ConSol Software GmbH

ConSol CM Administrator Manual

Version 6.13.0.5

Contents

Contents 2
0.1 ConSol CM for business process management 12
0.2 List of manuals 13
0.3 TecDoc server 14
0.4 Manual structure 15
0.5 Layout explanations 17
0.6 Legal notice 18
0.7 Gender disclaimer 18
0.8 Copyright 18
0.9 System components of ConSol CM 19
0.10 Technical principles of ConSol CM 21
0.10.1 Introduction 22
0.10.2 Engineers 22
0.10.3 Customers 23
0.10.4 Tickets 23
0.10.5 Resources 24
0.10.6 Queues 24
0.10.7 Workflows 25
0.10.8 Action Framework 25
0.10.9 Task Execution Framework 26
0.10.10 Accessing objects in ConSol CM 26
0.10.11 ConSol CM from a system administrator's point of view 26
0.11 Starting the Admin Tool 28
0.11.1 Login 28
0.11.2 Troubleshooting: When the Admin Tool Does Not Start 29
0.12 GUI of the Admin Tool 31
0.12.1 Introduction 31
0.12.2 Basic Principle 31
0.12.3 Inline Validation of Input Values 35
0.12.4 Icons and Other GUI Elements 38

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

0.12.5 Localization of Terms Displayed in the Web Client 40
A - Access and Roles 48
A.1 Engineers 49
A.1.1 Introduction to Engineer Administration 49
A.1.2 Engineer Administration Using the Admin Tool 50
A.2 Roles 54
A.2.1 Introduction to Role Administration 54
A.2.2 Role Administration Using the Admin Tool 55
A.2.3 Defining Additional Administrator Roles 71
A.3 Views 72
A.3.1 Introduction to View Administration 72
A.3.2 View Administration Using the Admin Tool 73
A.4 Engineer Functions 80
A.4.1 Introduction 80
A.4.2 Create or Edit an Engineer Function 82
A.4.3 Delete an Engineer Function 83
A.4.4 Disable or Enable an Engineer Function 83
A.4.5 Engineer Permissions Concerning Engineer Functions 83
B - Ticket Data Model 85
B.1 Ticket Fields (Setting Up the Ticket Data Model) 86
B.1.1 Introduction 86
B.1.2 Ticket Field Administration Using the Admin Tool 88
B.1.3 Tab Ticket Data 89
B.1.4 Tab Activity Form Data 101
B.1.5 Frequently Used Annotations 105
B.2 Managing Sorted Lists: Enums 111
B.2.1 Introduction 111
B.2.2 Enum Administration Using the Admin Tool 113
B.3 MLAs 120
B.3.1 Introduction 120
B.3.2 MLA Administration Using the Admin Tool 122

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

B.4 Ticket History 127
B.4.1 Introduction 127
B.4.2 Editing Comments in the Ticket History 128
B.4.3 Display Modes of Ticket History in the Web Client 129
B.4.4 General Information about the Visibility of Ticket History Entries in the Web Client 135
B.4.5 Ticket History Storage and Transfer to the Data Warehouse (DWH) 138
B.5 Ticket List 139
B.5.1 Introduction 139
B.5.2 Configuration of Grouping and Sorting of the Ticket List 141
B.5.3 Configuration of Parameters Which Are Displayed for One Ticket 146
B.5.4 The Definition of Customer Templates 147
B.5.5 The Annotation of Ticket Fields 147
B.5.6 The Page Customization for the Ticket List-Specific Attributes 148
C - Customer Data Model 156
C.1 Introduction to Customer Data Models 157
C.1.1 The Flexible Customer Data Model at a Glance 158
C.1.2 Introduction to the Objects in the Customer Data Model 161
C.1.3 Important Terms Used for Objects in the Customer Data Model 162
C.1.4 Management of Customer Objects Using the Admin Tool 163
C.2 A Short Introduction to Customer-Specific Web Client Functionalities 165
C.2.1 Introduction 165
C.2.2 Working with Customers in the ConSol CM Web Client 165
C.3 Setting Up the Customer Data Model 180
C.3.1 Introduction to Setting Up the Customer Data Model 180
C.3.2 Managing Contacts and Companies Using the Admin Tool 181
C.4 Customer Field Management and GUI Design for Customer Data 191
C.4.1 Introduction 191
C.4.2 Defining Customer Fields for Customer Data Using the Admin Tool 191
C.4.3 Scripting Using Customer Objects 202
C.4.4 Using Scripted Field Visualization for Customer Fields 205

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.5 Templates for Customer Data 206
C.5.1 Introduction to Using Templates for the Display of Customer Data 206
C.5.2 Coding Templates 207
C.5.3 Template Types 210
C.6 Managing Customer Groups 214
C.6.1 Basic Principles of Customer Data Models and Customer Groups 214
C.6.2 Managing Customer Groups Using the Admin Tool 214
C.6.3 Assigning Access Rights for Customer Groups 218
C.7 Customer Roles 222
C.7.1 Introduction 222
C.7.2 Defining Customer Roles Using the Admin Tool 223
C.8 Customer Relations 226
C.8.1 Introduction 226
C.8.2 Management of Customer Relations Using the Admin Tool 228
C.8.3 Creating Customer Relations Using the Web Client 231
C.8.4 Scripting Using Relations 232
C.8.5 Customer Relations to Resources 233
C.9 Action Framework - Customer Actions 234
C.9.1 Introduction 234
C.9.2 Managing Customer Actions Using the Admin Tool 236
C.9.3 Using Customer Actions as an Engineer (User) 242
C.9.4 Examples for Customer Action Scripts 242
C.9.5 Scripts for the Action Framework: Programming Customer Actions 245
C.10 Address Autocomplete 257
C.10.1 Introduction 258
C.10.2 Switch on the Address Autocomplete Feature Using the Admin Tool 261
C.10.3 Import Zip/City/Address Data into the ConSol CM Database 263
C.10.4 Define the Address Autocomplete Configuration Using the Admin Tool 264
C.10.5 Edit an Address Autocomplete Configuration 267
C.10.6 Delete an Address Autocomplete Configuration or Address Autocomplete Fields 267

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

D - Global Configuration 268
D.1 Languages 269
D.1.1 Languages 269
D.1.2 The Use of Locales 270
D.2 Queues 271
D.2.1 Introduction 271
D.2.2 Queue Administration Using the Admin Tool 272
D.3 Projects 280
D.3.1 Introduction 280
D.3.2 Managing Projects Using the Admin Tool 280
D.4 Working with Calendars 282
D.4.1 Business Calendars 283
D.4.2 Microsoft Exchange Calendar Integration 287
D.5 Classes of Text 299
D.5.1 Introduction 299
D.5.2 Managing Classes of Text Using the Admin Tool 301
E - GUI Design and Advanced Customization 307
E.1 Labels 308
E.1.1 Introduction 308
E.1.2 Configuring Labels Using the Admin Tool 308
E.2 Configuration of the Graphical Representation of Relations 319
E.2.1 Configuration of the Graphical Display in Standard Mode 319
E.2.2 Configuration of More Relation Displays - Expert Mode 321
E.3 Page Customization 322
E.3.1 General Introduction to Page Customization 323
E.3.2 Page Customization in the Web Client 324
E.3.3 Order and Priorities of Page Customization 333
E.3.4 Page Customization Using Attributes 334
E.3.5 Page Customization Attributes for Types, Scopes and Subscopes, in Alphabetical Order 343
E.3.6 Page Customization for the Graphical Representation of Relations 395

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.4 Dashboards 411
E.4.1 Introduction to dashboards in ConSol CM 411
E.4.2 Available widgets 413
E.4.3 Basic tasks 414
E.4.4 Advanced tasks 419
E.4.5 Calendar widgets 420
E.4.6 Chart widgets 424
E.4.7 HTML widgets 431
E.4.8 KPI widgets 432
E.4.9 News widgets 434
E.4.10 Recent changes widgets 435
E.4.11 Recently visited widgets 437
E.4.12 Table widgets 439
E.5 Layout Customization Using Skinning 442
E.5.1 Example 443
E.6 Design and Configuration of REST-based ConSol CM Client GUIs 444
E.6.1 Introduction 444
E.6.2 Configuration Principle 446
E.6.3 Configuration of Specific Pages in CM/Track 448
E.6.4 Creating New Configuration Files 458
E.6.5 Behavior Concerning System Installations and Updates 458
F - Expert 459
F.1 Ticket Administration 461
F.1.1 Introduction to Ticket Administration 461
F.1.2 Ticket Administration Using the Admin Tool 461
F.2 Data Warehouse (DWH) Management 466
F.2.1 Introduction 467
F.2.2 DWH Management Using the Admin Tool 468
F.2.3 DWH-Related System Properties 481
F.2.4 Transfer Mode 481
F.2.5 Expert DWH Information 483

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.3 CM Services 486
F.4 Search in ConSol CM 488
F.4.1 Search Configuration 489
F.4.2 Index 500
F.4.3 Action Framework - Search Actions 505
F.4.4 CSV Export of Search Results 520
F.5 The Task Execution Framework (TEF) 522
F.5.1 Introduction 522
F.5.2 Admin Tool Scripts of Type Task 524
F.5.3 Programming with Tasks 527
F.5.4 System Properties Relevant for the TEF 532
F.6 The ConSol CM Action Framework 533
F.6.1 Actions 533
F.6.2 Additional Components 533
F.6.3 Scripts for the Action Framework 534
F.6.4 Control Forms 545
F.7 Email Configuration 553
F.7.1 Email 554
F.7.2 Email Backups 565
F.7.3 Email Encryption 568
F.8 Scripts and Admin Tool Templates 573
F.8.1 Admin Tool Scripts 574
F.8.2 Admin Tool Templates 608
F.9 Deployment (Import/Export) 619
F.9.1 Introduction 619
F.9.2 Scenarios 619
F.9.3 Deployment (Import/Export) Using the Admin Tool 620
F.10 License Management 627
F.10.1 General Information about Licenses in ConSol CM 627
F.10.2 Sections of a License File 628
F.10.3 Managing the ConSol CM License Using the Admin Tool 629

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.10.4 Expert Information about Accessing Content of CM Licenses 631
F.11 System Properties 632
F.11.1 Introduction 632
F.11.2 System Property Overview 633
F.11.3 Setting System Properties 635
F.11.4 Programming with System Properties 636
F.12 System Documentation 637
F.13 Working with Templates 639
F.13.1 Text Templates 640
F.13.2 CM/Doc 652
F.14 Time Booking Using ConSol CM 667
F.14.1 General Introduction to Time Booking Using ConSol CM 668
F.14.2 Manual Time Bookings 668
F.14.3 Automatic Time Bookings 674
F.14.4 DWH Reports 677
F.14.5 Page Customization for Time Booking 677
F.14.6 Using Time Booking Data in Scripts 677
F.15 Authentication Methods in ConSol CM 680
F.15.1 Authentication Methods for Engineers in the Web Client 681
F.15.2 Authentication Methods for Customers in CM/Track 701
F.16 ConSol CM External Interfaces 708
F.16.1 Webhooks 709
F.17 System Architecture 716
F.17.1 Architecture of a CM System 717
G - Add-Ons 729
G.1 CM/Resource Pool 730
G.1.1 Introduction to CM/Resource Pool 731
G.1.2 CM/Resource Pool - Admin Tool Elements 735
G.1.3 A Short Introduction to CM/Resource Pool Functionality in the Web Client 738
G.1.4 CM/Resource Pool - Setting Up the Basic Resource Model 753
G.1.5 CM/Resource Pool - Templates for Resource Data 776

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.1.6 CM/Resource Pool - Resource Relations 780
G.1.7 CM/Resource Pool - Resource Actions 789
G.1.8 CM/Resource Pool - Assigning Permissions for Resources 809
G.1.9 CM/Resource Pool - The Resource Dashboard 810
G.2 CM/Track: The Customer Portal 812
G.2.1 Overview 812
G.2.2 Installing CM/Track 814
G.2.3 Authentication Methods for Customers in CM/Track 815
G.2.4 CM/Track: System Access for CM/Track Users (Customers) 822
G.2.5 CM/Track: Data Availability For Customers 833
G.2.6 CM/Track: FAQs in CM/Track 836
G.3 CM/EBIA 841
G.3.1 Introduction to CM/EBIA 841
G.3.2 Available reports in CM/EBIA 843
G.3.3 Basic tasks 844
G.3.4 Advanced tasks 846
G.4 CM/Forms 848
G.4.1 Introduction to CM/Forms 848
G.4.2 Available settings for web forms 849
G.4.3 Basic tasks 853
G.4.4 Advanced tasks 854
G.5 CM/Machine Learning 856
G.5.1 Introduction to machine learning in ConSol CM 856
G.5.2 Available machine learning features 858
G.5.3 Process for introducing machine learning 859
G.5.4 Advanced tasks 860
G.6 CM/Archive 863
G.6.1 Introduction 863
G.6.2 Setting up CM/Archive 864
G.6.3 Display of Archived Tickets 867

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.7 CM/Doc 869
G.8 CM/Phone: CTI with ConSol CM 870
G.8.1 Introduction to CM/Phone 870
G.8.2 CM/Phone Setup 872
G.8.3 Configuration of CM/Phone in the Admin Tool 873
H - Appendix 881
H.1 Annotations 882
H.1.1 List of Field Annotations 883
H.1.2 List of Group Annotations 898
H.2 System Properties 902
H.2.1 Alphabetical List of System Properties 903
H.2.2 List of System Properties by Area 1027
H.2.3 List of System Properties by Module 1069
H.3 Administrator and Notification Email Addresses 1181
H.3.1 Introduction 1181
H.3.2 Default Configuration 1181
H.3.3 Subsystem-Specific Notification Email Addresses 1183
H.4 Default Java Imports 1187
H.5 List of Code Examples 1191
H.6 Trademarks 1195
Glossary 1197
Index 1204

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

0.1 ConSol CM for business process management

ConSol CM is a process management platform, especially suited for use as customer service software.

Using ConSol CM you can control and steer business processes with a strong focus on human communication and interaction as required in all fields of customer service management. Well-known examples of huge ConSol CM systems comprise customer service desks, RMA processes, after sales services, call centers and support centers as well as claim and complaint management environments. You can also set up customer portals, including FAQ areas, using ConSol CM. Basically, every business process that is in operation in a company can be modeled and brought to life with ConSol CM.

ConSol CM also provides the functionality to cover adaptive case management. In this way, you can decide, if you would like to design and live a strictly controlled business process or if a rather high level of flexibility is required. You might also combine both concepts, depending on the team or department who work with the process.



