das Navigationselement *Adress-Vervollständigung* im Admin Tool ausgeblendet.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.9.2.0

automatic.booking.enabled

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Wenn aktiviert, wird die Zeit für das Erstellen eines Kommentars oder einer E-Mail gemessen und eine automatische Zeitbuchung hinzugefügt.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.9.4.2

batch-commit-interval

- **Modul**: cmas-dwh-server
- **Beschreibung**: Anzahl der Objekte in einer JMS-Nachricht. Höhere Werte bedeuten eine bessere Übertragungsperformance und größeren Speicherverbrauch.
  Ab ConSol CM-Version 6.11 wird diese Property nur verwendet, wenn die Paketgröße einer DWH-Operation nicht gesetzt ist. Dies kann nur der Fall sein, wenn der Befehl direkt an die Java MBean `consol.cmas.global.dwh.synchronizationService` gerichtet ist, z. B. durch Verwendung der Methode `update()`. Wenn eine DWH-Operation über das Admin Tool gestartet wird, gibt es immer einen Wert für die Paketgröße. Wenn dieser nicht explizit gesetzt ist, wird der Standardwert 1000 als Wert für `batch.commit.interval` verwendet.
- **Standardwert**: 1000
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 100
- **Seit**: 6.0.0

big.task.minimum.size

- **Modul**: cmas-core-index-common
- **Beschreibung**: Gibt die Minimalgröße eines Index-Tasks an (in Teilen, jeder Teil hat 100 Einheiten), um diesen Task als einen großen Task zu qualifizieren. Große Tasks haben eine niedrigere Priorität als normale.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 15 (Standardwert)
- **Seit**: 6.8.3

cache-cluster-name

- **Modul**: cmas-core-cache
- **Beschreibung**: Cache-Cluster-Name des JBoss.
- **Typ**: string
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein

- **Beispielwert**: 635a6de1-629a-4129-8299-2d98633310f0
- **Seit**: 6.4.0

calendar.csv.dateFormat

- **Modul**: cmas-core-server
- **Beschreibung**: Format des in der CSV-Datei mit der Liste der Feiertage angegebenen Datums.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: dd/MM/yyyy
- **Seit**: 6.9.3.2

calendar.csv.separator

- **Modul**: cmas-core-server
- **Beschreibung**: Trennzeichen, das in der CSV-Datei mit der Liste der Feiertage verwendet wird.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: ,
- **Seit**: 6.9.3.2

checkUserOnlineIntervalInSeconds

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Das Intervall (in Sekunden), in dem geprüft wird, welche Benutzer online sind (Standard 180 Sekunden = 3 Minuten).
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 180
- **Seit**: 6.0
- **Entfernt in**: 6.5 / 6.11.0.1

cluster.mode

- **Modul**: cmas-core-shared
- **Beschreibung**: Legt fest, ob CMAS in einem Cluster läuft.
- **Typ**: boolean
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.1.0

cluster.unicast

- **Modul**: cmas-core-shared
- **Beschreibung**: Flag zum Aktivieren des Jgroups-Unicast-Modus für ConSol CM-Cluster (im Gegensatz zum standardmäßigen Multicast-Modus, der in einigen Rechenzentrumsumgebungen zu Problemen führt). Wenn die Property auf „true“ gesetzt ist, müssen Sie die folgenden JVM-Startparameter setzen: `jgroups.bind.port`, `jgroups.bind.address` und `jgroups.initial_hosts`.
- **Typ**: boolean
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: ja
- **Beispielwert**: false (Standardwert)
- **Seit**: 6.11.0.0

cmas.dropSchemaBeforeSetup

- **Modul**: cmas-setup-hibernate
- **Beschreibung**: Gibt an, ob das Schema während des Setups gelöscht werden soll (wurde).
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: true
- **Seit**: 6.0

cmoffice.enabled

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Legt fest, ob CM/Doc (vorher CM/Office) aktiviert ist.

- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.4.0

cmoffice.oo.path.NUMBER

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Möglicher Speicherort der OpenOffice-Installation. Die Properties werden, beginnend mit 0, durchnummeriert.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: c:\Program Files (x86)\LibreOffice 3.6\program
- **Seit**: 6.10.1.0

cmoffice.strict.versioning.enabled

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Steuert, ob der Speichervorgang für Microsoft Word- / OpenOffice-Dokumente ein neues Attachment erzeugt (*true*) oder das vorhandene Attachment überschreibt (*false*). Dies betrifft das Verhalten innerhalb einer Session mit dem Textbearbeitungsprogramm. Wenn das Programm beendet wird, funktioniert der Mechanismus zum Überschreiben nicht mehr.
- **Typ**: Boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.10.5.4

comment.authors.disabled

- **Modul**: cmas-restapi-core
- **Beschreibung**: Deaktiviert die Anzeige des Autors des Inhalts über die REST-API. Der Standardwert ist „false“.
- **Typ**: boolean
- **Neustart erforderlich**: nein

- **System**: nein
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.11.0

commentRequiredForTicketCreation

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Legt fest, ob der Kommentar ein Pflichtfeld für die Erstellung eines Tickets ist.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: true (Standardwert)
- **Seit**: 6.2.0

communication.channel

- **Modul**: cmas-dwh-server
- **Beschreibung**: Kommunikationskanal, mögliche Werte sind DIRECT (Datenbank-Kommunikationskanal, Standardwert seit 6.9.4.1) oder JMS (Standardwert vor 6.9.4.1). Diese System-Property muss vor 6.9.4.1 extra hinzugefügt werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: DIRECT
- **Seit**: 6.8.5.0
- **Entfernt in**: 6.11.0.0 (Der einzige verfügbare und automatisch gesetzte Modus ist „DIRECT“.)

config.data.version

- **Modul**: cmas-core-server
- **Beschreibung**: Die interne Versionsnummer der aktuellen Systemkonfiguration. Diese Property wird intern gepflegt. Ändern Sie sie nur, wenn Sie von ConSol dazu aufgefordert werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein

- **Beispielwert**: 11
- **Seit**: 6.0

config.import.global.transaction.enabled

- **Modul**: cmas-core-server
- **Beschreibung**: Flag, mit dem entschieden wird, ob die Konfiguration (ohne Lokalisierungen) in einer einzigen Transaktion importiert werden soll.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.11.1.0

connection.release.mode

- **Modul**: cmas-setup-hibernate
- **Beschreibung**: Beschreibt die Strategie zur Handhabung der JEE-Verbindung bei Transaktionen. Wenn die Property auf „AFTER_TRANSACTION“ gesetzt ist, wird die Verbindung während der Transaktion zwischengespeichert und am Ende freigegeben. Wenn die Property auf „AFTER_ STATEMENT“ gesetzt ist, wird die Verbindung nach jeder Ausführung einer Anweisung freigegeben. Ändern Sie den Standardwert nur, wenn Sie von ConSol dazu aufgefordert werden.
- **Typ**: string
- **Neustart erforderlich**: ja
- **System**: nein
- **Optional**: ja
- **Beispielwert**: AFTER_STATEMENT (Standardwert für JEE-Umgebungen)
- **Seit**: 6.0

contact.authentication.method

- **Modul**: cmas-core-security
- **Beschreibung**: Definiert die Kontakt-Authentifizierungsmethode für CM/Track, mögliche Werte sind DATABASE oder LDAP oder LDAP,DATABASE oder DATABASE,LDAP.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Seit**: 6.9.3.0

contact.inherit.permissions.only.to.own.customer.group

- **Modul**: cmas-core-security
- **Beschreibung**: Legt fest, ob der authentifizierte Kontakt in CM/Track alle Kundengruppen-Berechtigungen vom repräsentierenden Bearbeiter erbt (false) oder nur die Berechtigungen für die eigene Kundengruppe hat (true).
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Seit**: 6.9.2.3

csrf.domain.white.list

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Die Liste der Domänen (durch "|" getrennt), die zugelassen sind und nicht vom CSRF (Cross-site request forgery)-Filter geprüft werden, z. B. "example.com | consol.de"\
- **Typ**: String
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: example.com | consol.de
- **Seit**: 6.10.7.0 und 6.11.0.5

csrf.request.filter.enabled

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Ermöglicht das Aktivieren/Deaktivieren des CSRF (Cross-site request forgery)-Anfragenfilters
- **Typ**: Boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.10.7.0 und 6.11.0.5

customizationVersion

- **Modul**: cmweb-server-adapter
- **Beschreibung**: UID der letzten Version der Web-Anpassung. Wird nur intern verwendet. Der Wert darf nicht geändert werden.

- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: cd58453e-f3cc-4538-8030-d15e8796a4a7
- **Seit**: 6.5.0

dao.log.threshold.milliseconds

- **Modul**: cmas-core-server
- **Beschreibung**: Zur Konfiguration der Protokollierung von Datenbankoperationszeiten verwendet. DAO-Methoden, deren Ausführung länger als die in dieser Property gesetzte Zeit (in Millisekunden) dauert, werden protokolliert.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 500 (Standardwert)
- **Seit**: 6.11.1.0

dao.log.username

- **Modul**: cmas-core-server
- **Beschreibung**: Zur Konfiguration der Protokollierung von Datenbankoperationszeiten verwendet. Die Ausführung von DAO-Methoden, die im Zusammenhang mit dem in dieser Property angegebenen Benutzernamen stehen werden, wird protokolliert. Es kann nur ein Benutzername angegeben werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: webadmin (Standardwert: leerer String)
- **Seit**: 6.11.1.0

data.directory

- **Modul**: cmas-core-shared
- **Beschreibung**: Verzeichnis für die CMAS-Daten (z. B. Index)
- **Typ**: string
- **Neustart erforderlich**: nein

- **System**: ja
- **Optional**: nein
- **Beispielwert**: C:\Users\user\cmas
- **Seit**: 6.0

data.optimization

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Definiert die Optimierung der Response-Daten. Bisher werden die folgenden Werte unterstützt (mehrere Werte können durch '|' getrennt gesetzt werden): MINIFICATION und COMPRESSION. MINIFICATION minimiert HTML-Daten, indem z. B. Leerzeichen und Kommentare entfernt werden. COMPRESSION wendet gzip-Komprimierung auf die HTTP-Response an. (Hinweis: Wenn das System im Cluster-Modus läuft und Sie parallel verschiedene Konfigurationen testen möchten, können Sie für jeden Cluster-Node verschiedene Werte setzen, indem Sie die System-Property nach dem Muster `data.optimization.nodeId` spezifizieren, um die Standard-Property zu überschreiben.)
- **Typ**: string
- **Neustart erforderlich**: COMPRESSION kann ohne Neustart ein- und ausgeschaltet werden, für MINIFICATION ist ein Neustart erforderlich.
- **System**: ja
- **Optional**: ja
- **Beispielwert**: MINIFICATION|COMPRESSION

database.notification.enabled

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, ob statt JMS der Database Notification Channel der Index-Aktualisierung verwendet werden soll.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.8.4.7

database.notification.redelivery.delay.seconds

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt bei Verwendung des Database Notification Channel der Index-Aktualisierung fest, mit welcher Verzögerung die Benachrichtigung erneut gesendet wird, wenn eine Exception auftritt.
- **Typ**: integer

- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 60
- **Seit**: 6.8.4.7

database.notification.redelivery.max.attempts

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt bei Verwendung des Database Notification Channel der Index-Aktualisierung fest, wie oft maximal versucht wird, die Benachrichtigung erneut zu senden, wenn eine Exception auftritt.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 60
- **Seit**: 6.8.4.7

defaultAttachmentEntryClassName

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Die Standardtextklasse für Attachments, wenn keine andere Klasse explizit gesetzt wurde.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: DefaultTextElement
- **Seit**: 6.9.2.0

defaultCommentClassName

- **Modul**: cmas-core-server
- **Beschreibung**: Standard-Textklasse für Kommentare.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja

- **Beispielwert**:
- **Seit**: 6.3.0

defaultContentEntryClassName

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Standard-Textklasse für neue ACIMs.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: default_class
- **Seit**: 6.3.0

defaultIncommingMailClassName

- **Modul**: cmas-core-server
- **Beschreibung**: Standard-Textklasse für eingehende E-Mails.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Seit**: 6.3.0

defaultNumberOfCustomFieldsColumns

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Standardanzahl an Spalten für Ticketfelder.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 3
- **Seit**: 6.2.0

defaultOutgoingMailClassName

- **Modul**: cmas-core-server
- **Beschreibung**: Standard-Textklasse für ausgehende E-Mails.
- **Typ**: string

- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**:
- **Seit**: 6.3.0

delete.ticket.enabled

- **Modul**: cmas-app-admin-tool
- **Beschreibung**: Steuert, ob in der Ticketverwaltung im Admin Tool der Menüpunkt *Tickets löschen* im Kontextmenü der Ticketliste angezeigt wird.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.9.4.0

diffTrackingEnabled

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Entfernt in ConSol CM-Version 6.11.
  Legt fest, ob das gleichzeitige Bearbeiten eines Tickets durch verschiedene Bearbeiter möglich ist. Der Standardwert ist *true*.
  false: Vorherige Art und Weise der Behandlung von Änderungen beim Editieren von Tickets. Wenn ein Ticket zwischendurch geändert wurde, kann der aktuelle Bearbeiter seine Änderungen nicht speichern, ohne die Seite vorher neu zu laden.
  true: Neuer Modus zur Behandlung von Änderungen. Wenn das Ticket geändert wurde, wird das Speichern von anderen Änderungen nicht mehr blockiert. Wenn der Teil des Tickets, der geändert wurde, genau der Teil ist, der vom speichernden Bearbeiter geändert wird, wird eine Informationsmeldung angezeigt, die Ticketänderungen werden aber trotzdem gespeichert.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: true
- **Seit**: 6.10.1
- **Entfernt in**: 6.11.0

diffTrackingEnabledForUnitAndResource

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Aktiviert die Verhinderung von gleichzeitigen Änderungen an Units/Ressourcen.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 3
- **Seit**: 6.11.0.0

diff.tracking.disabled

- **Modul**: cmas-restapi-core
- **Beschreibung**: Ausweich-Property zum Deaktivieren der Änderungsnachverfolgung für CM/Track. Diese ist protokollbasiert und kann viel Last erzeugen.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false (Standardwert)
- **Seit**: 6.10.5.6

disable.admin.task.auto.commit

- **Modul**: cmas-core-index-common
- **Beschreibung**: Alle Tasks, die für eine Index-Aktualisierung erstellt werden, werden automatisch direkt nach ihrer Erstellung ausgeführt.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.6.1

dwh.administration.refresh.interval.seconds

- **Modul**: cmas-app-admin-tool
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: integer

- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 10
- **Seit**: 6.11.0.1

dwh.mode

- **Modul**: cmas-dwh-server
- **Beschreibung**: Aktueller Modus der DWH-Datenübermittlung. Mögliche Werte sind OFF, ADMIN, LIVE.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: OFF
- **Seit**: 6.0.1

engineer.description.cache.enabled

- **Modul**: cmas-core-server
- **Beschreibung**: Legt fest, ob die Benutzerbeschreibungen zwischengespeichert werden. Der Standardwert ist „true“. Ändern Sie diesen Wert nur, wenn Sie von ConSol dazu aufgefordert werden.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.11.0

engineer.description.mode

- **Modul**: cmas-core-server
- **Beschreibung**: Legt fest, ob die Benutzernamen im Ticketprotokoll aus der Datenbank abgerufen oder dynamisch über Templates erzeugt werden. Der Standardwert „DYNAMIC“ ist aus Performance-Sicht etwas kostspieliger. „PROTOCOL“ ist schneller, gibt aber historische Namen zurück, die unter Umständen veraltet sind. Verwenden Sie „PROTOCOL“, wenn Sie viele Protokolleinträge von vielen unterschiedlichen Benutzern haben.
- **Typ**: string
- **Neustart erforderlich**: nein

- **System**: nein
- **Optional**: ja
- **Beispielwert**: PROTOCOL
- **Seit**: 6.11.0

engineer.description.template.name

- **Modul**: cmas-core-server
- **Beschreibung**: Definiert den Namen des Templates, mit dem die Bearbeiternamen im Web Client dargestellt werden. Das Template muss im Abschnitt *Templates* des Admin Tools gespeichert sein. Standardwert: „engineer description template name“.
- **Typ**: String
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: nein
- **Seit**: 6.11.0

eviction.event.queue.size

- **Modul**: cmas-core-cache
- **Beschreibung**: Die Größe der Queue mit den Cache-Events. Der Standardwert ist 200000. Auf Systemen mit viel Traffic oder Last wird empfohlen, diesen Wert leicht zu erhöhen (bis 400000).
- **Typ**: integer
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 200000
- **Seit**: 6.4.0

eviction.max.nodes

- **Modul**: cmas-core-cache
- **Beschreibung**: Setzt die maximale Größe der internen Caches. Der Standardwert ist 100000. Das Erhöhen dieses Wertes führt zu einem höheren Speicherverbrauch und wird nicht empfohlen, außer wenn ConSol explizit dazu auffordert.
- **Typ**: integer
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein

- **Beispielwert**: 100000
- **Seit**: 6.4.0

eviction.wakeup.interval

- **Modul**: cmas-core-cache
- **Beschreibung**: Setzt das Intervall (in Millisekunden) zwischen zwei Verarbeitungszyklen von Cache-Queue-Events. Der Standardwert ist 3000. Auf Systemen mit viel Traffic oder Last wird empfohlen, den Wert zu verringern (Minimalwert ist 1500).
- **Typ**: integer
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 3000
- **Seit**: 6.4.0

expert.mode

- **Modul**: cmas-core-shared
- **Beschreibung**: Schaltet den Expertenmodus ein bzw. aus, sodass Funktionen für Experten verfügbar gemacht bzw. blockiert werden. Zum Beispiel ist die CM-System-Property `initialized` nur im Expertenmodus verfügbar.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.0

external.line.access.prefix

- **Modul**: cmas-core-server
- **Beschreibung**: Allgemeine Vorwahl, die vor einer Ortsvorwahl gewählt wird. Wird für jede Kundengruppe einzeln eingestellt.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 411
- **Seit**: 6.9.3.0

favoritesSizeLimit

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Maximale Anzahl von Elementen in der Favoritenliste.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 10
- **Seit**: 6.0

fetchLock.interval

- **Modul**: cmas-workflow-jbpm
- **Beschreibung**:
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 5000
- **Entfernt in**: 6.8.0

fetchSize.strategy

- **Modul**: cmas-core-server
- **Beschreibung**: Auswahl der Strategie für die Abholgröße von JDBC-Ergebnissatzes.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: FetchSizePageBasedStrategy, FetchSizeThresholdStrategy, FetchSizeFixedStrategy
- **Seit**: 6.8.4.1

fetchSize.strategy.FetchSizeFixedStrategy.value

- **Modul**: cmas-core-server
- **Beschreibung**: Legt den Wert für Abholgrößen fest, wenn die ausgewählte Strategie für die Abholgröße `FetchSizeFixedStrategy` ist.
- **Typ**: integer

- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 150
- **Seit**: 6.8.4.1

fetchSize.strategy.FetchSizePageBasedStrategy.limit

- **Modul**: cmas-core-server
- **Beschreibung**: Legt den Wert für Abholgrößen fest, wenn die ausgewählte Strategie für die Abholgröße `FetchSizePageBasedStrategy` ist.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 10000
- **Seit**: 6.8.4.1

fetchSize.strategy.FetchSizeThresholdStrategy.value

- **Modul**: cmas-core-server
- **Beschreibung**: Gibt Grenzwerte für Abholgrößen an, wenn die ausgewählte Strategie für die Abholgröße `FetchSizeThresholdStrategy` ist.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 150,300,600,1000
- **Seit**: 6.8.4.1

filesystem.polling.threads.number

- **Modul**: cmas-nimh
- **Beschreibung**: Anzahl der Threads, die für die Abfrage der Datenbank-E-Mail-Warteschlange gestartet werden. Standardwert: 1.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja

- **Beispielwert**: 10
- **Seit**: 6.4.0

filesystem.polling.threads.shutdown.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Wartezeit nach dem Shutdown-Signal. Wenn der Timeout erreicht ist, wird der Thread beendet. Standardwert: 60.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

filesystem.polling.threads.watchdog.interval.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Intervall des Watchdog Threads. Standardwert: 30.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

filesystem.task.enabled

- **Modul**: cmas-nimh
- **Beschreibung**: Mit dieser System-Property kann der Service Thread eines bestimmten Pollers deaktiviert werden. Standardwert: true.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.4.0

filesystem.task.interval.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Standard-Intervall für den Abruf von Postfächern. Standardwert: 60 Sekunden
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

filesystem.task.polling.folder

- **Modul**: cmas-nimh
- **Beschreibung**: Speicherort des Ordners, der überwacht und nach E-Mails im Format von eml-Dateien durchsucht wird. Standard: Unterverzeichnis "mail" des cmas-Datenverzeichnisses
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: c://cmas//mail
- **Seit**: 6.4.0

filesystem.task.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Nach dieser Zeit (der Inaktivität) wird der Service Thread als beschädigt betrachtet und automatisch neu gestartet. Standardwert: 120 Sekunden
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

filesystem.task.transaction.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Standard-Transaktions-Timeout für Transaktionen, die E-Mails abrufen. Sollte mit der Anzahl der Nachrichten, die gleichzeitig abgeholt werden, korrelieren. Standardwert: 60 Sekunden
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

forward.mails.to.representatives

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Legt fest, ob E-Mails, die manuell aus dem Web Client gesendet werden, auch an die Vertreter eines Bearbeiters gesendet werden. Der Standardwert dieser Property ist „false“, d. h. diese Art von E-Mails wird nicht an den Vertreter weitergeleitet. Setzen Sie diese Property auf „true“, wenn Sie das bisherige Verhalten wiederherstellen wollen, d. h. alle E-Mails, die an den vertretenen Bearbeiter gesendet werden, werden automatisch an den Vertreter weitergeleitet. Berücksichtigen Sie, dass dies unter Umständen nicht erwünscht ist, wenn dieselbe Person gleichzeitig Bearbeiter und Kunde im CM-System ist.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: nein
- **Beispielwert**: false (Standardwert)
- **Seit**: 6.11.1.7



Diese Property konfiguriert nur die Verarbeitung von manuell gesendeten E-Mails. Die Verarbeitung von automatisch gesendeten E-Mails hängt von der verwendeten Java-Methode ab.

globalSearchResultSizeLimit

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Maximale Anzahl an Ergebnissen in der Schnellsuche.
- **Typ**: integer
- **Neustart erforderlich**: nein

- **System**: ja
- **Optional**: nein
- **Beispielwert**: 10
- **Seit**: 6.0

heartbeat

- **Modul**: cmas-core-server
- **Beschreibung**: Zeitstempel, der angibt, ob eine Instanz der Applikation mit dem Datenbankschema verbunden ist.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1234567899
- **Seit**: 6.10.5.3

helpFilePath

- **Modul**: cmweb-server-adapter
- **Beschreibung**: URL für die Onlinehilfe. Wenn der Wert nicht leer gelassen wird, wird der Hilfe-Button im Web Client angezeigt.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: http://www.consol.de
- **Seit**: 6.2.1

hibernate.dialect

- **Modul**: cmas-setup-hibernate
- **Beschreibung**: Der Hibernate-Dialekt. Normalerweise wird dieser Wert während des initialen Setups gesetzt (abhängig vom Datenbanksystem).
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein

- **Beispielwert**: org.hibernate.dialect.MySQL5InnoDBDialect
- **Seit**: 6.0

hideTicketSubject

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Wenn auf *true* gesetzt, wird das Ticketthema ausgeblendet.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.2.1

ignore-queues

- **Modul**: cmas-dwh-server
- **Beschreibung**: Durch eine durch Kommas getrennte Liste von Queue-Namen wird hier festgelegt, dass Tickets dieser Queues nicht ins DWH übermittelt werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: QueueName1,QueueName2,QueueName3
- **Seit**: 6.6.19
- **Entfernt in**: 6.8.1

index.attachment

- **Modul**: cmas-core-index-common
- **Beschreibung**: Beschreibt, ob der Inhalt von Attachments indiziert wird.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: true
- **Seit**: 6.4.3

index.history

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, ob das Unit- und das Ticketprotokoll indiziert werden.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.1.0
- **Entfernt in**: 6.11.0

index.status

- **Modul**: cmas-core-index-common
- **Beschreibung**: Status des Indexers, mögliche Werte sind RED, YELLOW, GREEN, werden im Admin Tool angezeigt.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: GREEN
- **Seit**: 6.6.1

index.task.worker.threads

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie viele Threads benutzt werden, um Index-Aufgaben auszuführen (Synchronisierung, Administrations- und Reparatur-Aufgaben).
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1 (Standardwert) (wir empfehlen einen Wert zu verwenden, der nicht größer als 2 ist)
- **Seit**: 6.6.14, 6.7.3. Seit 6.8.0 und auschließlich in 6.6.21 sind auch normale (live) Index-Aktualisierungen von dieser Property betroffen.

index.version.current

- **Modul**: cmas-core-index-common
- **Beschreibung**: Enthält Informationen über die derzeitige (möglicherweise veraltete) Index-version.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1 (Standardwert)
- **Seit**: 6.7.0

index.version.newest

- **Modul**: cmas-core-index-common
- **Beschreibung**: Enthält Informationen, welche Indexversion als die neueste betrachtet wird.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1 (Standardwert)
- **Seit**: 6.7.0

indexed.assets.per.thread.in.memory

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie viele Assets während des Indizierens pro Thread auf einmal in den Speicher geladen werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 200 (Standardwert)
- **Seit**: 6.8.0

indexed.engineers.per.thread.in.memory

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie viele Bearbeiter während des Indizierens pro Thread auf einmal in den Speicher geladen werden.

- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 300 (Standardwert)
- **Seit**: 6.6.14, 6.7.3

indexed.resources.per.thread.in.memory

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie viele Ressourcen während des Indizierens pro Thread auf einmal in den Speicher geladen werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 200 (Standardwert)
- **Seit**: 6.10.0.0

indexed.tickets.per.thread.in.memory

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie viele Tickets während des Indizierens pro Thread auf einmal in den Speicher geladen werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 100 (Standardwert)
- **Seit**: 6.6.14, 6.7.3

indexed.units.per.thread.in.memory

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie viele Units während des Indizierens pro Thread auf einmal in den Speicher geladen werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja

- **Optional**: nein
- **Beispielwert**: 200 (Standardwert)
- **Seit**: 6.6.14, 6.7.3

initialized

- **Modul**: cmas-setup-manager
- **Beschreibung**: Legt fest, ob CMAS initialisiert ist. Wenn dieser Wert fehlt oder nicht auf „true“ gesetzt ist, wird das Setup ausgeführt. Ab ConSol CM-Version 6.11 ist diese Property nur im Expertenmodus verfügbar.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: true
- **Seit**: 6.0

> ⚠ Seien Sie mit der Verwendung dieser System-Property sehr vorsichtig! Wenn Sie den Wert auf *false* setzen, wird der ConSol CM-Server beim nächsten Systemstart das System-Setup ausführen, d. h. alle Daten des bestehenden Systems gehen verloren, einschließlich der System-Properties!

internal.line.access.prefix

- **Modul**: cmas-core-server
- **Beschreibung**: Vorwahl, die das Telefonsystem des Unternehmens für externe Anrufe verwendet. Wird für jede Kundegruppe einzeln gesetzt.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 199
- **Seit**: 6.9.3.0

is.cmrf.alive

- **Modul**: cmas-dwh-server
- **Beschreibung**: Als Startpunkt sollte die Zeit genommen werden, bei der zuletzt eine Nachricht an das CMRF gesendet wurde. Wenn nach diesem Wert (in Sekunden) keine Antwort vom CMRF empfangen wird, wird ein DWH-Betriebsstatus mit der Fehlermeldung, dass das CMRF nicht erreichbar ist, erstellt.

- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1200
- **Seit**: 6.7.0

java.naming.factory.initial

- **Modul**: cmas-dwh-server
- **Beschreibung**: Factory-Java-Klasse für DWH context factory.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: org.jnp.interfaces.NamingContextFactory
- **Seit**: 6.0.1
- **Entfernt in**: 6.11.0.0

java.naming.factory.url.pkgs

- **Modul**: cmas-dwh-server
- **Beschreibung**:
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: org.jboss.naming:org.jnp.interfaces
- **Seit**: 6.0.1
- **Entfernt in**: 6.11.0.0

java.naming.provider.url

- **Modul**: cmas-dwh-server
- **Beschreibung**: URL des Naming Provider.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein

- **Beispielwert**: localhost
- **Seit**: 6.0.1
- **Entfernt in**: 6.11.0.0

jobExecutor.adminMail

- **Modul**: cmas-workflow-engine
- **Beschreibung**: E-Mail-Adresse, an die Benachrichtigungs-E-Mails, die Probleme der Jobausführung betreffen, geschickt werden (wenn die Anzahl der Neuversuche überschritten wurde).
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: admin@consol.de
- **Seit**: 6.8.0

jobExecutor.idleInterval

- **Modul**: cmas-workflow-jbpm
- **Beschreibung**:
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 45000
- **Entfernt in**: 6.8.0
- **Ersetzt durch**: jobExecutor.idleInterval.seconds

jobExecutor.idleInterval.seconds

- **Modul**: cmas-workflow-engine
- **Beschreibung**: Legt fest, wie oft der Job Executor Thread nach neuen Jobs zum Ausführen sucht.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja

- **Beispielwert**: 45 (Standardwert bis CM-Version 6.10.5.2. Der Standardwert für CM-Versionen 6.10.5.3 und höher ist 5)
- **Seit**: 6.8.0

jobExecutor.jobExecuteRetryNumber

- **Modul**: cmas-workflow-jbpm
- **Beschreibung**:
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 5
- **Entfernt in**: 6.8.0
- **Ersetzt durch**: jobExecutor.jobMaxRetries

jobExecutor.jobMaxRetries

- **Modul**: cmas-workflow-engine
- **Beschreibung**: Steuert die Anzahl der erneuten Versuche, die der Job Executor unternimmt, bevor er einen Job als fehlgeschlagen deklariert.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 5 (Standardwert)
- **Seit**: 6.8.0

jobExecutor.jobMaxRetriesReachedSubject

- **Modul**: cmas-workflow-engine
- **Beschreibung**: Der Betreff für Benachrichtigungs-E-Mails, die Administratoren über fehlgeschlagene Job-Ausführungen erhalten.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: Maximale Anzahl der Neuversuche für Job erreicht. Job wurde entfernt! (Standard)
- **Seit**: 6.8.0

jobExecutor.lockingLimit

- **Modul**: cmas-workflow-engine
- **Beschreibung**: Anzahl der gleichzeitig gelockten (als "in der Ausführung" markierten) Jobs des Job Executor Threads.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 5 (Standardwert seit CM-Version 6.10.5.3)
- **Seit**: 6.8.0

jobExecutor.lockTimeout.seconds

- **Modul**: cmas-workflow-engine
- **Beschreibung**: Legt fest, wie lange ein Job vom Job Executor (als "in der Ausführung" markiert) gelockt werden kann.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 360 (Standardwert)
- **Seit**: 6.8.0

jobExecutor.mailFrom

- **Modul**: cmas-workflow-engine
- **Beschreibung**: E-Mail-Adresse, die als From-Header für Admin-Benachrichtigungen eingesetzt wird.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: jobexecutor@consol.de
- **Seit**: 6.8.0

jobExecutor.maxInactivityInterval.minutes

- **Modul**: cmas-workflow-engine
- **Beschreibung**: Länge der erlaubten Inaktivität des Job Executors in Minuten (z. B. wenn er durch eine Langzeit-Ausführung gesperrt wird). Nach dieser Zeit werden die Executor Threads neu gestartet.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja Standardwert ist auf 30 Minuten gesetzt.
- **Beispielwert**: 15 (Standardwert)
- **Seit**: 6.9.2.0

jobExecutor.threads

- **Modul**: cmas-workflow-engine
- **Beschreibung**: Anzahl der Job Executor Threads.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 1 (Standardwert)
- **Seit**: 6.8.0

jobExecutor.timerRetryInterval

- **Modul**: cmas-workflow-jbpm
- **Beschreibung**:
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 10000
- **Entfernt in**: 6.8.0
- **Ersetzt durch**: jobExecutor.timerRetryInterval.seconds

jobExecutor.timerRetryInterval.seconds

- **Modul**: cmas-workflow-engine
- **Beschreibung**: Legt fest, wie lange der Job Executor Thread nach einem Fehler bei der Job-Ausführung wartet.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 10 (Standardwert bis CM-Version 6.10.5.2. Der Standardwert für CM-Versionen 6.10.5.3 und höher ist 30.)
- **Seit**: 6.8.0

jobExecutor.txTimeout.seconds

- **Modul**: cmas-workflow-engine
- **Beschreibung**: Transaktions-Timeout für die Job-Ausführung.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 60 (Standardwert)
- **Seit**: 6.8.0

kerberos.v5.enabled

- **Modul**: cmas-core-security
- **Beschreibung**: Legt fest, ob SSO über Kerberos aktiviert ist.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false (Standardwert, wenn Kerberos bei der Systemeinrichtung nicht aktiviert wurde)
- **Seit**: 6.2.0

kerberos.v5.username.regex

- **Modul**: cmas-core-security
- **Beschreibung**: Regulärer Ausdruck für die Zuordnung des Kerberos Principals zum Login des CM-Bearbeiters
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: (.*)@.*
- **Seit**: 6.2.0

last.config.change

- **Modul**: cmas-core-server
- **Beschreibung**: Zufällige UUID, die während der letzten Konfigurationsänderung generiert wurde. Dies ist ein intern verwalteter Wert. Ändern Sie diesen nur, wenn Sie von ConSol dazu aufgefordert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 2573c7b7-2bf5-47ff-b5a2-bad31951a266
- **Seit**: 6.1.0, 6.2.1

last.config.change.templates

- **Modul**: cmas-core-server
- **Beschreibung**: Zufällige UUID, die bei der letzten Änderung am Template erzeugt wird. Dies ist ein intern gepflegter Wert. Ändern Sie ihn nur, wenn Sie von ConSol dazu aufgefordert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 2573c7c7-2af5-4eff-b9c2-bad31951a266
- **Seit**: 6.10.5.0

last.ping.timestamp

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.

- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 32323214
- **Seit**: 6.11.0.1

ldap.authentication

- **Modul**: cmas-core-security
- **Beschreibung**: Verwendete Authentifizierungsmethode, wenn LDAP-Authentifizierung benutzt wird. Mögliche Werte sind "anonymous" und "simple" (Standardwert)
- **Typ**: string
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: simple
- **Seit**: 6.0

ldap.basedn

- **Modul**: cmas-core-security
- **Beschreibung**: Base DN für die Suche von LDAP-Benutzerkonten, wenn LDAP-Authentifizierung verwendet wird.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: ou=accounts,dc=consol,dc=de
- **Seit**: 6.0

ldap.certificate.basedn

- **Modul**: cmas-core-server
- **Beschreibung**: Base DN für den Speicherort der Zertifikate im LDAP-Verzeichnisbaum. Wenn nicht angegeben, wird `cmas-core-security, ldap.basedn` verwendet.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja

- **Optional**: ja
- **Beispielwert**: ou=accounts,dc=consol,dc=de
- **Seit**: 6.8.4

ldap.certificate.content.attribute

- **Modul**: cmas-core-server
- **Beschreibung**: Name des LDAP-Attributs, das angibt, wo Zertifikatsdaten im LDAP-Verzeichnisbaum gespeichert werden. Standardwert: usercertificate
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: usercertificate
- **Seit**: 6.8.4

ldap.certificate.password

- **Modul**: cmas-core-server
- **Beschreibung**: Passwort des LDAP-Zertifikatmanagers. Wenn nicht gesetzt, wird `cmas-core-security, ldap.password` verwendet.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.8.4

ldap.certificate.providerurl

- **Modul**: cmas-core-server
- **Beschreibung**: URL des LDAP-Zertifikat-Providers. Wenn nicht gesetzt, wird `cmas-core-security, ldap.providerurl` verwendet.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: ldap://ldap.consol.de:389
- **Seit**: 6.8.4

ldap.certificate.searchattr

- **Modul**: cmas-core-server
- **Beschreibung**: Name des LDAP-Attributs, mit dem Zertifikate im LDAP-Verzeichnisbaum gesucht werden. Standardwert: mail
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: mail
- **Seit**: 6.8.4

ldap.certificate.userdn

- **Modul**: cmas-core-server
- **Beschreibung**: DN des LDAP-Zertifikatmanagers. Wenn nicht gesetzt, wird `cmas-core-security, ldap.userdn` verwendet.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.8.4

ldap.contact.name.basedn

- **Modul**: cmas-core-security
- **Beschreibung**: Base DN für die Suche nach der Kontakt-DN mittels LDAP-ID (z. B. ou=accounts,dc=consol,dc=de).
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Seit**: 6.9.3.0

ldap.contact.name.password

- **Modul**: cmas-core-security
- **Beschreibung**: Passwort für die Suche nach der Kontakt-DN mittels LDAP-ID. Wenn nicht gesetzt, wird ein anonymes Konto verwendet.
- **Typ**: string
- **Neustart erforderlich**: nein

- **System**: nein
- **Optional**: ja
- **Seit**: 6.9.3.0

ldap.contact.name.providerurl

- **Modul**: cmas-core-security
- **Beschreibung**: Adresse des LDAP-Servers (ldap[s]://host:port).
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Seit**: 6.9.3.0

ldap.contact.name.searchattr

- **Modul**: cmas-core-security
- **Beschreibung**: Attribut für die Suche nach der Kontakt-DN mittels LDAP-ID (z. B. uid).
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Seit**: 6.9.3.0

ldap.contact.name.userdn

- **Modul**: cmas-core-security
- **Beschreibung**: Benutzer-DN für die Suche nach Kontakt-DN mittels LDAP-ID. Wenn nicht gesetzt, wird ein anonymes Konto verwendet.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Seit**: 6.9.3.0

ldap.initialcontextfactory

- **Modul**: cmas-core-security
- **Beschreibung**: Name der Java-Klasse für die Initial Context Factory der LDAP-Implementierung bei der Verwendung von LDAP-Authentifizierung. Ist üblicherweise `com.sun.jndi.ldap.LdapCtxFactory`.