Figure 1: *Overview of potential fields of use of the BPM software ConSol CM*

Using ConSol CM, you can handle all components which are relevant in business processes to represent and control your company's processes in an optimal way. ConSol CM is used in various different industries and branches ranging from insurances and banks over fashion designing companies to producers of ticket vending machines or car washes. The flexible process designing mechanism and workflow engine provide a perfect basis for the modeling and controlling of business processes, especially customer service processes, of different kinds.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

0.2 List of manuals

ConSol CM provides documentation for several groups of users. The following documents are available:

- **Administrator Manual**
  A detailed manual for CM administrators about the ConSol CM configuration using the Admin Tool.
- **DWH Manual**
  A detailed explanation of the ConSol CM data warehouse (DWH) concept, the database schema and a list of all table structures.
- **Operations Manual**
  A description of the ConSol CM infrastructure, the server integration into IT environments and the operation of the CM system, for IT administrators and operators.
- **Process Designer Manual**
  A guideline for workflow developers about the graphical user interface of the Process Designer and how to program workflow scripts.
- **Setup Manual**
  A technical description for ConSol CM setup in different IT environments. For expert CM administrators.
- **System Requirements**
  List of all requirements that have to be met to install ConSol CM, for IT administrators and CM administrators. Published for each ConSol CM version.
- **Technical Release Notes**
  Technical information about the new ConSol CM features. For CM administrators and key users. Published for each ConSol CM version.
- **User Manual**
  An introduction to the ConSol CM Web Client for end users.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

0.3 TecDoc server

For detailed information about all aspects of ConSol CM, please see also our **tecdoc server**, available at https://tecdoc.consol.de. You find there:

- All **ConSol CM manuals**
  The ConSol CM manuals for several target groups in English and in German
- The **release notes** for each ConSol CM version
  Detailed Release Notes documents which explain every new feature of every ConSol CM version
- The **system requirements**
  An overview of the required hardware, middleware, and software prerequisites which are required to install ConSol CM
- The **feature presentations**
  Not too technical presentations which provide an overview of the features of new ConSol CM versions

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

0.4 Manual structure

First, some basic principles of ConSol CM are explained to provide the theoretical background you need to become a CM administrator.

- In the **Introduction**, you will learn how ConSol CM is used in Business Process Management and you will get to know some basic principles of the application. Furthermore, you will learn how to start the main administration application for CM, the Admin Tool.

The following sections explain the features and functionality of the Admin Tool.

- **Access and Roles**
  In this section the basic principle and configuration concerning access permissions are explained. For example, you will learn how to define roles, assign roles to engineers (the users of the CM Web Client), and configure views (the to do lists in the system).
- **Ticket Data Model**
  Here, the set-up of the ticket data model and the placement of the data fields in the Web Client (GUI) are covered. For example, you will learn how to define the data fields which are required by a certain process and how to build different types of lists.
- **Customer Data Model**
  This section describes the setup of the data models for different customer groups and the respective designer GUIs. The representation of customer groups in CM is based on the flexible customer data model. For example, you will learn how to define one set of data fields for the customer group *Reseller* and another data model for the customer group *Direct Customers*. Furthermore, customer relations and customer actions are explained, two components which help you use CM as a CRM system.
- **Global Configuration**
  In this section some general configurations are explained. For example, you will learn about Queue Management, a queue being one of the core components of ConSol CM. Furthermore, working with business calendars and projects is explained.
- **GUI Design and Advanced Customization**
  This section describes the different features which you can use to design the user interface and customize ConSol CM.
- **Expert**
  This is the part of the book which is targeted at CM administrators who are responsible for advanced CM system configuration. You might want to work together with your CM consultant to change system settings. This section covers topics like
  - preparing the CM system for reporting
  - ConSol CM services
  - configuration of the search module
  - the Task Execution Framework (TEF)
  - email configuration
  - import and export of configurations
  - template management
  - working with system properties

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

  - working with Admin Tool scripts
  - license management
  - deployment of scenarios
- **Add-Ons**
  In this section, the five CM add-ons are explained. A CM add-on is a CM module for which a separate license has to be purchased. The following add-ons are available:
  - **CM/Resource Pool**
    In this section the set-up of the data model for the Resource Pool is explained. CM/Resource Pool is a distinct CM module which has to be licensed separately. If you have purchased this module, you can learn in this section how to represent different objects like IT assets, products, SLAs or other objects as CM resources. Besides the set-up of the data model, resource relations and resource actions are described.
  - **CM/Track**
    In this section, the customer portal, CM/Track, is explained.
  - **CM/Phone**
    In this section, the ConSol CM CTI solution is explained.
  - **CM/Archive**
    In this section, the ConSol CM ticket archiving feature is explained.
  - **CM/Forms**
    In this section, the ConSol CM feature for creating web forms is explained.
- **Appendix**
  Here, you find lists of all important terms that are used in the book (glossary), of all annotations (important for the GUI design), and system properties (important for the CM system management). Please see also the trademarks page.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

0.5 Layout explanations

The following icons and colors are used to emphasize and highlight information:

This is an additional information.

This is an important note. Be careful here!

This is a warning!

This is a recommendation from our in-the-field consultants.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

0.6 Legal notice

Since we would like to provide a manual for you which helps you manage your CM system, but which also provides additional information about connected topics, we have inserted external links into the manual. In this way, you can get some background information about a topic if you like. This can help you better understand the required CM configuration. Despite careful review, we assume no liability for the content of those external links. The operators of sites linked to are exclusively responsible for their content.

0.7 Gender disclaimer

As far as possible, ConSol CM manuals are written gender-neutral and often address the user with "you". When the phrasing "The user .... he ..." is used, this is always to be considered to refer to both, the feminine as well as the masculine form.

0.8 Copyright

© 2021 ConSol Consulting & Solutions Software GmbH - All rights are reserved.

0.9 System components of ConSol CM

ConSol CM comprises different client applications. Depending on your roles and tasks in your company you will use one or more of those applications.

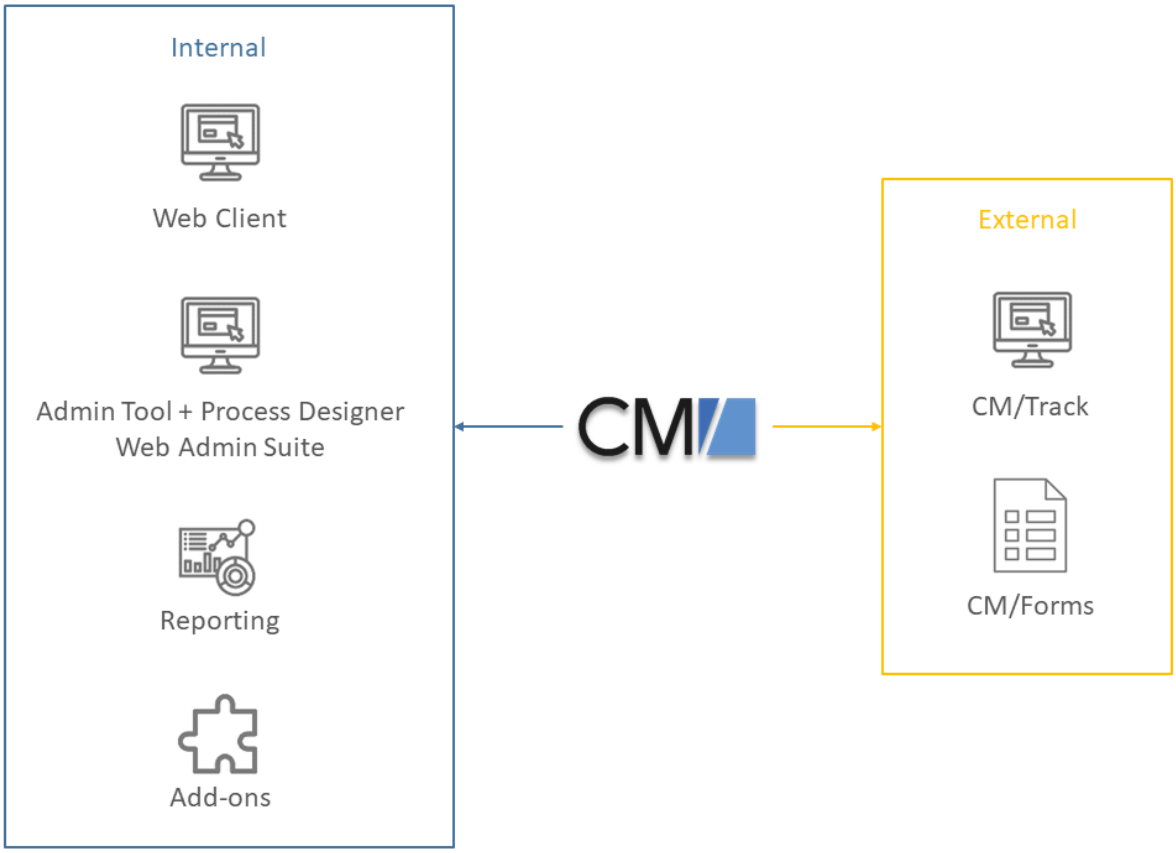


Figure 2: *ConSol CM system components*

- **Web Client**
  The Web Client is the primary access point to the system for internal users. They work with tickets, contacts and resources.
- **Admin Tool, Process Designer, Web Admin Suite**
  These three applications are used by administrators for setting up and maintaining the system. This includes both the system configuration and the workflows. The Web Admin Suite is a newly developed web application which will gradually replace the Admin Tool and Process Designer.
- **Reporting**
  ConSol CM comes with a data warehouse to consolidate data for reporting purposes. In addition, you can create dashboards using scripts, or purchase the add-on CM/EBIA for report creation using a graphical user interface.
- **Add-ons**
  ConSol CM comes with a number of add-ons which can be licensed separately to provide additional features, as for example an archive or machine learning.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **CM/Track**
  CM/Track is a portal which is the primary access point to the system for external users, either your customers or colleagues who do not work with the Web Client. CM/Track is an add-on which requires a separate license.
- **CM/Forms**
  CM/Forms is an add-on which allows to manipulate ConSol CM objects via web forms. The web forms can be filled by unauthenticated users.

Furthermore, ConSol CM is not an isolated application but can be easily integrated into your company's IT infrastructure, e.g. using webhooks or the REST interface.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

0.10 Technical principles of ConSol CM

This chapter discusses the following:

0.10.1 Introduction 22
0.10.2 Engineers 22
0.10.3 Customers 23
0.10.4 Tickets 23
0.10.5 Resources 24
0.10.6 Queues 24
0.10.7 Workflows 25
0.10.8 Action Framework 25
0.10.9 Task Execution Framework 26
0.10.10 Accessing objects in ConSol CM 26
0.10.11 ConSol CM from a system administrator's point of view 26

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

0.10.1 Introduction

In order to work efficiently and correctly with ConSol CM, you have to have a profound knowledge of all components which make up a CM system. The following section will give you a first introduction into the basic CM components.

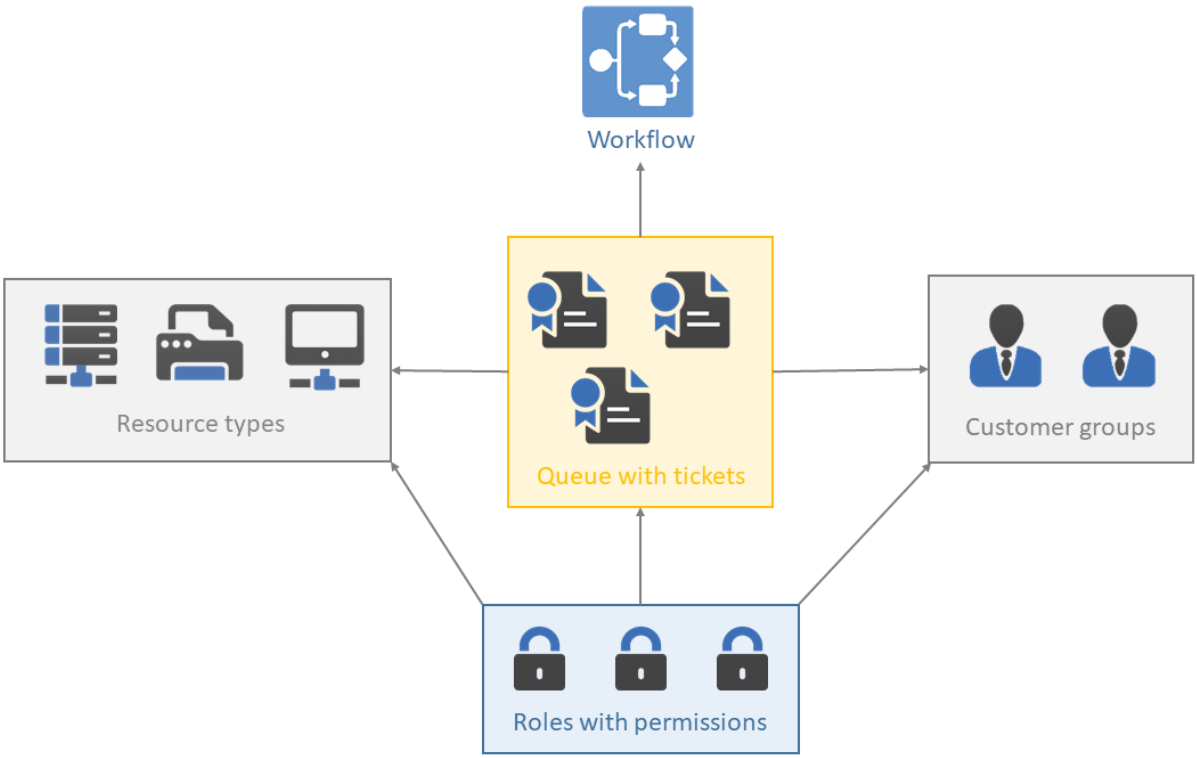


Figure 3: *ConSol CM - Basic principles (with CM/Resource Pool)*

In ConSol CM, the following rule applies regarding tickets:

> **External** side: If an external side is required for the queue, the tickets always have a **main customer**. This can be a contact or a company.
>
> **Internal** side: A ticket can have no or one main **engineer** who works on the ticket.

0.10.2 Engineers

The engineers represent the **internal** side of the CM system. All users of the Web Client are called engineers, regardless of their function within the company. The engineers work on the tickets to carry out the tasks defined in the business process. Every ticket can only be assigned to **one engineer**, who is currently responsible for the ticket. But a ticket may have any number of **additional engineers,** who all have an **engineer function** representing a specific task within the process.

All engineers have a user account consisting of a user name and password, which they use to log in to the Web Client. The engineers' access permissions are managed using roles. The roles, which contain access permissions for queues, customers, and resources, are defined in the Admin Tool and assigned to the engineers.

Please see the section about *Engineer Administration* for details.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

It depends on the configuration of your ConSol CM system if an engineer is called *engineer* within your system. Engineers might be called *agents*, *employees*, or similar in your Web Client. In this manual, engineers will always be called engineers for your convenience.

0.10.3 Customers

ConSol CM provides a customer data model which can define contact and company data in various constellations. In this way, you can define very simple, one-level data models which only contain contact data (e.g., name, phone number, email address, address) and complex, two-level models which contain contact data (e.g., name, phone number, email address) and company data (e.g., address, zip code, company size). You can define different models within one system, you can configure relations between customers, and add activities to contacts and companies.

Please refer to the *Customer Data Model* section, starting with chapter *Introduction to Customer Data Models*, for details about all components.

Each CM system uses customized customer groups and data models. Therefore, the available customer groups, hierarchical levels for customer objects, data fields, relations, and activities depend on the individual configuration of your CM system.

It depends on the configuration of your ConSol CM system if a customer is called *customer* within your system. Customers might be called *contacts* or similar in your Web Client. In this manual, the terms *customer*, *company* and *contact* will be used for your convenience.

0.10.4 Tickets

The ticket is the request of the customer which the engineers work on. This can be an incident, a service case, or any other request. For each request, a ticket is created. The engineers work on the ticket, which means that they carry out the necessary steps as defined in the business process. The progress, including internal and external communication, is documented in the ticket. The business process can involve several engineers and different teams. When the request is solved, the ticket is closed. Closed tickets are not lost, but they represent a powerful archive and knowledge base.

In ConSol CM there are the following rules for tickets:

- If an **external** side is required for the queue, the tickets must have a main customer. Every ticket can have only one main customer. It does not need to have additional customers, but it can have any number of additional customers. The customer represents the **external** side of a ticket.
- A ticket does not have to be assigned to an engineer, but if it is, it can only be assigned to one engineer at a time. A ticket does not have to have additional engineers, but it can have any number of additional engineers. Assigning a ticket to an engineer can be done manually or automatically. The engineers represent the **internal** side of a ticket.
- A ticket always has a name, often called *ticket number*, a subject, and a ticket icon. The ticket icon shows the current scope of the ticket and can have a color indicating the value of a given data field. Every ticket also has an ID that is used internally and cannot be seen by the user.
- The ticket header always shows the current queue and scope, assigned engineer, and creation date of a ticket.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- The ticket icon in the Web Client can have (and in most cases does have) a color that represents a certain value of a list. Often the priority is used, e.g., high priority tickets are displayed in red, medium tickets in orange, and low priority tickets in yellow.



It depends on the configuration of your ConSol CM system if a ticket is called *ticket* within your system. Tickets might be called *ticket*, *case*, *call*, *task*, or similar in your Web Client. Each queue can have its own term to refer to tickets. In this manual, tickets will always be called tickets for your convenience.

Each CM system uses customized ticket data. Therefore, the available fields, relations, and activities depend on the individual configuration of your CM system.

0.10.5 Resources

Resources can be used to manage objects which are related to the business process. Possible use cases are IT assets, SLAs, products or newsletters. All resources are saved in **CM/Resource Pool**, a separate CM module. The administrator defines the resource model, i.e., the resource types, resource data fields, the hierarchy of the resources, and the possible relations to tickets, customers, and other resources.

An engineer with the required permissions can create resources and link them to existing tickets, customers, and other resources. For example, you can link a computer to an incident ticket concerning this computer or to the customer who uses it.

Please see the section about CM/Resource Pool in the Add-on section for details about this topic.

0.10.6 Queues

The queue is the **core component** of ConSol CM administration. It contains thematically related tickets which should be handled in the same way and follow the same business process. Every queue has exactly one **workflow**, which implements the desired process. The ticket data fields needed for the process are assigned to the queue. In addition, the access permissions, which are granted to the engineers via roles, are based on queues.

For example, there is one queue for the user help desk with the *User Help Desk* workflow and data fields like *Customer Service Level*, *Device that does not work*, or *Priority*. Every incident ticket passes through this *User Help Desk* process. Another queue is the *Marketing and Sales* queue where fields like *probability of contract conclusion*, *next appointment* , or *budget [$]* are defined.

Therefore, the queue determines:

- if the tickets have customers
- how the ticket should look like (ticket data fields)
- whose tickets are processed (customer groups)
- how its tickets are processed (workflow)
- who can work on the tickets (permissions)

Queues often reflect the organizational structure of the company. For example, there can be one queue for each department, as each department has its own processes. A ticket can be passed from one queue to another. In this case it adapts to the new queue, i.e., it receives the data fields of the new queue and only engineers with permissions for the new queue can work on it.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH



It depends on the configuration of your ConSol CM system if a queue is called *queue* within your system. Queues might be called *processes*, *teams*, or similar in your Web Client. In this manual, queues will always be called *queue* for your convenience.

0.10.7 Workflows

A workflow is designed and created by a CM workflow developer using the ConSol CM Process Designer. It implements the business process which is executed in the Web Client. The workflow consists of several steps, which are called *activities*. There are manual activities, which are performed by the engineers, and automatic activities, which are performed by the system. The activities are arranged in **scopes** to illustrate the status of a ticket. The intelligence of the process, like conditions, decisions, escalations, reminders, automatically sent emails, or other actions, is also defined in the workflow. You can implement process chains or a hierarchical process structure by linking several workflows.

As an engineer, you will not work with the workflow itself, but you will see the current queue and scope of the ticket. The scope is indicated by the symbol of the ticket icon and scope name in the basic ticket data. Furthermore, you see the workflow activities which are available for the ticket at its current position. In this way, you always have a good overview of the current status of the ticket.

As an administrator, you might work with the ConSol CM Process Designer to model the business processes of your company. A process can be represented by one or more workflows.

Since we often deal with process chains rather than with single processes, in ConSol CM such process chains can be designed by defining a certain order for the processes. You can work with simple process chains or with a hierarchical structure. The intelligence of the process, like escalations, reminders, automatic generation of emails, or other actions during the process, is also defined in the workflow using Groovy scripts.

Please refer to the *ConSol CM Process Designer Manual* for a detailed introduction to process design and modeling using the ConSol CM Process Designer.

0.10.8 Action Framework

In addition to workflow activities, which are activities executed during a certain step of a business process, activities, here called actions, can also be triggered from other objects:

Customer actions are actions which are executed based on the customer object, i.e., for a contact or for a company. In this way, you can, for example, implement a company action which updates your company-specific sales figures every night.

Resource actions are actions which are executed based on a resource, i.e., on an object in the CM/Resource Pool. In this way, you could offer an action in the Web Client where the engineer can have a list with all customers for a certain newsletter.

Search actions are actions which can be executed based on a result of a Detailed Search. In this way you can, for example, trigger an email to all customers of a list of tickets which you have retrieved using the search interface.

All actions can be either executed manually or automatically. Manual actions are offered in the Web Client like workflow activities. Automatic actions run in the background without any engineer activity being involved.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

0.10.9 Task Execution Framework

The Task Execution Framework (TEF) stores and executes long-running tasks which should not be linked to any specific activity. For example, TEF tasks can be very helpful for import scripts.

0.10.10 Accessing objects in ConSol CM

In ConSol CM, the different objects (tickets, customers and resources) form a network. The objects are connected, e.g. a ticket is always linked to one or more customers. The connections which exist between the current object and other objects in ConSol CM are displayed on the object's page, where you can directly access the linked objects.

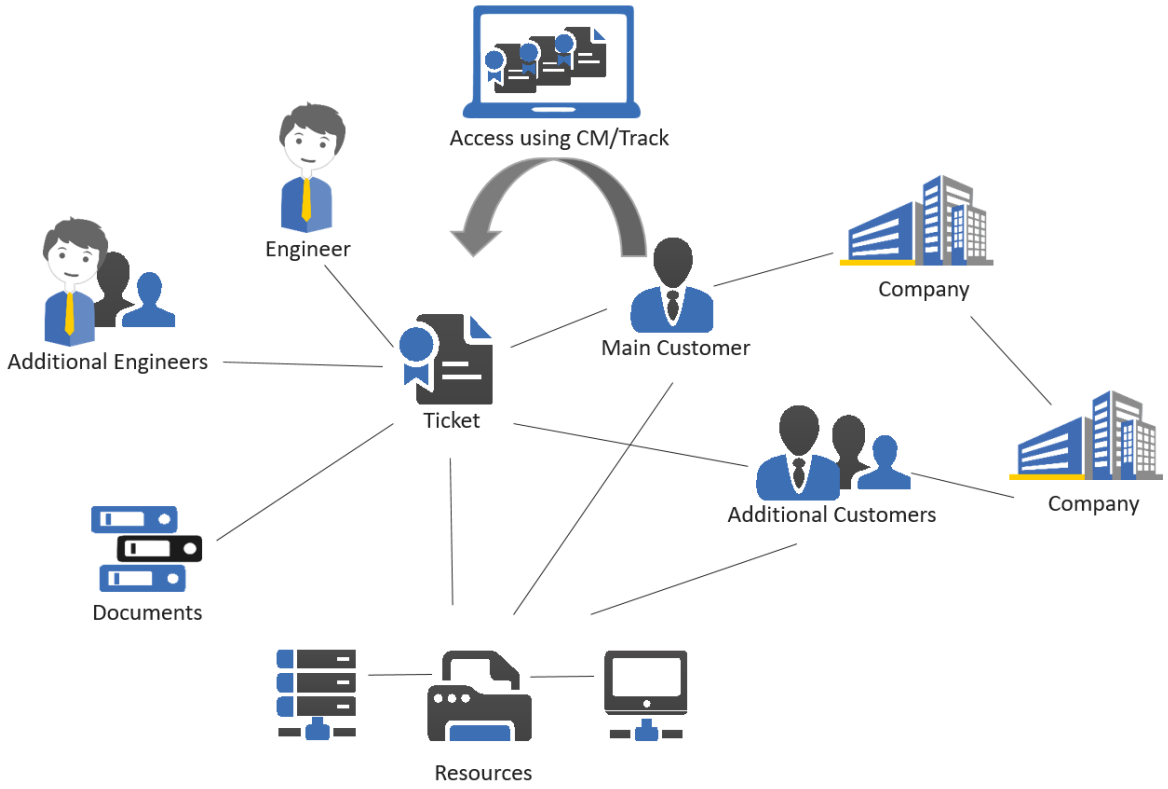


Figure 4: *Network of ConSol CM objects*

Since the ConSol CM Web Client always provides menus or links to open the objects which are related to another object, you, as a CM engineer, can move within the network easily, thus working efficiently with customer, ticket, and resource data. Once you have opened an object (e.g. a ticket), you can move from one related object (e.g. the main customer of the ticket) to other related objects (e.g. the company of the main customer or a resource which is related to the ticket or the customer). No further search is required.

All components for a certain ticket or service request can be accessed easily. If customer-customer relations are managed using CM, the CRM (customer relationship management) component is even stronger.

0.10.11 ConSol CM from a system administrator's point of view

ConSol CM is a Java EE application which runs in a standard application server. The data is stored in a relational database. ConSol CM connects to an email server to retrieve incoming emails and sends emails using an SMTP server. Please refer to the *ConSol CM Operations Manual* for a detailed

explanation of all aspects concerning running ConSol CM in an IT environment. A first introduction is provided in section *System Overview* in this manual.

> A detailed list of supported operation systems, application servers, database systems, and other systems, as well as storage and CPU requirements is given in the current *System Requirements*.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

0.11 Starting the Admin Tool

0.11.1 Login

Most of the ConSol CM system is managed using the ***Admin Tool***. By default, the Admin Tool is available as a Java Web Start application on the start page of the CM application server. To start the Admin Tool, you can either use the link on the page, or you can store the `jnlp` file locally and start it there.

> If Java Web Start is not available on your computer, please contact the ConSol CM support or your ConSol CM consultant. They can provide a local Java application which does not require Java Web Start. Remember to set the host name and port of your ConSol CM installation and, if you are using a Windows operating system, the path to the Java Runtime Environment in the start script delivered with the application.

After clicking `cm-admin-tool.jnlp`, the `jnlp` file is downloaded, the Admin Tool is started, and the login window is displayed.

Refer to the Troubleshooting: When the Admin Tool Does Not Start section, if the Admin Tool does not start.

Enter your login data (user name and password) to get access to the Admin Tool functions. An initial user name and password are created during system set-up. Further admin users can be configured later on in the Admin Tool.

After you have logged in successfully, the start page of the Admin Tool with basic information about the system is displayed.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

0.11.2 Troubleshooting: When the Admin Tool Does Not Start

0.11.2.1 Correct Process Using Java Web Start

If everything is set up properly, clicking on the Admin Tool hyperlink leads to the following:

1. In a pop-up window, you are prompted as to whether you would like to open the `jnlp` file - *Java (TM) Web Start Launcher* should be offered as default application for that - or to download the `jnlp` file to your system.
   Confirm with *Open with Java (TM) Web Start Launcher*.
2. The download of the Admin Tool `jnlp` file is started. During this process, the ConSol CM logo is displayed.
3. Java Web Start starts the Admin Tool. In a pop-up window the *Verifying application* message is displayed.
4. If the Java Web Console is activated, the console is opened and you can follow the progress.
5. The Admin Tool GUI is displayed with the login window in the foreground.

0.11.2.2 Process with Errors Using Java Web Start

If the Admin Tool cannot be started, check the following settings:

1. **Problems with step 1:**
   a. Is a supported version of Java installed on your machine?
   You can find a list of supported Java versions in the *System Requirements*.
   b. Is the correct Java version activated?
   Under Microsoft Windows use *System settings -> Java -> Java -> Display ...*
2. **Problems with step 2:**
   a. Can your client machine connect to the ConSol CM server over the network? Can the `jnlp` file be downloaded by the web browser?
   b. Check the Java Network connection settings.
   Under Microsoft Windows use *System settings -> Java -> General -> Network settings.*
3. **Problems with step 3:**
   a. Does Java Web Start load and verify all Admin Tool application files? If not, check the network connection.
   b. For all other errors, a pop-up window with a detailed error message will be displayed.
4. **Notes concerning step 4:**
   a. To find the cause of a problem, activate the Java Console.
   Under Microsoft Windows use *System settings -> Java -> Extended -> Display console, Debugging: Tracing enabled, Debugging enabled*.
5. **Problems with step 5:**
   a. When the login window is displayed, enter your login data. If a connection error occurs then, check the proxy settings.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

0.11.2.3 Correct Process Using the Local Java Application

Before starting the Admin Tool for the first time, you need to perform the following steps:

1. Unzip the file which you obtained from the ConSol CM support or your ConSol CM consultant.
2. Open the start script with a plain text editor, e.g., Notepad++. For Windows, the file is called `start_admintool.bat` and for Linux it is called `start_admintool.sh`.
3. Make the following settings:
    - Enter the path to your Java Runtime Environment executable (`java.exe`) in the line containing `Java=` (only Windows).
    - Enter the name of the host (without the protocol) in the line containing `Host=`.
    - Enter the port in the line containing `Port=`.

Start the Admin Tool:

- On Windows: Double-click the `start_admintool.bat` file. A command line window is opened and the Admin Tool is started. Do not close the command line window, as this will close the Admin Tool.
- On Linux: Execute the `start_admintool.sh` with the `-java` command. The JRE executable needs to be part of the path. The Admin Tool is started.

0.11.2.4 Process with Errors Using the Local Java Application

If the Admin Tool cannot be started, check the following:

- Is a supported version of Java installed on your machine?
  You can find a list of supported Java versions in the *System Requirements*.
- Is the correct Java version activated?
  Under Microsoft Windows use *System settings -> Java -> Java -> Display ...*
- Is the correct Java version set in the start script (Windows) or part of the path (Linux)?

If the Admin Tool can be started, but you cannot log in, check the following:

- Are the correct host name and port set in the start script?
  Remember to provide the host name without protocol, e.g., `myhost.consol.de`.
- Do you have the Java application for the correct CM version?
  The CM version is stated in the file name of the `jar` file.
- Does the CM version set in the start script match the version of the ConSol CM server which you are connecting to?
- Can your client machine connect to the ConSol CM server over the network?

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

0.12 GUI of the Admin Tool

This chapter discusses the following:

0.12.1 Introduction 31
0.12.2 Basic Principle 31
0.12.3 Inline Validation of Input Values 35
0.12.4 Icons and Other GUI Elements 38
0.12.5 Localization of Terms Displayed in the Web Client 40

0.12.1 Introduction

The following section provides an overview of the Admin Tool graphical user interface (GUI). The basic principle and all icons are explained.

In the run of this manual, we refer to the names of the GUI elements as they are explained here. In order to avoid redundancies, the icons are not always shown in each section.

0.12.2 Basic Principle

On the left-hand side, you see the navigation tree (1) with the navigation groups (2: extended navigation group, 4: closed navigation groups). Each navigation group contains several navigation items (3). Click the name of a group to expand the group in the tree. Click a navigation item to open the respective tab in the working area on the right-hand side.

In the bottom left corner (5), the name of the admin user who is currently logged in is displayed with all roles which are assigned to this user.