- **Typ**: string
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: com.sun.jndi.ldap.LdapCtxFactory
- **Seit**: 6.0

ldap.password

- **Modul**: cmas-core-security
- **Beschreibung**: Passwort für die Verbindung zum LDAP, um Benutzer zu suchen, wenn LDAP-Authentifizierung verwendet wird. Wird nur benötigt, wenn die Suche nicht anonym ausgeführt werden kann.
- **Typ**: password
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.1.2

ldap.providerurl

- **Modul**: cmas-core-security
- **Beschreibung**: LDAP-Provider, wenn LDAP-Authentifizierung verwendet wird.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: ldap://myserver.consol.de:389
- **Seit**: 6.0

ldap.searchattr

- **Modul**: cmas-core-security
- **Beschreibung**: Suchattribut für die Suche nach LDAP-Einträgen, die mit dem CM-Login verbunden sind.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein

- **Beispielwert**: uid
- **Seit**: 6.0

ldap.userdn

- **Modul**: cmas-core-security
- **Beschreibung**: LDAP-Benutzer für die Verbindung zum LDAP, um Benutzer zu suchen, wenn LDAP-Authentifizierung verwendet wird. Wird nur benötigt, wenn die Suche nicht anonym ausgeführt werden kann.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.1.2

live.start

- **Modul**: cmas-dwh-server
- **Beschreibung**: Wenn der DWH-Synchronisierungsmodus mit dem Admin Tool (Navigationsgruppe *Data Warehouse*, Navigationselement *Administration*, Button *Konfiguration*) auf „LIVE“ gesetzt wird, wird diese Property erstellt und auf das aktuelle Datum gesetzt. Wenn der LIVE-Modus nicht aktiviert ist und sich keine Daten in `cmas_dwh_ser_sync_object` befinden, wird die Property `live.start` gelöscht.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja (im DWH-Modus „LIVE“ automatisch hinzugefügt)
- **Beispielwert**: 15028802377645
- **Seit**: 6.7.0

local.country.prefix

- **Modul**: cmas-core-server
- **Beschreibung**: Vorwahl des Ländercodes. Wird für jede Kundengruppe einzeln eingestellt.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 48
- **Seit**: 6.9.3.0

mail.attachments.validation.info.sender

- **Modul**: cmas-nimh-extension
- **Beschreibung**: Setzt den From-Header bei Attachments vom Typ `error notification mail`.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: admin@mail.com
- **Seit**: 6.7.5

mail.attachments.validation.info.subject

- **Modul**: cmas-nimh-extension
- **Beschreibung**: Setzt den Betreff bei Attachments vom Typ error notification mail.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: E-Mail konnte nicht verarbeitet werden, weil ihre Attachments zurückgewiesen wurden!
- **Seit**: 6.7.5

mail.db.archive

- **Modul**: cmas-nimh-extension
- **Beschreibung**: Wenn dieser Wert auf *true* gesetzt ist, werden eingehende E-Mails in der Datenbank archiviert.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: false (Standardwert)
- **Seit**: 6.8.5.5

mail.encryption

- **Modul**: cmas-core-server
- **Beschreibung**: Wenn dieser Wert auf „true“ gesetzt ist, ist im Ticket-E-Mail-Editor die Checkbox zur Verschlüsselung der E-Mail standardmäßig aktiviert.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: true (Standardwert = false)
- **Seit**: 6.8.4.0

mail.error.from.address

- **Modul**: cmas-nimh-extension
- **Beschreibung**: From-Adresse für Fehler-E-Mails von NIMH
- **Typ**: email
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: myuser@consol.de
- **Seit**: 6.4.0

mail.error.to.address

- **Modul**: cmas-nimh-extension
- **Beschreibung**: To-Adresse für Fehler-E-Mails von NIMH. Standardmäßig wird die E-Mail-Adresse des Administrators, die Sie bei der Systemeinrichtung eingegeben haben, verwendet.
- **Typ**: email
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: myuser@consol.de
- **Seit**: 6.4.0

mail.from

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Diese E-Mail-Adresse wird anstelle der E-Mail-Adresse des Bearbeiters in E-Mail-Konversationen verwendet.

- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.1.2

mail.notification.engineerChange

- **Modul**: cmas-core-server
- **Beschreibung**: Legt fest, ob eine Benachrichtigungs-E-Mail verschickt wird, wenn der Bearbeiter eines Tickets wechselt.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: true
- **Seit**: 6.1.0

mail.notification.sender

- **Modul**: cmas-core-server
- **Beschreibung**: From-Adresse der Benachrichtigungs-E-Mails, die verschickt werden, wenn der Bearbeiter eines Tickets wechselt. Wenn nicht gesetzt, wird stattdessen `cmas-core-security, admin.email` verwendet.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: cm6notification@cm6installation
- **Seit**: 6.6.3

mail.on.error

- **Modul**: cmas-nimh-extension
- **Beschreibung**: Wenn diese Property auf *true* gesetzt ist, wird im Fall, dass eine E-Mail nicht verarbeitet werden konnte, ein Fehler-E-Mail an die oben konfigurierten E-Mail-Adressen gesendet. Standardwert: true.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein

- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.4.0

mail.process.error

- **Modul**: cmas-nimh-extension
- **Beschreibung**: To-Adresse für E-Mails mit Fehlermeldungen des Mule.
- **Typ**: email
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: myuser@consol.de
- **Seit**: 6.4.0

mail.redelivery.retry.count

- **Modul**: cmas-core-server
- **Beschreibung**: Anzahl der erneuten Zustellversuche einer ausgehenden E-Mail.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 3
- **Seit**: 6.1.0

mail.reply.to

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Wenn dieser Wert gesetzt ist, zeigt der Web Client diesen Wert beim Versenden einer E-Mail im Reply-To-Feld an.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.0.1

> ⚠ Bitte lesen Sie dazu die ausführlichen Informationen über Reply-To-Adressen in ConSol CM im Abschnitt Skripte des Typs E-Mail.

mail.sender.address

- **Modul**: cmas-workflow-jbpm
- **Beschreibung**: From-Adresse für E-Mails, die von der Workflow-Engine versendet werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: myuser@consol.de
- **Entfernt in**: 6.8.0
- **Ersetzt durch**: jobExecutor.mailFrom

mail.smtp.email

- **Modul**: cmas-core-server
- **Beschreibung**: SMTP-E-Mail-URL für ausgehende E-Mails.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: smtp://mail.mydomain.com:25
- **Seit**: 6.0

mail.smtp.envelopesender

- **Modul**: cmas-core-server
- **Beschreibung**: E-Mail-Adresse, die als Absender im SMTP-Envelope benutzt wird. Wenn nicht gesetzt, wird die From-Adresse der E-Mail benutzt.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: mysender@mydomain.com
- **Seit**: 6.5.7

mail.smtp.tls.enabled

- **Modul**: cmas-core-server
- **Beschreibung**: Aktiviert SMTP über SSL/TLS (SMTPS) für das Senden von E-Mails aus dem Web Client und aus Skripten. Der Standardwert ist „false“. Wenn Sie diese Poperty auf „true“ setzen, ist SMTPS zum Senden von E-Mails aktiviert.
- **Typ**: boolean
- **Neustart erforderlich**: ja
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.11.1.6

mail.ticketname.pattern

- **Modul**: cmas-nimh-extension
- **Beschreibung**: Regulärer Ausdruck, mit dem der Ticketname im Betreff eingehender E-Mails identifiziert wird.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: .*?Ticket\s+\((\S+)\).*
- **Seit**: 6.4.0

mailbox.1.connection.host

- **Modul**: cmas-nimh
- **Beschreibung**: Host (Server) für das erste konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.host`.

mailbox.1.connection.password

- **Modul**: cmas-nimh
- **Beschreibung**: Passwort für das erste konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.password`.

mailbox.1.connection.port

- **Modul**: cmas-nimh
- **Beschreibung**: Port für das erste konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.port`.

mailbox.1.connection.protocol

- **Modul**: cmas-nimh
- **Beschreibung**: Protokoll (z. B. IMAP oder POP3) für das erste konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.protocol`.

mailbox.1.connection.username

- **Modul**: cmas-nimh
- **Beschreibung**: Benutzername für das erste konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.username`.

mailbox.2.connection.host

- **Modul**: cmas-nimh
- **Beschreibung**: Host (Server) für das zweite konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.host`.

mailbox.2.connection.password

- **Modul**: cmas-nimh
- **Beschreibung**: Passwort für das zweite konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.password`.

mailbox.2.connection.port

- **Modul**: cmas-nimh
- **Beschreibung**: Port für das zweite konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.port`.

mailbox.2.connection.protocol

- **Modul**: cmas-nimh
- **Beschreibung**: Protokoll (z. B. IMAP oder POP3) für das zweite konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.protocol`.

mailbox.2.connection.username

- **Modul**: cmas-nimh
- **Beschreibung**: Benutzername für das zweite konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.username`.



Für alle Postfach-Properties im Zusammenhang mit NIMH wird folgendes Prinzip eingesetzt: Es wird eine Standard-Property definiert (z.B `mailbox.default.connection.port`). Wenn keine postfachspezifische Property konfiguriert ist, wird dieser Standardwert verwendet.

mailbox.default.connection.host

- **Modul**: cmas-nimh
- **Beschreibung**: Host (Servername) eines bestimmten Postfachs, aus dem der Poller E-Mails liest.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 10.10.1.157
- **Seit**: 6.4.0

mailbox.default.connection.password

- **Modul**: cmas-nimh
- **Beschreibung**: Passwort für ein bestimmtes Postfach, aus dem der Poller E-Mails liest.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: consol
- **Seit**: 6.4.0

mailbox.default.connection.port

- **Modul**: cmas-nimh
- **Beschreibung**: Port für ein bestimmtes Postfach, aus dem der Poller E-Mails liest.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 143
- **Seit**: 6.4.0

mailbox.default.connection.protocol

- **Modul**: cmas-nimh
- **Beschreibung**: Protokoll des Pollers, z. B. IMAP oder POP3. Kein Standardwert.
- **Typ**: string
- **Neustart erforderlich**: nein

- **System**: nein
- **Optional**: ja
- **Beispielwert**: imap
- **Seit**: 6.4.0

mailbox.default.connection.username

- **Modul**: cmas-nimh
- **Beschreibung**: Benutzername für ein bestimmtes Postfach, aus dem der Poller E-Mails liest.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: username
- **Seit**: 6.4.0

mailbox.default.session.mail.debug

- **Modul**: cmas-nimh
- **Beschreibung**: Beispiel für javax.mail-Property - ermöglicht detaillierteres Session-Debugging mit javax.mail.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.4.0

mailbox.default.session.mail.imap.timeout

- **Modul**: cmas-nimh
- **Beschreibung**: Beispiel für javax.mail-Property
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 120
- **Seit**: 6.4.0

mailbox.default.session.mail.mime.address.strict

- **Modul**: cmas-nimh
- **Beschreibung**: Beispiel für javax.mail-Property - Gegenstück zum alten `mail.mime.strict` von Mule, erlaubt das nicht so strikte Parsen des E-Mail-Headers.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.4.0

mailbox.default.session.mail.pop3.timeout

- **Modul**: cmas-nimh
- **Beschreibung**: Beispiel für javax.mail-Property
- **Typ**:
- **Neustart erforderlich**:
- **System**:
- **Optional**:
- **Beispielwert**:
- **Seit**: 6.4.0

mailbox.default.session.mail.PROTOCOL.fetchsize

- **Modul**: cmas-nimh
- **Beschreibung**: Setzt die JavaMail-Property für die Größe der `fetchsize` beim `partialfetch` in Bytes für das angegebene Protokoll.
  Für IMAP-Systeme: In CM-Versionen ab 6.10.7.0 wird der Wert von `mailbox.default.session.mail.imap.fetchsize` bei der initialen Einrichtung eines ConSol CM-Systems auf „1048576“ (entspricht 1 MB) gesetzt. Bei einer Aktualisierung eines vorhandenen ConSol CM-Systems bleibt der Wert der Property unverändert, wenn die Property bereits vorhanden ist. Wenn die Property noch nicht vorhanden ist, wird sie mit dem Standardwert hinzugefügt.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 1048576
- **Seit**: 6.9.4.0

mailbox.default.session.mail.PROTOCOL.partialfetch

- **Modul**: cmas-nimh
- **Beschreibung**: Setzt die JavaMail-Property für `partialfetch`, d. h. steuert, ob die PartialFetch-Funktion des Protokolls genutzt werden soll.
  Für IMAP-Systeme: In CM-Versionen ab 6.10.7.0 wird der Wert von `mailbox.default.session.mail.imap.partialfetch` bei der initialen Einrichtung eines ConSol CM-Systems auf „false“ gesetzt. Bei einer Aktualisierung eines vorhandenen ConSol CM-Systems bleibt der Wert der Property unverändert, wenn die Property bereits vorhanden ist. Wenn die Property noch nicht vorhanden ist, wird sie mit dem Standardwert hinzugefügt.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**:
- **Seit**: 6.9.4.0

mailbox.default.task.delete.read.messages

- **Modul**: cmas-nimh
- **Beschreibung**: Legt fest, ob Nachrichten nach dem Lesen aus dem Postfach entfernt werden sollen. Beim IMAP-Protokoll werden Nachrichten standardmäßig als SEEN gekennzeichnet. Bei POP3 wird die Nachricht nur gelöscht, wenn der Wert auf true gesetzt ist. Anderenfalls bleibt die Nachricht auf dem Server, was zu einer Endlos-Leseschleife führt. Standardwert: false.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.4.0

mailbox.default.task.enabled

- **Modul**: cmas-nimh
- **Beschreibung**: Mit dieser System-Property kann der Service Thread eines bestimmten Pollers deaktiviert werden. Standardwert: true.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja

- **Beispielwert**: false
- **Seit**: 6.4.0

mailbox.default.task.interval.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Standard-Intervall für den Abruf von Postfächern. Standardwert: 60 Sekunden
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

mailbox.default.task.max.message.size

- **Modul**: cmas-nimh
- **Beschreibung**: Maximale Größe von E-Mails (d. h. E-Mail plus Attachments). E-Mails, die größer als dieser Wert sind, werden nicht automatisch von NIMH verarbeitet. Sie werden in der Datenbank (Tabelle `cmas_nimh_archived_mail`) gespeichert und erscheinen daher in den E-Mail-Backups im Admin Tool (siehe Abschnitt E-Mail-Backups). Von dort können sie erneut gesendet, in das Dateisystem geladen oder gelöscht werden. Für diese Operationen ist die Größe der Nachricht nicht relevant. Standardmäßig auf 10 MB (10485760) gesetzt.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 10485760
- **Seit**: 6.4.0

mailbox.default.task.max.messages.per.run

- **Modul**: cmas-nimh
- **Beschreibung**: Anzahl der Nachrichten, die gleichzeitig aus dem Postfach abgeholt werden. Muss mit dem Transaktions-Timeout korrelieren. Standardwert: 20.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja

- **Beispielwert**: 60
- **Seit**: 6.4.0

mailbox.default.task.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Nach dieser Zeit (der Inaktivität) wird der Service Thread als beschädigt betrachtet und automatisch neu gestartet. Standardwert: 120 Sekunden
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

mailbox.default.task.transaction.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Standard-Transaktions-Timeout für Transaktionen, die E-Mails abrufen. Sollte mit der Anzahl der Nachrichten, die gleichzeitig abgeholt werden, korrelieren. Standardwert: 60 Sekunden
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

mailbox.polling.threads.mail.log.enabled

- **Modul**: cmas-nimh
- **Beschreibung**: Ermöglicht das Protokollieren von E-Mails, was in Cluster-Umgebungen besonders wichtig ist (wird dort als Semaphor genutzt).
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true (Standardwert)
- **Seit**: 6.9.4.1

mailbox.polling.threads.number

- **Modul**: cmas-nimh
- **Beschreibung**: Anzahl der Threads für den Zugriff auf Postfächer. Standardwert: 1.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 1
- **Seit**: 6.4.0

mailTemplateAboveQuotedText

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Legt das Verhalten der E-Mail-Vorlagen im Ticket-E-Mail-Editor fest, wenn eine andere E-Mail zitiert wird, d. h. auf diese geantwortet oder diese weitergeleitet wird. Wird oft verwendet, um die Signatur korrekt zu platzieren.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.2.4

max.licences.perUser

- **Modul**: cmas-core-server
- **Beschreibung**: Setzt die maximale Anzahl an Lizenzen, die ein einzelner Benutzer benutzen kann (z. B. durch Anmelden in einem anderen Browser). Standardmäßig ist dieser Wert nicht beschränkt.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 10
- **Seit**: 6.8.4.5

maxSizePerPagemapInMegaBytes

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Mit diesem Parameter wird die Größe (in MB) der Datei definiert, die vom Wicket-Framework pro Benutzer-Session, d. h. für jeden aktuell angemeldeten Bearbeiter, erstellt wird. Die Datei wird verwendet, um Seiten während der laufenden Session zu speichern. Wenn die definierte Größenbegrenzung erreicht wurde und neue Einträge hinzugefügt wurden, werden die ältesten Einträge entfernt. Aufgrund dieses Verhaltens wird ein Bearbeiter, der im Web Client mit einer "alten" Seite arbeitet, auf die Hauptseite (normalerweise die Dashboard-Seite) weitergeleitet, sobald die "alte" Seite aus der Datei entfernt wird.
  Wenn Bearbeiter in ConSol CM mit einer großen Anzahl an offenen Tabs arbeiten und sich darüber beschweren, dass Sie auf die Hauptseite weitergeleitet werden, kann es nützlich sein, diesen Parameter zu erhöhen. In großen Systemen können Sie zum Beispiel einen Wert von 45 oder 50 verwenden. Da dies die Größe der auf der Festplatte gespeicherten Datei ist, hängt der Maximalwert vom verfügbaren Speicherplatz ab, daher ist ein zu großer Wert nicht empfehlenswert.
- **Typ**: integer
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 15
- **Seit**: 6.3.5

monitoring.engineer.login

- **Modul**: cmas-core-server
- **Beschreibung**: Login des Monitoring-Bearbeiters.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: nagios
- **Seit**: 6.9.3.0

monitoring.unit.login

- **Modul**: cmas-core-server
- **Beschreibung**: Login der Monitoring-Unit.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja

- **Optional**: ja
- **Beispielwert**: nagios
- **Seit**: 6.9.3.0

nimh.enabled

- **Modul**: cmas-core-server
- **Beschreibung**: Aktiviert den NIMH-Dienst. Im Cluster muss die Node-ID angehängt werden, z. B. `nimh.enabled.NODEID` = „true“.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.9.4.0

notification.error.description

- **Modul**: cmas-dwh-server
- **Beschreibung**: Text für E-Mails mit Fehlermeldungen des DWH.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: Fehler aufgetreten
- **Seit**: 6.0.1

notification.error.from

- **Modul**: cmas-dwh-server
- **Beschreibung**: From-Adresse für E-Mails mit Fehlermeldungen des DWH.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.0.1

notification.error.subject

- **Modul**: cmas-dwh-server
- **Beschreibung**: Betreff für E-Mails mit Fehlermeldungen des DWH.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: Fehler aufgetreten
- **Seit**: 6.0.1

notification.error.to

- **Modul**: cmas-dwh-server
- **Beschreibung**: To-Adresse für E-Mails mit Fehlermeldungen des DWH.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: myuser@consol.de
- **Seit**: 6.0.1

notification.finished_successfully.description

- **Modul**: cmas-dwh-server
- **Beschreibung**: Text für E-Mails des DWHs, wenn eine Übertragung erfolgreich beendet wurde.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: Übertragung erfolgreich beendet
- **Seit**: 6.0.1

notification.finished_successfully.from

- **Modul**: cmas-dwh-server
- **Beschreibung**: From-Adresse für E-Mails des DWHs, wenn eine Übertragung erfolgreich beendet wurde.
- **Typ**: string
- **Neustart erforderlich**: nein

- **System**: ja
- **Optional**: ja
- **Seit**: 6.0.1

notification.finished_successfully.subject

- **Modul**: cmas-dwh-server
- **Beschreibung**: Betreff für E-Mails des DWHs, wenn eine Übertragung erfolgreich beendet wurde.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: Übertragung erfolgreich beendet
- **Seit**: 6.0.1

notification.finished_successfully.to

- **Modul**: cmas-dwh-server
- **Beschreibung**: To-Adresse für E-Mails des DWHs, wenn eine Übertragung erfolgreich beendet wurde.
- **Typ**: string
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: myuser@consol.de
- **Seit**: 6.0.1

notification.finished_unsuccessfully.description

- **Modul**: cmas-dwh-server
- **Beschreibung**: Text für E-Mails des DWHs, wenn eine Übertragung nicht erfolgreich beendet wurde.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: Übertragung nicht erfolgreich beendet
- **Seit**: 6.0.1

notification.finished_unsuccessfully.from

- **Modul**: cmas-dwh-server
- **Beschreibung**: From-Adresse für E-Mails des DWHs, wenn eine Übertragung nicht erfolgreich beendet wurde.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.0.1

notification.finished_unsuccessfully.subject

- **Modul**: cmas-dwh-server
- **Beschreibung**: Betreff für E-Mails des DWHs, wenn eine Übertragung nicht erfolgreich beendet wurde.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: Übertragung nicht erfolgreich beendet
- **Seit**: 6.0.1

notification.finished_unsuccessfully.to

- **Modul**: cmas-dwh-server
- **Beschreibung**: To-Adresse für E-Mails des DWHs, wenn eine Übertragung nicht erfolgreich beendet wurde.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: myuser@consol.de
- **Seit**: 6.0.1

notification.host

- **Modul**: cmas-dwh-server
- **Beschreibung**: Hostname des E-Mail-Servers (SMTP) für das Senden von DWH-E-Mails.
- **Typ**: string

- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: myserver.consol.de
- **Seit**: 6.0.1

notification.password

- **Modul**: cmas-dwh-server
- **Beschreibung**: Passwort für das Senden von DWH-E-Mails. (optional)
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.0.1

notification.port

- **Modul**: cmas-dwh-server
- **Beschreibung**: SMTP-Port für das Senden von DWH-E-Mails.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 25
- **Seit**: 6.0.1

notification.protocol

- **Modul**: cmas-dwh-server
- **Beschreibung**: Das Protokoll, das für das Senden von E-Mails aus dem DWH verwendet wird.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: pop3\

notification.tls.enabled

- **Modul**: cmas-dwh-server
- **Beschreibung**: Aktiviert SMTP über SSL/TLS (SMTPS) für das Senden von Benachrichtigungs-E-Mails aus dem DWH. Der Standardwert ist „false“. Wenn der Wert auf „true“ gesetzt ist, ist SMTPS für das Senden von DWH-Benachrichtigungen aktiviert.
- **Typ**: string
- **Neustart erforderlich**: ja
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.11.1.6

notification.username

- **Modul**: cmas-dwh-server
- **Beschreibung**: Benutzername (SMTP) für das Senden von DWH-E-Mails.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: myuser
- **Seit**: 6.0.1

number.of.tasks

- **Modul**: cmas-core-server
- **Beschreibung**: Anzahl der Threads, die das Task Execution Framework (TEF) verwendet.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 1
- **Seit**: 6.9.4.0

outdated.lock.age

- **Modul**: cmas-workflow-jbpm
- **Beschreibung**:
- **Typ**: integer

- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 60000
- **Entfernt in**: 6.8.0
- **Ersetzt durch**: cmas-workflow-engine, jobExecutor.lockTimeout.seconds

pagemapLockDurationInSeconds

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Anzahl der Sekunden, die vergehen müssen, bevor eine Pagemap als zu lange gelockt angesehen wird.
- **Typ**: integer
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.7.3

password.reset.mail.from

Modul: cmas-core-security

Beschreibung: Die From-Adresse der E-Mail, die an einen Kunden gesendet wird, der (über den Link *Passwort vergessen?*) in CM/Track ein neues Passwort anfordert bzw. die *From-Adresse* der E-Mail, die an einen Bearbeiter gesendet wird, der (über den Link *Passwort vergessen?*) im Web Client ein neues Passwort anfordert.

Typ: String

Neustart erforderlich: nein

System: nein

Optional: nein

Beispielwert: mypwreset@consol.de

Seit: 6.11.0.1

policy.password.age

- **Modul**: cmas-core-security
- **Beschreibung**: Maximaler Gültigkeitszeitraum, in Tagen, Beispiel „183“ (6 Monate), Standardwert: „5500“ (= 15 Jahre, d. h. es wird keine Passwortänderung erzwungen). Falls Sie möchten, dass der Bearbeiter sein Passwort sofort ändert, verwenden Sie einen der folgenden Werte:

  - **0**
    Der Bearbeiter muss sein Passwort bei der nächsten Anmeldung ändern.
  - **1**
    Der Bearbeiter muss sein Passwort am nächsten Tag ändern.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 5500 (15 Jahre, Standardwert)
- **Seit**: 6.10.1.0

policy.password.pattern

- **Modul**: cmas-core-security
- **Beschreibung**: Definiert das Passwortmuster.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: "^.3,$" (Standardwert)
- **Seit**: 6.10.1.0

policy.rotation.ratio

- **Modul**: cmas-core-security
- **Beschreibung**: Legt fest, wie oft sich das Passwort wiederholen darf. Ist der Wert z. B. 10, darf das neue Passwort nicht unter den zehn letzten Passwörtern des Benutzers sein.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 1 (Standardwert)
- **Seit**: 6.10.1.0

policy.track.username.case.sensitive

- **Modul**: cmas-core-security
- **Beschreibung**: Definiert, ob bei Kunden-Benutzernamen in CM/Track bei der Anmeldung zwischen Groß- und Kleinbuchstaben unterschieden wird.
- **Typ**: boolean

- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true (Standardwert)
- **Seit**: 6.11.0.0

policy.username.case.sensitive

- **Modul**: cmas-core-security
- **Beschreibung**: Legt fest, ob bei Benutzernamen die Groß- und Kleinschreibung beachtet werden muss.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true (Standardwert)
- **Seit**: 6.10.1.0

postActivityExecutionScriptName

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Definiert den Namen des Skripts, das nach jeder Workflow-Aktivität ausgeführt wird (siehe Abschnitt PostActivityExecutionScript. Wenn kein Skript ausgeführt werden soll, lassen Sie diesen Wert leer.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: postActivityExecutionHandler
- **Seit**: 6.2.0

queue.polling.threads.number

- **Modul**: cmas-nimh
- **Beschreibung**: Anzahl der Threads, die zur Überwachung der E-Mail-Warteschlange gestartet werden. Standardwert: 1.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja

- **Beispielwert**: 1
- **Seit**: 6.4.0

queue.polling.threads.shutdown.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Wartezeit nach dem Shutdown-Signal. Wenn der Timeout erreicht ist, wird der Thread beendet. Standardwert: 60.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

queue.polling.threads.watchdog.interval.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Intervall des Watchdog Threads. Standardwert: 30.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 30
- **Seit**: 6.4.0

queue.task.error.pause.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Anzahl der Sekunden, die der Queue-Poller nach einem Infrastrukturfehler (z. B. der Datenbank) wartet. Standardwert: 180 Sekunden
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 180
- **Seit**: 6.4.0

queue.task.interval.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Thread-Intervall zur Überwachung der Haupt-E-Mail-Warteschlange. Standardwert: 15.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 15
- **Seit**: 6.4.0

queue.task.max.retries

- **Modul**: cmas-nimh
- **Beschreibung**: Maximale Anzahl der Versuche, eine E-Mail nach einer Exception erneut zu verarbeiten. Ist diese erreicht, wird die E-Mail archiviert. Die archivierte E-Mail kann über das NIMH API (oder das Admin Tool) wieder aktiviert werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 10
- **Seit**: 6.4.0

queue.task.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Nach dieser Zeit (der Inaktivität) wird der Service Thread als beschädigt betrachtet und automatisch neu gestartet. Standardwert: 600 Sekunden
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 600
- **Seit**: 6.4.0

queue.task.transaction.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Transaktions-Timeout für die E-Mail-Verarbeitung in der Warteschlange. Standardwert: 60.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

queuesExcludedFromGS

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Durch Kommas getrennte Liste von Queue-Namen, die von der Schnellsuche ausgeschlossen werden sollen.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.0

recent.items.cleanup.cluster.node.id

- **Modul**: cmas-core-server
- **Beschreibung**: Wert einer `-Dcmas.clusternode.id`, die den Node angibt, der die zuletzt verwendeten Elemente bereinigt.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 1 (unter Annahme, dass der Cluster-Node mit dem Parameter `-Dcmas.clusternode.id=1` gestartet wurde)
- **Seit**: 6.11.0.1

recent.items.cleanup.interval.minutes

- **Modul**: cmas-core-server
- **Beschreibung**: Steuert das Zeitintervall (in Minuten), in dem das Entfernen von zuletzt verwendeten Elementen geprüft wird.

- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 60
- **Seit**: 6.11.0.1

recent.items.max.per.engineer

- **Modul**: cmas-core-server
- **Beschreibung**: Maximale Anzahl an zuletzt verwendeten Elementen pro Bearbeiter, die bei der Bereinigung (ältere Elemente werden gelöscht) beibehalten werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 500
- **Seit**: 6.11.0.1

recent.items.persistence.enabled

- **Modul**: cmas-core-server
- **Beschreibung**: Ermöglicht die Persistenz der zuletzt verwendeten Elemente, wenn false - verhindert das Speichern von neuen Elementen.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: true
- **Seit**: 6.11.1.0

recoverable.exceptions

- **Modul**: cmas-dwh-server
- **Beschreibung**: Durch Kommas getrennte Liste mit Exception-Definitionen: CLASS[+][:REGEX]. Die in der Liste enthaltenen Exceptions verhindern nicht, dass CM an den CMRF-Prozess sendet, sondern erzwingen einen erneuten Versuch. Wenn nach CLASS ein optionales '+' steht, gilt dies auch für Klassen, die CLASS erweitern.
- **Typ**: string
- **Neustart erforderlich**: nein

- **System**: ja
- **Optional**: ja
- **Beispielwert**: java.sql.SQLRecoverableException,java.lang.RuntimeException+:.*T.1\,2T.*
- **Seit**: 6.8.4.6

refreshTimeInCaseOfConcurrentRememberMeRequests

- **Modul**: cmas-workflow-jbpm
- **Beschreibung**: Legt die Aktualisierungszeit (in Sekunden) fest, nach der die Seite im Falle von gleichzeitigen Anfragen von Angemeldet bleiben neu geladen wird. Diese Funktion verhindert, dass ein Benutzer zu viele Lizenzen in Anspruch nimmt. Erhöhen Sie die Zeit, wenn immer noch Sessions belegt werden.
- **Typ**: integer
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 5
- **Seit**: 6.8.2

rememberMeLifetimeInMinutes

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Lebensdauer für *Angemeldet bleiben* in Minuten.
- **Typ**: integer
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1440
- **Seit**: 6.0

request.scope.transaction

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Ermöglicht die Deaktivierung von Scope-Transaktionen für Requests. Standardmäßig wird pro Request eine Transaktion verwendet. Wenn Sie diese Property auf „false“ setzen, wird eine Transaktion pro Aufruf einer Service-Methode erzeugt.
- **Typ**: boolean
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: ja

- **Beispielwert**: true
- **Seit**: 6.8.1

resetCode.expiriationPeriod

- **Modul**: cmas-core-security
- **Beschreibung**: Definiert die Gültigkeitsdauer des Links zum Zurücksetzen des Passworts in CM/Track.
- **Typ**: Integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 86400000 (Standardwert, 24 Stunden)
- **Seit**: 6.10.1

resource.replace.batchSize

- **Modul**: cmas-core-server
- **Beschreibung**: Legt die Anzahl der Objekte fest, die in einer Ressourcen-Ersetzen-Aktion verarbeitet werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 5
- **Seit**: 6.10.0.0

resource.replace.timeout

- **Modul**: cmas-core-server
- **Beschreibung**: Transaktions-Timeout (in Sekunden) für einen Schritt einer Ressourcen-Ersetzen-Aktion.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 120
- **Seit**: 6.10.0.0

scene

- **Modul**: cmas-setup-scene
- **Beschreibung**: Szenariodatei, die während des Setups importiert wurde (kann leer gelassen werden).
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: vfszip:/P:/dist/target/jboss/server/cmas/deploy/cm-dist-6.5.1-SNAPSHOT.ear/APP-INF/lib/dist-scene-6.5.1-SNAPSHOT.jar/META-INF/cmas/scenes/helpdesk-sales_scene.jar/
- **Seit**: 6.0

script.logging.threshold.seconds

- **Modul**: cmas-core-server
- **Beschreibung**: Legt die Zeit (in Sekunden) fest, nach deren Ablauf während einer Skript-Ausführung eine Warnung in der Log-Datei generiert wird.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 10 (Standardwert)
- **Seit**: 6.10.1.0

script.validation.interval.seconds

- **Modul**: cmas-app-admin-tool
- **Beschreibung**: Intervall in Sekunden zwischen zwei Code-Überprüfungen im Quelltext-Editor des Admin Tools oder Process Designers
- **Typ**: Integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: nein
- **Beispielwert**: 1 (Standardwert)
- **Seit**: 6.11.0.1

searchPageSize

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Standardgröße der Seiten für Suchergebnisse.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 20
- **Seit**: 6.0

searchPageSizeOptions

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Optionen für die Größe der Seiten für Suchergebnisse.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 10|20|30|40|50|75|100
- **Seit**: 6.0

security.fields.customer.exposure.check.enabled

- **Modul**: cmas-restapi-core
- **Beschreibung**: Aktiviert die Prüfung der Annotation `customer exposure` für Ticketfelder.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true (Standardwert)
- **Seit**: 6.10.5.4

security.restrict.unit.access.to.own.data

- **Modul**: cmas-restapi-core
- **Beschreibung**: Wenn die Property auf „true“ gesetzt ist, wird eine zusätzliche Prüfung durchgeführt, wenn sich ein Benutzer als Kunde über die REST-API anmeldet, z. B. in CM/Track.

Wenn er Kundendaten anfordert, werden nur die Firma des Benutzers oder andere Kontakte der Firma des Benutzers zurückgegeben. Wenn die Property auf „false“ gesetzt ist, wird keine zusätzliche Sicherheitsprüfung durchgeführt und die früheren Sicherheitsregeln gelten.

- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: true (Standardwert)
- **Seit**: 6.9.2.14

serial.mods.tracking.enabled

- **Modul**: cmas-core-server
- **Beschreibung**: Systemnahes technisches Flag, das entscheidet, ob die serielle Nachverfolgung von Änderungen für Entitäten aktiviert ist. Wenn diese Property aktiviert ist, gibt es keinen `StackOverflow`-Fehler, wenn Abhängigkeiten zwischen zwei Entitäten (zum Beispiel Bearbeiter und Ticket) eine Endlosschleife und infolgedessen einen StackOverflow auslösen. Diese Property muss manuell zur Konfiguration hinzugefügt werden. Sie wird nicht während der Einrichtung oder Aktualisierung zur Systemkonfiguration hinzugefügt.

  > ⚠ Aktivieren Sie das in diesem Abschnitt beschriebene eingeschränkte Verhalten bei Ticketänderungen nur, wenn Sie von einem ConSol-Vertreter dazu aufgefordert werden. Es handelt sich um ein systemnahes technisches Flag mit komplexen Auswirkungen auf das Systemverhalten und sollte daher nicht ohne genaue Untersuchung verwendet werden.

- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false (Standardwert)
- **Seit**: 6.10.7.0, 6.11.0.5

server.session.archive.reaper.interval

- **Modul**: cmas-core-server
- **Beschreibung**: Reaper-Intervall (in Sekunden) von archivierten Server-Sessions.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja

- **Beispielwert**: 60
- **Seit**: 6.7.1

server.session.archive.timeout

- **Modul**: cmas-core-server
- **Beschreibung**: Timeout der Gültigkeit der Server-Session-Archive (in Tagen). Nach diesem Zeitraum werden die Informationen zur Session aus der Datenbank entfernt.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 31
- **Seit**: 6.7.1

server.session.reaper.interval

- **Modul**: cmas-core-server
- **Beschreibung**: Interval (in Sekunden), in dem der sog. Reaper inaktive (= beendete) Server-Sessions löscht.
- **Typ**: integer
- **Neustart erforderlich**: Nur Session Service
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 60
- **Seit**: 6.6.1, 6.7.1

server.session.timeout

- **Modul**: cmas-core-server
- **Beschreibung**: Server-Session-Timeout (in Sekunden) für verbundene Clients. Jeder Client kann dieses Timeout mit benutzerdefinierten Werten mittels seiner ID (ADMIN_TOOL, WEB_CLIENT, WORKFLOW_EDITOR, TRACK (vor 6.8 bitte PORTER verwenden), ETL, REST) überschreiben. Diese wird an den Namen der System-Property angehängt, z. B. `server.session.timeout.ADMIN_TOOL`.
  Siehe auch die Attribute der Seitenanpassung *updateTimeServerSessionActivityEnabled* und *updateTimeServerSessionActivity*, beide vom Typ *cmApplicationCustomization*.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein

- **Beispielwert**: 1800
- **Seit**: 6.6.1, 6.7.1

Detaillierte Erklärung für das Admin Tool:

- server.session.timeout.ADMIN_TOOL
  Definiert den Zeitraum, den der Server eine Session im Admin Tool als gültig betrachtet, wenn keine Aktivität im Admin Tool der Session erfolgt. Das Admin Tool kennt diesen Wert nicht, es bemerkt nur eine ungültige Session, wenn es länger keine Aktivität gegeben hat.
- admin.tool.session.check.interval
  Definiert den Zeitraum zwischen zwei Überprüfungen durch das Admin Tool, ob der Server die Session noch als gültig betrachtet.