In the top right corner (6), you find the buttons to reload the Admin Tool data, to log in and to log out. The main working area (7) contains the data of the active/opened navigation item.

In the following figure, the *Home* tab is opened. This is the start page when you open the Admin Tool.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

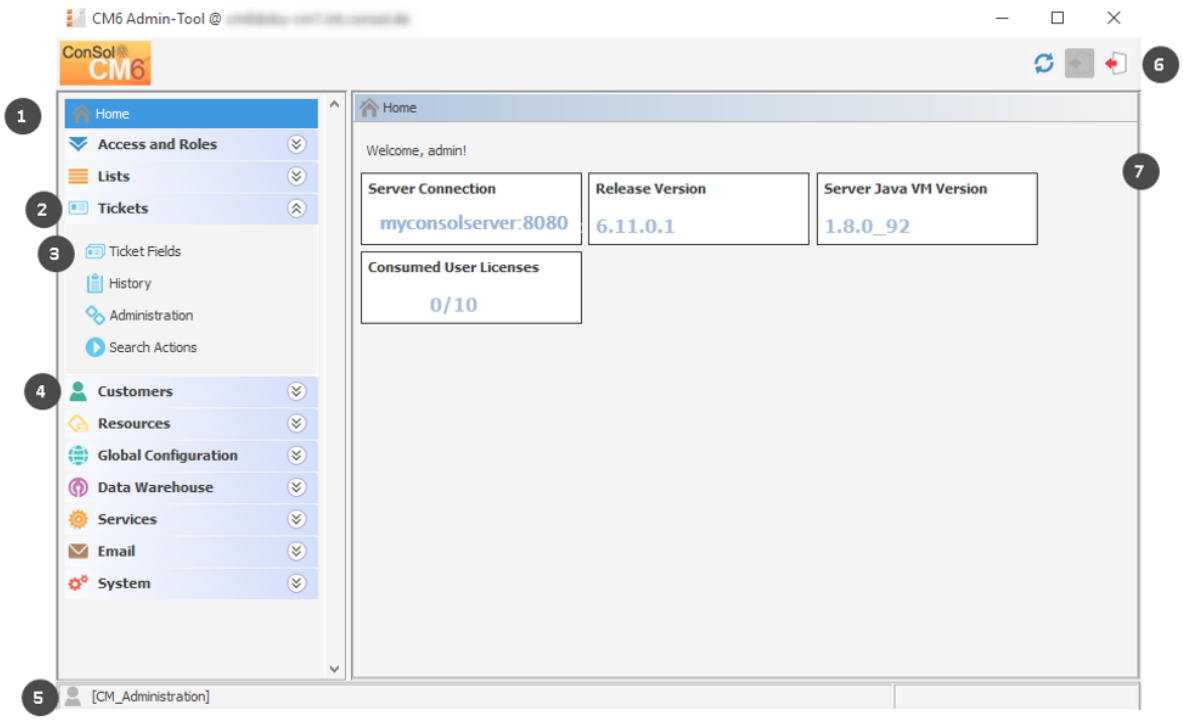


Figure 5: *ConSol CM Admin Tool - GUI overview*

The **Home tab** contains the following data:

- **Server Connection**
  The server name and port where the ConSol CM instance is running
- **Release Version**
  The exact version of the ConSol CM version
- **Server Java VM Version**
  The version of the Java VM (virtual machine) which is used for the current ConSol CM instance
- **Consumed User Licenses**
  The number of licenses which are used, compared to the number of available licenses.
  You can select one of two display modes here by manipulating the value of the CM system property cmas-app-admin-tool, admin.tool.consumed.licenses.pool.name. The value refers to a section in the license file:
  - **CONCURRENT_USERS**
    The number of engineers who are currently logged in using the Web Client is displayed.
  - **TRACK**
    Only possible if CM/Track is in operation. The number of customers who are currently logged in using CM/Track (the ConSol CM portal) is displayed.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

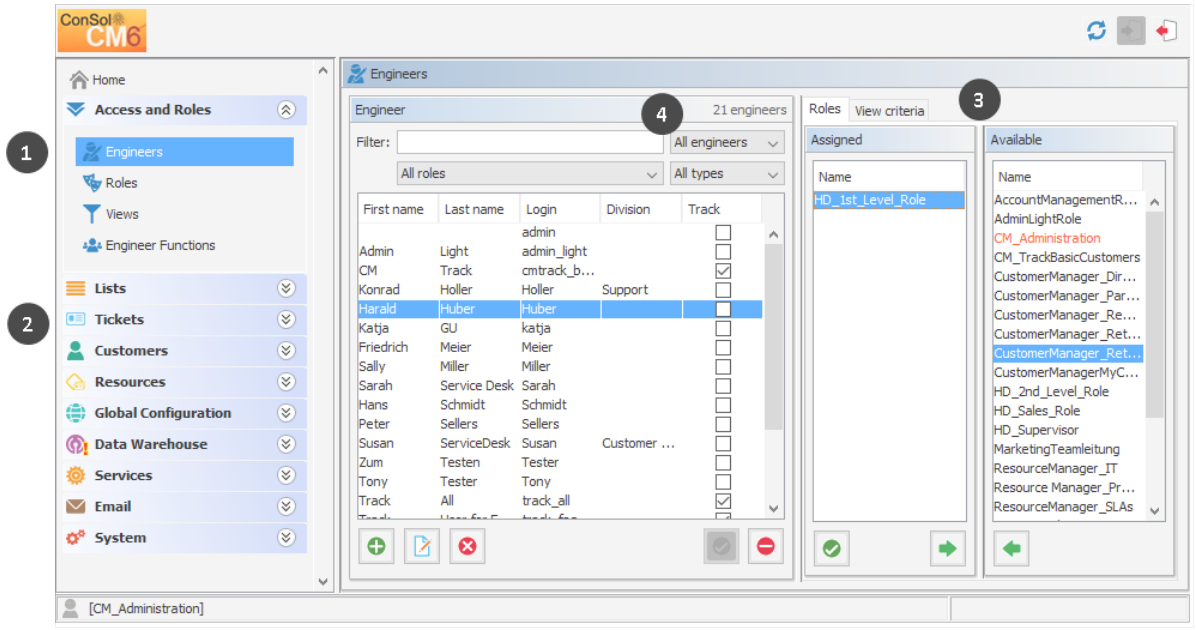


Figure 6: *ConSol CM Admin Tool - GUI overview: Navigation item open*

The user interface of all other navigation items (besides the *Home* tab) basically show the same structure. The following example shows the tab of the navigation item *Engineers*.

The left part of the screen shows the navigation tree (1, 2). The right part of the screen shows the settings for the active navigation item (Engineers, 1, in the example).

In the active tab, *Engineers* in the example, a list on the left shows the elements which can be modified. Elements can be added, edited, deleted, disabled, or enabled.

The attributes of an element are displayed on the right (3). You can move them from a list of available attributes to a list of assigned attributes either via double click or via clicking the *Assign* icon (example: *available roles* and *assigned roles*). Attributes can also be assigned via checkboxes or list boxes (not displayed here).

There are a couple of options to help you find the entries you want to edit more quickly:

- **Filters (4)**
  Filters help you find entries in lists (e.g. in the engineer list) rather quickly. There are two types of filters:
  - **Text filters**
    Type in the characters of the required word (e.g. the engineer name) and the list is updated automatically, displaying only matching entries.
  - **Drop-down menu filters**
    Select a category (e.g. *all engineers*) and only the matching list entries (e.g. engineer names) are displayed.
- **Sorting**
  You can sort the entries in ascending or descending order by clicking in one of the title fields of the list. The small up and down arrow icons denote the current sort order.

Usually all changes performed in the Admin Tool are submitted immediately without the need to synchronize any data. However, if changes have been performed in another module and the Admin Tool has to use the new data, synchronization is required. This can be achieved by clicking the *Refresh* button in the icon bar.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

One example for this is the deployment of a new workflow using the Process Designer. Before the new workflow can be assigned to a new queue, you have to synchronize the data in order to let the Admin Tool know that there **is** a new workflow. The Admin Tool loads all data from the database anew, including the new workflow. This new workflow can then be used for further operations, like assigning it to a new queue.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

0.12.3 Inline Validation of Input Values

For all data models (customer data model, resource data model, ticket data model), the input values are validated during input in order to avoid inconsistencies concerning the names of database fields. **Each value of a field has to be unique on the respective field level.** See the following examples:

- A new resource type cannot have the name of an existing resource type, but it can have the name of an existing resource field group.
- A resource field group cannot have the same name as another resource field group, no matter in which resource type (see example a below).
- A new ticket field cannot have the same name as another field in the same ticket field group, but it can have the same name as another field in another ticket field group (see examples b and c below).
- A new customer data model cannot have the same name as an existing customer data model.

A field which is not valid is marked in red. It is not possible to save this field value.



We recommend to use a defined terminology and syntax for data fields. Field names should be self-explanatory and unique as far as possible. There might be exceptions from this rule due to the implementation of certain special algorithms, but in general this is what you should keep in mind as a ConSol CM administrator!

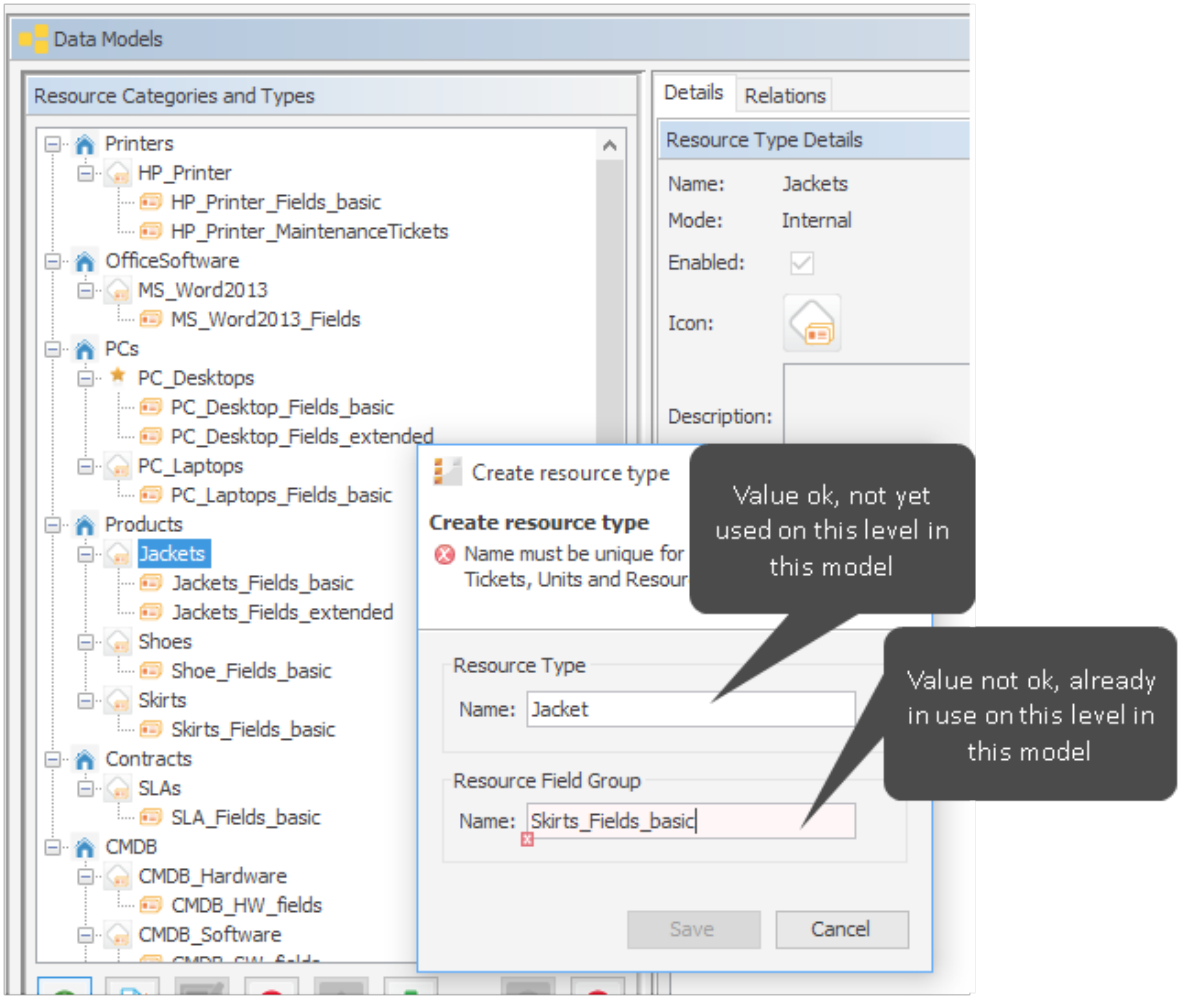


Figure 7: *Inline validation of input values during set up of the resource data model. Example a.*

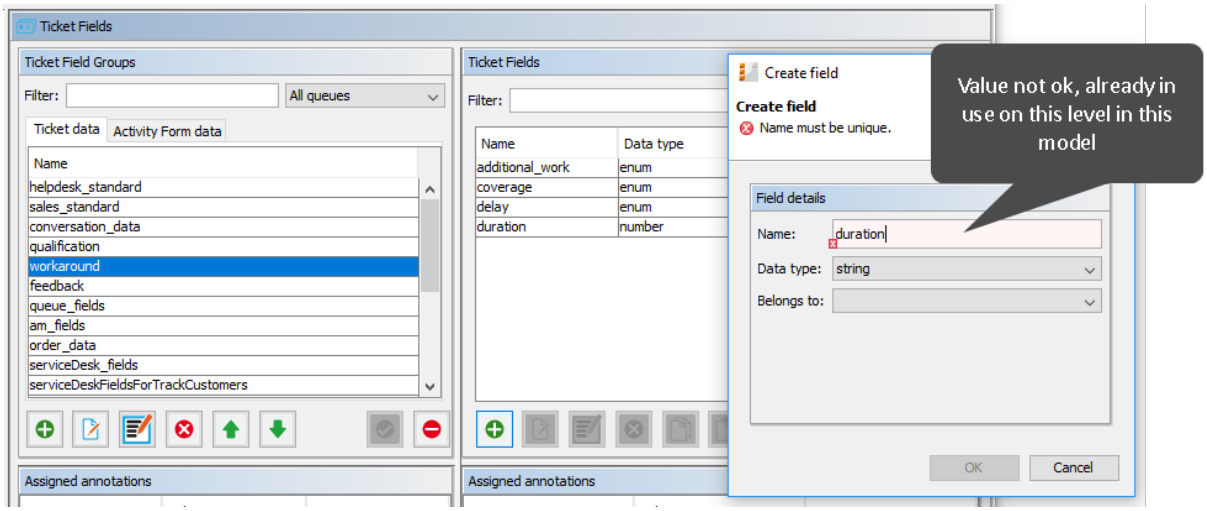


Figure 8: *Inline validation of input values during set up of the ticket data model. Example b.*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

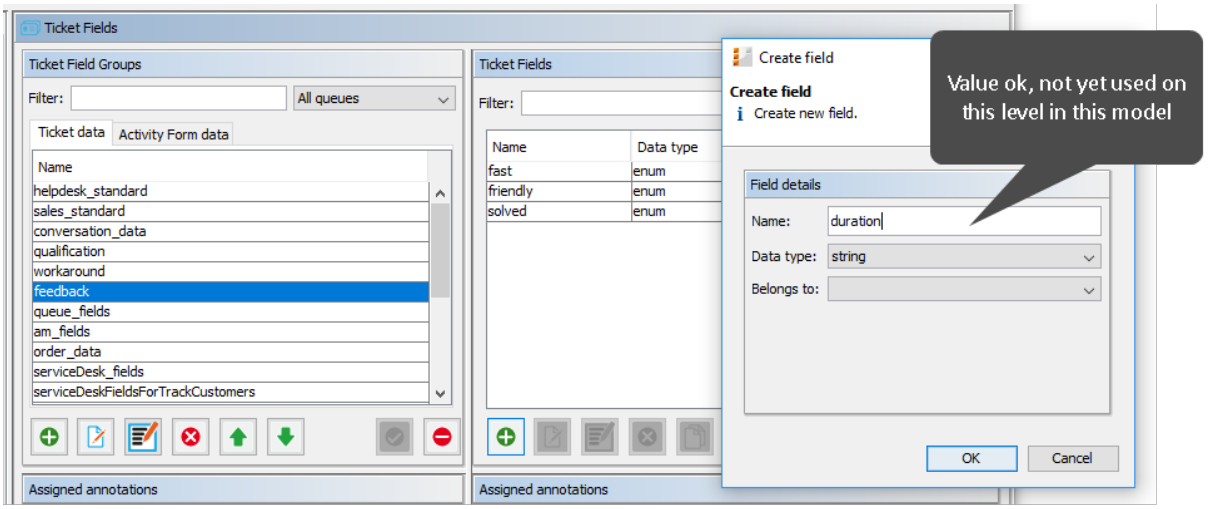


Figure 9: *Inline validation of input values during set up of the ticket data model. Example c.*

0.12.4 Icons and Other GUI Elements

You work with the following icons when you administrate a ConSol CM system. Here, the general explanations are provided, please see the respective sections for a detailed explanation of conditions and implications of applying the functionalities in the respective context.

Icon	Name	Meaning/Function	Examples
	Add / New	Add (= create) a new element of the respective type.	Add (create) a new engineer, a new view, a new script
	Edit	Edit the selected element. Usually a pop-up window is opened.	Edit an engineer, a class of text, a ticket field group
	Delete	Delete the selected element from the database. Cannot be restored.	Delete an engineer, a customer field group, a resource
	Copy	Copy the selected element	Copy a role
	Paste	Available for ticket, customer and resource fields	Paste the copied field into the selected (target) field group
	Activate	(Re-) activate an element which had been deactivated before.	(Re-) activate an engineer whose account had been deactivated.
	Deactivate	Deactivate the selected element. Might be safer than deleting it. Elements which are in use cannot be deleted, so it can be an alternative to deactivate them.	Deactivate an engineer (account), e.g. when an employee takes a sabbatical. Deactivate a customer who has canceled the contract.
	Move upwards	Move an element one step upwards in a list. This might have implications for the Web Client.	Move a view two steps upwards in the list. The view will then be displayed in the view list in the Web Client at the new position.
	Move downwards	Move an element one step downwards in a list. This might have implications for the Web Client.	Move a ticket field group one step downwards in the list. It will then be displayed in the ticket data section, maybe in the group section, at the new position.
	Unassign	Unassign/remove an element from the selected item.	Unassign a role from the selected engineer.
	Assign	Assign an element to the selected item.	Assign a role to an engineer. Assign a view to a role.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Icon	Name	Meaning/Function	Examples
	Annotate	Open the *Annotation* pop-up window	Used for ticket field groups, ticket fields, customer field groups, customer fields, resource field groups and resource fields
	Localize / Internationalize	Open the pop-up window to enter the localized / internationalized names of the technical objects. The languages which have been configured in the Admin Tool are offered.	Localize the name of a ticket field.
	Search	Open the Search GUI.	Start the ticket search.
	Select all	Marks all elements (often checkboxes), no database action is performed, only GUI helper	Mark all permissions for a role concerning queue access
	Deselect all	Deselects all elements (often checkboxes), no database action is performed, only GUI helper	Deselect all permissions for a role concerning queue access
	Start	Start the selected element (usually a service)	Start a CM service
	Stop	Stop the selected element (usually a service)	Stop a CM service
	Upload	Open the file browser to upload a file to the CM system	Upload a script
	Download / save	Save a file on the file system	Save a script as file in the file system.
	Save and close	Saves the element (usually a script) and closes the editor in edit mode. Switches to view mode.	Save a script in the script section and switch to view mode of the script.
	Close without saving	Does not save the element (usually a script) and closes the editor in edit mode. Switches to view mode.	Do not save the edited script in the script section and switch to view mode of the script.
	Refresh	Updates the data in the Admin Tool.	Update the data in the Admin Tool after making changes in the Process Designer.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

0.12.5 Localization of Terms Displayed in the Web Client

ConSol CM can be configured for international environments. In this way, every engineer can work with the country-specific language in the web browser. The following principles are important concerning localization/internationalization:

The CM administrator configures the languages which will be available in the entire system. This is done using the Admin Tool and is explained in section Languages.

The languages which are configured as available languages in the Admin Tool can then be applied for all terms which have been configured for the CM system, e.g., the terms for ticket fields. Using the Process Designer, the labels for workflow activities can also be localized. This is explained in the *ConSol CM Process Designer Manual*. Thus, there are two different categories of terms in the Web Client:

1. **Standard CM terms/labels** which cannot be modified using the Admin Tool or Process Designer (however, they might have been adapted to your system in case a customer specific software has been developed). Terms like the labels for the group headers in the ticket list (e.g., *Own tickets*, *Workgroup tickets*, *Unassigned tickets*) or the labels *Favorites* and *Workspace* belong to this category.
   For these terms/labels, two standard languages are available: English and German. English is also the overall standard language. You cannot modify standard CM terms! They are available in English and in German and represent a fixed set of terms and labels.
2. **System-specific terms, i.e., terms/labels which can be localized using the Admin Tool** or the Process Designer. Localization using the Admin Tool will be explained in the following section. For an explanation of localization using the Process Designer, please refer to the *ConSol CM Process Designer Manual*.
   For system-specific fields, the localized terms/labels have to be entered manually for each language.

In the Web Clients (CM Web Client and CM/Track), the labels are displayed according to the browser locale which has been set by the engineer or the customer.

0.12.5.1 The Different Modes of Localizing Terms and Labels Using the Admin Tool

General Principle of Localization in CM

Depending on the location in the Admin Tool, there are different GUI elements which are used to set localized values. However, the general principle is valid for all those locations. The general principle works as follows (explained for the example of a ticket field).

Technical name:

Each field or object has a technical name, e.g. the ticket field *priority*. This name is set when the field or object is created and should not be changed afterward because it might be used in Admin Tool or workflow scripts which would fail when the name has been modified. CM throws a warning message and blocks the action when someone tries to modify a technical field or object name. Only when the field or object is new and has not been used yet, it is possible to change the technical name.

Localized names:

For each field, field group or object a localized name can be set for each of the languages which have been configured for the CM system (using the settings in the navigation group *Global Configuration*, navigation item *Languages*).

For example, the ticket field with the technical name *priority* is named *HD Priority* in English and *HD Priorität* in German.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

In the web clients (CM Web Client and CM/Track), the labels of the fields and objects are displayed according to the browser locale of the engineer or customer. Some variants are possible:

1. Variant #1:
   The language of the browser locale has been configured and all labels and terms are displayed in this language. For example, the language *French* has been added in the Admin Tool, all system-specific terms have been localized manually for French (using the Admin Tool and the Process Designer). Then all system-specific terms are displayed in French, e.g., the workflow activities, the names of the ticket fields, and the names of the views.
2. Variant #2:
   The language of the browser locale has been configured, but not all labels and terms are displayed in this language. For example, the language *French* has been added in the Admin Tool, but only some system-specific terms have been localized manually for French (using the Admin Tool and the Process Designer). Then the localized system-specific terms are displayed in French, e.g., the workflow activities, the names of the ticket fields, and the names of the views. All non-localized terms (i.e. where the field for the localized term is empty) are displayed in the language which has been configured as default in the *Languages* settings.
3. Variant #3:
   The language of the browser locale has not been configured. For example, the localized terms and labels have been set for English, German and French, but the engineer has set the browser locale to ES (Spanish). In this case, all labels and terms are displayed in the language which has been configured as default in the *Languages* settings.

Localization of Data Fields

The localized terms for data fields are written directly into the *Labels* section in the Admin Tool. This applies to

- Ticket fields (for ticket data)
- Customer fields (for customer data)
- Resource fields (for resource data)

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

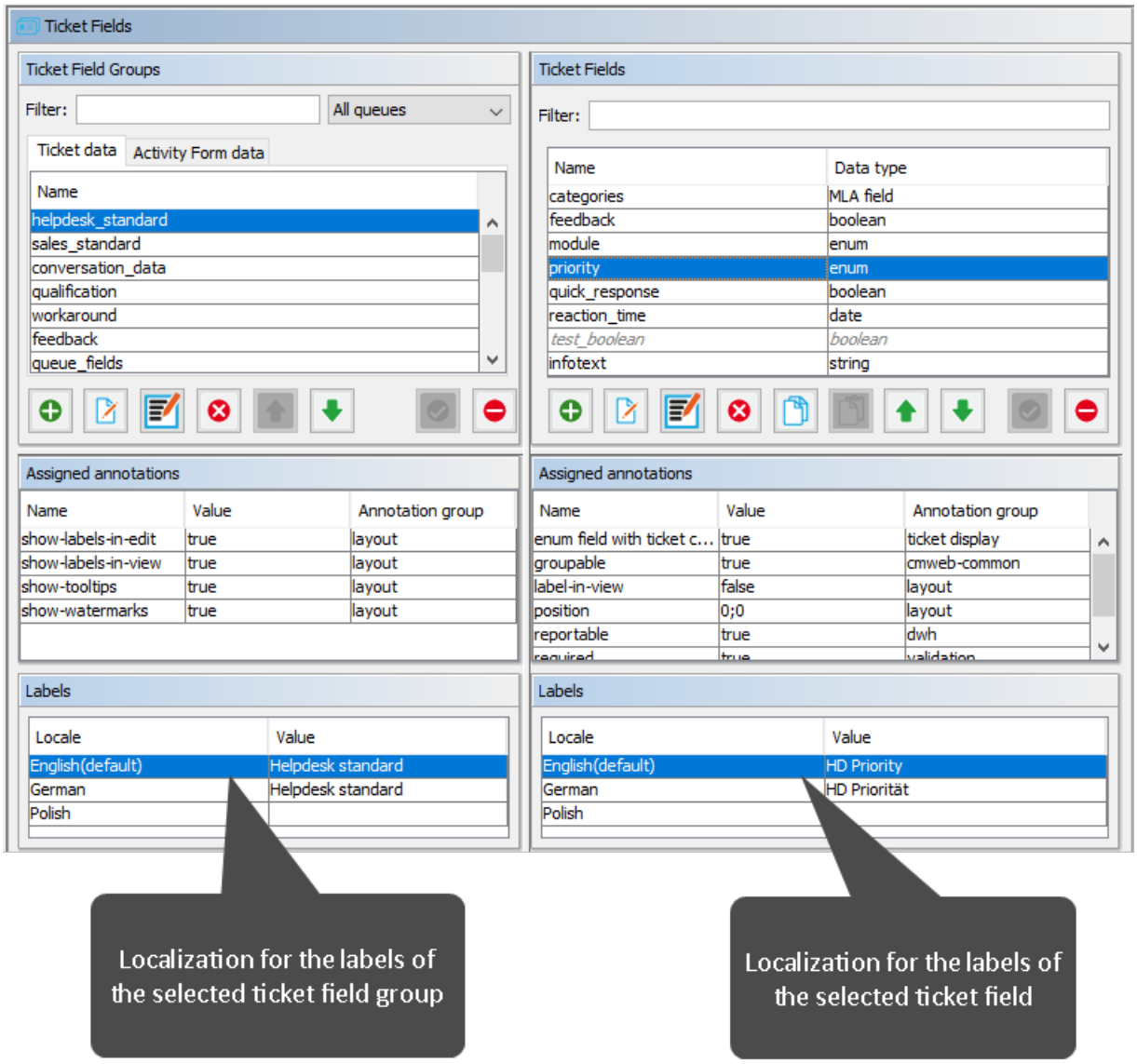


Figure 10: *ConSol CM Admin Tool - Localization of ticket fields*

In order to change the value of a field, click into the field and write the new term. Press ENTER to save the new value.

In the Web Client, the ticket field shown in the figure above is displayed as follows (in English and German):

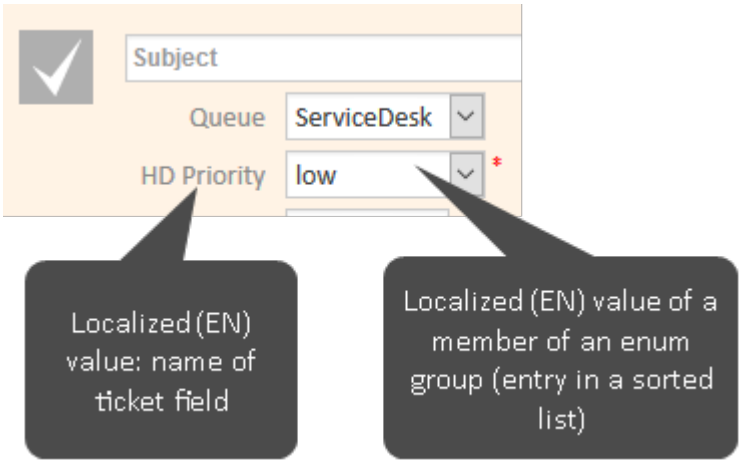


Figure 11: *ConSol CM Web Client - Ticket field displayed in English browser locale*

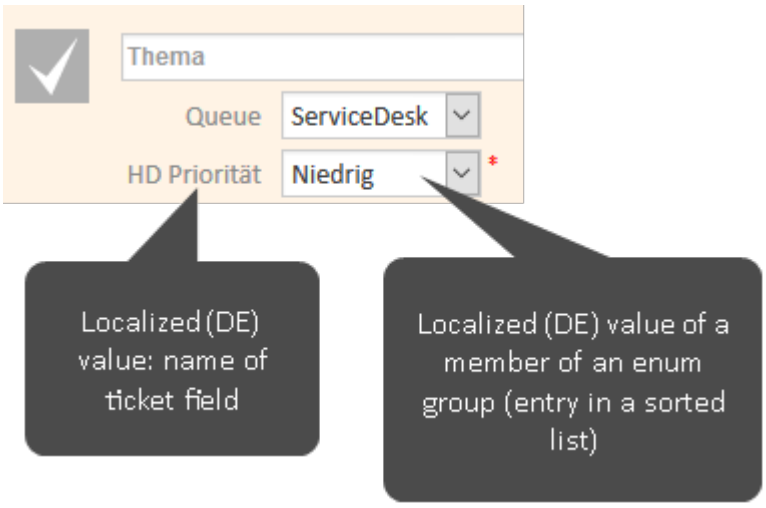


Figure 12: *ConSol CM Web Client - Ticket field displayed in German browser locale*

Localization of Objects in General, Type 1

A great number of objects can be localized using the *Localize / Internationalize* (globe icon) button which opens a pop-up window with the localization configuration.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