Wenn zum Beispiel admin.tool.session.check.interval = 60, fragt das Admin Tool den Server einmal pro Minute, ob die Session noch aktiv/gültig ist. Wenn server.session.timeout.ADMIN_TOOL = 600 erhält das Admin Tool die Antwort, dass die Session ungültig ist, nach zehn Minuten der Inaktivität.

serverPoolingInterval

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Definiert die Zeit in Sekunden, nach der der Pooling-Server die Caches auf dem Web-Layer ungültig macht.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 5
- **Seit**: 6.1.0

skip-ticket

- **Modul**: cmas-dwh-server
- **Beschreibung**: Tickets werden bei der Übertragung/Aktualisierung nicht übermittelt.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.6.19
- **Entfernt in**: 6.8.1

skip-ticket-history

- **Modul**: cmas-dwh-server
- **Beschreibung**: Ticketprotokoll wird bei der Übertragung/Aktualisierung nicht übermittelt.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.6.19
- **Entfernt in**: 6.8.1

skip-unit

- **Modul**: cmas-dwh-server
- **Beschreibung**: Units werden bei der Übertragung/Aktualisierung nicht übermittelt.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.6.19
- **Entfernt in**: 6.8.1

skip-unit-history

- **Modul**: cmas-dwh-server
- **Beschreibung**: Unit-Protokoll wird bei der Übertragung/Aktualisierung nicht übermittelt.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.6.19
- **Entfernt in**: 6.8.1

skip.wfl.transfer.cleanup

- **Modul**: cmas-core-server
- **Beschreibung**: Wenn diese System-Property auf *true* gesetzt wird, wird das Workflow-Cleanup nach der Übertragung übersprungen.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false (Standardwert)
- **Seit**: 6.9.4.1

skip.wfl.transfer.translations.cleanup

- **Modul**: cmas-core-server
- **Beschreibung**: Aktiviert das Überspringen der Bereinigung von lokalisierten Properties von entfernten Workflow-Elementen.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false (Standardwert)
- **Seit**: 6.10.5.5

split.history

- **Modul**: cmas-dwh-server
- **Beschreibung**: Ändert das SQL-Statement dahingehend, dass Ticketprotokolle während der Übertragung an das DWH nicht für alle Tickets auf einmal abgeholt werden, sondern für ein Ticket pro SQL-Statement.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.8.0

start.groovy.task.enabled

- **Modul**: cmas-app-admin-tool
- **Beschreibung**: Um Skripte vom Typ *Task* im Admin Tool (Navigationsgruppe *Dienste*, Navigationselement *Task-Ausführung*) ausführen zu können, ist es nötig, den Button *Start* zu aktivieren, der standardmäßig ausgeblendet ist. Setzen Sie hierfür diese System-Property auf *true*.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.9.4.0

statistics.calendar

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.client.group

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.contact.role

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string

- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.content.entry

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.content.entry.class

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.content.entry.history

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.customer.definition

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.engineer

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.enum.group

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.field.definition

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein

- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.group.definition

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.locale

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.localized.property

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.mla

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.project

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.queue

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.resource

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein

- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.resource.group

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.resource.history

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.resource.relation.definition

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.resource.type

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.ticket

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.ticket.function

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.ticket.history

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein

- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.time.booking

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.timestamp

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.unit

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.unit.history

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.unit.relation.definition

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.workflow

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

strict.utf.bmp.enabled

- **Modul**: cmas-core-server
- **Beschreibung**: In ConSol CM-Versionen unter 6.10.6 konnten eingehende E-Mails, deren Betreff 4-Byte UTF8-Zeichen enthält, von einigen Installationen mit einer MySQL-Datenbank-Engine nicht verarbeitet werden. Der Grund dafür ist, dass die Codierungs-/Collation-Konfiguration der Datenbank die 2-Byte BMP (Basic Multilingual Plane), Plane 0, verwendet und diese in einigen Installationen aus technischen Gründen nicht geändert werden kann. Andere

Datenbanken sind davon nicht betroffen. E-Mails mit dieser Codierung konnten in allen CM-Versionen unter 6.10.6 nicht ins System importiert werden. Um dieses Problem zu umgehen, ist diese System-Property zur Konfiguration verfügbar.
Mit der Einstellung „true“ werden alle 4-Byte UTF8-Zeichen herausgefiltert, bevor eine Interaktion mit der Datenbank stattfindet, sodass die oben beschriebenen Probleme nicht auftreten.
Der Property-Wert „true“ ist die Standardeinstellung für MySQL-Datenbanken und „false“ ist die Standardeinstellung für alle anderen Datenbanken, in denen diese Property nicht benötigt wird. Ändern Sie den Wert für eine MySQL-Datenbank nur, wenn die Datenbankeinstellungen 4-Byte-Zeichen unterstützen.

- **Typ**: Boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 100
- **Seit**: 6.10.6.0

supportEmail

- **Modul**: cmweb-server-adapter
- **Beschreibung**:
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.0
- **Entfernt in**: 6.11.0.1

synchronize.master.address

- **Modul**: cmas-core-index-common
- **Beschreibung**: Wert der Java-System-Property `-Dcmas.http.host.port`, die angibt, unter welcher URL der Index-Master-Server erreichbar ist. Standard ist Null. Seit CM-Version 6.6.17 ist dieser Wert beim Setup konfigurierbar, um den initialen Index-Master-Server zu bestimmen. Bitte beachten Sie, dass das Verändern dieses Wertes nur erlaubt ist, wenn alle Cluster-Nodes zum Empfang von Index-Veränderungen gestoppt sind.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja

- **Beispielwert**: 127.0.0.1:80
- **Seit**: 6.6.0

synchronize.master.security.token

- **Modul**: cmas-core-index-common
- **Beschreibung**: Das Passwort für den URL-Zugriff auf den Index-Snapshot, z. B. für die Index-Synchronisierung oder für Backups.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: token
- **Seit**: 6.6.0

synchronize.master.security.user

- **Modul**: cmas-core-index-common
- **Beschreibung**: Der Benutzername für den URL-Zugriff auf den Index-Snapshot, z. B. für die Index-Synchronisierung oder für Backups.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: user
- **Seit**: 6.6.0

synchronize.master.timeout.minutes

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie oft die Index-Synchronisierung ausgehend vom aktuellen Master-Server fehlschlagen darf, bis ein neuer Master für die Index-Reparatur ausgewählt wird. Standardwert ist 5. Seit CM-Version 6.6.17 ist dieser Wert im Setup konfigurierbar, wobei 0 bedeutet, dass der Master-Server nie geändert wird (Failover-Mechanismus deaktiviert).
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 5
- **Seit**: 6.6.0

synchronize.megabits.per.second

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie viel Bandbreite der Master-Server verbrauchen darf, um Index-Änderungen an die Slave-Server zu übermitteln. Standardwert ist 85. Nutzen Sie nicht die gesamte verfügbare Bandbreite, um die Index-Änderungen zwischen den Hosts zu übermitteln, da dies dafür sorgen kann, dass die Nodes des Clusters nicht mehr synchron sind.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 85
- **Seit**: 6.6.0

synchronize.sleep.millis

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie oft jeder Slave-Server den Master-Server auf Änderungen am Index abfragt. Standardwert ist 1000.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1000
- **Seit**: 6.6.0

task.execution.interval.seconds

- **Modul**: cmas-core-server
- **Beschreibung**: Zeit in Sekunden zwischen dem Ende eines abgeschlossenen Tasks im TEF (Task Execution Framework) und dem Beginn des nächsten Tasks.
- **Typ**: Integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: nein
- **Beispielwert**: 5
- **Seit**: 6

task.execution.node.id

Modul: cmas-core-server

Beschreibung: Nur in geclusterten Umgebungen relevant. Die ID des Nodes, auf dem automatisch gestartete Skripte des TEF (Task Execution Framework) ausgeführt werden. (Manuell über das Admin Tool gestartete Skripte werden auf dem Rechner ausgeführt, auf dem das Admin Tool läuft.)

Typ: Integer

Neustart erforderlich: nein

System: nein

Optional: nein

Beispielwert: 2

Seit: 6.11.0.1

task.panel.refresh.interval.seconds

- **Modul**: cmas-app-admin-tool
- **Beschreibung**: Zeit in Sekunden, nachdem die Task-Liste (im Admin Tool) des Task Execution Framework aktualisiert wird.
- **Typ**: Integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: nein
- **Beispielwert**: 10
- **Seit**: 6.10.5.3 (wird beim Update von einer Version vor 6.10.5.3 nicht automatisch hinzugefügt!)

themeOverlay

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Name des verwendeten Themen-Overlays.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: consolINT
- **Seit**: 6.0

ticket.delete.timeout

- **Modul**: cmas-core-server
- **Beschreibung**: Transaktions-Timeout (in Sekunden) beim Löschen von Tickets.
- **Typ**: integer
- **Neustart erforderlich**: nein

- **System**: ja
- **Optional**: nein
- **Beispielwert**: 60
- **Seit**: 6.1.3

ticket.from.incoming.message.accepted.links

- **Modul**: cmas-core-server
- **Beschreibung**: Liste mit Domänen, zu denen die Links in eingehenden E-Mails und Links in Kommentaren, die über die REST API hinzugefügt wurden, im Ticketprotokoll angeklickt werden können. Es können reguläre Ausdrücke verwendet werden, um die erlaubten URLs anzugeben. Es können mehrere, durch Leerzeichen getrennte URLs angegeben werden. Die URL muss mit einem der erlaubten Protokolle (http, https, ftp, ftps, file, mailto) beginnen. Alle anderen Links werden entfernt, d. h. der Link wird im Ticketprotokoll als Text dargestellt, kann aber nicht angeklickt werden. Ist die Property leer, werden alle Links entfernt. Der reguläre Ausdruck `.+` kann verwendet werden, um alle Domänen zu erlauben.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: nein
- **Beispielwert**: https://.*\.consol\.de (erlaubt Links auf „https://<beliebig>.consol.de“)
- **Seit**: 6.11.1.7

> ⚠ Beachten Sie, dass ConSol CM durch das Whitelisting von Domänen anfällig für Cross-Site-Scripting und andere Angriffe werden kann. Wählen Sie die Domänen in Ihrer Whitelist sorgfältig aus!

ticketListRefreshIntervalInSeconds

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Aktualisierungsintervall für die Ticketliste (in Sekunden).
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 180
- **Seit**: 6.0

ticketListSizeLimit

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Maximale Anzahl von Tickets in der Ticketliste.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 100
- **Seit**: 6.0

tickets.delete.size

- **Modul**: cmas-core-server
- **Beschreibung**: Definiert die Anzahl der Tickets, die pro Transaktion gelöscht werden. Standardmäßig ist dieser Wert 10.
- **Typ**: integer
- **Neustart erforderlich**: Nur Session Service
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 10
- **Seit**: 6.8.1

time.buffer

- **Modul**: cmas-dwh-server
- **Beschreibung**: Anzahl der Minuten zur Erweiterung des Datums des Starts des Live-Modus.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 5
- **Seit**: 6.8.1.11

transaction.timeout.minutes

- **Modul**: cmas-core-server
- **Beschreibung**: Setzt den Timeout für den Task Execution Service des TEF, d. h. ein Durchlauf eines Tasks muss vor dem Ablauf dieser Zeitspanne abgeschlossen sein. Die Änderungen sind nur für neue Aufgaben sichtbar, deren Ausführung nach der Konfigurationsänderung beginnt.

- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 10*3600 (10 Stunden - Standardwert)
- **Seit**: 6.10

tx.read.only.mode.enabled

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Aktiviert nur-lesende Transaktionen für ein schnelleres Laden der Seite. Dieses Transaktionsverhalten wurde in der 6.11.0 eingeführt und die Property ermöglicht es, das alte Verhalten wiederherzustellen. Ändern Sie diesen Wert nur, wenn Sie Transaktionsprobleme haben und von ConSol dazu aufgefordert werden.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true (Standardwert)
- **Seit**: 6.11

unit.description.mode

- **Modul**: cmas-core-server
- **Beschreibung**: Legt fest, ob die Kontaktbeschreibungen im Ticketprotokoll aus der Datenbank abgerufen oder dynamisch über Templates erzeugt werden. Der Wert „DYNAMIC“ ist aus Performance-Sicht etwas kostspieliger. „PROTOCOL“ ist schneller, gibt aber historische Namen zurück, die unter Umständen veraltet sind. Verwenden Sie „PROTOCOL“, wenn Sie viele Protokolleinträge von vielen unterschiedlichen Kontakten haben. Dies ist auch der Standardwert in CM-Versionen 6.11.1.1 und höher. In CM-Versionen bis 6.11.1.0 ist der Standardwert „DYNAMIC“.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: PROTOCOL
- **Seit**: 6.11.0

unit.replace.batchSize

- **Modul**: cmas-core-server
- **Beschreibung**: Legt die Anzahl der Objekte fest, die in einer Unit-Ersetzen-Aktion verarbeitet werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 5
- **Seit**: 6.8.2

unit.replace.timeout

- **Modul**: cmas-core-server
- **Beschreibung**: Transaktions-Timeout (in Sekunden) für einen Schritt einer Unit-Ersetzen-Aktion.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 120
- **Seit**: 6.8.2

unit.transfer.order

- **Modul**: cmas-dwh-server
- **Beschreibung**: Legt fest, in welcher Reihenfolge Kundenfeldgruppen an das DWH übertragen werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: company;customer
- **Seit**: 6.6.19
- **Entfernt in**: 6.8.1

unitIndexSearchResultSizeLimit

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Maximale Anzahl der Units in der Ergebnisliste, wenn nach Units gesucht wird (Beispiel: Kontaktsuche).
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 5
- **Seit**: 6.0

unused.content.remover.cluster.node.id

- **Modul**: cmas-core-server
- **Beschreibung**: Wert eines `cmas.clusternode.id`, der angibt, welcher Node nicht verwendete Ticket-Attachments und Unit-Inhaltseinträge entfernt.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 1 (in der Annahme, dass der Cluster-Node mit dem Parameter `-Dcmas.clusternode.id=1` gestartet ist)
- **Seit**: 6.9.0.0

unused.content.remover.enabled

- **Modul**: cmas-core-server
- **Beschreibung**: Legt fest, ob das Entfernen ungenutzter Ticket-Attachments und Unit-Inhaltseinträge durchgeführt werden soll.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: true
- **Seit**: 6.9.0.0

unused.content.remover.polling.minutes

- **Modul**: cmas-core-server
- **Beschreibung**: Legt fest, wie oft überprüft werden soll, ob ungenutzte Ticket-Attachments und Unit-Inhaltseinträge zum Entfernen vorhanden sind.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 15
- **Seit**: 6.9.0.0

unused.content.remover.ttl.minutes

- **Modul**: cmas-core-server
- **Beschreibung**: Mindestzeitraum, nach dem ungenutzte Ticket-Attachments und Unit-Inhaltseinträge entfernt werden können.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1440
- **Seit**: 6.9.0.0

update.6.11.0.0.sleep

- **Modul**: cmas-setup-hibernate
- **Beschreibung**: Hilfs-Property für die Skripte zur Update-Vorbereitung, die im Zusammenhang mit dem CM-Datenbank-Refactoring in Version 6.11 eingeführt wurden. Dies ist eine optionale Einstellung, die eine Verzögerung (in Millisekunden) nach jeder Schleifeniteration in den Vorbereitungsskripten ermöglicht. Das Einstellen einer Verzögerung sollte die Last auf der Datenbank verringern, zum Beispiel während der Arbeitszeiten. Diese Property kann entfernt werden, nachdem die Vorbereitungs-Tasks für das Update abgeschlossen sind.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.0, zur Verwendung in 6.10.5.x

update.6.11.0.0.timezone

- **Modul**: cmas-setup-hibernate
- **Beschreibung**: Hilfs-Property für die Migration des Ticketprotokolls (die neue Art zum Zählen der Protokollgruppen). Seit 6.11.0.0 sind die Gruppen konstant (Zeitraum vom 2 Stunden). Vor 6.11.0.0 waren die Gruppen nicht konstant und hingen von der Zeitzone des Kunden ab. Die Migrationsskripte verwenden einen alten Algorithmus zum Berechnen der Gruppen und benötigen daher die Information über die Zeitzone. Diese Property sollte auf die von den Kunden am häufigsten verwendete Zeitzone gesetzt werden. Wenn die Property nicht gesetzt ist, wird die Standardzeitzone des Servers verwendet (`TimeZone.getDefault()`). Die Property sollte vor dem Update auf die 6.11.0.0 gesetzt werden und wird nach der Migration automatisch gelöscht. Eine Liste der zulässigen Zeitzonen finden Sie zum Beispiel hier: http://joda-time.sourceforge.net/timezones.html.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: Europe/Berlin
- **Seit**: 6.11.0.0, verwendet vor einem Update auf diese Version

urlLogoutPath

- **Modul**: cmweb-server-adapter
- **Beschreibung**: URL die verwendet wird, wenn sich der Bearbeiter abmeldet. (wenn kein Wert gesetzt wird, wird nach dem Abmelden die Anmeldeseite angezeigt.)
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: http://intranet.consol.de
- **Seit**: 6.3.1

voCacheEnabled

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Diese Property aktiviert zusätzliches Caching, voCaching, von kompletten Objekten für den Web Client. Dies verbessert die Performance.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja

- **Beispielwert**: true
- **Seit**: 6.11.1.0

Hinweise:

- Seit ConSol CM-Version 6.11.1.1 ist der Standardwert „true“ in Umgebungen, die nicht zu einem Cluster gehören. Der Wert wird bei einer Neuinstallation oder einer Aktualisierung auf ConSol CM 6.11.1.1 automatisch auf „true“ gesetzt.
- Wenn voCaching aktiviert ist und lazy loading verwendet wird, um die Einträge des Ticketprotokolls einzuklappen, kann ein Bearbeiter diese Einträge nach dem Aufklappen nicht mehr einklappen, indem er die Seite neu lädt oder das Ticket aus dem Workspace öffnet.
- Wenn der dynamische Modus zum Anzeigen der Bearbeiter- und Kundennamen im Ticketprotokoll verwendet wird (konfiguriert in den System-Properties cmas-core-server, engineer.description.mode und cmas-core-server, unit.description.mode), wird die neue Version des Bearbeiter- bzw. Kundennamens erst angezeigt, nachdem das Ticket geändert wurde.
- Diese System-Property wird in Cluster-Umgebungen (Umgebungen, in denen cmas-core-shared, cluster.mode auf „true“ gesetzt ist) ignoriert. In Cluster-Umgebungen ist voCaching immer deaktiviert, um zu vermeiden, dass Änderungen an Objekten auf einem Node auf den anderen Nodes nicht sichtbar sind.

warmup.executor.enabled

- **Modul**: cmas-core-server
- **Beschreibung**: Legt fest, ob der Server beim Start asynchron anlaufen soll und nebenher andere Aufgaben erledigt (z. B. interne Caches füllen).
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: true
- **Seit**: 6.9.4.2

webSessionTimeoutInMinutes

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Session-Timeout in Minuten.
- **Typ**: integer
- **Neustart erforderlich**: ja
- **System**: ja

- **Optional**: nein
- **Beispielwert**: 180
- **Entfernt in**: 6.7.1
- **Ersetzt durch**: cmas-core-server, server.session.timeout

wfl.sticky.transfer.disabled

- **Modul**: cmas-core-server
- **Beschreibung**: Aktiviert die Verwendung von beibehaltenen Originalnamen von Workflow-Elementen.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.10.1.0

wicketAjaxRequestHeaderFilterEnabled

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Aktiviert Filter für Wicket AJAX-Anfragen, die auf veralteten Seiten mit Wicket 1.4-Skripten (CM-Version vor 6.8.0) stammen, nach einer Aktualisierung auf CM-Versionen ab 6.8.0.
- **Typ**: boolean
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.8.1

X-Frame-Options

- **Modul**: cmweb-server-http-headers
- **Beschreibung**: Beispiel-Property zum Verdeutlichen der Konfiguration von HTTP-Headern. In diesem Fall enthält der gesendete HTTP-Header das Feld *X-Frame-Options* mit dem Wert „SAMEORIGIN“.

  Jede Property im Modul `cmweb-server-http-headers` stellt ein Header-Feld dar. Der Name/Schlüssel der Property identifiziert das Header-Feld und der Wert der Property ist der im Header gesendete Feldwert.

> ⚠ Beachten Sie, dass zusätzliche HTTP-Antwort-Header korrekt und mit der offiziell vorgegebenen Schreibweise definiert werden müssen! Beachten Sie weiterhin, dass die korrekte Interpretation und Verarbeitung dieser Header komplett in Zuständigkeitsbereich des Client-Browsers liegt, der die Seite angefordert hat!

- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: SAMEORIGIN
- **Seit**: 6.10.8

H.2.2 Liste der System-Properties nach Bereich

Dieses Kapitel enthält eine Liste der System-Properties, die für die folgenden Bereiche relevant sind.

- CMRF- und DWH-Konfiguration
- Konfiguration von Indexer und Suche
- LDAP-Konfiguration
- E-Mail-Konfiguration
- Konfiguration des Aktivitätsintervalls
- Administrator-E-Mail-Adressen
- Konfiguration von HTTP-Headern

H.2.2.1 CMRF- und DWH-Konfiguration

autocommit.cf.changes

- **Modul**: cmas-dwh-server
- **Beschreibung**: Definiert, ob DWH-Aufgaben, die aufgrund von Konfigurationsänderungen an Ticketfeldern anfallen, automatisch ohne manuelle Interaktion im Admin Tool ausgeführt werden. Diese Property kann auch im Admin Tool im Navigationselement *DWH* gesetzt werden. Der Standardwert und empfohlene Wert ist *false*.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.7.0

batch-commit-interval

- **Modul**: cmas-dwh-server
- **Beschreibung**: Anzahl der Objekte in einer JMS-Nachricht. Höhere Werte bedeuten eine bessere Übertragungsperformance und größeren Speicherverbrauch.
  Ab ConSol CM-Version 6.11 wird diese Property nur verwendet, wenn die Paketgröße einer DWH-Operation nicht gesetzt ist. Dies kann nur der Fall sein, wenn der Befehl direkt an die Java MBean `consol.cmas.global.dwh.synchronizationService` gerichtet ist, z. B. durch Verwendung der Methode `update()`. Wenn eine DWH-Operation über das Admin Tool gestartet wird, gibt es immer einen Wert für die Paketgröße. Wenn dieser nicht explizit gesetzt ist, wird der Standardwert 1000 als Wert für `batch.commit.interval` verwendet.
- **Standardwert**: 1000
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 100
- **Seit**: 6.0.0

communication.channel

- **Modul**: cmas-dwh-server
- **Beschreibung**: Kommunikationskanal, mögliche Werte sind DIRECT (Datenbank-Kommunikationskanal, Standardwert seit 6.9.4.1) oder JMS (Standardwert vor 6.9.4.1). Diese System-Property muss vor 6.9.4.1 extra hinzugefügt werden.
- **Typ**: string

- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: DIRECT
- **Seit**: 6.8.5.0
- **Entfernt in**: 6.11.0.0 (Der einzige verfügbare und automatisch gesetzte Modus ist „DIRECT“.)

dwh.mode

- **Modul**: cmas-dwh-server
- **Beschreibung**: Aktueller Modus der DWH-Datenübermittlung. Mögliche Werte sind OFF, ADMIN, LIVE.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: OFF
- **Seit**: 6.0.1

ignore-queues

- **Modul**: cmas-dwh-server
- **Beschreibung**: Durch eine durch Kommas getrennte Liste von Queue-Namen wird hier festgelegt, dass Tickets dieser Queues nicht ins DWH übermittelt werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: QueueName1,QueueName2,QueueName3
- **Seit**: 6.6.19
- **Entfernt in**: 6.8.1

is.cmrf.alive

- **Modul**: cmas-dwh-server
- **Beschreibung**: Als Startpunkt sollte die Zeit genommen werden, bei der zuletzt eine Nachricht an das CMRF gesendet wurde. Wenn nach diesem Wert (in Sekunden) keine Antwort vom CMRF empfangen wird, wird ein DWH-Betriebsstatus mit der Fehlermeldung, dass das CMRF nicht erreichbar ist, erstellt.
- **Typ**: integer

- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1200
- **Seit**: 6.7.0

java.naming.factory.initial

- **Modul**: cmas-dwh-server
- **Beschreibung**: Factory-Java-Klasse für DWH context factory.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: org.jnp.interfaces.NamingContextFactory
- **Seit**: 6.0.1
- **Entfernt in**: 6.11.0.0

java.naming.factory.url.pkgs

- **Modul**: cmas-dwh-server
- **Beschreibung**:
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: org.jboss.naming:org.jnp.interfaces
- **Seit**: 6.0.1
- **Entfernt in**: 6.11.0.0

java.naming.provider.url

- **Modul**: cmas-dwh-server
- **Beschreibung**: URL des Naming Provider.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: localhost

- **Seit**: 6.0.1
- **Entfernt in**: 6.11.0.0

notification.error.description

- **Modul**: cmas-dwh-server
- **Beschreibung**: Text für E-Mails mit Fehlermeldungen des DWH.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: Fehler aufgetreten
- **Seit**: 6.0.1

notification.error.from

- **Modul**: cmas-dwh-server
- **Beschreibung**: From-Adresse für E-Mails mit Fehlermeldungen des DWH.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.0.1

notification.error.subject

- **Modul**: cmas-dwh-server
- **Beschreibung**: Betreff für E-Mails mit Fehlermeldungen des DWH.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: Fehler aufgetreten
- **Seit**: 6.0.1

notification.error.to

- **Modul**: cmas-dwh-server
- **Beschreibung**: To-Adresse für E-Mails mit Fehlermeldungen des DWH.
- **Typ**: string
- **Neustart erforderlich**: nein

- **System**: ja
- **Optional**: nein
- **Beispielwert**: myuser@consol.de
- **Seit**: 6.0.1

notification.finished_successfully.description

- **Modul**: cmas-dwh-server
- **Beschreibung**: Text für E-Mails des DWHs, wenn eine Übertragung erfolgreich beendet wurde.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: Übertragung erfolgreich beendet
- **Seit**: 6.0.1

notification.finished_successfully.from

- **Modul**: cmas-dwh-server
- **Beschreibung**: From-Adresse für E-Mails des DWHs, wenn eine Übertragung erfolgreich beendet wurde.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.0.1

notification.finished_successfully.subject

- **Modul**: cmas-dwh-server
- **Beschreibung**: Betreff für E-Mails des DWHs, wenn eine Übertragung erfolgreich beendet wurde.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: Übertragung erfolgreich beendet
- **Seit**: 6.0.1

notification.finished_successfully.to

- **Modul**: cmas-dwh-server
- **Beschreibung**: To-Adresse für E-Mails des DWHs, wenn eine Übertragung erfolgreich beendet wurde.
- **Typ**: string
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: myuser@consol.de
- **Seit**: 6.0.1

notification.finished_unsuccessfully.description

- **Modul**: cmas-dwh-server
- **Beschreibung**: Text für E-Mails des DWHs, wenn eine Übertragung nicht erfolgreich beendet wurde.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: Übertragung nicht erfolgreich beendet
- **Seit**: 6.0.1

notification.finished_unsuccessfully.from

- **Modul**: cmas-dwh-server
- **Beschreibung**: From-Adresse für E-Mails des DWHs, wenn eine Übertragung nicht erfolgreich beendet wurde.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.0.1

notification.finished_unsuccessfully.subject

- **Modul**: cmas-dwh-server
- **Beschreibung**: Betreff für E-Mails des DWHs, wenn eine Übertragung nicht erfolgreich beendet wurde.
- **Typ**: string

- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: Übertragung nicht erfolgreich beendet
- **Seit**: 6.0.1

notification.finished_unsuccessfully.to

- **Modul**: cmas-dwh-server
- **Beschreibung**: To-Adresse für E-Mails des DWHs, wenn eine Übertragung nicht erfolgreich beendet wurde.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: myuser@consol.de
- **Seit**: 6.0.1

notification.host

- **Modul**: cmas-dwh-server
- **Beschreibung**: Hostname des E-Mail-Servers (SMTP) für das Senden von DWH-E-Mails.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: myserver.consol.de
- **Seit**: 6.0.1

notification.password

- **Modul**: cmas-dwh-server
- **Beschreibung**: Passwort für das Senden von DWH-E-Mails. (optional)
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.0.1

notification.port

- **Modul**: cmas-dwh-server
- **Beschreibung**: SMTP-Port für das Senden von DWH-E-Mails.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 25
- **Seit**: 6.0.1

notification.protocol

- **Modul**: cmas-dwh-server
- **Beschreibung**: Das Protokoll, das für das Senden von E-Mails aus dem DWH verwendet wird.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: pop3\

notification.tls.enabled

- **Modul**: cmas-dwh-server
- **Beschreibung**: Aktiviert SMTP über SSL/TLS (SMTPS) für das Senden von Benachrichtigungs-E-Mails aus dem DWH. Der Standardwert ist „false“. Wenn der Wert auf „true“ gesetzt ist, ist SMTPS für das Senden von DWH-Benachrichtigungen aktiviert.
- **Typ**: string
- **Neustart erforderlich**: ja
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.11.1.6

notification.username

- **Modul**: cmas-dwh-server
- **Beschreibung**: Benutzername (SMTP) für das Senden von DWH-E-Mails.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja

- **Optional**: ja
- **Beispielwert**: myuser
- **Seit**: 6.0.1

skip-ticket

- **Modul**: cmas-dwh-server
- **Beschreibung**: Tickets werden bei der Übertragung/Aktualisierung nicht übermittelt.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.6.19
- **Entfernt in**: 6.8.1

skip-ticket-history

- **Modul**: cmas-dwh-server
- **Beschreibung**: Ticketprotokoll wird bei der Übertragung/Aktualisierung nicht übermittelt.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.6.19
- **Entfernt in**: 6.8.1

skip-unit

- **Modul**: cmas-dwh-server
- **Beschreibung**: Units werden bei der Übertragung/Aktualisierung nicht übermittelt.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.6.19
- **Entfernt in**: 6.8.1

skip-unit-history

- **Modul**: cmas-dwh-server
- **Beschreibung**: Unit-Protokoll wird bei der Übertragung/Aktualisierung nicht übermittelt.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.6.19
- **Entfernt in**: 6.8.1

split.history

- **Modul**: cmas-dwh-server
- **Beschreibung**: Ändert das SQL-Statement dahingehend, dass Ticketprotokolle während der Übertragung an das DWH nicht für alle Tickets auf einmal abgeholt werden, sondern für ein Ticket pro SQL-Statement.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.8.0

unit.transfer.order

- **Modul**: cmas-dwh-server
- **Beschreibung**: Legt fest, in welcher Reihenfolge Kundenfeldgruppen an das DWH übertragen werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: company;customer
- **Seit**: 6.6.19
- **Entfernt in**: 6.8.1

H.2.2.2 Konfiguration von Indexer und Suche

Indexer

big.task.minimum.size

- **Modul**: cmas-core-index-common
- **Beschreibung**: Gibt die Minimalgröße eines Index-Tasks an (in Teilen, jeder Teil hat 100 Einheiten), um diesen Task als einen großen Task zu qualifizieren. Große Tasks haben eine niedrigere Priorität als normale.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 15 (Standardwert)
- **Seit**: 6.8.3

database.notification.enabled

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, ob statt JMS der Database Notification Channel der Index-Aktualisierung verwendet werden soll.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.8.4.7

database.notification.redelivery.delay.seconds

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt bei Verwendung des Database Notification Channel der Index-Aktualisierung fest, mit welcher Verzögerung die Benachrichtigung erneut gesendet wird, wenn eine Exception auftritt.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 60
- **Seit**: 6.8.4.7

database.notification.redelivery.max.attempts

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt bei Verwendung des Database Notification Channel der Index-Aktualisierung fest, wie oft maximal versucht wird, die Benachrichtigung erneut zu senden, wenn eine Exception auftritt.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 60
- **Seit**: 6.8.4.7

disable.admin.task.auto.commit

- **Modul**: cmas-core-index-common
- **Beschreibung**: Alle Tasks, die für eine Index-Aktualisierung erstellt werden, werden automatisch direkt nach ihrer Erstellung ausgeführt.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.6.1

index.attachment

- **Modul**: cmas-core-index-common
- **Beschreibung**: Beschreibt, ob der Inhalt von Attachments indiziert wird.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: true
- **Seit**: 6.4.3

index.history

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, ob das Unit- und das Ticketprotokoll indiziert werden.
- **Typ**: boolean

- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.1.0
- **Entfernt in**: 6.11.0

index.status

- **Modul**: cmas-core-index-common
- **Beschreibung**: Status des Indexers, mögliche Werte sind RED, YELLOW, GREEN, werden im Admin Tool angezeigt.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: GREEN
- **Seit**: 6.6.1

index.task.worker.threads

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie viele Threads benutzt werden, um Index-Aufgaben auszuführen (Synchronisierung, Administrations- und Reparatur-Aufgaben).
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1 (Standardwert) (wir empfehlen einen Wert zu verwenden, der nicht größer als 2 ist)
- **Seit**: 6.6.14, 6.7.3. Seit 6.8.0 und auschließlich in 6.6.21 sind auch normale (live) Index-Aktualisierungen von dieser Property betroffen.

index.version.current

- **Modul**: cmas-core-index-common
- **Beschreibung**: Enthält Informationen über die derzeitige (möglicherweise veraltete) Index-version.
- **Typ**: integer
- **Neustart erforderlich**: nein

- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1 (Standardwert)
- **Seit**: 6.7.0

index.version.newest

- **Modul**: cmas-core-index-common
- **Beschreibung**: Enthält Informationen, welche Indexversion als die neueste betrachtet wird.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1 (Standardwert)
- **Seit**: 6.7.0

indexed.assets.per.thread.in.memory

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie viele Assets während des Indizierens pro Thread auf einmal in den Speicher geladen werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 200 (Standardwert)
- **Seit**: 6.8.0

indexed.engineers.per.thread.in.memory

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie viele Bearbeiter während des Indizierens pro Thread auf einmal in den Speicher geladen werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 300 (Standardwert)
- **Seit**: 6.6.14, 6.7.3

indexed.resources.per.thread.in.memory

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie viele Ressourcen während des Indizierens pro Thread auf einmal in den Speicher geladen werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 200 (Standardwert)
- **Seit**: 6.10.0.0

indexed.tickets.per.thread.in.memory

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie viele Tickets während des Indizierens pro Thread auf einmal in den Speicher geladen werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 100 (Standardwert)
- **Seit**: 6.6.14, 6.7.3

indexed.units.per.thread.in.memory

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie viele Units während des Indizierens pro Thread auf einmal in den Speicher geladen werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 200 (Standardwert)
- **Seit**: 6.6.14, 6.7.3

synchronize.master.address

- **Modul**: cmas-core-index-common
- **Beschreibung**: Wert der Java-System-Property `-Dcmas.http.host.port`, die angibt, unter welcher URL der Index-Master-Server erreichbar ist. Standard ist Null. Seit CM-Version 6.6.17 ist dieser Wert beim Setup konfigurierbar, um den initialen Index-Master-Server zu bestimmen. Bitte beachten Sie, dass das Verändern dieses Wertes nur erlaubt ist, wenn alle Cluster-Nodes zum Empfang von Index-Veränderungen gestoppt sind.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 127.0.0.1:80
- **Seit**: 6.6.0

synchronize.master.security.token

- **Modul**: cmas-core-index-common
- **Beschreibung**: Das Passwort für den URL-Zugriff auf den Index-Snapshot, z. B. für die Index-Synchronisierung oder für Backups.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: token
- **Seit**: 6.6.0

synchronize.master.security.user

- **Modul**: cmas-core-index-common
- **Beschreibung**: Der Benutzername für den URL-Zugriff auf den Index-Snapshot, z. B. für die Index-Synchronisierung oder für Backups.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: user
- **Seit**: 6.6.0

synchronize.master.timeout.minutes

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie oft die Index-Synchronisierung ausgehend vom aktuellen Master-Server fehlschlagen darf, bis ein neuer Master für die Index-Reparatur ausgewählt wird. Standardwert ist 5. Seit CM-Version 6.6.17 ist dieser Wert im Setup konfigurierbar, wobei 0 bedeutet, dass der Master-Server nie geändert wird (Failover-Mechanismus deaktiviert).
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 5
- **Seit**: 6.6.0

synchronize.megabits.per.second

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie viel Bandbreite der Master-Server verbrauchen darf, um Index-Änderungen an die Slave-Server zu übermitteln. Standardwert ist 85. Nutzen Sie nicht die gesamte verfügbare Bandbreite, um die Index-Änderungen zwischen den Hosts zu übermitteln, da dies dafür sorgen kann, dass die Nodes des Clusters nicht mehr synchron sind.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 85
- **Seit**: 6.6.0

synchronize.sleep.millis

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie oft jeder Slave-Server den Master-Server auf Änderungen am Index abfragt. Standardwert ist 1000.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1000
- **Seit**: 6.6.0

Search Results

globalSearchResultSizeLimit

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Maximale Anzahl an Ergebnissen in der Schnellsuche.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 10
- **Seit**: 6.0

searchPageSize

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Standardgröße der Seiten für Suchergebnisse.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 20
- **Seit**: 6.0

searchPageSizeOptions

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Optionen für die Größe der Seiten für Suchergebnisse.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 10|20|30|40|50|75|100
- **Seit**: 6.0

unitIndexSearchResultSizeLimit

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Maximale Anzahl der Units in der Ergebnisliste, wenn nach Units gesucht wird (Beispiel: Kontaktsuche).
- **Typ**: integer

- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 5
- **Seit**: 6.0

H.2.2.3 LDAP-Konfiguration

LDAP-Konfiguration (wenn LDAP als Authentifizierungsmethode im CM Web Client verwendet wird)

LDAP-Parameter sind nur dann wirksam, wenn die Authentifizierungsmethode für den CM Web Client auf „LDAP“ gesetzt wurde:

authentication.method

- **Modul**: cmas-core-security
- **Beschreibung**: Methode der Bearbeiter-Authentifizierung für den Web Client (interne CM-Datenbank oder LDAP-Authentifizierung). Mögliche Werte sind LDAP oder DATABASE.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: DATABASE
- **Seit**: 6.0

ldap.authentication

- **Modul**: cmas-core-security
- **Beschreibung**: Verwendete Authentifizierungsmethode, wenn LDAP-Authentifizierung benutzt wird. Mögliche Werte sind "anonymous" und "simple" (Standardwert)
- **Typ**: string
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: simple
- **Seit**: 6.0

ldap.basedn

- **Modul**: cmas-core-security
- **Beschreibung**: Base DN für die Suche von LDAP-Benutzerkonten, wenn LDAP-Authentifizierung verwendet wird.

- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: ou=accounts,dc=consol,dc=de
- **Seit**: 6.0

ldap.initialcontextfactory

- **Modul**: cmas-core-security
- **Beschreibung**: Name der Java-Klasse für die Initial Context Factory der LDAP-Implementierung bei der Verwendung von LDAP-Authentifizierung. Ist üblicherweise `com.sun.jndi.ldap.LdapCtxFactory`.
- **Typ**: string
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: com.sun.jndi.ldap.LdapCtxFactory
- **Seit**: 6.0

ldap.password

- **Modul**: cmas-core-security
- **Beschreibung**: Passwort für die Verbindung zum LDAP, um Benutzer zu suchen, wenn LDAP-Authentifizierung verwendet wird. Wird nur benötigt, wenn die Suche nicht anonym ausgeführt werden kann.
- **Typ**: password
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.1.2

ldap.providerurl

- **Modul**: cmas-core-security
- **Beschreibung**: LDAP-Provider, wenn LDAP-Authentifizierung verwendet wird.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein

- **Beispielwert**: ldap://myserver.consol.de:389
- **Seit**: 6.0

ldap.searchattr

- **Modul**: cmas-core-security
- **Beschreibung**: Suchattribut für die Suche nach LDAP-Einträgen, die mit dem CM-Login verbunden sind.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: uid
- **Seit**: 6.0

ldap.userdn

- **Modul**: cmas-core-security
- **Beschreibung**: LDAP-Benutzer für die Verbindung zum LDAP, um Benutzer zu suchen, wenn LDAP-Authentifizierung verwendet wird. Wird nur benötigt, wenn die Suche nicht anonym ausgeführt werden kann.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.1.2

LDAP-Konfiguration (wenn LDAP in CM/Track als Authentifizierungsmethode eingesetzt wird)

LDAP-Parameter sind nur dann wirksam, wenn die Authentifizierungsmethode für CM/Track auf „LDAP“ gesetzt wurde:

contact.authentication.method

- **Modul**: cmas-core-security
- **Beschreibung**: Definiert die Kontakt-Authentifizierungsmethode für CM/Track, mögliche Werte sind DATABASE oder LDAP oder LDAP,DATABASE oder DATABASE,LDAP.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Seit**: 6.9.3.0

ldap.contact.name.basedn

- **Modul**: cmas-core-security
- **Beschreibung**: Base DN für die Suche nach der Kontakt-DN mittels LDAP-ID (z. B. ou=accounts,dc=consol,dc=de).
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Seit**: 6.9.3.0

ldap.contact.name.password

- **Modul**: cmas-core-security
- **Beschreibung**: Passwort für die Suche nach der Kontakt-DN mittels LDAP-ID. Wenn nicht gesetzt, wird ein anonymes Konto verwendet.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Seit**: 6.9.3.0

ldap.contact.name.providerurl

- **Modul**: cmas-core-security
- **Beschreibung**: Adresse des LDAP-Servers (ldap[s]://host:port).
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Seit**: 6.9.3.0

ldap.contact.name.searchattr

- **Modul**: cmas-core-security
- **Beschreibung**: Attribut für die Suche nach der Kontakt-DN mittels LDAP-ID (z. B. uid).
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Seit**: 6.9.3.0

ldap.contact.name.userdn

- **Modul**: cmas-core-security
- **Beschreibung**: Benutzer-DN für die Suche nach Kontakt-DN mittels LDAP-ID. Wenn nicht gesetzt, wird ein anonymes Konto verwendet.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Seit**: 6.9.3.0

ldap.initialcontextfactory

- **Modul**: cmas-core-security
- **Beschreibung**: Name der Java-Klasse für die Initial Context Factory der LDAP-Implementierung bei der Verwendung von LDAP-Authentifizierung. Ist üblicherweise `com.sun.jndi.ldap.LdapCtxFactory`.
- **Typ**: string
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: com.sun.jndi.ldap.LdapCtxFactory
- **Seit**: 6.0

H.2.2.4 E-Mail-Konfiguration

Postausgang

mail.smtp.email

- **Modul**: cmas-core-server
- **Beschreibung**: SMTP-E-Mail-URL für ausgehende E-Mails.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: smtp://mail.mydomain.com:25
- **Seit**: 6.0

mail.smtp.envelopesender

- **Modul**: cmas-core-server
- **Beschreibung**: E-Mail-Adresse, die als Absender im SMTP-Envelope benutzt wird. Wenn nicht gesetzt, wird die From-Adresse der E-Mail benutzt.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: mysender@mydomain.com
- **Seit**: 6.5.7

mail.from

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Diese E-Mail-Adresse wird anstelle der E-Mail-Adresse des Bearbeiters in E-Mail-Konversationen verwendet.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.1.2

mail.reply.to

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Wenn dieser Wert gesetzt ist, zeigt der Web Client diesen Wert beim Versenden einer E-Mail im Reply-To-Feld an.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.0.1

> ⚠ Bitte lesen Sie dazu die ausführlichen Informationen über Reply-To-Adressen in ConSol CM im Abschnitt Skripte des Typs E-Mail.

mailTemplateAboveQuotedText

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Legt das Verhalten der E-Mail-Vorlagen im Ticket-E-Mail-Editor fest, wenn eine andere E-Mail zitiert wird, d. h. auf diese geantwortet oder diese weitergeleitet wird. Wird oft verwendet, um die Signatur korrekt zu platzieren.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.2.4

mail.sender.address

- **Modul**: cmas-workflow-jbpm
- **Beschreibung**: From-Adresse für E-Mails, die von der Workflow-Engine versendet werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: myuser@consol.de
- **Entfernt in**: 6.8.0
- **Ersetzt durch**: jobExecutor.mailFrom

mail.smtp.tls.enabled

- **Modul**: cmas-core-server
- **Beschreibung**: Aktiviert SMTP über SSL/TLS (SMTPS) für das Senden von E-Mails aus dem Web Client und aus Skripten. Der Standardwert ist „false“. Wenn Sie diese Poperty auf „true“ setzen, ist SMTPS zum Senden von E-Mails aktiviert.
- **Typ**: boolean
- **Neustart erforderlich**: ja
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.11.1.6

Posteingang

nimh.enabled

- **Modul**: cmas-core-server
- **Beschreibung**: Aktiviert den NIMH-Dienst. Im Cluster muss die Node-ID angehängt werden, z. B. `nimh.enabled.NODEID` = „true“.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.9.4.0

filesystem.polling.threads.number

- **Modul**: cmas-nimh
- **Beschreibung**: Anzahl der Threads, die für die Abfrage der Datenbank-E-Mail-Warteschlange gestartet werden. Standardwert: 1.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 10
- **Seit**: 6.4.0

filesystem.polling.threads.shutdown.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Wartezeit nach dem Shutdown-Signal. Wenn der Timeout erreicht ist, wird der Thread beendet. Standardwert: 60.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

filesystem.polling.threads.watchdog.interval.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Intervall des Watchdog Threads. Standardwert: 30.

- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

filesystem.task.enabled

- **Modul**: cmas-nimh
- **Beschreibung**: Mit dieser System-Property kann der Service Thread eines bestimmten Pollers deaktiviert werden. Standardwert: true.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.4.0

filesystem.task.interval.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Standard-Intervall für den Abruf von Postfächern. Standardwert: 60 Sekunden
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

filesystem.task.polling.folder

- **Modul**: cmas-nimh
- **Beschreibung**: Speicherort des Ordners, der überwacht und nach E-Mails im Format von eml-Dateien durchsucht wird. Standard: Unterverzeichnis "mail" des cmas-Datenverzeichnisses
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja

- **Beispielwert**: c://cmas//mail
- **Seit**: 6.4.0

filesystem.task.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Nach dieser Zeit (der Inaktivität) wird der Service Thread als beschädigt betrachtet und automatisch neu gestartet. Standardwert: 120 Sekunden
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

filesystem.task.transaction.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Standard-Transaktions-Timeout für Transaktionen, die E-Mails abrufen. Sollte mit der Anzahl der Nachrichten, die gleichzeitig abgeholt werden, korrelieren. Standardwert: 60 Sekunden
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

mailbox.1.connection.host

- **Modul**: cmas-nimh
- **Beschreibung**: Host (Server) für das erste konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.host`.

mailbox.1.connection.password

- **Modul**: cmas-nimh
- **Beschreibung**: Passwort für das erste konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.password`.

mailbox.1.connection.port

- **Modul**: cmas-nimh
- **Beschreibung**: Port für das erste konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.port`.

mailbox.1.connection.protocol

- **Modul**: cmas-nimh
- **Beschreibung**: Protokoll (z. B. IMAP oder POP3) für das erste konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.protocol`.

mailbox.1.connection.username

- **Modul**: cmas-nimh
- **Beschreibung**: Benutzername für das erste konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.username`.

mailbox.2.connection.host

- **Modul**: cmas-nimh
- **Beschreibung**: Host (Server) für das zweite konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.host`.

mailbox.2.connection.password

- **Modul**: cmas-nimh
- **Beschreibung**: Passwort für das zweite konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.password`.

mailbox.2.connection.port

- **Modul**: cmas-nimh
- **Beschreibung**: Port für das zweite konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.port`.

mailbox.2.connection.protocol

- **Modul**: cmas-nimh
- **Beschreibung**: Protokoll (z. B. IMAP oder POP3) für das zweite konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.protocol`.

mailbox.2.connection.username

- **Modul**: cmas-nimh
- **Beschreibung**: Benutzername für das zweite konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.username`.

Für alle Postfach-Properties im Zusammenhang mit NIMH gilt folgendes Prinzip: Es wird eine Standard-Property definiert (z.B `mailbox.default.connection.port`). Wenn keine postfachspezifische Property konfiguriert ist, wird dieser Standardwert verwendet.

mailbox.default.connection.host

- **Modul**: cmas-nimh
- **Beschreibung**: Host (Servername) eines bestimmten Postfachs, aus dem der Poller E-Mails liest.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 10.10.1.157
- **Seit**: 6.4.0

mailbox.default.connection.password

- **Modul**: cmas-nimh
- **Beschreibung**: Passwort für ein bestimmtes Postfach, aus dem der Poller E-Mails liest.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: consol
- **Seit**: 6.4.0

mailbox.default.connection.port

- **Modul**: cmas-nimh
- **Beschreibung**: Port für ein bestimmtes Postfach, aus dem der Poller E-Mails liest.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 143
- **Seit**: 6.4.0

mailbox.default.connection.protocol

- **Modul**: cmas-nimh
- **Beschreibung**: Protokoll des Pollers, z. B. IMAP oder POP3. Kein Standardwert.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: imap
- **Seit**: 6.4.0

mailbox.default.connection.username

- **Modul**: cmas-nimh
- **Beschreibung**: Benutzername für ein bestimmtes Postfach, aus dem der Poller E-Mails liest.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: username
- **Seit**: 6.4.0

mailbox.default.session.mail.debug

- **Modul**: cmas-nimh
- **Beschreibung**: Beispiel für javax.mail-Property - ermöglicht detaillierteres Session-Debugging mit javax.mail.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.4.0

mailbox.default.session.mail.imap.timeout

- **Modul**: cmas-nimh
- **Beschreibung**: Beispiel für javax.mail-Property
- **Typ**: integer
- **Neustart erforderlich**: nein

- **System**: nein
- **Optional**: ja
- **Beispielwert**: 120
- **Seit**: 6.4.0

mailbox.default.session.mail.mime.address.strict

- **Modul**: cmas-nimh
- **Beschreibung**: Beispiel für javax.mail-Property - Gegenstück zum alten `mail.mime.strict` von Mule, erlaubt das nicht so strikte Parsen des E-Mail-Headers.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.4.0

mailbox.default.session.mail.pop3.timeout

- **Modul**: cmas-nimh
- **Beschreibung**: Beispiel für javax.mail-Property
- **Typ**:
- **Neustart erforderlich**:
- **System**:
- **Optional**:
- **Beispielwert**:
- **Seit**: 6.4.0

mailbox.default.session.mail.<protocol>.partialfetch

- **Modul**: cmas-nimh
- **Beschreibung**: Setzt die JavaMail-Property für `partialfetch`, d. h. steuert, ob die PartialFetch-Funktion des Protokolls genutzt werden soll.
  Für IMAP-Systeme: In CM-Versionen ab 6.10.7.0 wird der Wert von `mailbox.default.session.mail.imap.partialfetch` bei der initialen Einrichtung eines ConSol CM-Systems auf „false“ gesetzt. Bei einer Aktualisierung eines vorhandenen ConSol CM-Systems bleibt der Wert der Property unverändert, wenn die Property bereits vorhanden ist. Wenn die Property noch nicht vorhanden ist, wird sie mit dem Standardwert hinzugefügt.
- **Typ**: boolean
- **Neustart erforderlich**: nein

- **System**: nein
- **Optional**: ja
- **Beispielwert**:
- **Seit**: 6.9.4.0

mailbox.default.task.delete.read.messages

- **Modul**: cmas-nimh
- **Beschreibung**: Legt fest, ob Nachrichten nach dem Lesen aus dem Postfach entfernt werden sollen. Beim IMAP-Protokoll werden Nachrichten standardmäßig als SEEN gekennzeichnet. Bei POP3 wird die Nachricht nur gelöscht, wenn der Wert auf true gesetzt ist. Anderenfalls bleibt die Nachricht auf dem Server, was zu einer Endlos-Leseschleife führt. Standardwert: false.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.4.0

mailbox.default.task.enabled

- **Modul**: cmas-nimh
- **Beschreibung**: Mit dieser System-Property kann der Service Thread eines bestimmten Pollers deaktiviert werden. Standardwert: true.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.4.0

mailbox.default.task.interval.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Standard-Intervall für den Abruf von Postfächern. Standardwert: 60 Sekunden
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja

- **Beispielwert**: 60
- **Seit**: 6.4.0

mailbox.default.task.max.message.size

- **Modul**: cmas-nimh
- **Beschreibung**: Maximale Größe von E-Mails (d. h. E-Mail plus Attachments). E-Mails, die größer als dieser Wert sind, werden nicht automatisch von NIMH verarbeitet. Sie werden in der Datenbank (Tabelle `cmas_nimh_archived_mail`) gespeichert und erscheinen daher in den E-Mail-Backups im Admin Tool (siehe Abschnitt E-Mail-Backups). Von dort können sie erneut gesendet, in das Dateisystem geladen oder gelöscht werden. Für diese Operationen ist die Größe der Nachricht nicht relevant. Standardmäßig auf 10 MB (10485760) gesetzt.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 10485760
- **Seit**: 6.4.0

mailbox.default.task.max.messages.per.run

- **Modul**: cmas-nimh
- **Beschreibung**: Anzahl der Nachrichten, die gleichzeitig aus dem Postfach abgeholt werden. Muss mit dem Transaktions-Timeout korrelieren. Standardwert: 20.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

mailbox.default.task.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Nach dieser Zeit (der Inaktivität) wird der Service Thread als beschädigt betrachtet und automatisch neu gestartet. Standardwert: 120 Sekunden
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja

- **Beispielwert**: 60
- **Seit**: 6.4.0

mailbox.default.task.transaction.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Standard-Transaktions-Timeout für Transaktionen, die E-Mails abrufen. Sollte mit der Anzahl der Nachrichten, die gleichzeitig abgeholt werden, korrelieren. Standardwert: 60 Sekunden
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

mailbox.polling.threads.mail.log.enabled

- **Modul**: cmas-nimh
- **Beschreibung**: Ermöglicht das Protokollieren von E-Mails, was in Cluster-Umgebungen besonders wichtig ist (wird dort als Semaphor genutzt).
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true (Standardwert)
- **Seit**: 6.9.4.1

mailbox.polling.threads.number

- **Modul**: cmas-nimh
- **Beschreibung**: Anzahl der Threads für den Zugriff auf Postfächer. Standardwert: 1.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 1
- **Seit**: 6.4.0

queue.polling.threads.number

- **Modul**: cmas-nimh
- **Beschreibung**: Anzahl der Threads, die zur Überwachung der E-Mail-Warteschlange gestartet werden. Standardwert: 1.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 1
- **Seit**: 6.4.0

queue.polling.threads.shutdown.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Wartezeit nach dem Shutdown-Signal. Wenn der Timeout erreicht ist, wird der Thread beendet. Standardwert: 60.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

queue.polling.threads.watchdog.interval.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Intervall des Watchdog Threads. Standardwert: 30.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 30
- **Seit**: 6.4.0

queue.task.error.pause.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Anzahl der Sekunden, die der Queue-Poller nach einem Infrastrukturfehler (z. B. der Datenbank) wartet. Standardwert: 180 Sekunden
- **Typ**: integer

- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 180
- **Seit**: 6.4.0

queue.task.interval.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Thread-Intervall zur Überwachung der Haupt-E-Mail-Warteschlange. Standardwert: 15.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 15
- **Seit**: 6.4.0

queue.task.max.retries

- **Modul**: cmas-nimh
- **Beschreibung**: Maximale Anzahl der Versuche, eine E-Mail nach einer Exception erneut zu verarbeiten. Ist diese erreicht, wird die E-Mail archiviert. Die archivierte E-Mail kann über das NIMH API (oder das Admin Tool) wieder aktiviert werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 10
- **Seit**: 6.4.0

queue.task.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Nach dieser Zeit (der Inaktivität) wird der Service Thread als beschädigt betrachtet und automatisch neu gestartet. Standardwert: 600 Sekunden
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja

- **Beispielwert**: 600
- **Seit**: 6.4.0

queue.task.transaction.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Transaktions-Timeout für die E-Mail-Verarbeitung in der Warteschlange. Standardwert: 60.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

mail.attachments.validation.info.sender

- **Modul**: cmas-nimh-extension
- **Beschreibung**: Setzt den From-Header bei Attachments vom Typ `error notification mail`.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: admin@mail.com
- **Seit**: 6.7.5

mail.attachments.validation.info.subject

- **Modul**: cmas-nimh-extension
- **Beschreibung**: Setzt den Betreff bei Attachments vom Typ error notification mail.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: E-Mail konnte nicht verarbeitet werden, weil ihre Attachments zurückgewiesen wurden!
- **Seit**: 6.7.5

mail.db.archive

- **Modul**: cmas-nimh-extension
- **Beschreibung**: Wenn dieser Wert auf *true* gesetzt ist, werden eingehende E-Mails in der Datenbank archiviert.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: false (Standardwert)
- **Seit**: 6.8.5.5

mail.error.from.address

- **Modul**: cmas-nimh-extension
- **Beschreibung**: From-Adresse für Fehler-E-Mails von NIMH
- **Typ**: email
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: myuser@consol.de
- **Seit**: 6.4.0

mail.error.to.address

- **Modul**: cmas-nimh-extension
- **Beschreibung**: To-Adresse für Fehler-E-Mails von NIMH. Standardmäßig wird die E-Mail-Adresse des Administrators, die Sie bei der Systemeinrichtung eingegeben haben, verwendet.
- **Typ**: email
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: myuser@consol.de
- **Seit**: 6.4.0

mail.on.error

- **Modul**: cmas-nimh-extension
- **Beschreibung**: Wenn diese Property auf *true* gesetzt ist, wird im Fall, dass eine E-Mail nicht verarbeitet werden konnte, ein Fehler-E-Mail an die oben konfigurierten E-Mail-Adressen gesendet. Standardwert: true.

- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.4.0

mail.process.error

- **Modul**: cmas-nimh-extension
- **Beschreibung**: To-Adresse für E-Mails mit Fehlermeldungen des Mule.
- **Typ**: email
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: myuser@consol.de
- **Seit**: 6.4.0

mail.ticketname.pattern

- **Modul**: cmas-nimh-extension
- **Beschreibung**: Regulärer Ausdruck, mit dem der Ticketname im Betreff eingehender E-Mails identifiziert wird.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: .*?Ticket\s+\((\S+)\).*
- **Seit**: 6.4.0

Attachments für eingehende E-Mails

attachment.allowed.types

- **Modul**: cmas-core-server
- **Beschreibung**: Durch Kommas getrennte Liste der erlaubten Dateinamenserweiterungen (wenn kein Werte definiert ist, sind alle Dateinamenserweiterungen erlaubt).
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja

- **Optional**: ja
- **Beispielwert**: txt,zip,doc
- **Seit**: 6.5.0

attachment.max.size

- **Modul**: cmas-core-server
- **Beschreibung**: Maximale Größe von Attachments in MB. Dies ist eine Validierungs-Property der CM-API. Sie steuert die Größe der Attachments in Tickets, Units und Ressourcen. Außerdem steuert sie die Größe der eingehenden (nicht ausgehenden!) E-Mail-Attachments.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 100
- **Seit**: 6.4.0

E-Mail-Verschlüsselung (Postausgang und Posteingang)

Diese Einstellungen sind nur dann wirksam, wenn die E-Mail-Verschlüsselung aktiv (true) ist.

mail.encryption

- **Modul**: cmas-core-server
- **Beschreibung**: Wenn dieser Wert auf „true“ gesetzt ist, ist im Ticket-E-Mail-Editor die Checkbox zur Verschlüsselung der E-Mail standardmäßig aktiviert.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: true (Standardwert = false)
- **Seit**: 6.8.4.0

Wenn Zertifikate in einem LDAP-Verzeichnis gespeichert sind, müssen folgende Einstellungen vorgenommen werden:

ldap.certificate.basedn

- **Modul**: cmas-core-server
- **Beschreibung**: Base DN für den Speicherort der Zertifikate im LDAP-Verzeichnisbaum. Wenn nicht angegeben, wird `cmas-core-security, ldap.basedn` verwendet.
- **Typ**: string
- **Neustart erforderlich**: nein

- **System**: ja
- **Optional**: ja
- **Beispielwert**: ou=accounts,dc=consol,dc=de
- **Seit**: 6.8.4

ldap.certificate.content.attribute

- **Modul**: cmas-core-server
- **Beschreibung**: Name des LDAP-Attributs, das angibt, wo Zertifikatsdaten im LDAP-Verzeichnisbaum gespeichert werden. Standardwert: usercertificate
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: usercertificate
- **Seit**: 6.8.4

ldap.certificate.password

- **Modul**: cmas-core-server
- **Beschreibung**: Passwort des LDAP-Zertifikatmanagers. Wenn nicht gesetzt, wird `cmas-core-security, ldap.password` verwendet.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.8.4

ldap.certificate.providerurl

- **Modul**: cmas-core-server
- **Beschreibung**: URL des LDAP-Zertifikat-Providers. Wenn nicht gesetzt, wird `cmas-core-security, ldap.providerurl` verwendet.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: ldap://ldap.consol.de:389
- **Seit**: 6.8.4

ldap.certificate.searchattr

- **Modul**: cmas-core-server
- **Beschreibung**: Name des LDAP-Attributs, mit dem Zertifikate im LDAP-Verzeichnisbaum gesucht werden. Standardwert: mail
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: mail
- **Seit**: 6.8.4

ldap.certificate.userdn

- **Modul**: cmas-core-server
- **Beschreibung**: DN des LDAP-Zertifikatmanagers. Wenn nicht gesetzt, wird `cmas-core-security, ldap.userdn` verwendet.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.8.4

H.2.2.5 Konfiguration des Aktivitätsintervalls

admin.tool.session.check.interval

- **Modul**: cmas-app-admin-tool
- **Beschreibung**: Intervall, in dem inaktive (beendete) Sitzungen im Admin Tool überprüft werden (in Sekunden).
- **Typ**: integer
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 30
- **Seit**: 6.7.5

server.session.timeout

- **Modul**: cmas-core-server
- **Beschreibung**: Server-Session-Timeout (in Sekunden) für verbundene Clients. Jeder Client kann dieses Timeout mit benutzerdefinierten Werten mittels seiner ID (ADMIN_TOOL, WEB_CLIENT, WORKFLOW_EDITOR, TRACK (vor 6.8 bitte PORTER verwenden), ETL, REST) überschreiben. Diese wird an den Namen der System-Property angehängt, z. B. `server.session.timeout.ADMIN_TOOL`.
  Siehe auch die Attribute der Seitenanpassung *updateTimeServerSessionActivityEnabled* und *updateTimeServerSessionActivity*, beide vom Typ *cmApplicationCustomization*.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1800
- **Seit**: 6.6.1, 6.7.1

Detaillierte Erklärung für das Admin Tool:

- server.session.timeout.ADMIN_TOOL
  Definiert den Zeitraum, den der Server eine Session im Admin Tool als gültig betrachtet, wenn keine Aktivität im Admin Tool der Session erfolgt. Das Admin Tool kennt diesen Wert nicht, es bemerkt nur eine ungültige Session, wenn es länger keine Aktivität gegeben hat.
- admin.tool.session.check.interval
  Definiert den Zeitraum zwischen zwei Überprüfungen durch das Admin Tool, ob der Server die Session noch als gültig betrachtet.

Wenn zum Beispiel admin.tool.session.check.interval = 60, fragt das Admin Tool den Server einmal pro Minute, ob die Session noch aktiv/gültig ist. Wenn server.session.timeout.ADMIN_TOOL = 600 erhält das Admin Tool die Antwort, dass die Session ungültig ist, nach zehn Minuten der Inaktivität.

H.2.2.6 Administrator-E-Mail-Adressen

ConSol CM kann je nach Untersystem verschiedene Administrator-E-Mail-Adressen verwenden. Eine detaillierte Erklärung über die Administrator-E-Mail-Adressen finden Sie in Administrator-E-Mail-Adressen. Wenn keine speziellen Administrator-E-Mail-Adressen konfiguriert sind, wird die globale E-Mail-Adresse (die Sie bei der Systemeinrichtung angegeben haben), verwendet.

H.2.2.7 Konfiguration von HTTP-Headern

Es ist möglich, die HTTP-Antwort-Header, die mit der Webseite im Web Client zurückgegeben werden, zu konfigurieren. Dies ermöglicht zum Beispiel die Definition von sicherheitsrelevanten Antwort-Headern entsprechend den Anforderungen von Richtlinien oder Umgebung. Diese Properties werden im Modul `cmweb-server-http-headers` verwaltet. Der Name der Property ist das Feld des Antwort-Headers und der Wert der Property ist der im Header gesendete Feldwert.

Sowohl der Name als auch der Wert müssen die genaue Schreibweise aus der HTTP-Spezifikation für das gewünschte Antwort-Header-Feld haben. Die richtige Interpretation und Verarbeitung der Header liegt im Zuständigkeitsbereich des Client-Browsers.

Beispiel-Property: X-Frame-Options

- **Modul**: cmweb-server-http-headers
- **Beschreibung**: Beispiel-Property zum Verdeutlichen der Konfiguration von HTTP-Headern. In diesem Fall enthält der gesendete HTTP-Header das Feld *X-Frame-Options* mit dem Wert „SAMEORIGIN“.

  Jede Property im Modul `cmweb-server-http-headers` stellt ein Header-Feld dar. Der Name/Schlüssel der Property identifiziert das Header-Feld und der Wert der Property ist der im Header gesendete Feldwert.

  Beachten Sie, dass zusätzliche HTTP-Antwort-Header korrekt und mit der offiziell vorgegebenen Schreibweise definiert werden müssen! Beachten Sie weiterhin, dass die korrekte Interpretation und Verarbeitung dieser Header komplett in Zuständigkeitsbereich des Client-Browsers liegt, der die Seite angefordert hat!

- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: SAMEORIGIN
- **Seit**: 6.10.8

H.2.3 Liste der System-Properties nach Modul

In diesem Kapitel sind die System-Properties der folgenden Module aufgeführt.

- cmas-app-admin-tool (Modul)
- cmas-core-cache (Modul)
- cmas-core-index-common (Modul)
- cmas-core-security (Modul)
- cmas-core-server (Modul)
- cmas-core-shared (Modul)
- cmas-dwh-server (Modul)
- cmas-nimh (Modul)
- cmas-nimh-extension (Modul)
- cmas-restapi-core (Modul)
- cmas-setup-hibernate (Modul)
- cmas-setup-manager (Modul)
- cmas-setup-scene (Modul)
- cmas-workflow-engine (Modul)
- cmas-workflow-jbpm (Modul)
- cmweb-server-adapter (Modul)
- cmweb-server-http-headers (Modul)

H.2.3.1 cmas-app-admin-tool (Modul)

admin.tool.consumed.licences.check.interval

- **Modul**: cmas-app-admin-tool
- **Beschreibung**: Setzt das Intervall (in Sekunden) zum Überwachen der Anzahl der verwendeten Lizenzen. Der Standardwert ist 30.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 30
- **Seit**: 6.11.0.0

admin.tool.consumed.licences.pool.name

- **Modul**: cmas-app-admin-tool
- **Beschreibung**: Setzt den Namen des Lizenzpools zum Überwachen der verwendeten Lizenzen. Der Standardwert ist „CONCURRENT_USERS“.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: CONCURRENT_USERS
- **Seit**: 6.11.0.0

admin.tool.session.check.interval

- **Modul**: cmas-app-admin-tool
- **Beschreibung**: Intervall, in dem inaktive (beendete) Sitzungen im Admin Tool überprüft werden (in Sekunden).
- **Typ**: integer
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 30
- **Seit**: 6.7.5

autocomplete.enabled

- **Modul**: cmas-app-admin-tool
- **Beschreibung**: Wenn diese System-Property fehlt oder der Wert „false“ ist, wird das Navigationselement *Adress-Vervollständigung* im Admin Tool ausgeblendet.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.9.2.0

delete.ticket.enabled

- **Modul**: cmas-app-admin-tool
- **Beschreibung**: Steuert, ob in der Ticketverwaltung im Admin Tool der Menüpunkt *Tickets löschen* im Kontextmenü der Ticketliste angezeigt wird.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.9.4.0

dwh.administration.refresh.interval.seconds

- **Modul**: cmas-app-admin-tool
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 10
- **Seit**: 6.11.0.1

script.validation.interval.seconds

- **Modul**: cmas-app-admin-tool
- **Beschreibung**: Intervall in Sekunden zwischen zwei Code-Überprüfungen im Quelltext-Editor des Admin Tools oder Process Designers
- **Typ**: Integer

- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: nein
- **Beispielwert**: 1 (Standardwert)
- **Seit**: 6.11.0.1

start.groovy.task.enabled

- **Modul**: cmas-app-admin-tool
- **Beschreibung**: Um Skripte vom Typ *Task* im Admin Tool (Navigationsgruppe *Dienste*, Navigationselement *Task-Ausführung*) ausführen zu können, ist es nötig, den Button *Start* zu aktivieren, der standardmäßig ausgeblendet ist. Setzen Sie hierfür diese System-Property auf *true*.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.9.4.0

task.panel.refresh.interval.seconds

- **Modul**: cmas-app-admin-tool
- **Beschreibung**: Zeit in Sekunden, nachdem die Task-Liste (im Admin Tool) des Task Execution Framework aktualisiert wird.
- **Typ**: Integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: nein
- **Beispielwert**: 10
- **Seit**: 6.10.5.3 (wird beim Update von einer Version vor 6.10.5.3 nicht automatisch hinzugefügt!)

H.2.3.2 cmas-core-cache (Modul)

cache-cluster-name

- **Modul**: cmas-core-cache
- **Beschreibung**: Cache-Cluster-Name des JBoss.
- **Typ**: string
- **Neustart erforderlich**: ja
- **System**: ja

- **Optional**: nein
- **Beispielwert**: 635a6de1-629a-4129-8299-2d98633310f0
- **Seit**: 6.4.0

eviction.event.queue.size

- **Modul**: cmas-core-cache
- **Beschreibung**: Die Größe der Queue mit den Cache-Events. Der Standardwert ist 200000. Auf Systemen mit viel Traffic oder Last wird empfohlen, diesen Wert leicht zu erhöhen (bis 400000).
- **Typ**: integer
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 200000
- **Seit**: 6.4.0

eviction.max.nodes

- **Modul**: cmas-core-cache
- **Beschreibung**: Setzt die maximale Größe der internen Caches. Der Standardwert ist 100000. Das Erhöhen dieses Wertes führt zu einem höheren Speicherverbrauch und wird nicht empfohlen, außer wenn ConSol explizit dazu auffordert.
- **Typ**: integer
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 100000
- **Seit**: 6.4.0

eviction.wakeup.interval

- **Modul**: cmas-core-cache
- **Beschreibung**: Setzt das Intervall (in Millisekunden) zwischen zwei Verarbeitungszyklen von Cache-Queue-Events. Der Standardwert ist 3000. Auf Systemen mit viel Traffic oder Last wird empfohlen, den Wert zu verringern (Minimalwert ist 1500).
- **Typ**: integer
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein

- **Beispielwert**: 3000
- **Seit**: 6.4.0

H.2.3.3 cmas-core-index-common (Modul)

big.task.minimum.size

- **Modul**: cmas-core-index-common
- **Beschreibung**: Gibt die Minimalgröße eines Index-Tasks an (in Teilen, jeder Teil hat 100 Einheiten), um diesen Task als einen großen Task zu qualifizieren. Große Tasks haben eine niedrigere Priorität als normale.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 15 (Standardwert)
- **Seit**: 6.8.3

database.notification.enabled

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, ob statt JMS der Database Notification Channel der Index-Aktualisierung verwendet werden soll.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.8.4.7

database.notification.redelivery.delay.seconds

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt bei Verwendung des Database Notification Channel der Index-Aktualisierung fest, mit welcher Verzögerung die Benachrichtigung erneut gesendet wird, wenn eine Exception auftritt.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein

- **Beispielwert**: 60
- **Seit**: 6.8.4.7

database.notification.redelivery.max.attempts

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt bei Verwendung des Database Notification Channel der Index-Aktualisierung fest, wie oft maximal versucht wird, die Benachrichtigung erneut zu senden, wenn eine Exception auftritt.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 60
- **Seit**: 6.8.4.7

disable.admin.task.auto.commit

- **Modul**: cmas-core-index-common
- **Beschreibung**: Alle Tasks, die für eine Index-Aktualisierung erstellt werden, werden automatisch direkt nach ihrer Erstellung ausgeführt.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.6.1

index.attachment

- **Modul**: cmas-core-index-common
- **Beschreibung**: Beschreibt, ob der Inhalt von Attachments indiziert wird.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: true
- **Seit**: 6.4.3

index.history

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, ob das Unit- und das Ticketprotokoll indiziert werden.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.1.0
- **Entfernt in**: 6.11.0

index.status

- **Modul**: cmas-core-index-common
- **Beschreibung**: Status des Indexers, mögliche Werte sind RED, YELLOW, GREEN, werden im Admin Tool angezeigt.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: GREEN
- **Seit**: 6.6.1

index.task.worker.threads

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie viele Threads benutzt werden, um Index-Aufgaben auszuführen (Synchronisierung, Administrations- und Reparatur-Aufgaben).
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1 (Standardwert) (wir empfehlen einen Wert zu verwenden, der nicht größer als 2 ist)
- **Seit**: 6.6.14, 6.7.3. Seit 6.8.0 und auschließlich in 6.6.21 sind auch normale (live) Index-Aktualisierungen von dieser Property betroffen.

index.version.current

- **Modul**: cmas-core-index-common
- **Beschreibung**: Enthält Informationen über die derzeitige (möglicherweise veraltete) Index-version.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1 (Standardwert)
- **Seit**: 6.7.0

index.version.newest

- **Modul**: cmas-core-index-common
- **Beschreibung**: Enthält Informationen, welche Indexversion als die neueste betrachtet wird.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1 (Standardwert)
- **Seit**: 6.7.0

indexed.assets.per.thread.in.memory

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie viele Assets während des Indizierens pro Thread auf einmal in den Speicher geladen werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 200 (Standardwert)
- **Seit**: 6.8.0

indexed.engineers.per.thread.in.memory

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie viele Bearbeiter während des Indizierens pro Thread auf einmal in den Speicher geladen werden.
- **Typ**: integer

- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 300 (Standardwert)
- **Seit**: 6.6.14, 6.7.3

indexed.resources.per.thread.in.memory

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie viele Ressourcen während des Indizierens pro Thread auf einmal in den Speicher geladen werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 200 (Standardwert)
- **Seit**: 6.10.0.0

indexed.tickets.per.thread.in.memory

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie viele Tickets während des Indizierens pro Thread auf einmal in den Speicher geladen werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 100 (Standardwert)
- **Seit**: 6.6.14, 6.7.3

indexed.units.per.thread.in.memory

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie viele Units während des Indizierens pro Thread auf einmal in den Speicher geladen werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein

- **Beispielwert**: 200 (Standardwert)
- **Seit**: 6.6.14, 6.7.3

synchronize.master.address

- **Modul**: cmas-core-index-common
- **Beschreibung**: Wert der Java-System-Property `-Dcmas.http.host.port`, die angibt, unter welcher URL der Index-Master-Server erreichbar ist. Standard ist Null. Seit CM-Version 6.6.17 ist dieser Wert beim Setup konfigurierbar, um den initialen Index-Master-Server zu bestimmen. Bitte beachten Sie, dass das Verändern dieses Wertes nur erlaubt ist, wenn alle Cluster-Nodes zum Empfang von Index-Veränderungen gestoppt sind.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 127.0.0.1:80
- **Seit**: 6.6.0

synchronize.master.security.token

- **Modul**: cmas-core-index-common
- **Beschreibung**: Das Passwort für den URL-Zugriff auf den Index-Snapshot, z. B. für die Index-Synchronisierung oder für Backups.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: token
- **Seit**: 6.6.0

synchronize.master.security.user

- **Modul**: cmas-core-index-common
- **Beschreibung**: Der Benutzername für den URL-Zugriff auf den Index-Snapshot, z. B. für die Index-Synchronisierung oder für Backups.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: user
- **Seit**: 6.6.0

synchronize.master.timeout.minutes

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie oft die Index-Synchronisierung ausgehend vom aktuellen Master-Server fehlschlagen darf, bis ein neuer Master für die Index-Reparatur ausgewählt wird. Standardwert ist 5. Seit CM-Version 6.6.17 ist dieser Wert im Setup konfigurierbar, wobei 0 bedeutet, dass der Master-Server nie geändert wird (Failover-Mechanismus deaktiviert).
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 5
- **Seit**: 6.6.0

synchronize.megabits.per.second

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie viel Bandbreite der Master-Server verbrauchen darf, um Index-Änderungen an die Slave-Server zu übermitteln. Standardwert ist 85. Nutzen Sie nicht die gesamte verfügbare Bandbreite, um die Index-Änderungen zwischen den Hosts zu übermitteln, da dies dafür sorgen kann, dass die Nodes des Clusters nicht mehr synchron sind.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 85
- **Seit**: 6.6.0

synchronize.sleep.millis

- **Modul**: cmas-core-index-common
- **Beschreibung**: Legt fest, wie oft jeder Slave-Server den Master-Server auf Änderungen am Index abfragt. Standardwert ist 1000.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1000
- **Seit**: 6.6.0

H.2.3.4 cmas-core-security (Modul)

admin.email

- **Modul**: cmas-core-security
- **Beschreibung**: Die E-Mail-Adresse des ConSol CM-Administrators. Anfänglich wird hier der Wert genommen, den Sie bei der Systemeinrichtung eingegeben haben.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: myuser@consol.de
- **Seit**: 6.0

admin.login

- **Modul**: cmas-core-security
- **Beschreibung**: Der Name des ConSol CM-Administrators. Anfänglich wird hier der Wert genommen, den Sie bei der Systemeinrichtung eingegeben haben.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: admin
- **Seit**: 6.0

authentication.method

- **Modul**: cmas-core-security
- **Beschreibung**: Methode der Bearbeiter-Authentifizierung für den Web Client (interne CM-Datenbank oder LDAP-Authentifizierung). Mögliche Werte sind LDAP oder DATABASE.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: DATABASE
- **Seit**: 6.0

contact.authentication.method

- **Modul**: cmas-core-security
- **Beschreibung**: Definiert die Kontakt-Authentifizierungsmethode für CM/Track, mögliche Werte sind DATABASE oder LDAP oder LDAP,DATABASE oder DATABASE,LDAP.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Seit**: 6.9.3.0

contact.inherit.permissions.only.to.own.customer.group

- **Modul**: cmas-core-security
- **Beschreibung**: Legt fest, ob der authentifizierte Kontakt in CM/Track alle Kundengruppen-Berechtigungen vom repräsentierenden Bearbeiter erbt (false) oder nur die Berechtigungen für die eigene Kundengruppe hat (true).
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Seit**: 6.9.2.3

kerberos.v5.enabled

- **Modul**: cmas-core-security
- **Beschreibung**: Legt fest, ob SSO über Kerberos aktiviert ist.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false (Standardwert, wenn Kerberos bei der Systemeinrichtung nicht aktiviert wurde)
- **Seit**: 6.2.0

kerberos.v5.username.regex

- **Modul**: cmas-core-security
- **Beschreibung**: Regulärer Ausdruck für die Zuordnung des Kerberos Principals zum Login des CM-Bearbeiters
- **Typ**: string

- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: (.*)@.*
- **Seit**: 6.2.0

ldap.authentication

- **Modul**: cmas-core-security
- **Beschreibung**: Verwendete Authentifizierungsmethode, wenn LDAP-Authentifizierung benutzt wird. Mögliche Werte sind "anonymous" und "simple" (Standardwert)
- **Typ**: string
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: simple
- **Seit**: 6.0

ldap.basedn

- **Modul**: cmas-core-security
- **Beschreibung**: Base DN für die Suche von LDAP-Benutzerkonten, wenn LDAP-Authentifizierung verwendet wird.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: ou=accounts,dc=consol,dc=de
- **Seit**: 6.0

ldap.contact.name.basedn

- **Modul**: cmas-core-security
- **Beschreibung**: Base DN für die Suche nach der Kontakt-DN mittels LDAP-ID (z. B. ou=a-ccounts,dc=consol,dc=de).
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Seit**: 6.9.3.0

ldap.contact.name.password

- **Modul**: cmas-core-security
- **Beschreibung**: Passwort für die Suche nach der Kontakt-DN mittels LDAP-ID. Wenn nicht gesetzt, wird ein anonymes Konto verwendet.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Seit**: 6.9.3.0

ldap.contact.name.providerurl

- **Modul**: cmas-core-security
- **Beschreibung**: Adresse des LDAP-Servers (ldap[s]://host:port).
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Seit**: 6.9.3.0

ldap.contact.name.searchattr

- **Modul**: cmas-core-security
- **Beschreibung**: Attribut für die Suche nach der Kontakt-DN mittels LDAP-ID (z. B. uid).
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Seit**: 6.9.3.0

ldap.contact.name.userdn

- **Modul**: cmas-core-security
- **Beschreibung**: Benutzer-DN für die Suche nach Kontakt-DN mittels LDAP-ID. Wenn nicht gesetzt, wird ein anonymes Konto verwendet.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Seit**: 6.9.3.0

ldap.initialcontextfactory

- **Modul**: cmas-core-security
- **Beschreibung**: Name der Java-Klasse für die Initial Context Factory der LDAP-Implementierung bei der Verwendung von LDAP-Authentifizierung. Ist üblicherweise `com.sun.jndi.ldap.LdapCtxFactory`.
- **Typ**: string
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: com.sun.jndi.ldap.LdapCtxFactory
- **Seit**: 6.0

ldap.password

- **Modul**: cmas-core-security
- **Beschreibung**: Passwort für die Verbindung zum LDAP, um Benutzer zu suchen, wenn LDAP-Authentifizierung verwendet wird. Wird nur benötigt, wenn die Suche nicht anonym ausgeführt werden kann.
- **Typ**: password
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.1.2

ldap.providerurl

- **Modul**: cmas-core-security
- **Beschreibung**: LDAP-Provider, wenn LDAP-Authentifizierung verwendet wird.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: ldap://myserver.consol.de:389
- **Seit**: 6.0

ldap.searchattr

- **Modul**: cmas-core-security
- **Beschreibung**: Suchattribut für die Suche nach LDAP-Einträgen, die mit dem CM-Login verbunden sind.

- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: uid
- **Seit**: 6.0

ldap.userdn

- **Modul**: cmas-core-security
- **Beschreibung**: LDAP-Benutzer für die Verbindung zum LDAP, um Benutzer zu suchen, wenn LDAP-Authentifizierung verwendet wird. Wird nur benötigt, wenn die Suche nicht anonym ausgeführt werden kann.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.1.2

password.reset.mail.from

Modul: cmas-core-security

Beschreibung: Die From-Adresse der E-Mail, die an einen Kunden gesendet wird, der (über den Link *Passwort vergessen?*) in CM/Track ein neues Passwort anfordert bzw. die *From-Adresse* der E-Mail, die an einen Bearbeiter gesendet wird, der (über den Link *Passwort vergessen?*) im Web Client ein neues Passwort anfordert.

Typ: String

Neustart erforderlich: nein

System: nein

Optional: nein

Beispielwert: mypwreset@consol.de

Seit: 6.11.0.1

policy.password.age

- **Modul**: cmas-core-security
- **Beschreibung**: Maximaler Gültigkeitszeitraum, in Tagen, Beispiel „183“ (6 Monate), Standardwert: „5500“ (= 15 Jahre, d. h. es wird keine Passwortänderung erzwungen). Falls Sie möchten, dass der Bearbeiter sein Passwort sofort ändert, verwenden Sie einen der folgenden Werte:

  - **0**
    Der Bearbeiter muss sein Passwort bei der nächsten Anmeldung ändern.
  - **1**
    Der Bearbeiter muss sein Passwort am nächsten Tag ändern.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 5500 (15 Jahre, Standardwert)
- **Seit**: 6.10.1.0

policy.password.pattern

- **Modul**: cmas-core-security
- **Beschreibung**: Definiert das Passwortmuster.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: "^.3,$" (Standardwert)
- **Seit**: 6.10.1.0

policy.rotation.ratio

- **Modul**: cmas-core-security
- **Beschreibung**: Legt fest, wie oft sich das Passwort wiederholen darf. Ist der Wert z. B. 10, darf das neue Passwort nicht unter den zehn letzten Passwörtern des Benutzers sein.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 1 (Standardwert)
- **Seit**: 6.10.1.0

policy.username.case.sensitive

- **Modul**: cmas-core-security
- **Beschreibung**: Legt fest, ob bei Benutzernamen die Groß- und Kleinschreibung beachtet werden muss.
- **Typ**: boolean

- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true (Standardwert)
- **Seit**: 6.10.1.0

policy.track.username.case.sensitive

- **Modul**: cmas-core-security
- **Beschreibung**: Definiert, ob bei Kunden-Benutzernamen in CM/Track bei der Anmeldung zwischen Groß- und Kleinbuchstaben unterschieden wird.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true (Standardwert)
- **Seit**: 6.11.0.0

resetCode.expiriationPeriod

- **Modul**: cmas-core-security
- **Beschreibung**: Definiert die Gültigkeitsdauer des Links zum Zurücksetzen des Passworts in CM/Track.
- **Typ**: Integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 86400000 (Standardwert, 24 Stunden)
- **Seit**: 6.10.1

H.2.3.5 cmas-core-server (Modul)

attachment.allowed.types

- **Modul**: cmas-core-server
- **Beschreibung**: Durch Kommas getrennte Liste der erlaubten Dateinamenserweiterungen (wenn kein Werte definiert ist, sind alle Dateinamenserweiterungen erlaubt).
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja

- **Optional**: ja
- **Beispielwert**: txt,zip,doc
- **Seit**: 6.5.0

attachment.max.size

- **Modul**: cmas-core-server
- **Beschreibung**: Maximale Größe von Attachments in MB. Dies ist eine Validierungs-Property der CM-API. Sie steuert die Größe der Attachments in Tickets, Units und Ressourcen. Außerdem steuert sie die Größe der eingehenden (nicht ausgehenden!) E-Mail-Attachments.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 100
- **Seit**: 6.4.0

calendar.csv.dateFormat

- **Modul**: cmas-core-server
- **Beschreibung**: Format des in der CSV-Datei mit der Liste der Feiertage angegebenen Datums.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: dd/MM/yyyy
- **Seit**: 6.9.3.2

calendar.csv.separator

- **Modul**: cmas-core-server
- **Beschreibung**: Trennzeichen, das in der CSV-Datei mit der Liste der Feiertage verwendet wird.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: ,
- **Seit**: 6.9.3.2

config.data.version

- **Modul**: cmas-core-server
- **Beschreibung**: Die interne Versionsnummer der aktuellen Systemkonfiguration. Diese Property wird intern gepflegt. Ändern Sie sie nur, wenn Sie von ConSol dazu aufgefordert werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 11
- **Seit**: 6.0

config.import.global.transaction.enabled

- **Modul**: cmas-core-server
- **Beschreibung**: Flag, mit dem entschieden wird, ob die Konfiguration (ohne Lokalisierungen) in einer einzigen Transaktion importiert werden soll.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.11.1.0

dao.log.threshold.milliseconds

- **Modul**: cmas-core-server
- **Beschreibung**: Zur Konfiguration der Protokollierung von Datenbankoperationszeiten verwendet. DAO-Methoden, deren Ausführung länger als die in dieser Property gesetzte Zeit (in Millisekunden) dauert, werden protokolliert.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 500 (Standardwert)
- **Seit**: 6.11.1.0

dao.log.username

- **Modul**: cmas-core-server
- **Beschreibung**: Zur Konfiguration der Protokollierung von Datenbankoperationszeiten verwendet. Die Ausführung von DAO-Methoden, die im Zusammenhang mit dem in dieser Property angegebenen Benutzernamen stehen werden, wird protokolliert. Es kann nur ein Benutzername angegeben werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: webadmin (Standardwert: leerer String)
- **Seit**: 6.11.1.0

defaultCommentClassName

- **Modul**: cmas-core-server
- **Beschreibung**: Standard-Textklasse für Kommentare.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**:
- **Seit**: 6.3.0

defaultIncommingMailClassName

- **Modul**: cmas-core-server
- **Beschreibung**: Standard-Textklasse für eingehende E-Mails.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Seit**: 6.3.0

defaultOutgoingMailClassName

- **Modul**: cmas-core-server
- **Beschreibung**: Standard-Textklasse für ausgehende E-Mails.
- **Typ**: string
- **Neustart erforderlich**: nein

- **System**: nein
- **Optional**: ja
- **Beispielwert**:
- **Seit**: 6.3.0

engineer.description.cache.enabled

- **Modul**: cmas-core-server
- **Beschreibung**: Legt fest, ob die Benutzerbeschreibungen zwischengespeichert werden. Der Standardwert ist „true“. Ändern Sie diesen Wert nur, wenn Sie von ConSol dazu aufgefordert werden.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.11.0

engineer.description.mode

- **Modul**: cmas-core-server
- **Beschreibung**: Legt fest, ob die Benutzernamen im Ticketprotokoll aus der Datenbank abgerufen oder dynamisch über Templates erzeugt werden. Der Standardwert „DYNAMIC“ ist aus Performance-Sicht etwas kostspieliger. „PROTOCOL“ ist schneller, gibt aber historische Namen zurück, die unter Umständen veraltet sind. Verwenden Sie „PROTOCOL“, wenn Sie viele Protokolleinträge von vielen unterschiedlichen Benutzern haben.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: PROTOCOL
- **Seit**: 6.11.0

engineer.description.template.name

- **Modul**: cmas-core-server
- **Beschreibung**: Definiert den Namen des Templates, mit dem die Bearbeiternamen im Web Client dargestellt werden. Das Template muss im Abschnitt *Templates* des Admin Tools gespeichert sein. Standardwert: „engineer description template name“.
- **Typ**: String
- **Neustart erforderlich**: nein

- **System**: nein
- **Optional**: nein
- **Seit**: 6.11.0

external.line.access.prefix

- **Modul**: cmas-core-server
- **Beschreibung**: Allgemeine Vorwahl, die vor einer Ortsvorwahl gewählt wird. Wird für jede Kundengruppe einzeln eingestellt.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 411
- **Seit**: 6.9.3.0

fetchSize.strategy

- **Modul**: cmas-core-server
- **Beschreibung**: Auswahl der Strategie für die Abholgröße von JDBC-Ergebnissatzes.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: FetchSizePageBasedStrategy, FetchSizeThresholdStrategy, FetchSizeFixedStrategy
- **Seit**: 6.8.4.1

fetchSize.strategy.FetchSizeFixedStrategy.value

- **Modul**: cmas-core-server
- **Beschreibung**: Legt den Wert für Abholgrößen fest, wenn die ausgewählte Strategie für die Abholgröße `FetchSizeFixedStrategy` ist.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 150
- **Seit**: 6.8.4.1

fetchSize.strategy.FetchSizePageBasedStrategy.limit

- **Modul**: cmas-core-server
- **Beschreibung**: Legt den Wert für Abholgrößen fest, wenn die ausgewählte Strategie für die Abholgröße `FetchSizePageBasedStrategy` ist.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 10000
- **Seit**: 6.8.4.1

fetchSize.strategy.FetchSizeThresholdStrategy.value

- **Modul**: cmas-core-server
- **Beschreibung**: Gibt Grenzwerte für Abholgrößen an, wenn die ausgewählte Strategie für die Abholgröße `FetchSizeThresholdStrategy` ist.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 150,300,600,1000
- **Seit**: 6.8.4.1

heartbeat

- **Modul**: cmas-core-server
- **Beschreibung**: Zeitstempel, der angibt, ob eine Instanz der Applikation mit dem Datenbankschema verbunden ist.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1234567899
- **Seit**: 6.10.5.3

internal.line.access.prefix

- **Modul**: cmas-core-server
- **Beschreibung**: Vorwahl, die das Telefonsystem des Unternehmens für externe Anrufe verwendet. Wird für jede Kundegruppe einzeln gesetzt.

- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 199
- **Seit**: 6.9.3.0

last.config.change

- **Modul**: cmas-core-server
- **Beschreibung**: Zufällige UUID, die während der letzten Konfigurationsänderung generiert wurde. Dies ist ein intern verwalteter Wert. Ändern Sie diesen nur, wenn Sie von ConSol dazu aufgefordert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 2573c7b7-2bf5-47ff-b5a2-bad31951a266
- **Seit**: 6.1.0, 6.2.1

last.config.change.templates

- **Modul**: cmas-core-server
- **Beschreibung**: Zufällige UUID, die bei der letzten Änderung am Template erzeugt wird. Dies ist ein intern gepflegter Wert. Ändern Sie ihn nur, wenn Sie von ConSol dazu aufgefordert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 2573c7c7-2af5-4eff-b9c2-bad31951a266
- **Seit**: 6.10.5.0

ldap.certificate.basedn

- **Modul**: cmas-core-server
- **Beschreibung**: Base DN für den Speicherort der Zertifikate im LDAP-Verzeichnisbaum. Wenn nicht angegeben, wird `cmas-core-security, ldap.basedn` verwendet.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja

- **Optional**: ja
- **Beispielwert**: ou=accounts,dc=consol,dc=de
- **Seit**: 6.8.4

ldap.certificate.content.attribute

- **Modul**: cmas-core-server
- **Beschreibung**: Name des LDAP-Attributs, das angibt, wo Zertifikatsdaten im LDAP-Verzeichnisbaum gespeichert werden. Standardwert: usercertificate
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: usercertificate
- **Seit**: 6.8.4

ldap.certificate.password

- **Modul**: cmas-core-server
- **Beschreibung**: Passwort des LDAP-Zertifikatmanagers. Wenn nicht gesetzt, wird `cmas-core-security, ldap.password` verwendet.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.8.4

ldap.certificate.providerurl

- **Modul**: cmas-core-server
- **Beschreibung**: URL des LDAP-Zertifikat-Providers. Wenn nicht gesetzt, wird `cmas-core-security, ldap.providerurl` verwendet.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: ldap://ldap.consol.de:389
- **Seit**: 6.8.4

ldap.certificate.searchattr

- **Modul**: cmas-core-server
- **Beschreibung**: Name des LDAP-Attributs, mit dem Zertifikate im LDAP-Verzeichnisbaum gesucht werden. Standardwert: mail
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: mail
- **Seit**: 6.8.4

ldap.certificate.userdn

- **Modul**: cmas-core-server
- **Beschreibung**: DN des LDAP-Zertifikatmanagers. Wenn nicht gesetzt, wird `cmas-core-security, ldap.userdn` verwendet.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.8.4

local.country.prefix

- **Modul**: cmas-core-server
- **Beschreibung**: Vorwahl des Ländercodes. Wird für jede Kundengruppe einzeln eingestellt.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 48
- **Seit**: 6.9.3.0

mail.encryption

- **Modul**: cmas-core-server
- **Beschreibung**: Wenn dieser Wert auf „true“ gesetzt ist, ist im Ticket-E-Mail-Editor die Checkbox zur Verschlüsselung der E-Mail standardmäßig aktiviert.
- **Typ**: boolean
- **Neustart erforderlich**: nein

- **System**: ja
- **Optional**: nein
- **Beispielwert**: true (Standardwert = false)
- **Seit**: 6.8.4.0

mail.notification.engineerChange

- **Modul**: cmas-core-server
- **Beschreibung**: Legt fest, ob eine Benachrichtigungs-E-Mail verschickt wird, wenn der Bearbeiter eines Tickets wechselt.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: true
- **Seit**: 6.1.0

mail.notification.sender

- **Modul**: cmas-core-server
- **Beschreibung**: From-Adresse der Benachrichtigungs-E-Mails, die verschickt werden, wenn der Bearbeiter eines Tickets wechselt. Wenn nicht gesetzt, wird stattdessen `cmas-core-security, admin.email` verwendet.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: cm6notification@cm6installation
- **Seit**: 6.6.3

mail.redelivery.retry.count

- **Modul**: cmas-core-server
- **Beschreibung**: Anzahl der erneuten Zustellversuche einer ausgehenden E-Mail.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 3
- **Seit**: 6.1.0

mail.smtp.email

- **Modul**: cmas-core-server
- **Beschreibung**: SMTP-E-Mail-URL für ausgehende E-Mails.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: smtp://mail.mydomain.com:25
- **Seit**: 6.0

mail.smtp.envelopesender

- **Modul**: cmas-core-server
- **Beschreibung**: E-Mail-Adresse, die als Absender im SMTP-Envelope benutzt wird. Wenn nicht gesetzt, wird die From-Adresse der E-Mail benutzt.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: mysender@mydomain.com
- **Seit**: 6.5.7

mail.smtp.tls.enabled

- **Modul**: cmas-core-server
- **Beschreibung**: Aktiviert SMTP über SSL/TLS (SMTPS) für das Senden von E-Mails aus dem Web Client und aus Skripten. Der Standardwert ist „false“. Wenn Sie diese Poperty auf „true“ setzen, ist SMTPS zum Senden von E-Mails aktiviert.
- **Typ**: boolean
- **Neustart erforderlich**: ja
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.11.1.6

max.licences.perUser

- **Modul**: cmas-core-server
- **Beschreibung**: Setzt die maximale Anzahl an Lizenzen, die ein einzelner Benutzer benutzen kann (z. B. durch Anmelden in einem anderen Browser). Standardmäßig ist dieser Wert nicht beschränkt.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 10
- **Seit**: 6.8.4.5

monitoring.engineer.login

- **Modul**: cmas-core-server
- **Beschreibung**: Login des Monitoring-Bearbeiters.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: nagios
- **Seit**: 6.9.3.0

monitoring.unit.login

- **Modul**: cmas-core-server
- **Beschreibung**: Login der Monitoring-Unit.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: nagios
- **Seit**: 6.9.3.0

nimh.enabled

- **Modul**: cmas-core-server
- **Beschreibung**: Aktiviert den NIMH-Dienst. Im Cluster muss die Node-ID angehängt werden, z. B. `nimh.enabled.NODEID` = „true“.
- **Typ**: boolean

- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.9.4.0

number.of.tasks

- **Modul**: cmas-core-server
- **Beschreibung**: Anzahl der Threads, die das Task Execution Framework (TEF) verwendet.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 1
- **Seit**: 6.9.4.0

recent.items.cleanup.cluster.node.id

- **Modul**: cmas-core-server
- **Beschreibung**: Wert einer `-Dcmas.clusternode.id`, die den Node angibt, der die zuletzt verwendeten Elemente bereinigt.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 1 (unter Annahme, dass der Cluster-Node mit dem Parameter `-Dcmas.clusternode.id=1` gestartet wurde)
- **Seit**: 6.11.0.1

recent.items.cleanup.interval.minutes

- **Modul**: cmas-core-server
- **Beschreibung**: Steuert das Zeitintervall (in Minuten), in dem das Entfernen von zuletzt verwendeten Elementen geprüft wird.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein

- **Beispielwert**: 60
- **Seit**: 6.11.0.1

recent.items.max.per.engineer

- **Modul**: cmas-core-server
- **Beschreibung**: Maximale Anzahl an zuletzt verwendeten Elementen pro Bearbeiter, die bei der Bereinigung (ältere Elemente werden gelöscht) beibehalten werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 500
- **Seit**: 6.11.0.1

recent.items.persistence.enabled

- **Modul**: cmas-core-server
- **Beschreibung**: Ermöglicht die Persistenz der zuletzt verwendeten Elemente, wenn false - verhindert das Speichern von neuen Elementen.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: true
- **Seit**: 6.11.1.0

resource.replace.batchSize

- **Modul**: cmas-core-server
- **Beschreibung**: Legt die Anzahl der Objekte fest, die in einer Ressourcen-Ersetzen-Aktion verarbeitet werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 5
- **Seit**: 6.10.0.0

resource.replace.timeout

- **Modul**: cmas-core-server
- **Beschreibung**: Transaktions-Timeout (in Sekunden) für einen Schritt einer Ressourcen-Ersetzen-Aktion.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 120
- **Seit**: 6.10.0.0

script.logging.threshold.seconds

- **Modul**: cmas-core-server
- **Beschreibung**: Legt die Zeit (in Sekunden) fest, nach deren Ablauf während einer Skript-Ausführung eine Warnung in der Log-Datei generiert wird.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 10 (Standardwert)
- **Seit**: 6.10.1.0

serial.mods.tracking.enabled

- **Modul**: cmas-core-server
- **Beschreibung**: Systemnahes technisches Flag, das entscheidet, ob die serielle Nachverfolgung von Änderungen für Entitäten aktiviert ist. Wenn diese Property aktiviert ist, gibt es keinen `StackOverflow`-Fehler, wenn Abhängigkeiten zwischen zwei Entitäten (zum Beispiel Bearbeiter und Ticket) eine Endlosschleife und infolgedessen einen StackOverflow auslösen. Diese Property muss manuell zur Konfiguration hinzugefügt werden. Sie wird nicht während der Einrichtung oder Aktualisierung zur Systemkonfiguration hinzugefügt.

  > ⚠ Aktivieren Sie das in diesem Abschnitt beschriebene eingeschränkte Verhalten bei Ticketänderungen nur, wenn Sie von einem ConSol-Vertreter dazu aufgefordert werden. Es handelt sich um ein systemnahes technisches Flag mit komplexen Auswirkungen auf das Systemverhalten und sollte daher nicht ohne genaue Untersuchung verwendet werden.

- **Typ**: boolean
- **Neustart erforderlich**: nein

- **System**: nein
- **Optional**: ja
- **Beispielwert**: false (Standardwert)
- **Seit**: 6.10.7.0, 6.11.0.5

server.session.archive.reaper.interval

- **Modul**: cmas-core-server
- **Beschreibung**: Reaper-Intervall (in Sekunden) von archivierten Server-Sessions.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.7.1

server.session.archive.timeout

- **Modul**: cmas-core-server
- **Beschreibung**: Timeout der Gültigkeit der Server-Session-Archive (in Tagen). Nach diesem Zeitraum werden die Informationen zur Session aus der Datenbank entfernt.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 31
- **Seit**: 6.7.1

server.session.reaper.interval

- **Modul**: cmas-core-server
- **Beschreibung**: Interval (in Sekunden), in dem der sog. Reaper inaktive (= beendete) Server-Sessions löscht.
- **Typ**: integer
- **Neustart erforderlich**: Nur Session Service
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 60
- **Seit**: 6.6.1, 6.7.1

server.session.timeout

- **Modul**: cmas-core-server
- **Beschreibung**: Server-Session-Timeout (in Sekunden) für verbundene Clients. Jeder Client kann dieses Timeout mit benutzerdefinierten Werten mittels seiner ID (ADMIN_TOOL, WEB_CLIENT, WORKFLOW_EDITOR, TRACK (vor 6.8 bitte PORTER verwenden), ETL, REST) überschreiben. Diese wird an den Namen der System-Property angehängt, z. B. `server.session.timeout.ADMIN_TOOL`.
  Siehe auch die Attribute der Seitenanpassung *updateTimeServerSessionActivityEnabled* und *updateTimeServerSessionActivity*, beide vom Typ *cmApplicationCustomization*.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1800
- **Seit**: 6.6.1, 6.7.1

Detaillierte Erklärung für das Admin Tool:

- server.session.timeout.ADMIN_TOOL
  Definiert den Zeitraum, den der Server eine Session im Admin Tool als gültig betrachtet, wenn keine Aktivität im Admin Tool der Session erfolgt. Das Admin Tool kennt diesen Wert nicht, es bemerkt nur eine ungültige Session, wenn es länger keine Aktivität gegeben hat.
- admin.tool.session.check.interval
  Definiert den Zeitraum zwischen zwei Überprüfungen durch das Admin Tool, ob der Server die Session noch als gültig betrachtet.

Wenn zum Beispiel admin.tool.session.check.interval = 60, fragt das Admin Tool den Server einmal pro Minute, ob die Session noch aktiv/gültig ist. Wenn server.session.timeout.ADMIN_TOOL = 600 erhält das Admin Tool die Antwort, dass die Session ungültig ist, nach zehn Minuten der Inaktivität.

skip.wfl.transfer.cleanup

- **Modul**: cmas-core-server
- **Beschreibung**: Wenn diese System-Property auf *true* gesetzt wird, wird das Workflow-Cleanup nach der Übertragung übersprungen.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false (Standardwert)
- **Seit**: 6.9.4.1

skip.wfl.transfer.translations.cleanup

- **Modul**: cmas-core-server
- **Beschreibung**: Aktiviert das Überspringen der Bereinigung von lokalisierten Properties von entfernten Workflow-Elementen.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false (Standardwert)
- **Seit**: 6.10.5.5

strict.utf.bmp.enabled

- **Modul**: cmas-core-server
- **Beschreibung**: In ConSol CM-Versionen unter 6.10.6 konnten eingehende E-Mails, deren Betreff 4-Byte UTF8-Zeichen enthält, von einigen Installationen mit einer MySQL-Datenbank-Engine nicht verarbeitet werden. Der Grund dafür ist, dass die Codierungs-/Collation-Konfiguration der Datenbank die 2-Byte BMP (Basic Multilingual Plane), Plane 0, verwendet und diese in einigen Installationen aus technischen Gründen nicht geändert werden kann. Andere Datenbanken sind davon nicht betroffen. E-Mails mit dieser Codierung konnten in allen CM-Versionen unter 6.10.6 nicht ins System importiert werden. Um dieses Problem zu umgehen, ist diese System-Property zur Konfiguration verfügbar.
  Mit der Einstellung „true“ werden alle 4-Byte UTF8-Zeichen herausgefiltert, bevor eine Interaktion mit der Datenbank stattfindet, sodass die oben beschriebenen Probleme nicht auftreten.
  Der Property-Wert „true“ ist die Standardeinstellung für MySQL-Datenbanken und „false“ ist die Standardeinstellung für alle anderen Datenbanken, in denen diese Property nicht benötigt wird. Ändern Sie den Wert für eine MySQL-Datenbank nur, wenn die Datenbankeinstellungen 4-Byte-Zeichen unterstützen.
- **Typ**: Boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 100
- **Seit**: 6.10.6.0

task.execution.interval.seconds

- **Modul**: cmas-core-server
- **Beschreibung**: Zeit in Sekunden zwischen dem Ende eines abgeschlossenen Tasks im TEF (Task Execution Framework) und dem Beginn des nächsten Tasks.
- **Typ**: Integer

- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: nein
- **Beispielwert**: 5
- **Seit**: 6

task.execution.node.id

Modul: cmas-core-server

Beschreibung: Nur in geclusterten Umgebungen relevant. Die ID des Nodes, auf dem automatisch gestartete Skripte des TEF (Task Execution Framework) ausgeführt werden. (Manuell über das Admin Tool gestartete Skripte werden auf dem Rechner ausgeführt, auf dem das Admin Tool läuft.)

Typ: Integer

Neustart erforderlich: nein

System: nein

Optional: nein

Beispielwert: 2

Seit: 6.11.0.1

tickets.delete.size

- **Modul**: cmas-core-server
- **Beschreibung**: Definiert die Anzahl der Tickets, die pro Transaktion gelöscht werden. Standardmäßig ist dieser Wert 10.
- **Typ**: integer
- **Neustart erforderlich**: Nur Session Service
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 10
- **Seit**: 6.8.1

ticket.delete.timeout

- **Modul**: cmas-core-server
- **Beschreibung**: Transaktions-Timeout (in Sekunden) beim Löschen von Tickets.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein

- **Beispielwert**: 60
- **Seit**: 6.1.3

ticket.from.incoming.message.accepted.links

- **Modul**: cmas-core-server
- **Beschreibung**: Liste mit Domänen, zu denen die Links in eingehenden E-Mails und Links in Kommentaren, die über die REST API hinzugefügt wurden, im Ticketprotokoll angeklickt werden können. Es können reguläre Ausdrücke verwendet werden, um die erlaubten URLs anzugeben. Es können mehrere, durch Leerzeichen getrennte URLs angegeben werden. Die URL muss mit einem der erlaubten Protokolle (http, https, ftp, ftps, file, mailto) beginnen. Alle anderen Links werden entfernt, d. h. der Link wird im Ticketprotokoll als Text dargestellt, kann aber nicht angeklickt werden. Ist die Property leer, werden alle Links entfernt. Der reguläre Ausdruck `.+` kann verwendet werden, um alle Domänen zu erlauben.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: nein
- **Beispielwert**: https://.*\.consol\.de (erlaubt Links auf „https://<beliebig>.consol.de“)
- **Seit**: 6.11.1.7



Beachten Sie, dass ConSol CM durch das Whitelisting von Domänen anfällig für Cross-Site-Scripting und andere Angriffe werden kann. Wählen Sie die Domänen in Ihrer Whitelist sorgfältig aus!

transaction.timeout.minutes

- **Modul**: cmas-core-server
- **Beschreibung**: Setzt den Timeout für den Task Execution Service des TEF, d. h. ein Durchlauf eines Tasks muss vor dem Ablauf dieser Zeitspanne abgeschlossen sein. Die Änderungen sind nur für neue Aufgaben sichtbar, deren Ausführung nach der Konfigurationsänderung beginnt.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 10*3600 (10 Stunden - Standardwert)
- **Seit**: 6.10

unit.description.mode

- **Modul**: cmas-core-server
- **Beschreibung**: Legt fest, ob die Kontaktbeschreibungen im Ticketprotokoll aus der Datenbank abgerufen oder dynamisch über Templates erzeugt werden. Der Wert „DYNAMIC“ ist aus Performance-Sicht etwas kostspieliger. „PROTOCOL“ ist schneller, gibt aber historische Namen zurück, die unter Umständen veraltet sind. Verwenden Sie „PROTOCOL“, wenn Sie viele Protokolleinträge von vielen unterschiedlichen Kontakten haben. Dies ist auch der Standardwert in CM-Versionen 6.11.1.1 und höher. In CM-Versionen bis 6.11.1.0 ist der Standardwert „DYNAMIC“.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: PROTOCOL
- **Seit**: 6.11.0

unit.replace.batchSize

- **Modul**: cmas-core-server
- **Beschreibung**: Legt die Anzahl der Objekte fest, die in einer Unit-Ersetzen-Aktion verarbeitet werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 5
- **Seit**: 6.8.2

unit.replace.timeout

- **Modul**: cmas-core-server
- **Beschreibung**: Transaktions-Timeout (in Sekunden) für einen Schritt einer Unit-Ersetzen-Aktion.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 120
- **Seit**: 6.8.2

unused.content.remover.cluster.node.id

- **Modul**: cmas-core-server
- **Beschreibung**: Wert eines `cmas.clusternode.id`, der angibt, welcher Node nicht verwendete Ticket-Attachments und Unit-Inhaltseinträge entfernt.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 1 (in der Annahme, dass der Cluster-Node mit dem Parameter `-Dcmas.clusternode.id=1` gestartet ist)
- **Seit**: 6.9.0.0

unused.content.remover.enabled

- **Modul**: cmas-core-server
- **Beschreibung**: Legt fest, ob das Entfernen ungenutzter Ticket-Attachments und Unit-Inhaltseinträge durchgeführt werden soll.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: true
- **Seit**: 6.9.0.0

unused.content.remover.polling.minutes

- **Modul**: cmas-core-server
- **Beschreibung**: Legt fest, wie oft überprüft werden soll, ob ungenutzte Ticket-Attachments und Unit-Inhaltseinträge zum Entfernen vorhanden sind.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 15
- **Seit**: 6.9.0.0

unused.content.remover.ttl.minutes

- **Modul**: cmas-core-server
- **Beschreibung**: Mindestzeitraum, nach dem ungenutzte Ticket-Attachments und Unit-Inhalt-seinträge entfernt werden können.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1440
- **Seit**: 6.9.0.0

warmup.executor.enabled

- **Modul**: cmas-core-server
- **Beschreibung**: Legt fest, ob der Server beim Start asynchron anlaufen soll und nebenher andere Aufgaben erledigt (z. B. interne Caches füllen).
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: true
- **Seit**: 6.9.4.2

wfl.sticky.transfer.disabled

- **Modul**: cmas-core-server
- **Beschreibung**: Aktiviert die Verwendung von beibehaltenen Originalnamen von Workflow-Elementen.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.10.1.0

H.2.3.6 cmas-core-shared (Modul)

cluster.mode

- **Modul**: cmas-core-shared
- **Beschreibung**: Legt fest, ob CMAS in einem Cluster läuft.
- **Typ**: boolean
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.1.0

cluster.unicast

- **Modul**: cmas-core-shared
- **Beschreibung**: Flag zum Aktivieren des Jgroups-Unicast-Modus für ConSol CM-Cluster (im Gegensatz zum standardmäßigen Multicast-Modus, der in einigen Rechenzentrumsumgebungen zu Problemen führt). Wenn die Property auf „true“ gesetzt ist, müssen Sie die folgenden JVM-Startparameter setzen: `jgroups.bind.port`, `jgroups.bind.address` und `jgroups.initial_hosts`.
- **Typ**: boolean
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: ja
- **Beispielwert**: false (Standardwert)
- **Seit**: 6.11.0.0

data.directory

- **Modul**: cmas-core-shared
- **Beschreibung**: Verzeichnis für die CMAS-Daten (z. B. Index)
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: C:\Users\user\cmas
- **Seit**: 6.0

expert.mode

- **Modul**: cmas-core-shared
- **Beschreibung**: Schaltet den Expertenmodus ein bzw. aus, sodass Funktionen für Experten verfügbar gemacht bzw. blockiert werden. Zum Beispiel ist die CM-System-Property `initialized` nur im Expertenmodus verfügbar.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.0

H.2.3.7 cmas-dwh-server (Modul)

autocommit.cf.changes

- **Modul**: cmas-dwh-server
- **Beschreibung**: Definiert, ob DWH-Aufgaben, die aufgrund von Konfigurationsänderungen an Ticketfeldern anfallen, automatisch ohne manuelle Interaktion im Admin Tool ausgeführt werden. Diese Property kann auch im Admin Tool im Navigationselement *DWH* gesetzt werden. Der Standardwert und empfohlene Wert ist *false*.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.7.0

batch-commit-interval

- **Modul**: cmas-dwh-server
- **Beschreibung**: Anzahl der Objekte in einer JMS-Nachricht. Höhere Werte bedeuten eine bessere Übertragungsperformance und größeren Speicherverbrauch.
  Ab ConSol CM-Version 6.11 wird diese Property nur verwendet, wenn die Paketgröße einer DWH-Operation nicht gesetzt ist. Dies kann nur der Fall sein, wenn der Befehl direkt an die Java MBean `consol.cmas.global.dwh.synchronizationService` gerichtet ist, z. B. durch Verwendung der Methode `update()`. Wenn eine DWH-Operation über das Admin Tool gestartet wird, gibt es immer einen Wert für die Paketgröße. Wenn dieser nicht explizit gesetzt ist, wird der Standardwert 1000 als Wert für `batch.commit.interval` verwendet.
- **Standardwert**: 1000
- **Typ**: integer

- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 100
- **Seit**: 6.0.0

communication.channel

- **Modul**: cmas-dwh-server
- **Beschreibung**: Kommunikationskanal, mögliche Werte sind DIRECT (Datenbank-Kommunikationskanal, Standardwert seit 6.9.4.1) oder JMS (Standardwert vor 6.9.4.1). Diese System-Property muss vor 6.9.4.1 extra hinzugefügt werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: DIRECT
- **Seit**: 6.8.5.0
- **Entfernt in**: 6.11.0.0 (Der einzige verfügbare und automatisch gesetzte Modus ist „DIRECT“.)