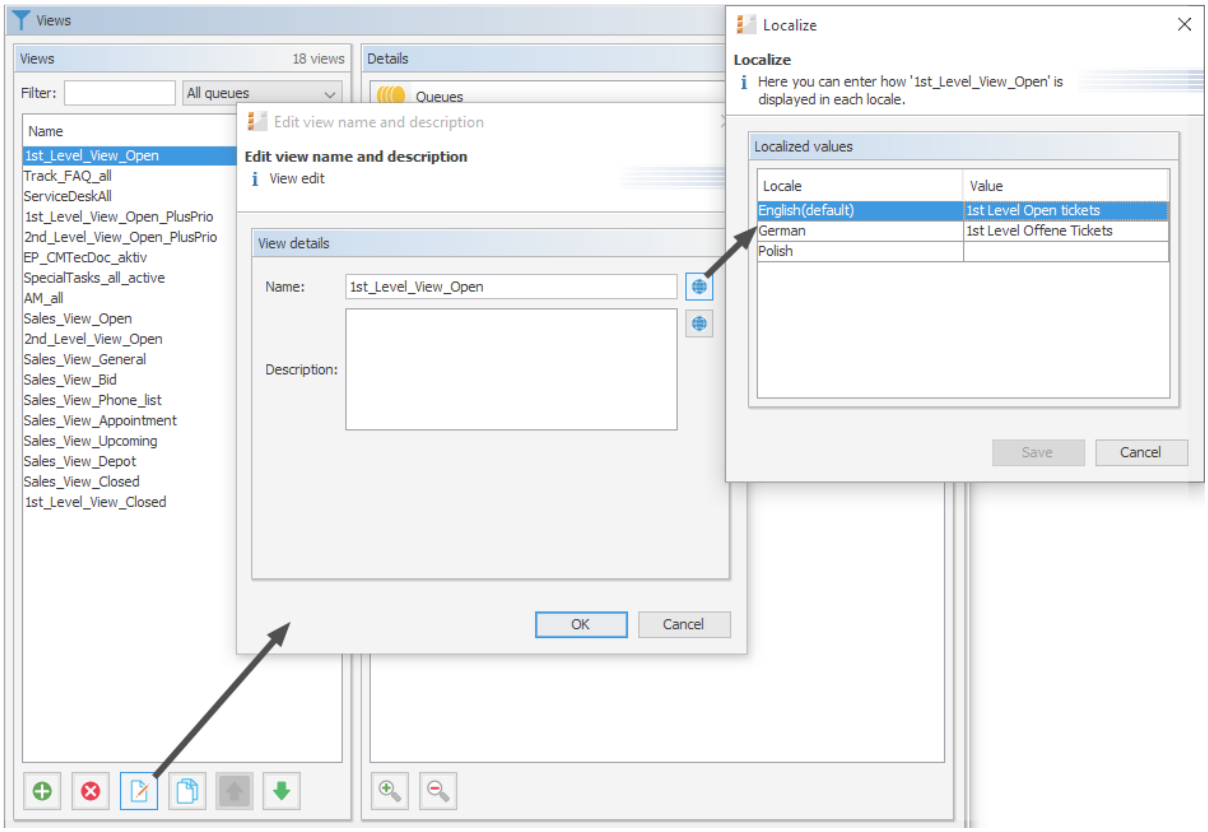


Figure 13: *ConSol CM Admin Tool - Localization of objects, example: name of a view*

In order to change the value of a field, click into the field and write the new term. Press ENTER to save the new value.

Localization of Objects in General, Type 2

Some objects are localized using the localization table within the pop-up window which is used to edit the object.

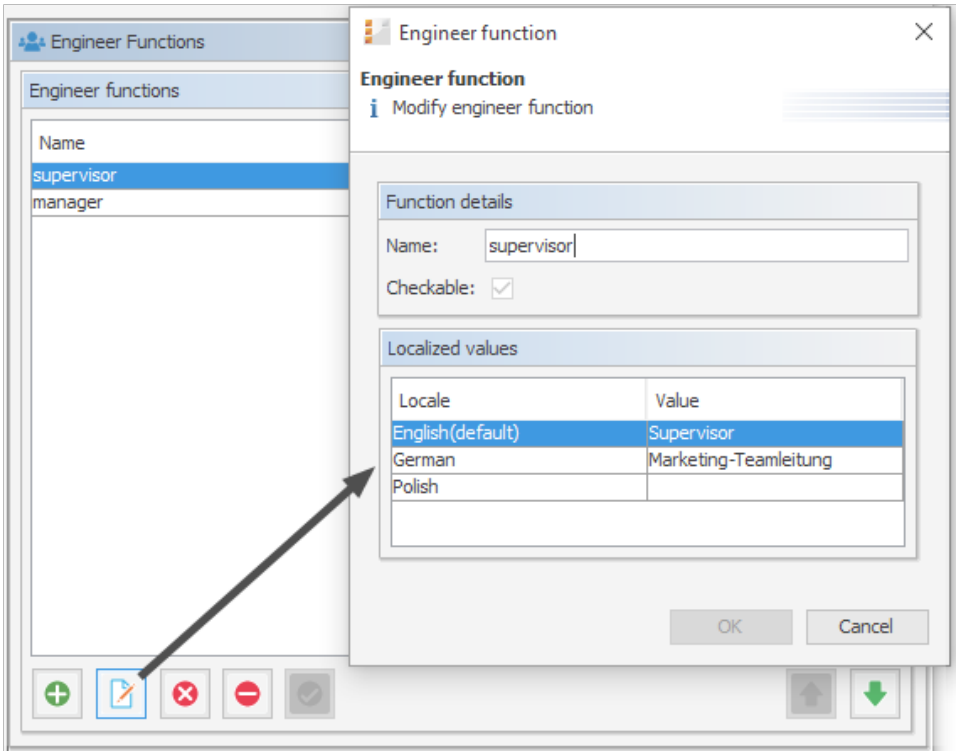


Figure 14: *ConSol CM Admin Tool - Localization of objects, example: name of an engineer function*

Using UTF-8 Symbols in Localized Fields

In order to have graphical symbols displayed in text fields, you can use UTF-8 symbols. This works for all localized fields, e.g., string fields, queue names, or enums. Just copy the UTF-8 symbol from a unicode table page and paste it into the text field in the Admin Tool.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Example:

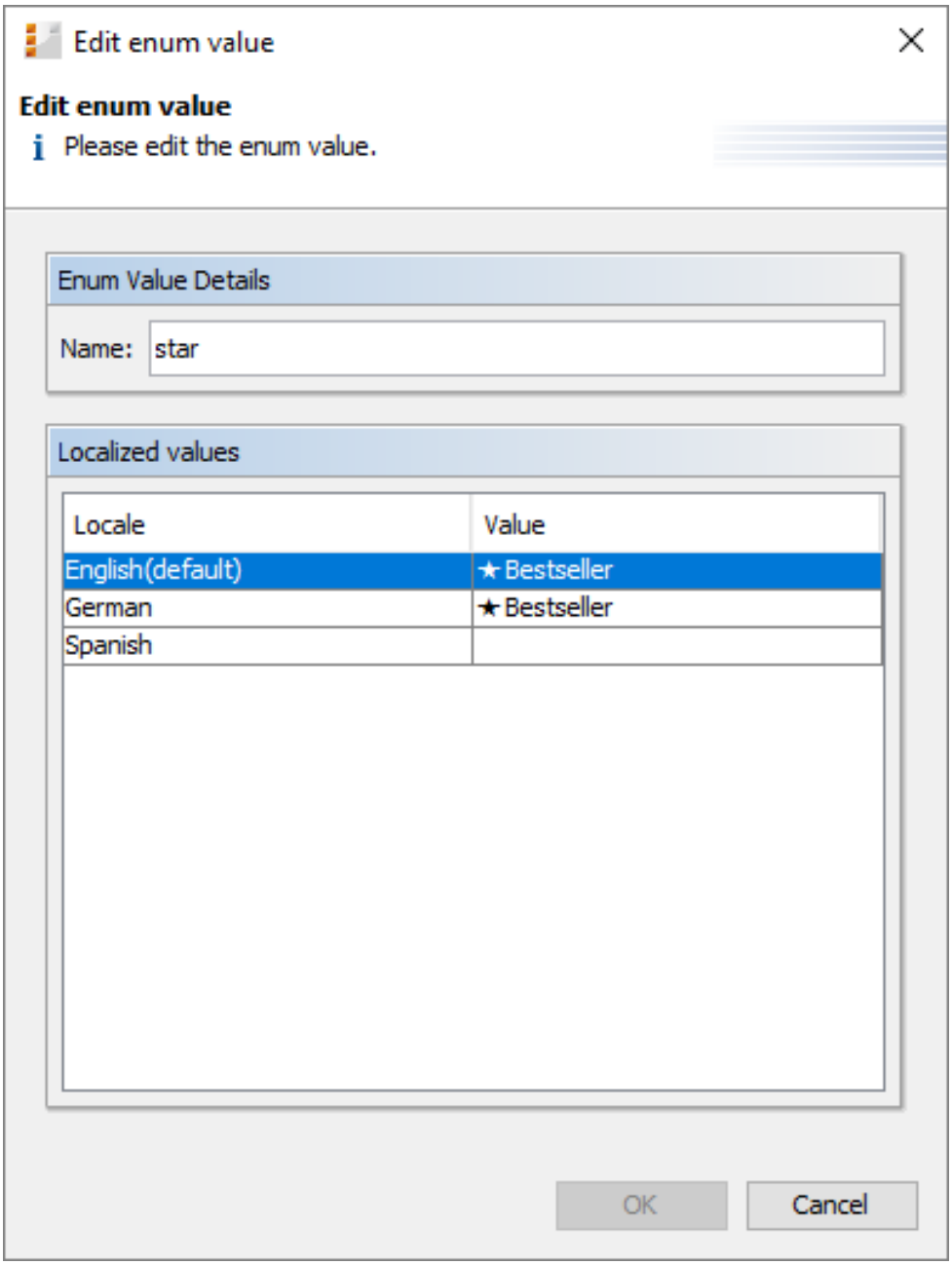


Figure 15: *UTF-8 symbol in a localized field in the Admin Tool*

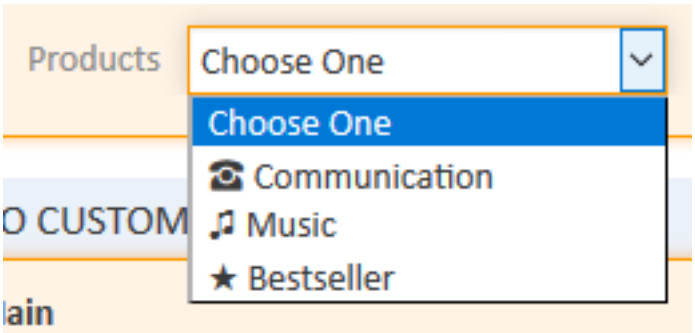


Figure 16: *Display of UTF-8 graphical symbols in the Web Client, here: enum*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

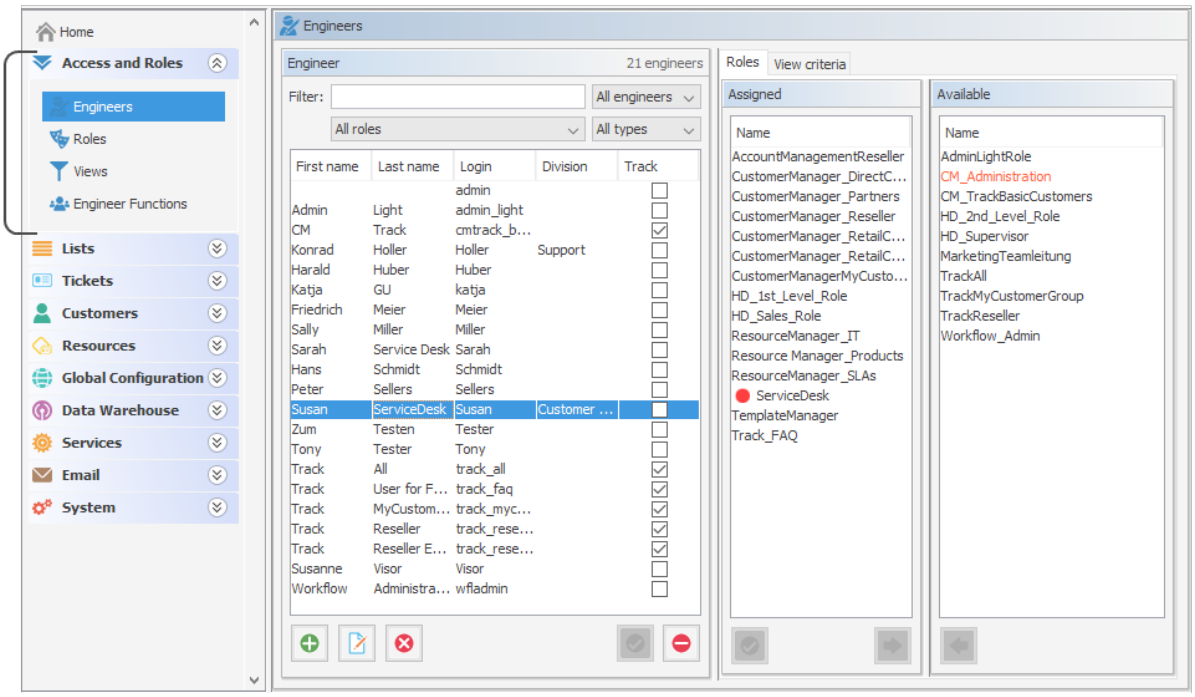
In case you work with a MySQL database, use the following settings to work with the UTF-8 graphical symbols:

- CHARACTER SET = utf8mb4
- COLLATE = utf8mb4_unicode_ci

Localization of Labels

Several terms in the Web Client can be localized using labels. This is explained in section Labels.

A - Access and Roles



This section provides some background knowledge about the general principle of ConSol CM access permissions and shows you how to define the engineer accounts with the required parameters.

Please read the following sections:

- Engineers
- Roles
- Views
- Engineer Functions

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

A.1 Engineers

This chapter discusses the following:

A.1.1 Introduction to Engineer Administration 49

A.1.2 Engineer Administration Using the Admin Tool 50

A.1.1 Introduction to Engineer Administration

An engineer account is the basic access object which allows an engineer or administrator to access the Web Client, Admin Tool, or Process Designer. During system set-up an administrator account for the first access to the Admin Tool is created. Using this account, you can set up further accounts.

Newly created engineer accounts do not have any permissions. These permissions have to be assigned through one or more roles displayed in the *Roles* tab. If you have not created any roles yet, you only see the administrator role (see Tab Roles - Assign Roles to an Engineer Account).

Views define which tickets engineers will see in the ticket list (to-do list) of the Web Client. They are created in the Views and assigned via roles. On the engineer administration page you can preset dynamic view criteria for specific engineers (see Tab View Criteria - Define Engineer-Specific View Criteria).



We would recommend that you create at least one role and one view first before you create any engineer accounts.

A.1.2 Engineer Administration Using the Admin Tool

To open the Engineer Administration, open the navigation group *Access and Roles* in the Admin Tool and click the navigation item *Engineers*.

A.1.2.1 List of All Engineers

When you have opened the Engineer Administration, a list of all engineers is displayed. Engineer accounts which are currently deactivated are displayed in gray. You can easily narrow down the list of engineers to engineers of only one role (for details, please refer to section Roles) by using the drop-down menu (filter) for roles.