dwh.mode

- **Modul**: cmas-dwh-server
- **Beschreibung**: Aktueller Modus der DWH-Datenübermittlung. Mögliche Werte sind OFF, ADMIN, LIVE.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: OFF
- **Seit**: 6.0.1

ignore-queues

- **Modul**: cmas-dwh-server
- **Beschreibung**: Durch eine durch Kommas getrennte Liste von Queue-Namen wird hier festgelegt, dass Tickets dieser Queues nicht ins DWH übermittelt werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja

- **Optional**: ja
- **Beispielwert**: QueueName1,QueueName2,QueueName3
- **Seit**: 6.6.19
- **Entfernt in**: 6.8.1

is.cmrf.alive

- **Modul**: cmas-dwh-server
- **Beschreibung**: Als Startpunkt sollte die Zeit genommen werden, bei der zuletzt eine Nachricht an das CMRF gesendet wurde. Wenn nach diesem Wert (in Sekunden) keine Antwort vom CMRF empfangen wird, wird ein DWH-Betriebsstatus mit der Fehlermeldung, dass das CMRF nicht erreichbar ist, erstellt.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1200
- **Seit**: 6.7.0

java.naming.factory.initial

- **Modul**: cmas-dwh-server
- **Beschreibung**: Factory-Java-Klasse für DWH context factory.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: org.jnp.interfaces.NamingContextFactory
- **Seit**: 6.0.1
- **Entfernt in**: 6.11.0.0

java.naming.factory.url.pkgs

- **Modul**: cmas-dwh-server
- **Beschreibung**:
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: org.jboss.naming:org.jnp.interfaces

- **Seit**: 6.0.1
- **Entfernt in**: 6.11.0.0

java.naming.provider.url

- **Modul**: cmas-dwh-server
- **Beschreibung**: URL des Naming Provider.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: localhost
- **Seit**: 6.0.1
- **Entfernt in**: 6.11.0.0

last.ping.timestamp

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 32323214
- **Seit**: 6.11.0.1

live.start

- **Modul**: cmas-dwh-server
- **Beschreibung**: Wenn der DWH-Synchronisierungsmodus mit dem Admin Tool (Navigationsgruppe *Data Warehouse*, Navigationselement *Administration*, Button *Konfiguration*) auf „LIVE“ gesetzt wird, wird diese Property erstellt und auf das aktuelle Datum gesetzt. Wenn der LIVE-Modus nicht aktiviert ist und sich keine Daten in `cmas_dwh_ser_sync_object` befinden, wird die Property `live.start` gelöscht.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja (im DWH-Modus „LIVE“ automatisch hinzugefügt)
- **Beispielwert**: 15028802377645
- **Seit**: 6.7.0

notification.error.description

- **Modul**: cmas-dwh-server
- **Beschreibung**: Text für E-Mails mit Fehlermeldungen des DWH.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: Fehler aufgetreten
- **Seit**: 6.0.1

notification.error.from

- **Modul**: cmas-dwh-server
- **Beschreibung**: From-Adresse für E-Mails mit Fehlermeldungen des DWH.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.0.1

notification.error.subject

- **Modul**: cmas-dwh-server
- **Beschreibung**: Betreff für E-Mails mit Fehlermeldungen des DWH.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: Fehler aufgetreten
- **Seit**: 6.0.1

notification.error.to

- **Modul**: cmas-dwh-server
- **Beschreibung**: To-Adresse für E-Mails mit Fehlermeldungen des DWH.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein

- **Beispielwert**: myuser@consol.de
- **Seit**: 6.0.1

notification.finished_successfully.description

- **Modul**: cmas-dwh-server
- **Beschreibung**: Text für E-Mails des DWHs, wenn eine Übertragung erfolgreich beendet wurde.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: Übertragung erfolgreich beendet
- **Seit**: 6.0.1

notification.finished_successfully.from

- **Modul**: cmas-dwh-server
- **Beschreibung**: From-Adresse für E-Mails des DWHs, wenn eine Übertragung erfolgreich beendet wurde.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.0.1

notification.finished_successfully.subject

- **Modul**: cmas-dwh-server
- **Beschreibung**: Betreff für E-Mails des DWHs, wenn eine Übertragung erfolgreich beendet wurde.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: Übertragung erfolgreich beendet
- **Seit**: 6.0.1

notification.finished_successfully.to

- **Modul**: cmas-dwh-server
- **Beschreibung**: To-Adresse für E-Mails des DWHs, wenn eine Übertragung erfolgreich beendet wurde.
- **Typ**: string
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: myuser@consol.de
- **Seit**: 6.0.1

notification.finished_unsuccessfully.description

- **Modul**: cmas-dwh-server
- **Beschreibung**: Text für E-Mails des DWHs, wenn eine Übertragung nicht erfolgreich beendet wurde.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: Übertragung nicht erfolgreich beendet
- **Seit**: 6.0.1

notification.finished_unsuccessfully.from

- **Modul**: cmas-dwh-server
- **Beschreibung**: From-Adresse für E-Mails des DWHs, wenn eine Übertragung nicht erfolgreich beendet wurde.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.0.1

notification.finished_unsuccessfully.subject

- **Modul**: cmas-dwh-server
- **Beschreibung**: Betreff für E-Mails des DWHs, wenn eine Übertragung nicht erfolgreich beendet wurde.
- **Typ**: string

- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: Übertragung nicht erfolgreich beendet
- **Seit**: 6.0.1

notification.finished_unsuccessfully.to

- **Modul**: cmas-dwh-server
- **Beschreibung**: To-Adresse für E-Mails des DWHs, wenn eine Übertragung nicht erfolgreich beendet wurde.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: myuser@consol.de
- **Seit**: 6.0.1

notification.host

- **Modul**: cmas-dwh-server
- **Beschreibung**: Hostname des E-Mail-Servers (SMTP) für das Senden von DWH-E-Mails.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: myserver.consol.de
- **Seit**: 6.0.1

notification.password

- **Modul**: cmas-dwh-server
- **Beschreibung**: Passwort für das Senden von DWH-E-Mails. (optional)
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.0.1

notification.port

- **Modul**: cmas-dwh-server
- **Beschreibung**: SMTP-Port für das Senden von DWH-E-Mails.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 25
- **Seit**: 6.0.1

notification.protocol

- **Modul**: cmas-dwh-server
- **Beschreibung**: Das Protokoll, das für das Senden von E-Mails aus dem DWH verwendet wird.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: pop3\

notification.tls.enabled

- **Modul**: cmas-dwh-server
- **Beschreibung**: Aktiviert SMTP über SSL/TLS (SMTPS) für das Senden von Benachrichtigungs-E-Mails aus dem DWH. Der Standardwert ist „false“. Wenn der Wert auf „true“ gesetzt ist, ist SMTPS für das Senden von DWH-Benachrichtigungen aktiviert.
- **Typ**: string
- **Neustart erforderlich**: ja
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.11.1.6

notification.username

- **Modul**: cmas-dwh-server
- **Beschreibung**: Benutzername (SMTP) für das Senden von DWH-E-Mails.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja

- **Optional**: ja
- **Beispielwert**: myuser
- **Seit**: 6.0.1

recoverable.exceptions

- **Modul**: cmas-dwh-server
- **Beschreibung**: Durch Kommas getrennte Liste mit Exception-Definitionen: CLASS[+][:REGEX]. Die in der Liste enthaltenen Exceptions verhindern nicht, dass CM an den CMRF-Prozess sendet, sondern erzwingen einen erneuten Versuch. Wenn nach CLASS ein optionales '+' steht, gilt dies auch für Klassen, die CLASS erweitern.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: java.sql.SQLRecoverableException,java.lang.RuntimeException+:.*T.1\,2T.*
- **Seit**: 6.8.4.6

skip-ticket

- **Modul**: cmas-dwh-server
- **Beschreibung**: Tickets werden bei der Übertragung/Aktualisierung nicht übermittelt.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.6.19
- **Entfernt in**: 6.8.1

skip-ticket-history

- **Modul**: cmas-dwh-server
- **Beschreibung**: Ticketprotokoll wird bei der Übertragung/Aktualisierung nicht übermittelt.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false

- **Seit**: 6.6.19
- **Entfernt in**: 6.8.1

skip-unit

- **Modul**: cmas-dwh-server
- **Beschreibung**: Units werden bei der Übertragung/Aktualisierung nicht übermittelt.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.6.19
- **Entfernt in**: 6.8.1

skip-unit-history

- **Modul**: cmas-dwh-server
- **Beschreibung**: Unit-Protokoll wird bei der Übertragung/Aktualisierung nicht übermittelt.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.6.19
- **Entfernt in**: 6.8.1

split.history

- **Modul**: cmas-dwh-server
- **Beschreibung**: Ändert das SQL-Statement dahingehend, dass Ticketprotokolle während der Übertragung an das DWH nicht für alle Tickets auf einmal abgeholt werden, sondern für ein Ticket pro SQL-Statement.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.8.0

statistics.calendar

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.client.group

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.contact.role

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.content.entry

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja

- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.content.entry.class

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.content.entry.history

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.customer.definition

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.engineer

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.

- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.enum.group

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.field.definition

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.group.definition

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.locale

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.localized.property

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.mla

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.project

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja

- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.queue

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.resource

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.resource.group

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.resource.history

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.

- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.resource.relation.definition

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.resource.type

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.ticket

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.ticket.function

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.ticket.history

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.time.booking

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.timestamp

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja

- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.unit

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.unit.history

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.unit.relation.definition

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

statistics.workflow

- **Modul**: cmas-dwh-server
- **Beschreibung**: Interne DWH-Property, sollte nicht manuell geändert werden.

- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.1

time.buffer

- **Modul**: cmas-dwh-server
- **Beschreibung**: Anzahl der Minuten zur Erweiterung des Datums des Starts des Live-Modus.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 5
- **Seit**: 6.8.1.11

unit.transfer.order

- **Modul**: cmas-dwh-server
- **Beschreibung**: Legt fest, in welcher Reihenfolge Kundenfeldgruppen an das DWH übertragen werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: company;customer
- **Seit**: 6.6.19
- **Entfernt in**: 6.8.1

H.2.3.8 cmas-nimh (Modul)

filesystem.polling.threads.number

- **Modul**: cmas-nimh
- **Beschreibung**: Anzahl der Threads, die für die Abfrage der Datenbank-E-Mail-Warteschlange gestartet werden. Standardwert: 1.
- **Typ**: integer
- **Neustart erforderlich**: nein

- **System**: nein
- **Optional**: ja
- **Beispielwert**: 10
- **Seit**: 6.4.0

filesystem.polling.threads.shutdown.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Wartezeit nach dem Shutdown-Signal. Wenn der Timeout erreicht ist, wird der Thread beendet. Standardwert: 60.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

filesystem.polling.threads.watchdog.interval.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Intervall des Watchdog Threads. Standardwert: 30.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

filesystem.task.enabled

- **Modul**: cmas-nimh
- **Beschreibung**: Mit dieser System-Property kann der Service Thread eines bestimmten Pollers deaktiviert werden. Standardwert: true.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.4.0

filesystem.task.interval.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Standard-Intervall für den Abruf von Postfächern. Standardwert: 60 Sekunden
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

filesystem.task.polling.folder

- **Modul**: cmas-nimh
- **Beschreibung**: Speicherort des Ordners, der überwacht und nach E-Mails im Format von eml-Dateien durchsucht wird. Standard: Unterverzeichnis "mail" des cmas-Datenverzeichnisses
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: c://cmas//mail
- **Seit**: 6.4.0

filesystem.task.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Nach dieser Zeit (der Inaktivität) wird der Service Thread als beschädigt betrachtet und automatisch neu gestartet. Standardwert: 120 Sekunden
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

filesystem.task.transaction.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Standard-Transaktions-Timeout für Transaktionen, die E-Mails abrufen. Sollte mit der Anzahl der Nachrichten, die gleichzeitig abgeholt werden, korrelieren. Standardwert: 60 Sekunden

- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

mailbox.1.connection.host

- **Modul**: cmas-nimh
- **Beschreibung**: Host (Server) für das erste konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.host`.

mailbox.1.connection.password

- **Modul**: cmas-nimh
- **Beschreibung**: Passwort für das erste konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.password`.

mailbox.1.connection.port

- **Modul**: cmas-nimh
- **Beschreibung**: Port für das erste konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.port`.

mailbox.1.connection.protocol

- **Modul**: cmas-nimh
- **Beschreibung**: Protokoll (z. B. IMAP oder POP3) für das erste konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.protocol`.

mailbox.1.connection.username

- **Modul**: cmas-nimh
- **Beschreibung**: Benutzername für das erste konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.username`.

mailbox.2.connection.host

- **Modul**: cmas-nimh
- **Beschreibung**: Host (Server) für das zweite konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.host`.

mailbox.2.connection.password

- **Modul**: cmas-nimh
- **Beschreibung**: Passwort für das zweite konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.password`.

mailbox.2.connection.port

- **Modul**: cmas-nimh
- **Beschreibung**: Port für das zweite konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.port`.

mailbox.2.connection.protocol

- **Modul**: cmas-nimh
- **Beschreibung**: Protokoll (z. B. IMAP oder POP3) für das zweite konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.protocol`.

mailbox.2.connection.username

- **Modul**: cmas-nimh
- **Beschreibung**: Benutzername für das zweite konfigurierte Postfach. Überschreibt den Standardparameter `mailbox.default.connection.username`.



Für alle Postfach-Properties im Zusammenhang mit NIMH gilt folgendes Prinzip: Es wird eine Standard-Property definiert (z.B `mailbox.default.connection.port`). Wenn keine postfachspezifische Property konfiguriert ist, wird dieser Standardwert verwendet.

mailbox.default.connection.host

- **Modul**: cmas-nimh
- **Beschreibung**: Host (Servername) eines bestimmten Postfachs, aus dem der Poller E-Mails liest.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 10.10.1.157
- **Seit**: 6.4.0

mailbox.default.connection.password

- **Modul**: cmas-nimh
- **Beschreibung**: Passwort für ein bestimmtes Postfach, aus dem der Poller E-Mails liest.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: consol
- **Seit**: 6.4.0

mailbox.default.connection.port

- **Modul**: cmas-nimh
- **Beschreibung**: Port für ein bestimmtes Postfach, aus dem der Poller E-Mails liest.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 143
- **Seit**: 6.4.0

mailbox.default.connection.protocol

- **Modul**: cmas-nimh
- **Beschreibung**: Protokoll des Pollers, z. B. IMAP oder POP3. Kein Standardwert.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: imap
- **Seit**: 6.4.0

mailbox.default.connection.username

- **Modul**: cmas-nimh
- **Beschreibung**: Benutzername für ein bestimmtes Postfach, aus dem der Poller E-Mails liest.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: username
- **Seit**: 6.4.0

mailbox.default.session.mail.debug

- **Modul**: cmas-nimh
- **Beschreibung**: Beispiel für javax.mail-Property - ermöglicht detaillierteres Session-Debugging mit javax.mail.
- **Typ**: boolean
- **Neustart erforderlich**: nein

- **System**: nein
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.4.0

mailbox.default.session.mail.imap.timeout

- **Modul**: cmas-nimh
- **Beschreibung**: Beispiel für javax.mail-Property
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 120
- **Seit**: 6.4.0

mailbox.default.session.mail.mime.address.strict

- **Modul**: cmas-nimh
- **Beschreibung**: Beispiel für javax.mail-Property - Gegenstück zum alten `mail.mime.strict` von Mule, erlaubt das nicht so strikte Parsen des E-Mail-Headers.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.4.0

mailbox.default.session.mail.pop3.timeout

- **Modul**: cmas-nimh
- **Beschreibung**: Beispiel für javax.mail-Property
- **Typ**:
- **Neustart erforderlich**:
- **System**:
- **Optional**:
- **Beispielwert**:
- **Seit**: 6.4.0

mailbox.default.session.mail.<protocol>.fetchsize

- **Modul**: cmas-nimh
- **Beschreibung**: Setzt die JavaMail-Property für die Größe der `fetchsize` beim `partialfetch` in Bytes für das angegebene Protokoll.
  Für IMAP-Systeme: In CM-Versionen ab 6.10.7.0 wird der Wert von `mailbox.default.session.mail.imap.fetchsize` bei der initialen Einrichtung eines ConSol CM-Systems auf „1048576“ (entspricht 1 MB) gesetzt. Bei einer Aktualisierung eines vorhandenen ConSol CM-Systems bleibt der Wert der Property unverändert, wenn die Property bereits vorhanden ist. Wenn die Property noch nicht vorhanden ist, wird sie mit dem Standardwert hinzugefügt.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 1048576
- **Seit**: 6.9.4.0

mailbox.default.session.mail.<protocol>.partialfetch

- **Modul**: cmas-nimh
- **Beschreibung**: Setzt die JavaMail-Property für `partialfetch`, d. h. steuert, ob die PartialFetch-Funktion des Protokolls genutzt werden soll.
  Für IMAP-Systeme: In CM-Versionen ab 6.10.7.0 wird der Wert von `mailbox.default.session.mail.imap.partialfetch` bei der initialen Einrichtung eines ConSol CM-Systems auf „false“ gesetzt. Bei einer Aktualisierung eines vorhandenen ConSol CM-Systems bleibt der Wert der Property unverändert, wenn die Property bereits vorhanden ist. Wenn die Property noch nicht vorhanden ist, wird sie mit dem Standardwert hinzugefügt.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**:
- **Seit**: 6.9.4.0

mailbox.default.task.delete.read.messages

- **Modul**: cmas-nimh
- **Beschreibung**: Legt fest, ob Nachrichten nach dem Lesen aus dem Postfach entfernt werden sollen. Beim IMAP-Protokoll werden Nachrichten standardmäßig als SEEN gekennzeichnet. Bei POP3 wird die Nachricht nur gelöscht, wenn der Wert auf true gesetzt ist. Anderenfalls bleibt die Nachricht auf dem Server, was zu einer Endlos-Leseschleife führt. Standardwert: false.

- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.4.0

mailbox.default.task.enabled

- **Modul**: cmas-nimh
- **Beschreibung**: Mit dieser System-Property kann der Service Thread eines bestimmten Pollers deaktiviert werden. Standardwert: true.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.4.0

mailbox.default.task.interval.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Standard-Intervall für den Abruf von Postfächern. Standardwert: 60 Sekunden
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

mailbox.default.task.max.message.size

- **Modul**: cmas-nimh
- **Beschreibung**: Maximale Größe von E-Mails (d. h. E-Mail plus Attachments). E-Mails, die größer als dieser Wert sind, werden nicht automatisch von NIMH verarbeitet. Sie werden in der Datenbank (Tabelle `cmas_nimh_archived_mail`) gespeichert und erscheinen daher in den E-Mail-Backups im Admin Tool (siehe Abschnitt E-Mail-Backups). Von dort können sie erneut gesendet, in das Dateisystem geladen oder gelöscht werden. Für diese Operationen ist die Größe der Nachricht nicht relevant. Standardmäßig auf 10 MB (10485760) gesetzt.
- **Typ**: integer
- **Neustart erforderlich**: nein

- **System**: nein
- **Optional**: ja
- **Beispielwert**: 10485760
- **Seit**: 6.4.0

mailbox.default.task.max.messages.per.run

- **Modul**: cmas-nimh
- **Beschreibung**: Anzahl der Nachrichten, die gleichzeitig aus dem Postfach abgeholt werden. Muss mit dem Transaktions-Timeout korrelieren. Standardwert: 20.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

mailbox.default.task.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Nach dieser Zeit (der Inaktivität) wird der Service Thread als beschädigt betrachtet und automatisch neu gestartet. Standardwert: 120 Sekunden
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

mailbox.default.task.transaction.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Standard-Transaktions-Timeout für Transaktionen, die E-Mails abrufen. Sollte mit der Anzahl der Nachrichten, die gleichzeitig abgeholt werden, korrelieren. Standardwert: 60 Sekunden
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja

- **Beispielwert**: 60
- **Seit**: 6.4.0

mailbox.polling.threads.mail.log.enabled

- **Modul**: cmas-nimh
- **Beschreibung**: Ermöglicht das Protokollieren von E-Mails, was in Cluster-Umgebungen besonders wichtig ist (wird dort als Semaphor genutzt).
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true (Standardwert)
- **Seit**: 6.9.4.1

mailbox.polling.threads.number

- **Modul**: cmas-nimh
- **Beschreibung**: Anzahl der Threads für den Zugriff auf Postfächer. Standardwert: 1.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 1
- **Seit**: 6.4.0

queue.polling.threads.number

- **Modul**: cmas-nimh
- **Beschreibung**: Anzahl der Threads, die zur Überwachung der E-Mail-Warteschlange gestartet werden. Standardwert: 1.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 1
- **Seit**: 6.4.0

queue.polling.threads.shutdown.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Wartezeit nach dem Shutdown-Signal. Wenn der Timeout erreicht ist, wird der Thread beendet. Standardwert: 60.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.4.0

queue.polling.threads.watchdog.interval.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Intervall des Watchdog Threads. Standardwert: 30.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 30
- **Seit**: 6.4.0

queue.task.error.pause.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Anzahl der Sekunden, die der Queue-Poller nach einem Infrastrukturfehler (z. B. der Datenbank) wartet. Standardwert: 180 Sekunden
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 180
- **Seit**: 6.4.0

queue.task.interval.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Thread-Intervall zur Überwachung der Haupt-E-Mail-Warteschlange. Standardwert: 15.
- **Typ**: integer

- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 15
- **Seit**: 6.4.0

queue.task.max.retries

- **Modul**: cmas-nimh
- **Beschreibung**: Maximale Anzahl der Versuche, eine E-Mail nach einer Exception erneut zu verarbeiten. Ist diese erreicht, wird die E-Mail archiviert. Die archivierte E-Mail kann über das NIMH API (oder das Admin Tool) wieder aktiviert werden.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 10
- **Seit**: 6.4.0

queue.task.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Nach dieser Zeit (der Inaktivität) wird der Service Thread als beschädigt betrachtet und automatisch neu gestartet. Standardwert: 600 Sekunden
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 600
- **Seit**: 6.4.0

queue.task.transaction.timeout.seconds

- **Modul**: cmas-nimh
- **Beschreibung**: Transaktions-Timeout für die E-Mail-Verarbeitung in der Warteschlange. Standardwert: 60.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja

- **Beispielwert**: 60
- **Seit**: 6.4.0

H.2.3.9 cmas-nimh-extension (Modul)

mail.attachments.validation.info.sender

- **Modul**: cmas-nimh-extension
- **Beschreibung**: Setzt den From-Header bei Attachments vom Typ `error notification mail`.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: admin@mail.com
- **Seit**: 6.7.5

mail.attachments.validation.info.subject

- **Modul**: cmas-nimh-extension
- **Beschreibung**: Setzt den Betreff bei Attachments vom Typ error notification mail.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: E-Mail konnte nicht verarbeitet werden, weil ihre Attachments zurückgewiesen wurden!
- **Seit**: 6.7.5

mail.db.archive

- **Modul**: cmas-nimh-extension
- **Beschreibung**: Wenn dieser Wert auf *true* gesetzt ist, werden eingehende E-Mails in der Datenbank archiviert.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: false (Standardwert)
- **Seit**: 6.8.5.5

mail.error.from.address

- **Modul**: cmas-nimh-extension
- **Beschreibung**: From-Adresse für Fehler-E-Mails von NIMH
- **Typ**: email
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: myuser@consol.de
- **Seit**: 6.4.0

mail.error.to.address

- **Modul**: cmas-nimh-extension
- **Beschreibung**: To-Adresse für Fehler-E-Mails von NIMH. Standardmäßig wird die E-Mail-Adresse des Administrators, die Sie bei der Systemeinrichtung eingegeben haben, verwendet.
- **Typ**: email
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: myuser@consol.de
- **Seit**: 6.4.0

mail.on.error

- **Modul**: cmas-nimh-extension
- **Beschreibung**: Wenn diese Property auf *true* gesetzt ist, wird im Fall, dass eine E-Mail nicht verarbeitet werden konnte, ein Fehler-E-Mail an die oben konfigurierten E-Mail-Adressen gesendet. Standardwert: true.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.4.0

mail.process.error

- **Modul**: cmas-nimh-extension
- **Beschreibung**: To-Adresse für E-Mails mit Fehlermeldungen des Mule.
- **Typ**: email

- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: myuser@consol.de
- **Seit**: 6.4.0

mail.ticketname.pattern

- **Modul**: cmas-nimh-extension
- **Beschreibung**: Regulärer Ausdruck, mit dem der Ticketname im Betreff eingehender E-Mails identifiziert wird.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: .*?Ticket\s+\((\S+)\).*
- **Seit**: 6.4.0

H.2.3.10 cmas-restapi-core (Modul)

comment.authors.disabled

- **Modul**: cmas-restapi-core
- **Beschreibung**: Deaktiviert die Anzeige des Autors des Inhalts über die REST-API. Der Standardwert ist „false“.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: false
- **Seit**: 6.11.0

diff.tracking.disabled

- **Modul**: cmas-restapi-core
- **Beschreibung**: Ausweich-Property zum Deaktivieren der Änderungsnachverfolgung für CM/Track. Diese ist protokollbasiert und kann viel Last erzeugen.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein

- **Optional**: ja
- **Beispielwert**: false (Standardwert)
- **Seit**: 6.10.5.6

security.fields.customer.exposure.check.enabled

- **Modul**: cmas-restapi-core
- **Beschreibung**: Aktiviert die Prüfung der Annotation `customer exposure` für Ticketfelder.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true (Standardwert)
- **Seit**: 6.10.5.4

security.restrict.unit.access.to.own.data

- **Modul**: cmas-restapi-core
- **Beschreibung**: Wenn die Property auf „true“ gesetzt ist, wird eine zusätzliche Prüfung durchgeführt, wenn sich ein Benutzer als Kunde über die REST-API anmeldet, z. B. in CM/Track. Wenn er Kundendaten anfordert, werden nur die Firma des Benutzers oder andere Kontakte der Firma des Benutzers zurückgegeben. Wenn die Property auf „false“ gesetzt ist, wird keine zusätzliche Sicherheitsprüfung durchgeführt und die früheren Sicherheitsregeln gelten.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: true (Standardwert)
- **Seit**: 6.9.2.14

H.2.3.11 cmas-setup-hibernate (Modul)

cmas.dropSchemaBeforeSetup

- **Modul**: cmas-setup-hibernate
- **Beschreibung**: Gibt an, ob das Schema während des Setups gelöscht werden soll (wurde).
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein

- **Beispielwert**: true
- **Seit**: 6.0

connection.release.mode

- **Modul**: cmas-setup-hibernate
- **Beschreibung**: Beschreibt die Strategie zur Handhabung der JEE-Verbindung bei Transaktionen. Wenn die Property auf „AFTER_TRANSACTION“ gesetzt ist, wird die Verbindung während der Transaktion zwischengespeichert und am Ende freigegeben. Wenn die Property auf „AFTER_ STATEMENT“ gesetzt ist, wird die Verbindung nach jeder Ausführung einer Anweisung freigegeben. Ändern Sie den Standardwert nur, wenn Sie von ConSol dazu aufgefordert werden.
- **Typ**: string
- **Neustart erforderlich**: ja
- **System**: nein
- **Optional**: ja
- **Beispielwert**: AFTER_STATEMENT (Standardwert für JEE-Umgebungen)
- **Seit**: 6.0

hibernate.dialect

- **Modul**: cmas-setup-hibernate
- **Beschreibung**: Der Hibernate-Dialekt. Normalerweise wird dieser Wert während des initialen Setups gesetzt (abhängig vom Datenbanksystem).
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: org.hibernate.dialect.MySQL5InnoDBDialect
- **Seit**: 6.0

update.6.11.0.0.sleep

- **Modul**: cmas-setup-hibernate
- **Beschreibung**: Hilfs-Property für die Skripte zur Update-Vorbereitung, die im Zusammenhang mit dem CM-Datenbank-Refactoring in Version 6.11 eingeführt wurden. Dies ist eine optionale Einstellung, die eine Verzögerung (in Millisekunden) nach jeder Schleifeniteration in den Vorbereitungsskripten ermöglicht. Das Einstellen einer Verzögerung sollte die Last auf der Datenbank verringern, zum Beispiel während der Arbeitszeiten. Diese Property kann entfernt werden, nachdem die Vorbereitungs-Tasks für das Update abgeschlossen sind.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: nein

- **Optional**: ja
- **Beispielwert**: 0
- **Seit**: 6.11.0.0, zur Verwendung in 6.10.5.x

update.6.11.0.0.timezone

- **Modul**: cmas-setup-hibernate
- **Beschreibung**: Hilfs-Property für die Migration des Ticketprotokolls (die neue Art zum Zählen der Protokollgruppen). Seit 6.11.0.0 sind die Gruppen konstant (Zeitraum vom 2 Stunden). Vor 6.11.0.0 waren die Gruppen nicht konstant und hingen von der Zeitzone des Kunden ab. Die Migrationsskripte verwenden einen alten Algorithmus zum Berechnen der Gruppen und benötigen daher die Information über die Zeitzone. Diese Property sollte auf die von den Kunden am häufigsten verwendete Zeitzone gesetzt werden. Wenn die Property nicht gesetzt ist, wird die Standardzeitzone des Servers verwendet (`TimeZone.getDefault()`). Die Property sollte vor dem Update auf die 6.11.0.0 gesetzt werden und wird nach der Migration automatisch gelöscht. Eine Liste der zulässigen Zeitzonen finden Sie zum Beispiel hier: http://joda-time.sourceforge.net/timezones.html.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: Europe/Berlin
- **Seit**: 6.11.0.0, verwendet vor einem Update auf diese Version

H.2.3.12 cmas-setup-manager (Modul)

initialized

- **Modul**: cmas-setup-manager
- **Beschreibung**: Legt fest, ob CMAS initialisiert ist. Wenn dieser Wert fehlt oder nicht auf „true“ gesetzt ist, wird das Setup ausgeführt. Ab ConSol CM-Version 6.11 ist diese Property nur im Expertenmodus verfügbar.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: true
- **Seit**: 6.0

> ⚠ Seien Sie mit der Verwendung dieser System-Property sehr vorsichtig! Wenn Sie den Wert auf *false* setzen, wird der ConSol CM-Server beim nächsten Systemstart das System-Setup ausführen, d. h. alle Daten des bestehenden Systems gehen verloren, einschließlich der System-Properties!

H.2.3.13 cmas-setup-scene (Modul)

scene

- **Modul**: cmas-setup-scene
- **Beschreibung**: Szenariodatei, die während des Setups importiert wurde (kann leer gelassen werden).
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: vfszip:/P:/dist/target/jboss/server/cmas/deploy/cm-dist-6.5.1-SNAPSHOT.ear/APP-INF/lib/dist-scene-6.5.1-SNAPSHOT.jar/META-INF/cmas/scenes/helpdesk-sales_scene.jar/
- **Seit**: 6.0

H.2.3.14 cmas-workflow-engine (Modul)

jobExecutor.adminMail

- **Modul**: cmas-workflow-engine
- **Beschreibung**: E-Mail-Adresse, an die Benachrichtigungs-E-Mails, die Probleme der Jobausführung betreffen, geschickt werden (wenn die Anzahl der Neuversuche überschritten wurde).
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: admin@consol.de
- **Seit**: 6.8.0

jobExecutor.idleInterval.seconds

- **Modul**: cmas-workflow-engine
- **Beschreibung**: Legt fest, wie oft der Job Executor Thread nach neuen Jobs zum Ausführen sucht.
- **Typ**: integer

- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 45 (Standardwert bis CM-Version 6.10.5.2. Der Standardwert für CM-Versionen 6.10.5.3 und höher ist 5)
- **Seit**: 6.8.0

jobExecutor.jobMaxRetries

- **Modul**: cmas-workflow-engine
- **Beschreibung**: Steuert die Anzahl der erneuten Versuche, die der Job Executor unternimmt, bevor er einen Job als fehlgeschlagen deklariert.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 5 (Standardwert)
- **Seit**: 6.8.0

jobExecutor.jobMaxRetriesReachedSubject

- **Modul**: cmas-workflow-engine
- **Beschreibung**: Der Betreff für Benachrichtigungs-E-Mails, die Administratoren über fehlgeschlagene Job-Ausführungen erhalten.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: Maximale Anzahl der Neuversuche für Job erreicht. Job wurde entfernt! (Standard)
- **Seit**: 6.8.0

jobExecutor.lockingLimit

- **Modul**: cmas-workflow-engine
- **Beschreibung**: Anzahl der gleichzeitig gelockten (als "in der Ausführung" markierten) Jobs des Job Executor Threads.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja

- **Optional**: ja
- **Beispielwert**: 5 (Standardwert seit CM-Version 6.10.5.3)
- **Seit**: 6.8.0

jobExecutor.lockTimeout.seconds

- **Modul**: cmas-workflow-engine
- **Beschreibung**: Legt fest, wie lange ein Job vom Job Executor (als "in der Ausführung" markiert) gelockt werden kann.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 360 (Standardwert)
- **Seit**: 6.8.0

jobExecutor.mailFrom

- **Modul**: cmas-workflow-engine
- **Beschreibung**: E-Mail-Adresse, die als From-Header für Admin-Benachrichtigungen eingesetzt wird.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: jobexecutor@consol.de
- **Seit**: 6.8.0

jobExecutor.maxInactivityInterval.minutes

- **Modul**: cmas-workflow-engine
- **Beschreibung**: Länge der erlaubten Inaktivität des Job Executors in Minuten (z. B. wenn er durch eine Langzeit-Ausführung gesperrt wird). Nach dieser Zeit werden die Executor Threads neu gestartet.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja Standardwert ist auf 30 Minuten gesetzt.
- **Beispielwert**: 15 (Standardwert)
- **Seit**: 6.9.2.0

jobExecutor.threads

- **Modul**: cmas-workflow-engine
- **Beschreibung**: Anzahl der Job Executor Threads.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 1 (Standardwert)
- **Seit**: 6.8.0

jobExecutor.timerRetryInterval.seconds

- **Modul**: cmas-workflow-engine
- **Beschreibung**: Legt fest, wie lange der Job Executor Thread nach einem Fehler bei der Job-Ausführung wartet.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 10 (Standardwert bis CM-Version 6.10.5.2. Der Standardwert für CM-Versionen 6.10.5.3 und höher ist 30.)
- **Seit**: 6.8.0

jobExecutor.txTimeout.seconds

- **Modul**: cmas-workflow-engine
- **Beschreibung**: Transaktions-Timeout für die Job-Ausführung.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 60 (Standardwert)
- **Seit**: 6.8.0

H.2.3.15 cmas-workflow-jbpm (Modul)

fetchLock.interval

- **Modul**: cmas-workflow-jbpm
- **Beschreibung**:

- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 5000
- **Entfernt in**: 6.8.0

jobExecutor.idleInterval

- **Modul**: cmas-workflow-jbpm
- **Beschreibung**:
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 45000
- **Entfernt in**: 6.8.0
- **Ersetzt durch**: jobExecutor.idleInterval.seconds

jobExecutor.jobExecuteRetryNumber

- **Modul**: cmas-workflow-jbpm
- **Beschreibung**:
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 5
- **Entfernt in**: 6.8.0
- **Ersetzt durch**: jobExecutor.jobMaxRetries

jobExecutor.timerRetryInterval

- **Modul**: cmas-workflow-jbpm
- **Beschreibung**:
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein

- **Beispielwert**: 10000
- **Entfernt in**: 6.8.0
- **Ersetzt durch**: jobExecutor.timerRetryInterval.seconds

mail.sender.address

- **Modul**: cmas-workflow-jbpm
- **Beschreibung**: From-Adresse für E-Mails, die von der Workflow-Engine versendet werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: myuser@consol.de
- **Entfernt in**: 6.8.0
- **Ersetzt durch**: jobExecutor.mailFrom

outdated.lock.age

- **Modul**: cmas-workflow-jbpm
- **Beschreibung**:
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 60000
- **Entfernt in**: 6.8.0
- **Ersetzt durch**: cmas-workflow-engine, jobExecutor.lockTimeout.seconds

refreshTimeInCaseOfConcurrentRememberMeRequests

- **Modul**: cmas-workflow-jbpm
- **Beschreibung**: Legt die Aktualisierungszeit (in Sekunden) fest, nach der die Seite im Falle von gleichzeitigen Anfragen von Angemeldet bleiben neu geladen wird. Diese Funktion verhindert, dass ein Benutzer zu viele Lizenzen in Anspruch nimmt. Erhöhen Sie die Zeit, wenn immer noch Sessions belegt werden.
- **Typ**: integer
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: ja

- **Beispielwert**: 5
- **Seit**: 6.8.2

H.2.3.16 cmweb-server-http-headers (Modul)

X-Frame-Options

- **Modul**: cmweb-server-http-headers
- **Beschreibung**: Beispiel-Property zum Verdeutlichen der Konfiguration von HTTP-Headern. In diesem Fall enthält der gesendete HTTP-Header das Feld *X-Frame-Options* mit dem Wert „SAMEORIGIN“.

  Jede Property im Modul `cmweb-server-http-headers` stellt ein Header-Feld dar. Der Name/Schlüssel der Property identifiziert das Header-Feld und der Wert der Property ist der im Header gesendete Feldwert.

  > ⚠ Beachten Sie, dass zusätzliche HTTP-Antwort-Header korrekt und mit der offiziell vorgegebenen Schreibweise definiert werden müssen! Beachten Sie weiterhin, dass die korrekte Interpretation und Verarbeitung dieser Header komplett in Zuständigkeitsbereich des Client-Browsers liegt, der die Seite angefordert hat!

- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: SAMEORIGIN
- **Seit**: 6.10.8

H.2.3.17 cmweb-server-adapter (Modul)

attachment.upload.timeout

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Definiert das Transaktions-Timeout in Minuten für das Hinzufügen von Attachments zu einem Ticket, einer Ressource oder einem Kunden. Dabei zählt die Zeit für den Upload aller Attachments einer Transaktion. Wenn das Timeout eintritt, werden alle Dateien, die temporär auf dem Server gespeichert waren, gelöscht. Es wird keine Datei hochgeladen.
- **Typ**: Integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 3
- **Seit**: 6.10.5.3

automatic.booking.enabled

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Wenn aktiviert, wird die Zeit für das Erstellen eines Kommentars oder einer E-Mail gemessen und eine automatische Zeitbuchung hinzugefügt.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.9.4.2

checkUserOnlineIntervalInSeconds

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Das Intervall (in Sekunden), in dem geprüft wird, welche Benutzer online sind (Standard 180 Sekunden = 3 Minuten).
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 180
- **Seit**: 6.0
- **Entfernt in**: 6.5 / 6.11.0.1

cmoffice.enabled

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Legt fest, ob CM/Doc (vorher CM/Office) aktiviert ist.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.4.0

cmoffice.oo.path.NUMBER

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Möglicher Speicherort der OpenOffice-Installation. Die Properties werden, beginnend mit 0, durchnummeriert.

- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: c:\Program Files (x86)\LibreOffice 3.6\program
- **Seit**: 6.10.1.0

cmoffice.strict.versioning.enabled

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Steuert, ob der Speichervorgang für Microsoft Word- / OpenOffice-Dokumente ein neues Attachment erzeugt (*true*) oder das vorhandene Attachment überschreibt (*false*). Dies betrifft das Verhalten innerhalb einer Session mit dem Textbearbeitungsprogramm. Wenn das Programm beendet wird, funktioniert der Mechanismus zum Überschreiben nicht mehr.
- **Typ**: Boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.10.5.4

commentRequiredForTicketCreation

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Legt fest, ob der Kommentar ein Pflichtfeld für die Erstellung eines Tickets ist.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: true (Standardwert)
- **Seit**: 6.2.0

csrf.domain.white.list

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Die Liste der Domänen (durch "|" getrennt), die zugelassen sind und nicht vom CSRF (Cross-site request forgery)-Filter geprüft werden, z. B. "example.com | consol.de"\
- **Typ**: String
- **Neustart erforderlich**: nein
- **System**: nein

- **Optional**: ja
- **Beispielwert**: example.com | consol.de
- **Seit**: 6.10.7.0 und 6.11.0.5

csrf.request.filter.enabled

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Ermöglicht das Aktivieren/Deaktivieren des CSRF (Cross-site request forgery)-Anfragenfilters
- **Typ**: Boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.10.7.0 und 6.11.0.5

customizationVersion

- **Modul**: cmweb-server-adapter
- **Beschreibung**: UID der letzten Version der Web-Anpassung. Wird nur intern verwendet. Der Wert darf nicht geändert werden.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: cd58453e-f3cc-4538-8030-d15e8796a4a7
- **Seit**: 6.5.0

data.optimization

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Definiert die Optimierung der Response-Daten. Bisher werden die folgenden Werte unterstützt (mehrere Werte können durch '|' getrennt gesetzt werden): MINIFICATION und COMPRESSION. MINIFICATION minimiert HTML-Daten, indem z. B. Leerzeichen und Kommentare entfernt werden. COMPRESSION wendet gzip-Komprimierung auf die HTTP-Response an. (Hinweis: Wenn das System im Cluster-Modus läuft und Sie parallel verschiedene Konfigurationen testen möchten, können Sie für jeden Cluster-Node verschiedene Werte setzen, indem Sie die System-Property nach dem Muster `data.optimization.nodeId` spezifizieren, um die Standard-Property zu überschreiben.)
- **Typ**: string
- **Neustart erforderlich**: COMPRESSION kann ohne Neustart ein- und ausgeschaltet werden, für MINIFICATION ist ein Neustart erforderlich.

- **System**: ja
- **Optional**: ja
- **Beispielwert**: MINIFICATION|COMPRESSION

defaultAttachmentEntryClassName

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Die Standardtextklasse für Attachments, wenn keine andere Klasse explizit gesetzt wurde.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: DefaultTextElement
- **Seit**: 6.9.2.0

defaultContentEntryClassName

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Standard-Textklasse für neue ACIMs.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: default_class
- **Seit**: 6.3.0

defaultNumberOfCustomFieldsColumns

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Standardanzahl an Spalten für Ticketfelder.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 3
- **Seit**: 6.2.0

diffTrackingEnabled

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Entfernt in ConSol CM-Version 6.11.
  Legt fest, ob das gleichzeitige Bearbeiten eines Tickets durch verschiedene Bearbeiter möglich ist. Der Standardwert ist *true*.
  false: Vorherige Art und Weise der Behandlung von Änderungen beim Editieren von Tickets. Wenn ein Ticket zwischendurch geändert wurde, kann der aktuelle Bearbeiter seine Änderungen nicht speichern, ohne die Seite vorher neu zu laden.
  true: Neuer Modus zur Behandlung von Änderungen. Wenn das Ticket geändert wurde, wird das Speichern von anderen Änderungen nicht mehr blockiert. Wenn der Teil des Tickets, der geändert wurde, genau der Teil ist, der vom speichernden Bearbeiter geändert wird, wird eine Informationsmeldung angezeigt, die Ticketänderungen werden aber trotzdem gespeichert.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: true
- **Seit**: 6.10.1
- **Entfernt in**: 6.11.0

diffTrackingEnabledForUnitAndResource

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Aktiviert die Verhinderung von gleichzeitigen Änderungen an Units/Ressourcen.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: 3
- **Seit**: 6.11.0.0

favoritesSizeLimit

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Maximale Anzahl von Elementen in der Favoritenliste.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein

- **Beispielwert**: 10
- **Seit**: 6.0

forward.mails.to.representatives

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Legt fest, ob E-Mails, die manuell aus dem Web Client gesendet werden, auch an die Vertreter eines Bearbeiters gesendet werden. Der Standardwert dieser Property ist „false“, d. h. diese Art von E-Mails wird nicht an den Vertreter weitergeleitet. Setzen Sie diese Property auf „true“, wenn Sie das bisherige Verhalten wiederherstellen wollen, d. h. alle E-Mails, die an den vertretenen Bearbeiter gesendet werden, werden automatisch an den Vertreter weitergeleitet. Berücksichtigen Sie, dass dies unter Umständen nicht erwünscht ist, wenn dieselbe Person gleichzeitig Bearbeiter und Kunde im CM-System ist.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: nein
- **Beispielwert**: false (Standardwert)
- **Seit**: 6.11.1.7



Diese Property konfiguriert nur die Verarbeitung von manuell gesendeten E-Mails. Die Verarbeitung von automatisch gesendeten E-Mails hängt von der verwendeten Java-Methode ab.

globalSearchResultSizeLimit

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Maximale Anzahl an Ergebnissen in der Schnellsuche.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 10
- **Seit**: 6.0

helpFilePath

- **Modul**: cmweb-server-adapter
- **Beschreibung**: URL für die Onlinehilfe. Wenn der Wert nicht leer gelassen wird, wird der Hilfe-Button im Web Client angezeigt.
- **Typ**: string

- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: http://www.consol.de
- **Seit**: 6.2.1

hideTicketSubject

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Wenn auf *true* gesetzt, wird das Ticketthema ausgeblendet.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.2.1

mail.from

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Diese E-Mail-Adresse wird anstelle der E-Mail-Adresse des Bearbeiters in E-Mail-Konversationen verwendet.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.1.2

mail.reply.to

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Wenn dieser Wert gesetzt ist, zeigt der Web Client diesen Wert beim Versenden einer E-Mail im Reply-To-Feld an.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.0.1

> ⚠ Bitte lesen Sie dazu die ausführlichen Informationen über Reply-To-Adressen in ConSol CM im Abschnitt Skripte des Typs E-Mail.

mailTemplateAboveQuotedText

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Legt das Verhalten der E-Mail-Vorlagen im Ticket-E-Mail-Editor fest, wenn eine andere E-Mail zitiert wird, d. h. auf diese geantwortet oder diese weitergeleitet wird. Wird oft verwendet, um die Signatur korrekt zu platzieren.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: false
- **Seit**: 6.2.4

maxSizePerPagemapInMegaBytes

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Mit diesem Parameter wird die Größe (in MB) der Datei definiert, die vom Wicket-Framework pro Benutzer-Session, d. h. für jeden aktuell angemeldeten Bearbeiter, erstellt wird. Die Datei wird verwendet, um Seiten während der laufenden Session zu speichern. Wenn die definierte Größenbegrenzung erreicht wurde und neue Einträge hinzugefügt wurden, werden die ältesten Einträge entfernt. Aufgrund dieses Verhaltens wird ein Bearbeiter, der im Web Client mit einer "alten" Seite arbeitet, auf die Hauptseite (normalerweise die Dashboard-Seite) weitergeleitet, sobald die "alte" Seite aus der Datei entfernt wird.
  Wenn Bearbeiter in ConSol CM mit einer großen Anzahl an offenen Tabs arbeiten und sich darüber beschweren, dass Sie auf die Hauptseite weitergeleitet werden, kann es nützlich sein, diesen Parameter zu erhöhen. In großen Systemen können Sie zum Beispiel einen Wert von 45 oder 50 verwenden. Da dies die Größe der auf der Festplatte gespeicherten Datei ist, hängt der Maximalwert vom verfügbaren Speicherplatz ab, daher ist ein zu großer Wert nicht empfehlenswert.
- **Typ**: integer
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 15
- **Seit**: 6.3.5

pagemapLockDurationInSeconds

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Anzahl der Sekunden, die vergehen müssen, bevor eine Pagemap als zu lange gelockt angesehen wird.
- **Typ**: integer
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: ja
- **Beispielwert**: 60
- **Seit**: 6.7.3

postActivityExecutionScriptName

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Definiert den Namen des Skripts, das nach jeder Workflow-Aktivität ausgeführt wird (siehe Abschnitt PostActivityExecutionScript. Wenn kein Skript ausgeführt werden soll, lassen Sie diesen Wert leer.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: postActivityExecutionHandler
- **Seit**: 6.2.0

queuesExcludedFromGS

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Durch Kommas getrennte Liste von Queue-Namen, die von der Schnellsuche ausgeschlossen werden sollen.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.0

rememberMeLifetimeInMinutes

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Lebensdauer für *Angemeldet bleiben* in Minuten.
- **Typ**: integer

- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 1440
- **Seit**: 6.0

request.scope.transaction

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Ermöglicht die Deaktivierung von Scope-Transaktionen für Requests. Standardmäßig wird pro Request eine Transaktion verwendet. Wenn Sie diese Property auf „false“ setzen, wird eine Transaktion pro Aufruf einer Service-Methode erzeugt.
- **Typ**: boolean
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.8.1

searchPageSize

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Standardgröße der Seiten für Suchergebnisse.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 20
- **Seit**: 6.0

searchPageSizeOptions

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Optionen für die Größe der Seiten für Suchergebnisse.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 10|20|30|40|50|75|100
- **Seit**: 6.0

serverPoolingInterval

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Definiert die Zeit in Sekunden, nach der der Pooling-Server die Caches auf dem Web-Layer ungültig macht.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 5
- **Seit**: 6.1.0

supportEmail

- **Modul**: cmweb-server-adapter
- **Beschreibung**:
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Seit**: 6.0
- **Entfernt in**: 6.11.0.1

themeOverlay

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Name des verwendeten Themen-Overlays.
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: consolINT
- **Seit**: 6.0

ticketListRefreshIntervalInSeconds

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Aktualisierungsintervall für die Ticketliste (in Sekunden).
- **Typ**: integer
- **Neustart erforderlich**: nein

- **System**: ja
- **Optional**: nein
- **Beispielwert**: 180
- **Seit**: 6.0

ticketListSizeLimit

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Maximale Anzahl von Tickets in der Ticketliste.
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 100
- **Seit**: 6.0

tx.read.only.mode.enabled

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Aktiviert nur-lesende Transaktionen für ein schnelleres Laden der Seite. Dieses Transaktionsverhalten wurde in der 6.11.0 eingeführt und die Property ermöglicht es, das alte Verhalten wiederherzustellen. Ändern Sie diesen Wert nur, wenn Sie Transaktionsprobleme haben und von ConSol dazu aufgefordert werden.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: nein
- **Optional**: ja
- **Beispielwert**: true (Standardwert)
- **Seit**: 6.11

unitIndexSearchResultSizeLimit

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Maximale Anzahl der Units in der Ergebnisliste, wenn nach Units gesucht wird (Beispiel: Kontaktsuche).
- **Typ**: integer
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: nein

- **Beispielwert**: 5
- **Seit**: 6.0

urlLogoutPath

- **Modul**: cmweb-server-adapter
- **Beschreibung**: URL die verwendet wird, wenn sich der Bearbeiter abmeldet. (wenn kein Wert gesetzt wird, wird nach dem Abmelden die Anmeldeseite angezeigt.)
- **Typ**: string
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: http://intranet.consol.de
- **Seit**: 6.3.1

voCacheEnabled

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Diese Property aktiviert zusätzliches Caching, voCaching, von kompletten Objekten für den Web Client. Dies verbessert die Performance.
- **Typ**: boolean
- **Neustart erforderlich**: nein
- **System**: ja
- **Optional**: ja
- **Beispielwert**: true
- **Seit**: 6.11.1.0

Hinweise:

- Seit ConSol CM-Version 6.11.1.1 ist der Standardwert „true“ in Umgebungen, die nicht zu einem Cluster gehören. Der Wert wird bei einer Neuinstallation oder einer Aktualisierung auf ConSol CM 6.11.1.1 automatisch auf „true“ gesetzt.
- Wenn voCaching aktiviert ist und lazy loading verwendet wird, um die Einträge des Ticketprotokolls einzuklappen, kann ein Bearbeiter diese Einträge nach dem Aufklappen nicht mehr einklappen, indem er die Seite neu lädt oder das Ticket aus dem Workspace öffnet.
- Wenn der dynamische Modus zum Anzeigen der Bearbeiter- und Kundennamen im Ticketprotokoll verwendet wird (konfiguriert in den System-Properties cmas-core-server, engineer.description.mode und cmas-core-server, unit.description.mode), wird die neue Version des Bearbeiter- bzw. Kundennamens erst angezeigt, nachdem das Ticket geändert wurde.
- Diese System-Property wird in Cluster-Umgebungen (Umgebungen, in denen cmas-core-shared, cluster.mode auf „true“ gesetzt ist) ignoriert. In Cluster-Umgebungen ist voCaching immer deaktiviert, um zu vermeiden, dass Änderungen an Objekten auf einem Node auf den anderen Nodes nicht sichtbar sind.

webSessionTimeoutInMinutes

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Session-Timeout in Minuten.
- **Typ**: integer
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: nein
- **Beispielwert**: 180
- **Entfernt in**: 6.7.1
- **Ersetzt durch**: cmas-core-server, server.session.timeout

wicketAjaxRequestHeaderFilterEnabled

- **Modul**: cmweb-server-adapter
- **Beschreibung**: Aktiviert Filter für Wicket AJAX-Anfragen, die auf veralteten Seiten mit Wicket 1.4-Skripten (CM-Version vor 6.8.0) stammen, nach einer Aktualisierung auf CM-Versionen ab 6.8.0.
- **Typ**: boolean
- **Neustart erforderlich**: ja
- **System**: ja
- **Optional**: ja

- **Beispielwert**: false
- **Seit**: 6.8.1

H.3 Administrator-E-Mail-Adressen

In diesem Kapitel werden folgende Themen behandelt:

H.3.1 Einleitung 1402
H.3.2 Standardkonfiguration 1402
H.3.3 Spezifische Benachrichtigungs-E-Mail-Adressen für einzelne Untersysteme 1404

H.3.1 Einleitung

In ConSol CM können mehrere Administrator-E-Mail-Adressen (oder Benachrichtigungs-E-Mail-Adressen) konfiguriert werden. Hier finden Sie eine Übersicht über alle diese Adressen.

H.3.2 Standardkonfiguration

Wenn Sie ein ConSol CM-System aufsetzen, müssen Sie eine globale Administrator-E-Mail-Adresse eingeben.

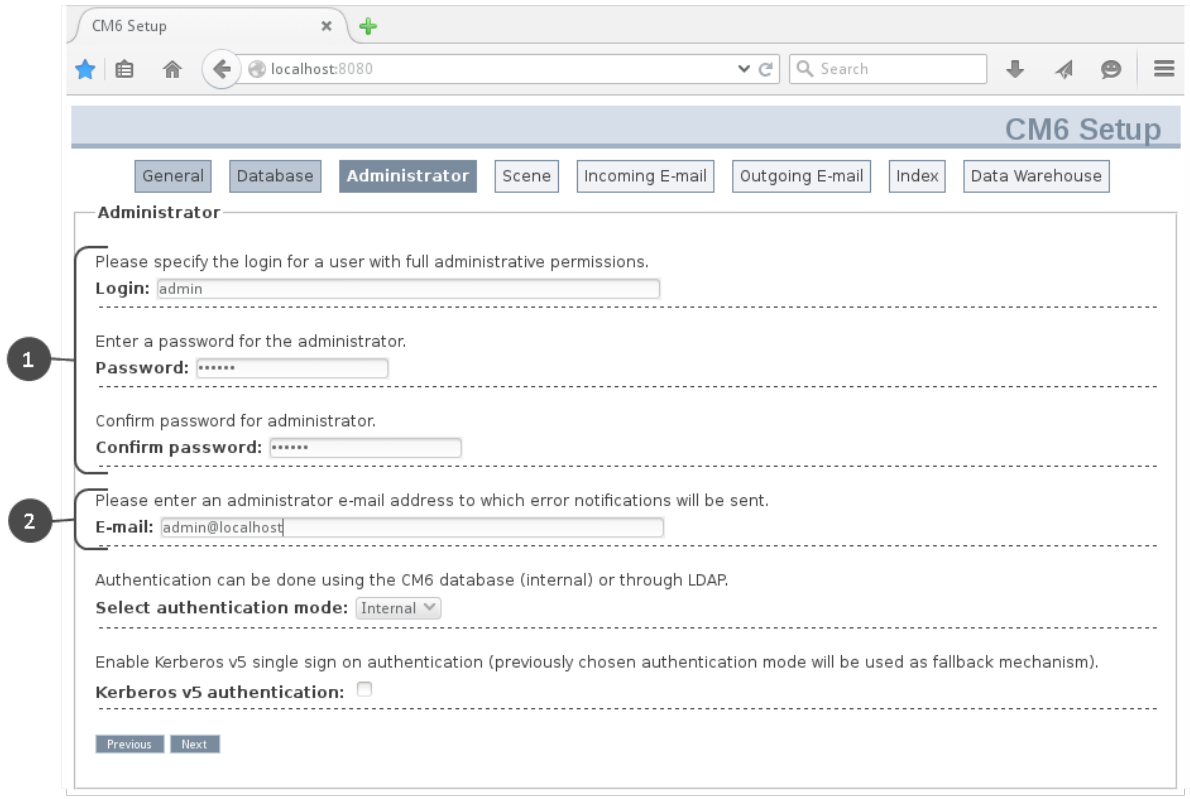


Abbildung 712: *Konfiguration der Administrator-E-Mail-Adresse bei der Systemeinrichtung*

- Login für den CM-Administratorbenutzer (1)
- Globale E-Mail-Adresse des CM-Administrator (2)
  Diese globale E-Mail-Adresse (System-Property cmas-core-security, admin.email) wird für alle Benachrichtigungen verwendet und automatisch auch für alle spezifischen E-Mail-Adressen der Untersysteme eingetragen. Das bedeutet, dass diese E-Mail-Adresse initial automatisch für alle System-Properties gesetzt wird, die Administrator- oder Benachrichtigungs-E-Mail-Adressen enthalten. Die Properties können über die GUI des Admin Tools geändert werden, um das jeweilige Untersystem zu konfigurieren. Sie können zum Beispiel im Abschnitt *DWH-Konfiguration* des Admin Tools eine spezielle Benachrichtigungs-E-Mail-Adresse für DWH-Operationen konfigurieren.

Durch die Konfiguration von unterschiedlichen Administrator-/Benachrichtigungs-E-Mail-Adressen für unterschiedliche Untersysteme können Sie die Verantwortlichkeiten entsprechend den Zuständigkeiten und Rollen innerhalb Ihres Unternehmens verteilen. Für einige Benachrichtigungen können die From-Adresse, der Text und der Betreff konfiguriert werden.

H.3.2.1 Sonderfälle, in denen die Administrator-E-Mail-Adresse verwendet wird

Queue-spezifische E-Mails beim Zuweisen/Entfernen eines Bearbeiters

Die E-Mails, die gesendet werden, wenn für eine Queue Templates für die Ticketzuweisung gesetzt sind (siehe Abschnitt *Queue-Verwaltung*), verwenden die Adresse `admin.email` als:

- From-Adresse
- ReplyTo-Adresse

Die queue-spezifischen E-Mail-Skripte werden nicht für diese automatischen E-Mails verwendet. Es ist daher nicht möglich, die From- und ReplyTo-Adresse für die E-Mails beim Zuweisen/Entfernen eines Bearbeiters in diesem Skript zu setzen.



Beachten Sie Folgendes: Wenn die CM-System-Property cmas-core-server, mail.notification.sender gesetzt ist, wird der Wert dieser Property als From-Adresse und als ReplyTo-Adresse für E-Mails beim Zuweisen/Entfernen des Bearbeiters verwendet.

H.3.3 Spezifische Benachrichtigungs-E-Mail-Adressen für einzelne Untersysteme

H.3.3.1 DWH (Data Warehouse) - Spezifische Benachrichtigungs-E-Mail-Adressen und E-Mail-Konfigurationen

System-Properties

- **Fehler**
  - cmas-dwh-server, notification.error.to
    Wenn eine DWH-Operation fehlgeschlagen ist, wird eine E-Mail an diese Adresse gesendet. Wenn die Property nicht gesetzt ist, wird keine E-Mail gesendet.
  - cmas-dwh-server, notification.error.from
    Wenn eine DWH-Operation fehlgeschlagen ist, wird eine E-Mail von dieser From-Adresse gesendet.
  - cmas-dwh-server, notification.error.subject
    Betreff für E-Mails mit Fehlermeldungen des DWH.
  - cmas-dwh-server, notification.error.description
    Text für E-Mails mit Fehlermeldungen des DWH.
- **Erfolgreich**
  - cmas-dwh-server, notification.finished_successfully.to
    Es wird eine E-Mail an diese E-Mail-Adresse gesendet, wenn eine DWH-Übertragung erfolgreich abgeschlossen wurde, z. B. wenn eine Übertragung ohne Fehler abgeschlossen wurde. Wenn die Property nicht gesetzt ist, wird keine E-Mail gesendet.
  - cmas-dwh-server, notification.finished_successfully.from
    From-Adresse für E-Mails des DWHs, wenn eine Übertragung erfolgreich beendet wurde.
  - cmas-dwh-server, notification.finished_successfully.subject
    Betreff für E-Mails des DWHs, wenn eine Übertragung erfolgreich beendet wurde.
  - cmas-dwh-server, notification.finished_successfully.description
    Text für E-Mails des DWHs, wenn eine Übertragung erfolgreich beendet wurde.
- **Fehlgeschlagen**
  - cmas-dwh-server, notification.finished_unsuccessfully.to
    Es wird eine E-Mail an diese E-Mail-Adresse gesendet, wenn eine DWH-Übertragung abgeschlossen wurde, aber nicht erfolgreich war, z. B. wenn eine Übertragung mit Fehlern abgeschlossen wurde. Wenn die Property nicht gesetzt ist, wird keine E-Mail gesendet.
  - cmas-dwh-server, notification.finished_unsuccessfully.from
    From-Adresse für E-Mails des DWHs, wenn eine Übertragung nicht erfolgreich beendet wurde.
  - cmas-dwh-server, notification.finished_unsuccessfully.subject
    Betreff für E-Mails des DWHs, wenn eine Übertragung nicht erfolgreich beendet wurde.
  - cmas-dwh-server, notification.finished_unsuccessfully.description

Text für E-Mails des DWHs, wenn eine Übertragung nicht erfolgreich beendet wurde.

- cmas-dwh-server, cmas-dwh-server, notification.error.description
  Text für E-Mails mit Fehlermeldungen des DWH.

Eine Übersicht mit allen System-Properties, die für DWH-Benachrichtigungen gesetzt werden können, finden Sie im Abschnitt CMRF- und DWH-Konfiguration des Kapitels Liste der System-Properties nach Bereich. Alle Properties, die in diesem Kontext relevant sind, beginnen mit **notification**.

Grafische Konfiguration

Sie gelangen auf diese Seite, indem Sie das Navigationselement *Administration* der Navigationsgruppe *Data Warehouse* öffnen und auf das Werkzeugsymbol klicken.

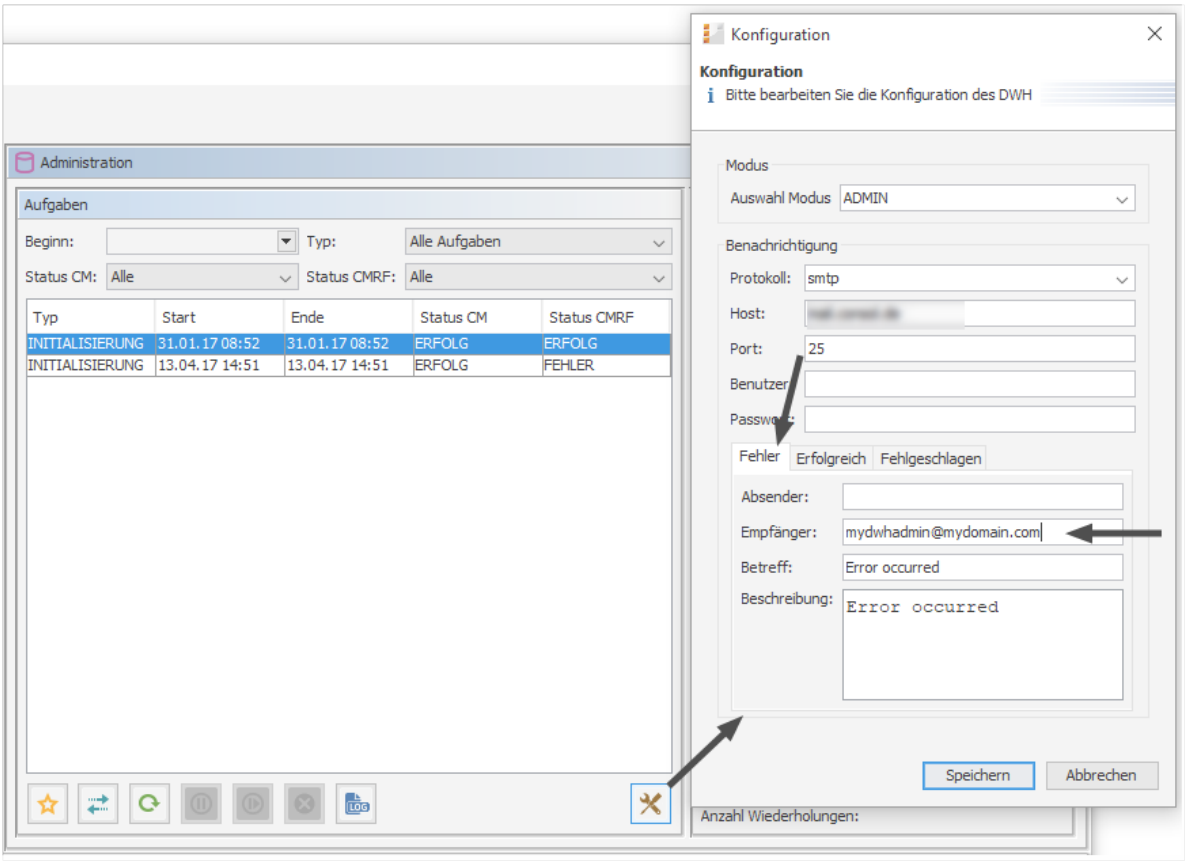


Abbildung 713: *ConSol CM Admin Tool - Data Warehouse, Administration: Benachrichtigungs-E-Mail-Adresse für DWH-Fehler*

Die E-Mail-Adresse wird überprüft, wenn Sie auf *Speichern* klicken. Wenn die E-Mail-Adresse nicht gültig ist, wird eine Meldung angezeigt. Es werden keine E-Mails an diese E-Mail-Adresse gesendet.

H.3.3.2 E-Mail-Spezifische Benachrichtigungs-E-Mail-Adressen

- cmas-nimh-extension, mail.error.to.address
  Im Fall, dass eine E-Mail nicht verarbeitet werden konnte, wird eine Fehler-E-Mail an die E-Mail-

Adresse gesendet. Ab CM-Version 6.10.5.1 wird auch bei E-Mail-Timeouts eine E-Mail an diese Adresse gesendet.
Wenn die Property nicht gesetzt ist, wird keine E-Mail gesendet und es wird eine Exception in die Log-Datei geschrieben.

- cmas-nimh-extension, mail.attachments.validation.info.sender
  Setzt den From-Header bei Attachments vom Typ *error notification e-mail*.

Beachten Sie, dass das Versenden von der System-Property cmas-nimh-extension, mail.on.error gesteuert wird. Nur wenn diese Property auf „true“ gesetzt ist, wird im Fall, dass eine E-Mail nicht verarbeitet werden konnte, ein Fehler-E-Mail an die oben konfigurierte E-Mail-Adresse gesendet.

Grafische Konfiguration

Der Wert, der hier auf der grafischen Benutzeroberfläche eingegeben wird (*E-Mail-Adresse des Administrators* im Abschnitt *Konfiguration*), wird als allgemeine E-Mail-Adresse für Benachrichtigungen verwendet.

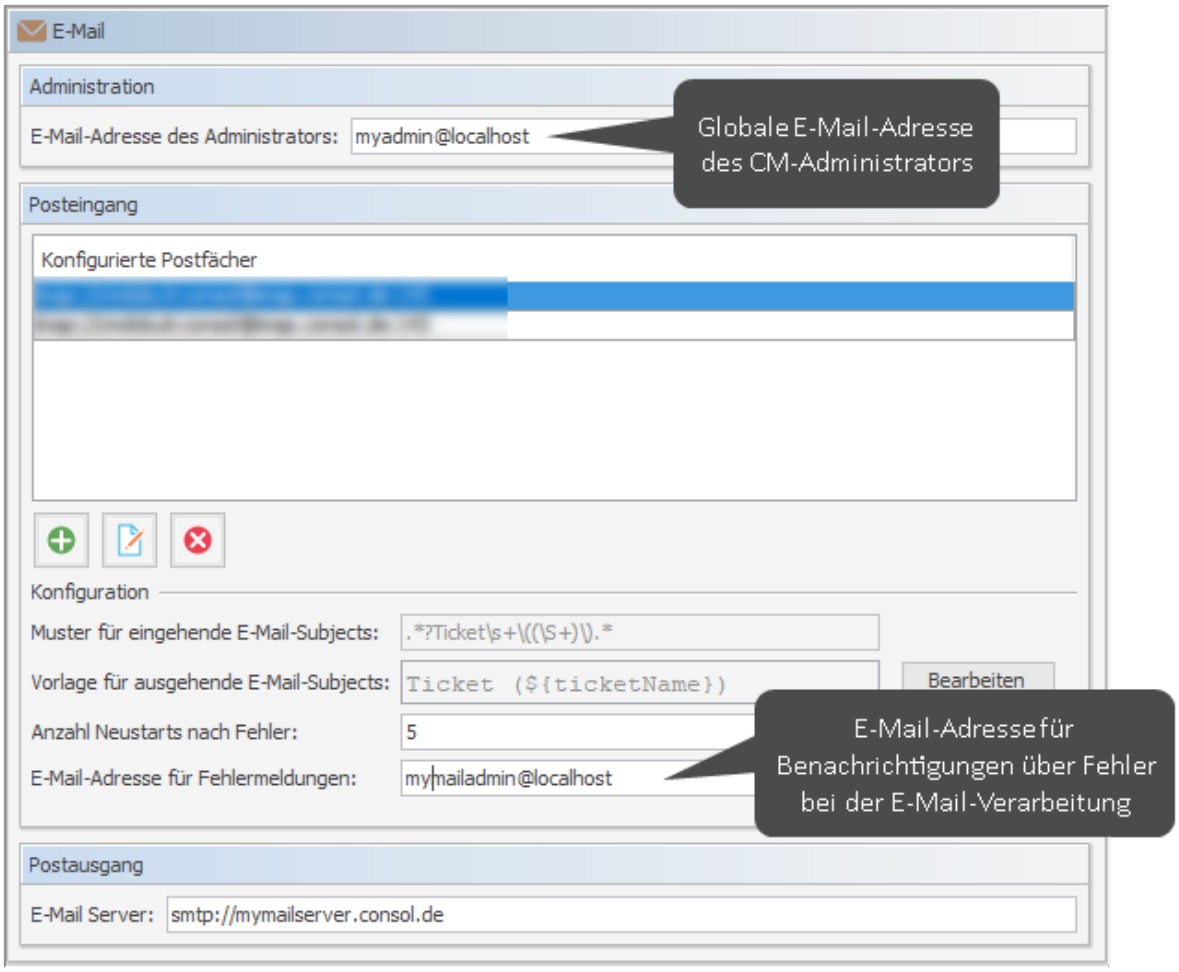


Abbildung 714: *ConSol CM Admin Tool - E-Mail, E-Mail: Konfiguration der globalen Administrator-E-Mail-Adresse und Benachrichtigungs-E-Mail-Adresse für E-Mail-Verarbeitungsprobleme*

Verwechseln Sie die beiden E-Mail-Adressen, die im Tab des Navigationselements *E-Mail* konfiguriert werden können, nicht!

- **E-Mail-Adresse des Administrators**
  ist die globale E-Mail-Adresse des Administrators (bei der Einrichtung angegeben).
- **E-Mail-Adresse für Fehlermeldungen**
  ist die E-Mail-Adresse des E-Mail-Administrators oder einer anderen Person, die die Fehlermeldungen erhalten soll, die auftreten, wenn die Verarbeitung der (eingehenden oder ausgehenden) E-Mails nicht richtig funktioniert.

H.3.3.3 Workflow-Engine - Spezifische Benachrichtigungs-E-Mail-Adressen

- cmas-workflow-engine, jobExecutor.adminMail
  E-Mail-Adresse, an die Benachrichtigungs-E-Mails, die Probleme der Jobausführung betreffen, geschickt werden (wenn die Anzahl der Neuversuche überschritten wurde).
- cmas-workflow-engine, jobExecutor.mailFrom
  E-Mail-Adresse, die als From-Header für Admin-Benachrichtigungen eingesetzt wird.

H.4 Standardimports von Java-Klassen

Damit alle erforderlichen Klassen und Methoden in ConSol CM-Skripten verfügbar sind, müssen Sie unter Umständen Java/Groovy-Klassen und -Pakete importieren. Dies ist für das Admin Tool und den Process Designer relevant. Ab ConSol CM-Version 6.11.1 müssen Sie nur die Klassen und Pakete importieren, die in den meisten Standardfällen nicht verwendet werden. Alle Pakete, die häufig verwendete Klassen enthalten, werden implizit vom Admin Tool und Process Designer importiert. So basiert sowohl die Code-Validierung beim Schreiben der Skripte als auch die Ausführung des Codes zur Laufzeit auf den implizit importierten Skripten, wodurch das Schreiben von CM-Skripten bequemer wird.

Die folgenden Klassen werden implizit ins Admin Tool importiert:

- com.consol.cmas.common.model.*
- com.consol.cmas.common.model.calendar.*
- com.consol.cmas.common.model.configuration.*
- com.consol.cmas.common.model.content.*
- com.consol.cmas.common.model.content.unit.*
- com.consol.cmas.common.model.customer.*
- com.consol.cmas.common.model.customfield.*
- com.consol.cmas.common.model.customfield.cfel.*
- com.consol.cmas.common.model.customfield.enums.*
- com.consol.cmas.common.model.customfield.meta.*
- com.consol.cmas.common.model.event.*
- com.consol.cmas.common.model.event.configuration.*
- com.consol.cmas.common.model.event.content.*
- com.consol.cmas.common.model.event.content.support.*
- com.consol.cmas.common.model.event.customfield.*
- com.consol.cmas.common.model.event.delete.*
- com.consol.cmas.common.model.event.engineer.*
- com.consol.cmas.common.model.event.engineer.support.*
- com.consol.cmas.common.model.event.localization.*
- com.consol.cmas.common.model.event.localization.support.*
- com.consol.cmas.common.model.event.ticket.*
- com.consol.cmas.common.model.event.ticket.support.*
- com.consol.cmas.common.model.event.unit.*
- com.consol.cmas.common.model.event.unit.support.*
- com.consol.cmas.common.model.event.util.*
- com.consol.cmas.common.model.event.workflow.*

- com.consol.cmas.common.model.history.*
- com.consol.cmas.common.model.history.custom.*
- com.consol.cmas.common.model.history.ticket.*
- com.consol.cmas.common.model.inventory.*
- com.consol.cmas.common.model.inventory.history.*
- com.consol.cmas.common.model.inventory.meta.*
- com.consol.cmas.common.model.localization.*
- com.consol.cmas.common.model.mail.*
- com.consol.cmas.common.model.resource.*
- com.consol.cmas.common.model.resource.history.*
- com.consol.cmas.common.model.resource.meta.*
- com.consol.cmas.common.model.scripting.OperationMessage
- com.consol.cmas.common.model.scripting.OperationResponse
- com.consol.cmas.common.model.scripting.OperationType
- com.consol.cmas.common.model.search.*
- com.consol.cmas.common.model.task.*
- com.consol.cmas.common.model.ticket.*
- com.consol.cmas.common.model.ticket.user.*
- com.consol.cmas.common.model.ticket.user.function.*
- com.consol.cmas.common.model.ticket.view.*
- com.consol.cmas.common.security.authentication.UserType
- com.consol.cmas.workflow.common.model.*
- java.io.*
- java.util.*
- javax.activation.DataSource
- com.consol.cmas.common.model.form.*

Die folgenden Klassen werden implizit in den Process Designer importiert:

- com.consol.cmas.common.model.*
- com.consol.cmas.common.model.calendar.*
- com.consol.cmas.common.model.configuration.*
- com.consol.cmas.common.model.content.*
- com.consol.cmas.common.model.content.unit.*
- com.consol.cmas.common.model.customer.*
- com.consol.cmas.common.model.customfield.*

- com.consol.cmas.common.model.customfield.cfel.*
- com.consol.cmas.common.model.customfield.enums.*
- com.consol.cmas.common.model.customfield.meta.*
- com.consol.cmas.common.model.event.*
- com.consol.cmas.common.model.event.configuration.*
- com.consol.cmas.common.model.event.content.*
- com.consol.cmas.common.model.event.content.support.*
- com.consol.cmas.common.model.event.customfield.*
- com.consol.cmas.common.model.event.delete.*
- com.consol.cmas.common.model.event.engineer.*
- com.consol.cmas.common.model.event.engineer.support.*
- com.consol.cmas.common.model.event.localization.*
- com.consol.cmas.common.model.event.localization.support.*
- com.consol.cmas.common.model.event.ticket.*
- com.consol.cmas.common.model.event.ticket.support.*
- com.consol.cmas.common.model.event.unit.*
- com.consol.cmas.common.model.event.unit.support.*
- com.consol.cmas.common.model.event.util.*
- com.consol.cmas.common.model.event.workflow.*
- com.consol.cmas.common.model.history.*
- com.consol.cmas.common.model.history.custom.*
- com.consol.cmas.common.model.history.ticket.*
- com.consol.cmas.common.model.inventory.*
- com.consol.cmas.common.model.inventory.history.*
- com.consol.cmas.common.model.inventory.meta.*
- com.consol.cmas.common.model.localization.*
- com.consol.cmas.common.model.mail.*
- com.consol.cmas.common.model.resource.*
- com.consol.cmas.common.model.resource.history.*
- com.consol.cmas.common.model.resource.meta.*
- com.consol.cmas.common.model.scripting.OperationMessage
- com.consol.cmas.common.model.scripting.OperationResponse
- com.consol.cmas.common.model.scripting.OperationType
- com.consol.cmas.common.model.search.*

- com.consol.cmas.common.model.task.*
- com.consol.cmas.common.model.ticket.*
- com.consol.cmas.common.model.ticket.user.*
- com.consol.cmas.common.model.ticket.user.function.*
- com.consol.cmas.common.model.ticket.view.*
- com.consol.cmas.common.security.authentication.UserType
- com.consol.cmas.workflow.common.model.*
- java.io.*
- java.util.*
- javax.activation.DataSource
- com.consol.cmas.common.model.permission.*
- com.consol.cmas.common.model.scripting.*
- com.consol.cmas.common.model.util.*

H.5 Liste der Code-Beispiele

In diesem Abschnitt finden Sie eine Liste der Code-Beispiele aus diesem Handbuch.