A.1.2.2 Create or Edit an Engineer Account

You can create a new engineer account or edit the settings of an existing account. Both actions are performed in the same pop-up window. To create a new engineer account, click the *Add* icon below the list of engineers. You can open the window to edit an existing engineer account by either double-clicking the respective entry in the list of engineers or by clicking the *Edit* icon.

The window shows the parameters describing an engineer account:

- **Login:**
  Mandatory. This field contains the account name which has to be entered on the login page of the Web Client. Please use only international alphabetic and numeric characters, no blanks, punctuation marks, or special characters such as umlauts, hyphens, or the like.
- **First name:**
  Optional. The engineer's first name. This field is optional but will be displayed in the Web Client for the engineer. The entry may contain alphabetic characters, blanks, comma, periods, and hyphens. Please do not use other characters.
- **Last name:**
  Optional. The engineer's last name. This field is optional but will be displayed in the Web Client for the engineer. The entry may contain alphabetic characters, blanks, comma, periods, and hyphens. Please do not use other characters.
- **Email:**
  Mandatory. The engineer's email address. Please use only international alphabetic and numeric characters, hyphens, underscores, periods, and the @ sign. The entry of multiple email addresses in one line is not allowed.
- **Position:**
  Optional. The engineer's position or function in the company. This field is optional and has a descriptive function only. The entry may contain alphabetic characters, blanks, comma, periods, and hyphens. Please do not use other characters.
- **Company:**
  Optional. The engineer's company. This field is optional and has a descriptive function at the moment. The entry may contain alphabetic characters, blanks, comma, periods, and hyphens. Please do not use other characters.
- **Division:**
  Optional. The division in which the engineer works. This field is optional and has a descriptive function. The entry may contain alphabetic characters, blanks, comma, periods, and hyphens. Please do not use other characters.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Description:**
  Optional. An additional description for the engineer account. This field is optional and will **not** be displayed in the Web Client. The entry may contain alphabetic characters, blanks, comma, periods, and hyphens. Please do not use other characters.
- **Phone:**
  Optional. The engineer's phone number. This field is optional and has a descriptive function at the moment.
- **Mobile:**
  Optional. The engineer's mobile phone number. This field is optional and has a descriptive function at the moment.
- **Fax:**
  Optional. The engineer's fax number. This field is optional and has a descriptive function at the moment.

> ⚠ Several fields which contain engineer data (like *Company*, *Division*, or *Phone*) are optional fields. However, if you work with text templates which contain engineer data fields (see section Text Templates) the emails or comments will not be formed correctly if the data is missing. For example, the field *ticket-engineer, phone* cannot be filled-in in the template if it has not been set for the engineer in Engineer Administration! So please make sure all data which will be required later on is filled-in correctly in the first place!

- **LDAP ID**
  The LDAP user ID if LDAP is used for authentication. No password has to be set here.

> ⓘ If you do not enter an LDAP ID here, the login will be used as authentication login parameter for the LDAP server (if LDAP authentication is activated)!
>
> For detailed information about the LDAP authentication, please see section LDAP Authentication for Engineers in the Web Client.

- **Kerberos Principal Name**
  The Kerberos principal name if Kerberos V5 protocol is used for authentication. Engineers can log in to the Web Client by using their Windows credentials. For detailed information about Single Sign-On with Kerberos, please see section Single Sign-On in a Windows Domain Using Kerberos .
- **Password:**
  Mandatory. The engineer's password is mandatory. Please use only international alphabetic and numeric characters, and punctuation marks, do **not** use any special characters such as, e.g., umlauts. The password entered will be shown as a string of asterisks. Please see section Configuring the Password Policy for information about the optional password policy.
- **Password (again):**
  Mandatory. Please repeat the password here. This security query helps to avoid erroneous entries which would not be noticed otherwise because the password is shown as a string of asterisks. Please see section Configuring the Password Policy for information about the optional password policy.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

> This field will only appear if the engineer authenticates against the Web Client via the CM database, i.e., if LDAP or Kerberos authentication is used, the field is not visible.

- **Track user**
  This checkbox has to be ticked if you want to create a technical engineer (or CM/Track user profile) used to define access permissions for CM/Track users. The available CM/Track users (user profiles) are shown in the Web Client when creating or modifying a customer. So, by ticking this checkbox, you do not define a real engineer (a person) with access permissions to the system but rather a user profile for CM/Track which is then assigned to one or more customers who should access the portal ***CM/Track*** using those access permissions. For a detailed description of the CM/Track access definition see also section CM/Track: System Access for CM/Track Users (Customers).

Click *Save* afterwards to store your entries and to close the window.

A.1.2.3 Delete an Engineer Account

To delete an engineer account, select the account in the list and click the *Delete* button. Since an engineer account can only be deleted if there are no tickets (open or closed) for it anymore, you have to assign its tickets to another engineer. The name of the deleted engineer is still displayed in all history entries in tickets and customer pages which were performed by this engineer.

In case you do not want to transfer any tickets to another engineer, you can deactivate the engineer account. See next section.

A.1.2.4 Disable or Enable an Engineer Account

If engineers should not have access to the system for a certain period of time (e.g., because they have taken a sabbatical), an account can be disabled. There will be no change regarding the tickets of these engineers, but they cannot log in anymore and other engineers cannot assign any tickets to their accounts.

To disable an engineer account, select the account and click the *Deactivate* button. The entry in the list is shown in gray italics afterwards. It is not possible to create new tickets or to edit existing tickets for this account. To re-enable the account, just click the *Activate* button at the bottom of the page.

> If you try to disable an engineer who is assigned as a main engineer to open or closed tickets, or who is currently representing another engineer, a warning message is displayed. Any representations are deleted upon confirming the warning message.

A.1.2.5 Tab *Roles* - Assign Roles to an Engineer Account

On this tab you can assign roles to an engineer account. Select the account on the left and then the desired role(s) in the list of available roles on the right. Click the *Assign* button to move the selected roles into the list of assigned roles. Now an engineer with this account can act in the system according to the permissions set in the role(s) (see also Roles).

Set Roles as Main Roles

From the list of assigned roles you can choose one role as the main role for each engineer account. Select the desired role in the list and click the *Activate* button below the list. Afterwards the main role is marked with a red dot. Now the views of the main role always appear at the top of the view list in the Web Client for this engineer account.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Edit One of the Roles of an Engineer

Each role in the list of roles of an engineer offers a context menu with two items:

- **(Un-)Assign role**
  You can use this menu item to (un-)assign a role to/from an engineer. It has exactly the same effect as using the arrow buttons below the list.
- **Jump to role**
  You can use this menu item as a shortcut to quickly open the navigation element *Roles* of this role (see Roles).

A.1.2.6 Tab *View Criteria* - Define Engineer-Specific View Criteria

Here you can change the dynamic view criteria for an engineer. Dynamic criteria are used to give the engineer the possibility to adjust a view interactively in the Web Client (see also Views).

> This tab only shows view criteria if you have created a view with dynamic criteria and assigned it to the engineer's role.

Select the engineer account on the left and then the desired criterion in the list of available view criteria on the right. Click the *Assign* button to move it to the list of assigned view criteria. You will see the possible values below the criterion in the list. Tick the checkboxes of the values you want to change or preset. The engineer can change these settings in the Web Client (profile page) and changes you have made in the Admin Tool are immediately visible in the engineer's profile page.

Example

You have assigned the dynamic criterion *priority*. The list shows the values “Not set”, “low”, “normal”, and “high”. If you tick the values “normal” and “high” the engineer will only see tickets with normal and high priority after logging in to the Web Client. If you do not tick any values the engineer will see no tickets for this view. See section Views for details.

> Please note that in a view with a dynamic criterion, only the tickets are displayed which match this criterion. So if engineers have not selected any criteria in their engineer profiles or if the administrator has removed all selections using the Admin Tool, the engineer's view will be empty! Make sure your users know about this fact and make sure you as an administrator are always aware of that fact.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

A.2 Roles

This chapter discusses the following:

A.2.1 Introduction to Role Administration 54

A.2.2 Role Administration Using the Admin Tool 55

A.2.3 Defining Additional Administrator Roles 71

A.2.1 Introduction to Role Administration

Roles provide access rights and views, they specify what an engineer is allowed to do or to see. Without a role, an engineer can log in to the system but cannot perform any actions. Only by being assigned one or more role(s) does an engineer obtain system permissions. For each task in a company using the system there should be a role which defines its permissions. Engineers fulfilling the task should have this role.



When engineers log in to the system, they will have all permissions from all roles they have been assigned. So all permissions are added! There is no way of explicitly preventing access to objects in ConSol CM - access can only be **granted**! The sum of all granted permissions defines the final permissions for the engineer.

Roles define:

- **Access permissions for one or more queue(s)**
  E.g. read, write, and append rights are granted. The permissions are valid for all tickets in the queue(s).
- **Global permissions**
  Several system-wide permissions are managed here, e.g., the rights concerning template management, workflow design, and system administration. Using the option *Administrate access and roles*, it is possible to define an administrator "light" who can manage CM engineers with their system access permissions but who cannot modify technical system-wide settings. This is explained in section Defining Additional Administrator Roles.
- **Access permissions for customer data**
  Read, write, modify, and delete permissions for each distinct customer group.
- **Access permissions for resource data**
  Read, write, modify, create permissions, assigned on the basis of resource types.
- **Views**
  To do lists of tickets which are displayed in the ticket list in the Web Client.
- **Engineer functions**
  Additional engineer functions which can be assigned to members of this role, e.g., *approver.*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

A.2.2 Role Administration Using the Admin Tool

You reach this screen by opening the navigation item *Roles* in the navigation group *Access and Roles*.

You might also see this screen when you have worked in the Engineers and have selected *Jump to role* for a role in the role list of an engineer.

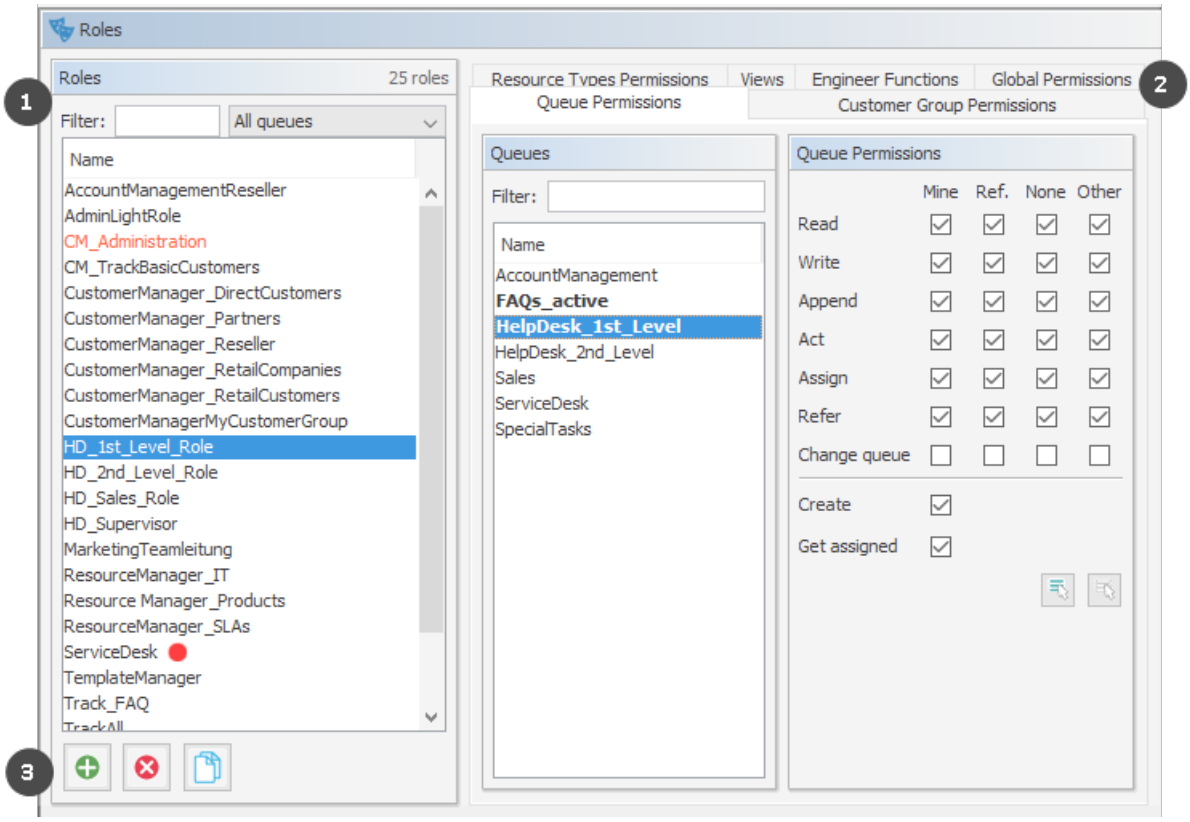


Figure 17: *ConSol CM Admin Tool - Access and Roles, Roles: Queue permissions*

You see the list of all available roles on the left-hand side (1) and the permissions which can be granted on the right-hand side (2). In the list of roles, all roles which have been set as *main role* for at least one engineer are marked with a red dot. You always work on the access permissions of the role which has been selected in the list of roles. Only one role can be selected at a time. You can use the buttons below the list of roles (3) to add, delete or copy a role.

On the right-hand side, several tabs are available:

- Tab Queue Permissions
- Tab Global Permissions
- Tab Customer Group Permissions
- Tab Resource Types Permissions
- Tab Views
- Tab Engineer Functions

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH



All changes in the *Role Management* tabs take effect immediately or after clicking the *OK* button. You do not have to click the *Synchronize* button in the icon bar.

In the Web Client, engineers have to log in again to use their new roles. Views become effective after pressing F5 (page refresh).

Working in Role Administration, you always mark a role and then can display and modify the parameters of this role. However, it is not possible to display a list of all engineers who have been assigned this role. In order to have such a list displayed, please change to Engineers (navigation item *Engineers*) and filter the engineer list for a certain role.

Please note that you can edit the name of a role by clicking on it in the list of roles and by modifying the name as required.

A.2.2.1 Create a Role

Click the *Add* button below the role list to create a new role. A pop-up window appears where you can enter the role name. Since the role name is used only for admin purposes and not displayed in the Web Client, no localization is required here. Afterwards you have to set the permissions of this role using the tabs on the right side of the page (see also the preceding picture).

Tab *Queue Permissions*

The permissions set in this tab apply to the selected role (left part of page) and the selected queue (center part of page). Without an entry here, an engineer with this role is not able to see tickets nor to perform any actions in the system.