Code-Beispiel 1: Beispielskript für die Konfiguration der Ticketliste (view-SpecificTicketListConfig.groovy) 186
Code-Beispiel 2: JSon-Objekt für das Kundendatenformat in der Vorschau 276
Code-Beispiel 3: Beispielwert für das Layout der Kundendatenvorschau 276
Code-Beispiel 4: ConSol CM-Standardskript für Chart-Widgets ticketsInViewDataWidget.groovy ... 292
Code-Beispiel 5: Sichtbarkeit ausgeschaltet (im Admin-Tool-Skript definiert) 294
Code-Beispiel 6: Sichtbarkeit abhängig von der Bearbeiterfunktion (im Admin-Tool-Skript definiert) 294
Code-Beispiel 7: Chart-Objekt 297
Code-Beispiel 8: Label-Objekt 298
Code-Beispiel 9: Groovy-Skript für die Implementierung eines Vertriebstrichter-Charts auf dem Web Client Dashboard (mit festen Zahlen als Beispiel) 302
Code-Beispiel 10: Admin-Tool-Skript für Tabellen-Widget 304
Code-Beispiel 11: Admin-Tool-Skript für ein KPI-Widget, mit dem die in der letzten Woche geöffneten Tickets berechnet werden 307
Code-Beispiel 12: Layout-Attribut für zusammengesetztes Dashboard 312
Code-Beispiel 13: Admin-Tool-Skript, das mit dem Tabellen-Widget verknüpft ist 314
Code-Beispiel 14: Admin-Tool-Skript, das mit dem Tortendigramm-Widget verknüpft ist 316
Code-Beispiel 15: Anzeigen des Standard-Charts in der 3D-Ansicht 319
Code-Beispiel 16: Drill-down-Funktion für Chart-Widget 322
Code-Beispiel 17: Einfaches graphScript 330
Code-Beispiel 18: Einfaches graphScript mit einfacher Knotenlayoutdefinition 331
Code-Beispiel 19: Skript für customRelationGraph , showGraphicalRelationsOfResource2.groovy .. 335
Code-Beispiel 20: Beispiel für die Datei config.json 352
Code-Beispiel 21: Ausschnitt aus einer Datei localization_en.json 356
Code-Beispiel 22: Ausschnitt aus einer Datei localization_de.json 357
Code-Beispiel 23: Ausschnitt aus localization_en.json mit den Standardwerten für den auf der Startseite angezeigten Text und der für die Queue-Beschreibung erstellten benutzerdefinierten Bezeichnung 360
Code-Beispiel 24: Beispiel einer Datei public.json für die Konfiguration des CM-REST-Clients (dieses Beispiel ist auch im Admin Tool enthalten) 363
Code-Beispiel 25: Ausschnitt aus config.json zum Deaktivieren der Funktion zum Ändern des Passworts 365
Code-Beispiel 26: Ausschnitt aus public.json zum Deaktivieren der Funktion zum Zurücksetzen des Passworts 365

Code-Beispiel 27: Beispiel für den Abruf des Firmennamens eines Kontakts 423
Code-Beispiel 28: Beispiel 1: Suche nach den Tickets eines Kontakts oder einer Firma 425
Code-Beispiel 29: Suche nach den Tickets eines Kontakts, der zu einer bestimmten Firma gehört 425
Code-Beispiel 30: Suche nach Kontakten einer bestimmten Firma 425
Code-Beispiel 31: Beispiel für ein Kunden-Template mit dem Kundenname (muss in eine Zeile geschrieben werden!) 431
Code-Beispiel 32: Beispiel für ein Template mit dem Firmenkontakt (muss in eine Zeile geschrieben werden!) 431
Code-Beispiel 33: Beispiel für ein Kundensuche-Template (muss in einer Zeile geschrieben werden!) 431
Code-Beispiel 34: Einstellen des Zahlenformats: Entfernen von "." in der Anzeige von Zahlen 431
Code-Beispiel 35: REST-Template 434
Code-Beispiel 36: E-Mail-Template (muss in eine Zeile geschrieben werden!) 437
Code-Beispiel 37: Suchergebnis-Template von ResellerCompany (muss in eine Zeile geschrieben werden!) 439
Code-Beispiel 38: Kundenaktionsskript für CM-Version 6.9.4 470
Code-Beispiel 39: Kundenaktionsskript für CM-Version 6.10 471
Code-Beispiel 40: Kundenaktionsskript, CM-Version 6.10 478
Code-Beispiel 41: Kundenskript (CM-Version 6.9.4) 479
Code-Beispiel 42: Kundenskript (CM-Version 6.10) 479
Code-Beispiel 43: Setzen eines Wertes in den Kundendaten und Aktualisieren der Unit 480
Code-Beispiel 44: Firmenskript, mit dem einige Unit-Daten gefüllt werden, CM-Version 6.9.4 481
Code-Beispiel 45: Firmenskript, mit dem einige Unit-Daten gefüllt werden, CM-Version 6.10 482
Code-Beispiel 46: Mit dem Skript wird ein Aktionsergebnis erzeugt und zurückgegeben, das den Web Client anweist, ein neues Ticket mit der Unit als Hauptkontakt zu erstellen, CM-Version 6.9.4 484
Code-Beispiel 47: Mit dem Skript wird ein Aktionsergebnis erzeugt und zurückgegeben, das den Web Client anweist, ein neues Ticket mit der Unit als Hauptkontakt zu erstellen, CM-Version 6.10 485
Code-Beispiel 48: Skript, das die Firmenseite öffnet, CM-Version 6.9.4 487
Code-Beispiel 49: Skript, das die Firmenseite öffnet, CM-Version 6.10 488
Code-Beispiel 50: Kundenbedingungsskript, das überprüft, ob es eine Reseller-Relation gibt 489
Code-Beispiel 51: Öffnen einer Ticketseite im Ansichtsmodus, CM-Version 6.9 491
Code-Beispiel 52: Öffnen einer Ticketseite im Ansichtsmodus, CM-Version 6.10 492
Code-Beispiel 53: Skript, das eine bestimmte Website (URL) öffnet, CM-Version 6.9.4 493
Code-Beispiel 54: Skript, das eine bestimmte Website (URL) öffnet, CM-Version 6.10 493
Code-Beispiel 55: Kundenbedingungsskript 494

Code-Beispiel 56: Unit-Aktualisierungsskript, in dem Änderungen überwacht und in die Datei server.log geschrieben werden ... 496
Code-Beispiel 57: Log-Ausgabe für das obige Skript ... 497
Code-Beispiel 58: Skript der Aktivität „Neues Ticket (Ticket annehmen)“, Auslastung des Bearbeiters wird geprüft ... 521
Code-Beispiel 59: Beispiel für Kalenderintegrationsskript ... 546
Code-Beispiel 60: Admin-Tool-Skript, Beispiel für ein calendarEventHandlerScript ... 555
Code-Beispiel 61: Suche-Aktionsskript für Tickets ... 622
Code-Beispiel 62: Suche-Aktionsskript für Ressourcen ... 624
Code-Beispiel 63: Suche-Aktionsskript für Units ... 628
Code-Beispiel 64: Suche-Bedingungsskript für Units ... 629
Code-Beispiel 65: Positives Feedback ... 635
Code-Beispiel 66: Negatives Feedback ... 636
Code-Beispiel 67: Beispiel für ein Admin-Tool-Skript des Typs Task ... 643
Code-Beispiel 68: Erstellen eines Task Descriptors ... 646
Code-Beispiel 69: Abbrechen eines Tasks ... 647
Code-Beispiel 70: Wiederholen eines Tasks ... 647
Code-Beispiel 71: Planen eines Tasks ... 647
Code-Beispiel 72: Wiederholen eines Tasks nach einem Fehler ... 648
Code-Beispiel 73: Workflow-Aktivitätsskript für Task-Ausführung ... 649
Code-Beispiel 74: Öffnen der Seite Neues Ticket ... 659
Code-Beispiel 75: Öffnen der Ticketseite im Anzeigemodus ... 659
Code-Beispiel 76: Öffnen der Ticketseite im Anzeigemodus, nach Anzeige eines ACFs ... 659
Code-Beispiel 77: Kundenaktionsskript (hier: Firmenskript -> Firma implizit verfügbar), um die Seite Neuer Kunde zu öffnen ... 660
Code-Beispiel 78: Öffnen einer URL ... 660
Code-Beispiel 79: Relationsaktions-Skript, mit dem eine E-Mail gesendet wird, wenn eine Relation zwischen Ressource und Firma erstellt oder gelöscht wird ... 664
Code-Beispiel 80: Aktionsformular Bedingungsskript, das überprüft, ob Felder ausgefüllt sind ... 677
Code-Beispiel 81: Template für Admin-Tool-Skripte des Typs Text-Autovervollständigung ... 712
Code-Beispiel 82: Skript einer Autocomplete-Liste für ein Ressourcenfeld ... 714
Code-Beispiel 83: Skript, das die Werte eines Autocomplete-Ticketfeldes mit Bearbeiterdaten in die Datei server.log schreibt ... 716
Code-Beispiel 84: Skript einer Autocomplete-Liste, die in CM/Track V2 angezeigt werden kann ... 717
Code-Beispiel 85: E-Mail-Skript ... 724

Code-Beispiel 86: Duplizieren-Skript zum Zurücksetzen des Ticketfelds für den gewünschten Termin 730
Code-Beispiel 87: Feldvisualisierungsskript zum Anzeigen eines Bildes auf der Kundenseite 734
Code-Beispiel 88: Beispiel für Datei hightlight.css 736
Code-Beispiel 89: Admin-Tool-Skript des Typs Feld-Visualisierung, zur Hervorhebung von bestimmten Anzeigewerten in einem Ticketfeld verwendet 737
Code-Beispiel 90: Admin-Tool-Skript des Typs Feld-Visualisierung für die Verwendung von statistischen Daten basierend auf der Highcharts-Bibliothek 740
Code-Beispiel 91: Admin-Tool-Skript des Typs Feld-Visualisierung, zur Angabe eines Ortes in OpenStreetMap verwendet 745
Code-Beispiel 92: PostActivityExecutionScript 754
Code-Beispiel 93: PostActivityExecutionHandler 754
Code-Beispiel 94: Template für die Darstellung von Kundendaten 761
Code-Beispiel 95: Text des Beispiel-Templates engineer-assigned-default-mail 763
Code-Beispiel 96: Template zum Zurücksetzen des Passworts eines Bearbeiters 765
Code-Beispiel 97: Zurücksetzen des Passworts eines Kunden in CM/Track 767
Code-Beispiel 98: Aus dem Workflow aufgerufenes Admin-Tool-Skript: alle auf das Ticket gebuchten Zeiten werden als Kommentar ins Ticketprotokoll geschrieben 870
Code-Beispiel 99: JBoss 7 889
Code-Beispiel 100: WildFly 8.2 889
Code-Beispiel 101: cm6-kerberos.properties 890
Code-Beispiel 102: krb5.ini 890
Code-Beispiel 103: Beispiel für das Zurückstufen der Session auf die Berechtigungen eines Bearbeiters 914
Code-Beispiel 104: Admin-Tool-Skript, das einen Webhooks-Service implementiert (mit einigen log.-info-Anweisungen zu Testzwecken) 915
Code-Beispiel 105: GET-Abfrage zum Abrufen von Ressourcen im System 1003
Code-Beispiel 106: GET-Abfrage zum Abrufen der Ressource mit der ID 2 1004
Code-Beispiel 107: Ressourcenaktionsskript 1017
Code-Beispiel 108: Ressourcenaktionsskript für PC_Desktops zur Erstellung eines neuen ServiceDesk-Tickets für den Kontakt, der für den PC zuständig ist 1022
Code-Beispiel 109: Ressourcenaktionsskript, das ein Ticket mit ACF öffnet 1028
Code-Beispiel 110: Ressourcen-Aktualisierungsskript, in dem Änderungen überwacht und in die Datei server.log geschrieben werden 1032
Code-Beispiel 111: Log-Ausgabe für das obige Skript 1033
Code-Beispiel 112: ResourceDashboardOverview1.groovy 1043

H.6 Marken

- Apache OpenOffice™ – Apache und die Apache-Federlogos sind Marken von The Apache Software Foundation. OpenOffice.org und das Mövenlogo sind eingetragene Marken von The Apache Software Foundation. Siehe Website von Apache OpenOffice zu Markenrichtlinien.
- Google Maps™ – Google Maps ist eine Marke von Google Inc. Siehe Google-Website über Marken.
- HAProxy – HAProxy ist Copyright von Willy Tarreau. Siehe HAProxy-Website.
- Microsoft® – Microsoft und Windows sind entweder eingetragene Marken oder Marken der Microsoft Corporation in den USA und/oder anderen Ländern. Siehe Website von Microsoft zu Markenrichtlinien.
- Microsoft® Active Directory® – Microsoft und Microsoft Active Directory sind entweder eingetragene Marken oder Marken der Microsoft Corporation in den USA und/oder anderen Ländern. Siehe Website von Microsoft zu Markenrichtlinien.
- Microsoft® Exchange Server – Microsoft und Microsoft Exchange Server sind entweder eingetragene Marken oder Marken der Microsoft Corporation in den USA und/oder anderen Ländern. Siehe Website von Microsoft zu Markenrichtlinien.
- Microsoft® Office – Microsoft und Microsoft Office sind entweder eingetragene Marken oder Marken der Microsoft Corporation in den USA und/oder anderen Ländern. Siehe Website von Microsoft zu Markenrichtlinien.
- Windows® Betriebssystem – Microsoft und Windows sind entweder eingetragene Marken oder Marken der Microsoft Corporation in den USA und/oder anderen Ländern. Siehe Website von Microsoft zu Markenrichtlinien.
- Microsoft® SQL Server® – Microsoft und Microsoft SQL Server sind entweder eingetragene Marken oder Marken der Microsoft Corporation in den USA und/oder anderen Ländern. Siehe Website von Microsoft zu Markenrichtlinien.
- Microsoft® Word® – Microsoft und Microsoft Word sind entweder eingetragene Marken oder Marken der Microsoft Corporation in den USA und/oder anderen Ländern. Siehe Website von Microsoft zu Markenrichtlinien.
- NGiNX – NGiNX ist Copyright von Igor Sysoev und Nginx, Inc. Siehe NGiNX-Lizenzseite.
- OpenStreetMap - OpenStreetMap® sind frei verfügbare Daten (Open Data), die unter Open Data Commons Open Database License (ODbL) von der OpenStreetMap Foundation (OSMF) lizenziert wurden. Siehe OpenStreetMap Copyright- und Lizenzseite.
- Oracle® – Oracle ist eine eingetragene Marke von Oracle Corporation und/oder ihren verbundenen Unternehmen. Siehe Website von Oracle zu Markenrichtlinien.
- Oracle® WebLogic – Oracle ist eine eingetragene Marke von Oracle Corporation und/oder ihren verbundenen Unternehmen. Siehe Website von Oracle zu Markenrichtlinien.
- Pentaho® – Pentaho und das Pentaho-Logo sind eingetragene Marken von Pentaho Inc. Siehe Website von Pentaho zu Markenrichtlinien.
- Vis.js – Vis.js ist Copyright von Almende B.V. Siehe Vis.js-Lizenzseite.

- Wicket™ - Apache Wicket und Wicket, Apache, das Apache-Federlogo und das Apache Wicket-Projektlogo sind Marken von The Apache Software Foundation. Siehe zum Beispiel den Hinweis unten auf der Wicket-Homepage

Glossar

A

ACF

ACF ist die Abkürzung von Activity Control Form (auf Deutsch Aktivitätsformular). ACFs können in Workflow-Aktivitäten verwendet werden, um den Bearbeiter zu zwingen, bestimmte Datenfelder auszufüllen, bevor er fortfahren kann.

ACIM

Activity Item - Eintrag im Ticketbereich Protokoll eines Tickets (z.B. Kommentar, E-Mail, Attachment, Zeitbuchungseintrag).

AD

Microsoft Active Directory - ein LDAP-basierter Verzeichnisservice für Microsoft Windows-Domänennetzwerke.

Admin Tool

ConSol CM-Kompenente, grafische Applikation, um ein ConSol CM-System zu konfigurieren und zu verwalten. Verwendet Java Web Start.

AJP

Apache JServ Protocol, siehe zum Beispiel https://en.wikipedia.org/wiki/Apache_JServ_Protocol

B

Bearbeiter

Bearbeiter sind die Benutzer, die im Web Client an Tickets arbeiten.

Berechtigung

Mit Berechtigungen wird festgelegt, welche Tickets der Bearbeiter im Web Client sehen kann und welche Aktionen er durchführen darf. Berechtigungen werden immer über Rollen erteilt, d.h. sie werden nicht einem einzelnen Benutzer zugewiesen, sondern einer Gruppe von Benutzern, die die gleiche Rolle haben. Normalerweise gehören diese Benutzer zum gleichen Team und/oder haben ähnliche Funktionen im Unternehmen.

BI

Business Intelligence - Methoden, Technologien und Architekturen, um Daten in für Geschäftszwecke nützliche Informationen umzuwandeln.

C

CFEL

Custom Field Expression Language - Java-Klassen und -Methoden des ConSol CM APIs, um auf Daten von Ticketfeldern, Kundenfeldern und Ressourcenfeldern zuzugreifen.

CIDR

Classless Inter-Domain Routing, Notation für IP-Adressbereiche

CM/Doc

Ein ConSol CM-Standard-Modul, das es Bearbeitern ermöglicht, über den Web Client mit Microsoft Word- oder OpenOffice-Dokumenten zu arbeiten, die mit ConSol CM-Ticket- oder Kundendaten vorausgefüllt sind.

CM/Phone

Das ConSol CM-Modul, das CTI für CM bereitstellt

CM/Resource Pool

CM/Resource Pool ist ein optionales Add-on, das es ermöglicht, unterschiedliche Arten von Objekten als Ressourcen in ConSol CM zu speichern.

CM/Track

CM/Track ist das Portal von ConSol CM. Kunden erhalten über CM/Track Zugriff auf ihre Tickets.

CMDB

ConSol CM-Datenbank - die Arbeitsdatenbank des CM-Systems.

CMRF

ConSol CM Reporting Framework - eine Java EE-Applikation, die Daten zwischen der ConSol CM-Datenbank und dem DWH synchronisiert.

CTI

Computer Telephony Integration - übergeordnete Bezeichnung für Technologien, die eine Verknüpfung von Telefonanlagen mit Computersystemen ermöglichen.

D

DWH

Data Warehouse - ConSol CM-Datenbank, die für Reporting und Datenanalyse verwendet wird.

E

ERP-System

Enterprise Resource Planning - häufig verwendet für diese Art von Enterprise Management Software.

ESB

Enterprise Service Bus - eine Software-Architektur, die für die Kommunikation zwischen gemeinsam interagierenden Software-Applikationen in einer service-orientierten Architektur (SOA) verwendet wird.

ETL

Extract Transform Load - extrahiert Daten aus einer Quelle (dies kann eine Datenbank oder eine andere Quelle sein), wandelt diese um und lädt sie in eine andere Datenbank, z.B. ein Data Warehouse.

CM/ ist eine geschützte Marke der ConSol Consulting & Solutions Software GmbH

F

Firma

Die Firma stellt die obere Hierarchiestufe in einem zweistufigen Kundenmodell dar. Eine Firma kann mehrere Kontakte haben.

FlexCDM

Das Flexible Kundendatenmodell - das Kundendatenmodell, welches in ConSol CM-Version 6.9 eingeführt wurde. Für jede Kundengruppe kann ein eigenes Kundendatenmodell definiert werden.

G

GUI

Graphical User Interface

H

Hauptkunde

Der Hauptkunde ist der Kunde, der der Grund für die Erstellung des Tickets ist. In einem Ticket muss ein Hauptkunde angegeben sein.

HMAC

Hash-based Message Authentication Code, Funktion zur Authentifizierung von Nachrichten unter Verwendung von Hashes

I

IMAP

Internet Message Access Protocol - Internet-Standardprotokoll für den Zugriff auf E-Mails auf einem Remote-E-Mail-Server. Kann als einfaches (plain) IMAP oder sicheres (secure) IMAP (IMAPs) verwendet werden. Im letzteren Fall wird ein gültiges Zertifikat benötigt.

J

Java EE

Java Enterprise Edition

JMS

Java Message Service - Java EE Komponente, um Nachrichten zwischen JMS-Clients zu versenden.

JRE

Java Runtime Environment. Java-Laufzeitumgebung, enthält die virtuelle Maschine, die für die Ausführung von Java-Anwendungen nötig ist.

K

Kerberos

Ein Protokoll zur Authentifizierung im Netzwerk, das auf Kerberos-Tickets basiert und eine spezielle Infrastruktur benötigt

Kontakt

Der Kontakt stellt die untere Hierarchiestufe in einem zweistufigen Kundenmodell dar. Ein Kontakt kann nur zu einer Firma gehören.

KPI

Key Performance Indicator - Parameter für die Performance-Messung für Firmen, Projekte usw.

Kunde

Der Kunde stellt die externe Seite eines Tickets dar. Er ist die Person oder das Objekt, das den Grund für die Erstellung eines Tickets bildet. Ein Kunde kann entweder eine Firma oder ein Kontakt sein.

Kundenaktion

Teil des Action Frameworks - Aktion, die für ein Kundenobjekt (d.h. eine Firma oder einen Kontakt) ausgeführt wird.

Kundendatenmodell

Das Kundendatenmodell ist die Definition der Kunden. Es bestimmt die verfügbaren Datenfelder und möglichen Relationen.

Kundenfeld

Ein Feld, in dem Daten für Kunden (Kontakten oder Firmen) gespeichert werden. Ähnlich den Ticketfeldern für Ticketdaten.

Kundenfeldgruppe

Eine Gruppe von Feldern, in denen Daten für Kunden (Kontakte oder Firmen) gespeichert werden. Ähnlich den Ticketfeldgruppen für Ticketdaten.

Kundengruppe

Die Kundengruppe bestimmt, welches Kundendatenmodell für ihre Kunden verwendet wird und welche Aktionen verfügbar sind.

Kundenobjekt

Ein Kunde (Kontakt oder Firma). Ehemals Unit oder Datenobjekt.

L

LDAP

LDAP ist die Abkürzung für Lightweight Directory Access Protocol. Das ist ein Protokoll, mit dem Anmeldeinformationen für mehrere Applikationen verwaltet werden.

LDAPS

LDAP über SSL

M

Mule

Ein Open Source Java-basierter Enterprise Service Bus (ESB).

N

NIMH

New Incoming Mail Handler - Modul für das Abrufen von eingehenden E-Mails, neu seit Version 6.9.4.

P

PCDS

Page Customization Definition Section - Definitionsbereich der Seitenanpassung.

Pentaho

PentahoTM ist eine Business Intelligence-Produktsuite, die als Open-Source- und als Enterprise-Version verfügbar ist.

POP

Post Office Protocol - Standard-Internetprotokoll für den Fernzugriff auf einen E-Mail-Server mittels TCP/IP. Kann als einfaches POP oder als verschlüsseltes POP (POPS) benutzt werden. Im letzteren Fall werden die geeigneten Zertifikate benötigt.

Portal

CM/Track - ermöglicht Kunden Zugriff auf ConSol CM.

Postfach

Zielort, an den E-Mails zugestellt werden. Postfächer werden auf einem E-Mail-Server verwaltet. ConSol CM kann E-Mails von einem oder mehreren getrennten Postfächern abrufen.

Process Designer

ConSol CM-Komponente für das Design, die Erstellung und das Installieren von Workflows.

Protokoll

Das Protokoll enthält alle Änderungen, die an dem Ticket, Kunden oder der Ressource ausgeführt wurden.

Q

Queue

Die Queue enthält thematisch ähnliche Tickets, die gleich behandelt werden sollen und dem gleichen Geschäftsprozess (Workflow) folgen. Berechtigungen und andere Parameter werden immer auf der Basis von Queues definiert.

R

RDBMS

Relational Database Management System (relationales Datenbankmanagementsystem) - z.B. Oracle ®, MS SQL Server ®, MySQL.

Relation

Relationen sind Verknüpfungen zwischen unterschiedlichen Datenobjekten in ConSol CM. Es gibt Relationen zwischen Objekten des gleichen Typs, z.B. zwischen Tickets, Kunden und Ressourcen, und Relationen zwischen Objekten unterschiedlicher Typen, z.B. zwischen einem Ticket und einer Ressource oder einem Kunden und einer Ressource.

Ressource

Ressourcen sind Objekte, die in CM/Resource Pool verwaltet werden.

Ressourcenaktion

Teil des Action Frameworks - Aktion, die für ein Ressourcenobjekt ausgeführt wird.

Ressourcenfeld

Feld, in dem Ressourcendaten gespeichert werden.

Ressourcenfeldgruppe

Eine Gruppe von Feldern, in denen Daten für Ressourcen gespeichert werden. Ähnlich den Ticketfeldgruppen für Ticketdaten.

Ressourcentyp

Der Ressourcentyp ist die Definition der Ressourcen. Es bestimmt die verfügbaren Datenfelder und möglichen Relationen und Aktionen.

REST

Representational State Transfer - Methode, um Daten mittels eines Netzwerks zu übermitteln, basierend auf HTTP.

Rolle

Rollen werden den Bearbeitern zugewiesen. Sie bestimmen die Zugangsberechtigungen und Sichten der Bearbeiter.

S

Sicht

Sichten beschränken die in der Ticketliste im ConSol CM Web Client angezeigten Tickets auf die Tickets, die bestimmte Kriterien (Bereiche eines oder mehrerer Workflows) erfüllen. Sichten werden Rollen zugewiesen.

Skript

Programme, die für eine spezielle Laufzeitumgebung geschrieben werden, die die Ausführung von Aufgaben interpretieren und automatisieren können. In ConSol CM werden Skripte im Admin Tool gespeichert und als Skripte für Aktivitäten in Workflows gespeichert.

SMTP

Simple Message Transfer Protocol - Standard-Protokoll für den Versand von E-Mails.

Suche-Aktion

Teil des Action Frameworks - Aktion, die für das Ergebnis einer Suche ausgeführt wird.

T

TAPI

Telephony Application Programming Interface - Microsoft Windows API, welche Computer-Telefonie-Integration bereitstellt und es PCs unter Microsoft Windows ermöglicht, Telefondienste zu verwenden.

TEF

Task Execution Framework - ein ConSol CM-Modul, das Aufgaben (Tasks) asynchron ausführen kann. Neu seit Version 6.9.4.

Ticket

Das Ticket ist die Kundenanfrage, an der der Bearbeiter arbeitet. Es ist das Objekt, das den vom Workflow definierten Geschäftsprozess durchläuft.

Ticketfeld

Ein Feld, in dem Ticketdaten gespeichert werden.

Ticketfeldgruppe

Eine Gruppe von Ticketfeldern, in denen Ticketdaten gespeichert werden.

U

Unit

Java-Klasse, die einen Kunden darstellt. Ein Kontakt ist ein Objekt der Klasse Unit, und eine Firma ist ein Objekt der Klasse Unit.

V

Vorlage

Vorlagen enthalten vordefinierten und vorformatierten Text. Sie können für Kommentare, E-Mails und Dokumente verwendet werden.

W

Web Client

Der Web Client ist der Hauptzugang zum System für die Bearbeiter.

Wicket

Apache Wicket ist ein serverseitiges, komponentenorientiertes Open-Source-Framework für Java-Web-Applikationen. Details unter https://wicket.apache.org/.

Workflow

Der Workflow ist die Umsetzung des in ConSol CM verwalteten Geschäftsprozesses. Er enthält eine Reihe von Schritten, die von den Bearbeitern durchgeführt werden.

Z

Zeitbuchung

Zeitbuchungen ermöglichen es den Bearbeitern, die Zeit festzuhalten, die sie an einem Ticket oder Projekt gearbeitet haben.

Zusatzkunde

Zusatzkunden sind Kunden (Firmen oder Kontakte), die am Ticket interessiert sind. Sie sind optional und haben normalerweise eine Rolle, die anzeigt, wieso sie hinzugefügt wurden.

zusätzlicher Bearbeiter

Zusätzliche Bearbeiter sind Bearbeiter, die einen bestimmten, im Geschäftsprozess definierten Zweck erfüllen. Normalerweise müssen sie im Prozess bestimmte Aufgaben ausführen.

Index

A

abhängige sortierte Liste
  Ticketfeldgruppe 111
ACF 125
  Layout 128
acim 222
Action Framework
  Skript 654
activity item 222
Administrator
  Bearbeiter und Rollen 86
Adress-Vervollständigung 498
Aktivitätsformular 125
Annotation 1113
  Ressourcenfeld 989
  String-Feld 118
  Ticketfeld 122, 131
  Ticketfeldgruppe 112
Arbeitszeitkalender 536
Authentifizierung
  Kerberos 880
Autocomplete-Suche 603

B

Bearbeiter 60
  Rollen 65
  Sichtenmerkmale 66
Bearbeiterfunktion 98
  Berechtigungen 102
Bearbeiterkonto 61
Berechtigungen
  Administrator 74
  Kundengruppe 76
  Vorlage 75
  Workflow 74
Bezeichnung 514
boolean 115, 411, 983

C

Chart
  3D 317
  Drilldown 319
Chart-Widget 296
  3D 317
  Drilldown 319
CM-Service 594
CM/Doc 838
  Berechtigung 840
  MergeFields 849
CM/Phone 1101
CM/Resource Pool 945
  Bezeichnung 516
CM/Track 1046
  Anmeldeinformationen 896, 1049, 1063, 1081
  Anmeldung 1066, 1085
  Berechtigung 1061, 1067, 1076, 1086
  Berechtigungen 76
  FAQ 1068, 1087
  Passwort zurücksetzen 1067
  Systemzugang 1060, 1075

Textklasse 562
CM/Track-Benutzer 64
contact data reference 118, 414, 986
CSV-Export 637
CTI 1101

D

Dashboard 187
Chart 188, 296
drucken 316
Layout 284
Seitenanpassung 281
Skript 289
Tabelle 188, 302
Widget 188
Data Warehouse 573
date 116, 412, 984
Dateistruktur 929
Datentyp 115, 983
contact data reference 118, 414, 986
date 116, 412, 984
enum 116, 412, 984
fixed point number 117, 413, 985
list 116, 412, 984
long string 118, 414, 986
MLA field 118, 414, 986
number 117, 413, 985
short string 118, 414, 986
string 117-118, 413, 985
struct 117, 413, 985
Datentype
boolean 115, 411, 983
Datum 116, 412, 984
Datumsformat
acim 224-227
Detailsuche 598
Ergebnisse 600
Export 637
DWH 573
Admin-Modus 581
Aufgabe 586
Live-Modus 581
Lokalisierung 592
Synchronisierung 583
Ticketprotokoll 168
Übertragungsmodus 590

E

E-Mail 680, 927
automatisch 681
Backup 691
manuell 680
NIMH 688, 693
Postausgang 686
Posteingang 685
TO-Adresse 231
Verschlüsselung 694
E-Mail-Backup 691
E-Mail-Vorlage 795
enum 116, 412, 984
Enum 138
Enum-Parameter 144
Export 770, 773
Einzelticket 775
Konfiguration 776
Laufzeitdaten 775

F

FAQ 1068, 1087
Festkommazahl 117, 413, 985
Firmenseite 379
fixed point number 117, 413, 985
FlexCM 369
Flexibles Kundendatenmodell 369

G

GUI 41

H

Hauptkunde 384

I

Import 770, 777
Indexer 610
Indizierung 605

K

Kalender
  Arbeitszeitkalender 536
  Feiertage 540
  Feiertage importieren 541
  Microsoft Exchange 543
  Queue 526
  Zeitzone 538
Kalenderskript 546
Kerberos 880
Kerberos Principal Name 64
Kontakt
  anonymisieren 389
Kontaktseite 382
Kunde
  deaktivieren 388
  löschen 389
  Template 428
Kunden
  Datenobjekt 396
Kunden-Template 428, 760
Kundenaktion 466
  Admin Tool 469
  Aktionsskript 472, 475, 480
  Bedingungsskript 472, 494
  Web Client 385
Kundenaktionsskript 480
Kundenbedingungsskript 494
Kundendatenmodell 369
  Einrichtung 394
Kundenfeld 411
Kundenfeldgruppe 410
Kundengruppe 445
  Berechtigungen 451
  Schnellsuche 393
Kundengruppenfilter 377
Kundenrelation 458
  Admin Tool 461
  Web Client 387
Kundenrolle 454
Kundenseite
  Ticketfilter 390

L

Label 514
LDAP 928
LDAP-ID 64
LDAPS 877, 903, 1056

list 116, 412, 984
Liste 116, 412, 984
Lizenz 782
Log-Datei 934
log file 934
long string 118, 414, 986

M

Microsoft Exchange-Kalender 543
  Skript 546
MLA 118, 149, 414, 986
MLA field 118, 414, 986
Multi Level Attribute 149

N

Nachricht
  Bezeichnung 518
NIMH 688, 722
number 117, 413, 985

P

Passwort-Richtlinie 873
Passwort zurücksetzen 874
  CM/Track 766-767, 1067
  Web Client 765
PostActivityExecutionScript 752
Projekt 533

Q

Queue 523
  FAQ 526
  Kalender 526
  Skript 528

R

Relationen
  Graphendarstellung 193
Reporting 574
Ressource
  Berechtigungen 1034
  Detailsuche 966
  Schnellsuche 966
  Template 995
Ressourcen-Dashboard 1035
Ressourcen-Template 995
Ressourcenaktion 1014
  Admin Tool 1015
  Web Client 963
Ressourcenfeld 982
Ressourcenfeldgruppe 979
Ressourcenkategorie 970
Ressourcenmmodell
  Einrichten 969
Ressourcenmodell 947
Ressourcenmodus 976
Ressourcenpool-Dashboard 1035
Ressourcenrelation 1005
  Admin Tool 1007
Ressourcenrelationen
  Web Client 960
Ressourcentyp 974
Ressourcentypseite 958
Rolle 68
  Administrator 74, 86
  allgemeine Berechtigungen 73
  Bearbeiterfunktionen 84
  Queue-Berechtigungen 70

Ressourcenberechtigungen 80
Sichten 82

S

Schnellsuche 597
Kundengruppe 393
Seitenanpassung 196
Bereich 203, 209
Typ 202
Unterbereich 204
Service
CM 594
short string 118, 414, 986
Sicht 89
Bereichsfilter 91
dynamisches Merkmal 93
Queue-Filter 91
statisches Merkmal 92
Sichtbarkeit 159, 166
Skript 701
Abhängige sortierte Liste 718
Duplizieren 727
E-Mail 722
Feedback 635
PostActivityExecutionScript 752
Queue 528
Standardwerte 746
Task 643
Workflow 751
Sortierte Liste 116, 138, 412, 984
Gruppe 140, 142
Typ 140-141
Wert 140, 144
Sprache 511
SSO 880
string 117, 413, 985
struct 117, 413, 985
Suche-Aktion 620
Kunde 628
Ressourcen 623
Ticket 621
Suchergebnisse
Export 637
Symbol 48
System-Properties 788, 1135
Systemarchitektur 920
Szenario 771

T

Tabellen-Widget 302
Task-Skript 643
Task Execution Framework 640
TEF 640
Template 757
Kunde 760
Passwort zurücksetzen 764
Ticketzuweisung 762
Textklasse 556
CM/Track 562
Farbe 559
Icon 562
Sichtbarkeit 560
Verfügbarkeit 560
Textvorlage 795-796
Typ 801
Ticket
löschen 568
wieder öffnen 568

Ticket-Icon
Farbe 146
Ticketdaten
Layout 124
Ticketfeld 108, 114
Annotationen 131
Ticketfeldgruppe 107, 110
abhängige sortierte Liste 111
Ticketliste 221
Kunde 221
laden 221
Link zum Zuweisen 221
Ticketprotokoll 157
Anzeigemodus 165
DWH 168
Sichtbarkeit 159, 165-166

V

Verschlüsselung
Client-Zertifikat 697
Server-Zertifikat 696
Vorlage
Berechtigungen 798
Brief 806
E-Mail 795
Include 817
Microsoft Word 841
Open Office 856
Parameter 813
Skript 825
Text 796
Textblock 820
Zuordnung 834
Vorlagen-Manager 795

W

Web Client Dashboard 187
Webhooks 908
Widget
Chart 296
Tabelle 302
Workflow 526

Z

Zahl 117, 413, 985
Zeitbuchung 859
aktivieren 232
automatisch 866
Bearbeiterprofil 863
manuell 860