You reach this screen by clicking the *Queue Permissions* tab in the navigation item *Roles*, navigation group *Access and Roles*.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

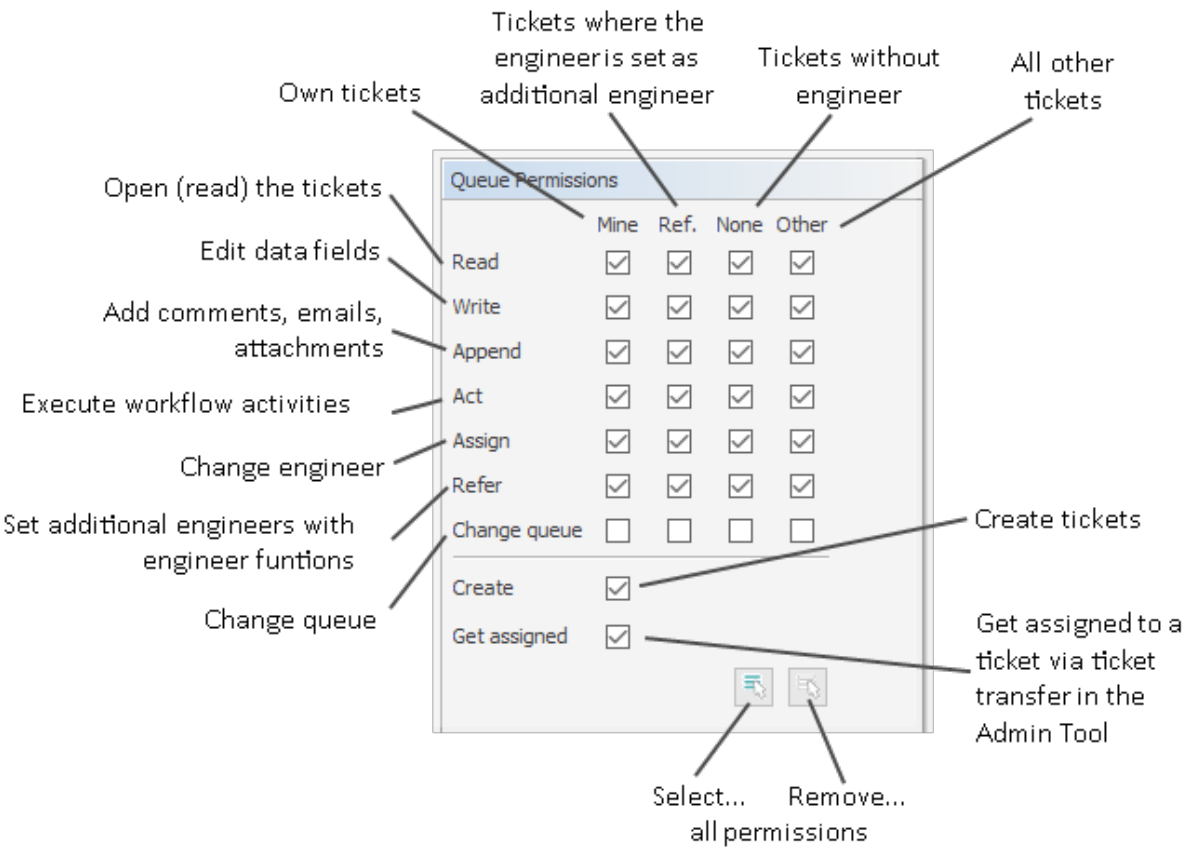


Figure 18: *ConSol CM Admin Tool - Access and Roles, Roles: Setting queue permissions*

The following permissions can be set:

- **Read**
  Read tickets.
- **Write**
  Edit data fields (default fields, ticket fields, etc.) of a ticket. The fields might be located in the ticket header section or in the group section.
- **Append**
  Add information to a ticket (comments, emails, attachments, time booking entries), i.e. add content in the ticket history.
- **Act**
  Execute workflow activities, i.e. move the ticket forward in the workflow.
- **Assign**
  Assign tickets to another engineer. The permission to assign a ticket to oneself and to accept a ticket is not relevant in this context.
  The engineer who should receive the ticket has to have at least one role with the *Get assigned* permission!
- **Refer**
  Assign an additional engineer (with engineer function, see Tab Engineer Functions) for a ticket.
- **Change queue**
  Move a ticket from this queue to another queue.
  If the current engineer has the *Change queue* permission for the respective range of tickets

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

(mine, referenced etc., see below), the pull-down menu where the queue can be changed is displayed. All queues where the current engineer has the *Change queue* permission are listed. This means, the engineer needs the *Change queue* permission in the source as well as in the target queue. However, the engineer can only perform the entire operation if he has the permission for the right range of tickets in the target queue. For example, if an engineer wants to move a ticket which is assigned to himself to another queue, the minimum permission which is required for this operation is the *Change queue* permission for the range *mine* in the target queue. If another range was set (e.g. *referenced* or *none*), the operation would not succeed, because the ticket engineer is not changed during the operation and thus the suitable range has to be used.

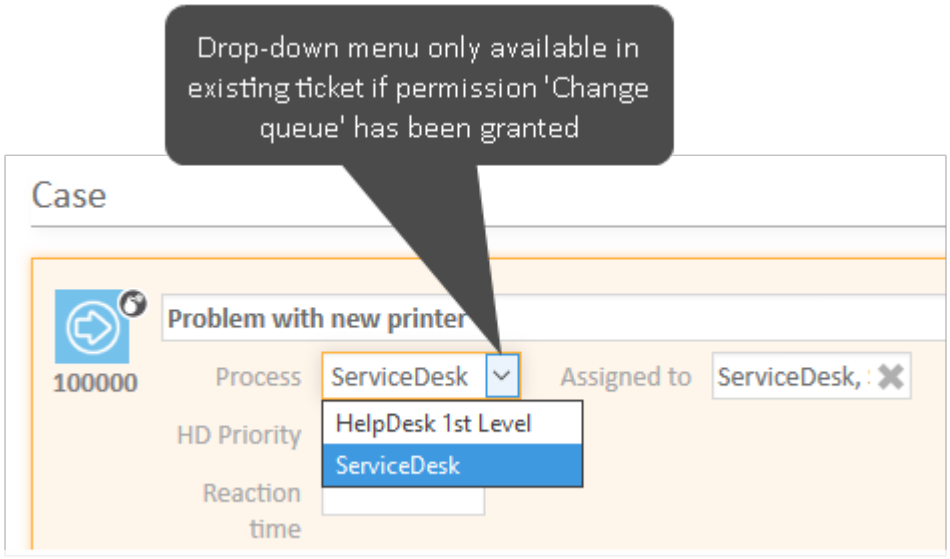


Figure 19: *ConSol CM Web Client - Drop-down menu to change queue*

It depends on the workflow of the target queue where the processing of the ticket will continue:

- If the source queue and the target queue have the same workflow, the ticket will start its processing in the target queue at the original position (i.e., its last position in the source queue).
- If the source queue and the target queue have different workflows, the ticket will start the process in the target queue at the START node.

> ⚠ Be very careful when granting the *Change queue* permission!!! Usually it is not required. On the contrary, it can destroy your process chain definition where tickets are passed from one process to another using process/workflow components, namely the Jump-in and Jump-out nodes.
>
> This permission should only be granted if it is absolutely necessary and when all side-effects have been considered thoroughly!

You can define for which range of tickets the permissions are valid:

- **Mine**
  Own tickets.

- **Ref**
  Tickets to which the engineer is assigned as an additional engineer (with engineer function, see Tab Engineer Functions).
- **None**
  Tickets without assigned engineer.
- **Other**
  Tickets assigned to other engineers.

Click the corresponding checkbox to assign one or more permissions for the desired ticket range.

Two general permissions can also be set:

- **Create**
  An engineer is allowed to create tickets in this queue.
- **Get assigned**
  Other engineers can assign tickets to an engineer who has a role with this permission (if the other engineers have the *Assign* permission!)
  An engineer can receive tickets by ticket transfer which is performed using the Admin Tool.

If you want to select all permissions simultaneously just click the *Select all* button below the list. Clicking *Deselect all* removes all selections.

Tab *Global Permissions*

Global permissions are general and queue-independent rights for a role. Setting these permissions is optional.

You reach this screen by clicking the *Global Permissions* tab in the navigation item *Roles*, navigation group *Access and Roles*.

You can specify the following:

- **Administrator Permissions**
  - **Administrate system (full access)**
    Provides administrator access to the entire CM system, this applies to the Admin Tool, the Process Designer, and admin access to the Web Client. An administrator with this role has access to all navigation groups and items in the Admin Tool.
  - **Administrate system (configuration only)**
    Provides access to the complete system configuration in the Admin Tool and Process Designer, and the page customization, text templates and web forms in the Web Client. An administrator with this role does not have access to runtime data (runtime data in the Web Client and navigation item *Administration* in the navigation group *Tickets* in the Admin Tool).
  - **Administrate access and roles**
    Provides administrator access only to the navigation group *Access and Roles*.

> Please see Defining Additional Administrator Roles for detailed information about the impact of the different administrator permissions.

- **Workflow Permissions**
  Provides permissions concerning workflow design and management. These are
  - **Read**
  - **Write** (modify and store)
  - **Deploy** (install and put in operation).
- **Template Permissions**
  - **Write Template** provides the permission
    - to use the Text Template Manager, which is used to create and edit email and comment templates. See section Text Templates for details.
    - to use the Document Template Manager, which is required to define templates for CM/Doc. Only available if CM/Doc is active in the CM system.
- **Representation Permissions**
  - **Configure representation**
    If this permission is set, engineers with this role can configure themselves as a representation for other engineers, e.g., who are ill and have not defined other engineers to represent them resp. if the defined engineers are not available at the moment. On the Web Client the engineers that can be represented by an engineer with this permission are shown in a list within the engineer profile.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Important information about representation configurations

Please note that there are two different scenarios for sending emails and that the CM system behavior concerning sending representation mails might differ for the two scenarios!

- **An engineer writes an email using the Ticket Email Editor**

  It depends on the value of the property cmweb-server-adapter, forward.mails.to.representatives if the representation rule is applied and the representing engineer receives a copy of the email. By default, this property is set to "false", meaning that this email is **not** sent to the representing engineer. If the property is set to "true", all emails which are sent manually using CM are sent to the original recipient **and** his current representative. The CM system checks if a representation rule is active for the respective (recipient) email address. Please keep this in mind when you configure the representation permissions in the Admin Tool and inform your CM users (engineers) about this behavior! It might lead to unwanted effects, especially when persons are registered as engineers and as contacts in the ConSol CM system (e.g., for an internal help desk).

- **An email is sent automatically from the CM system**

  It depends on the specific configuration of the CM system which engineers receive a copy of the email, the email is (!) not sent to the representing engineers automatically!
  It might be implemented that the representing engineer gets a copy, but this is not mandatory. The automatic email might be sent from a workflow script or from an Admin Tool script (which might also be called from a workflow). It depends on the implementation in this script who receives a copy of the email. For details, please refer to the *ConSol CM Process Designer Manual*.

- **Track User Permissions**
  - **Access tickets of the own company**
    Users with this permission are allowed to access not only their own tickets in CM/Track, but all tickets of the company they belong to. This permission makes only sense for roles that define access rights of CM/Track users/user profiles, not for single users.
- **Archive Permissions**
  Provides permissions for CM/Archive:
  - **Read**
    View tickets in CM/Archive
  - **Write**
    Archive tickets
  - **Delete**
    Remove tickets from CM/Archive
  - **Admin**
    Display statistics in CM/Archive

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

> ⚠ The engineers also need read permissions to the queues which the archived tickets belong to. Any of the read permissions for the queue (*Mine*, *Ref.*, *None*, or *Other*) is sufficient to view tickets from the queue in CM/Archive.

- **Analytics Permissions:**
  Provides permissions for CM/EBIA:
  - **EBIA full access**
    Users with this permission see the menu item *Analytics* for CM/EBIA in the Web Client. They are allowed to ask questions, create dashboards and explore data in CM/EBIA.

Tab *Customer Group Permissions*

In order to let engineers work with customer data from one or more customer groups, e.g. to edit reseller data sets or to create new contact data within the customer group, you have to grant access permissions concerning the customer group(s) to one or more roles.

You reach this screen by clicking the *Customer Group Permissions* tab in the navigation item *Roles*, navigation group *Access and Roles*.

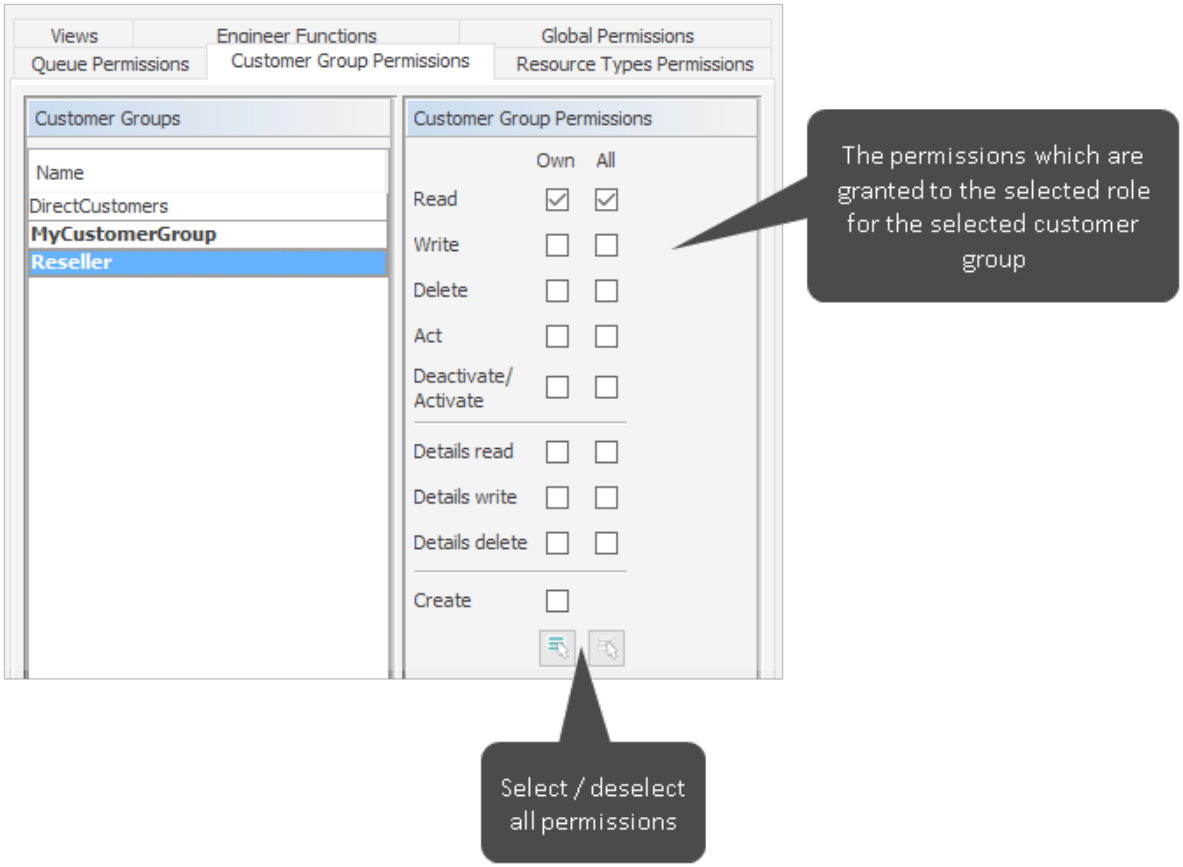


Figure 20: *ConSol CM Admin Tool - Access and Roles, Roles: Assigning permissions for customer groups to a role*

A concept which has proven very useful in various customer environments is the set-up of specific roles for customer data management. For example, there could be a role *CustomerManager_CustomerGroup1* and another role *CustomerManager_CustomerGroup2*. You can even differentiate between CustomerManager_CustomerGroup1_full and CustomerManager_CustomerGroup1 _light. In this way, you can use the assignment of the customer manager roles as a toggle and you do not mix up queue access permissions and customer management permissions. This can be very helpful in case you have a heterogeneous team in which not everyone is allowed to edit the complete customer data.

However, do not forget to grant read permissions to customer data of the required customer groups to all engineers of the respective queues! Otherwise, they cannot open their tickets at all!

Customer permissions can be granted separately to the general sections (1) and the Comments and Attachments section (2).

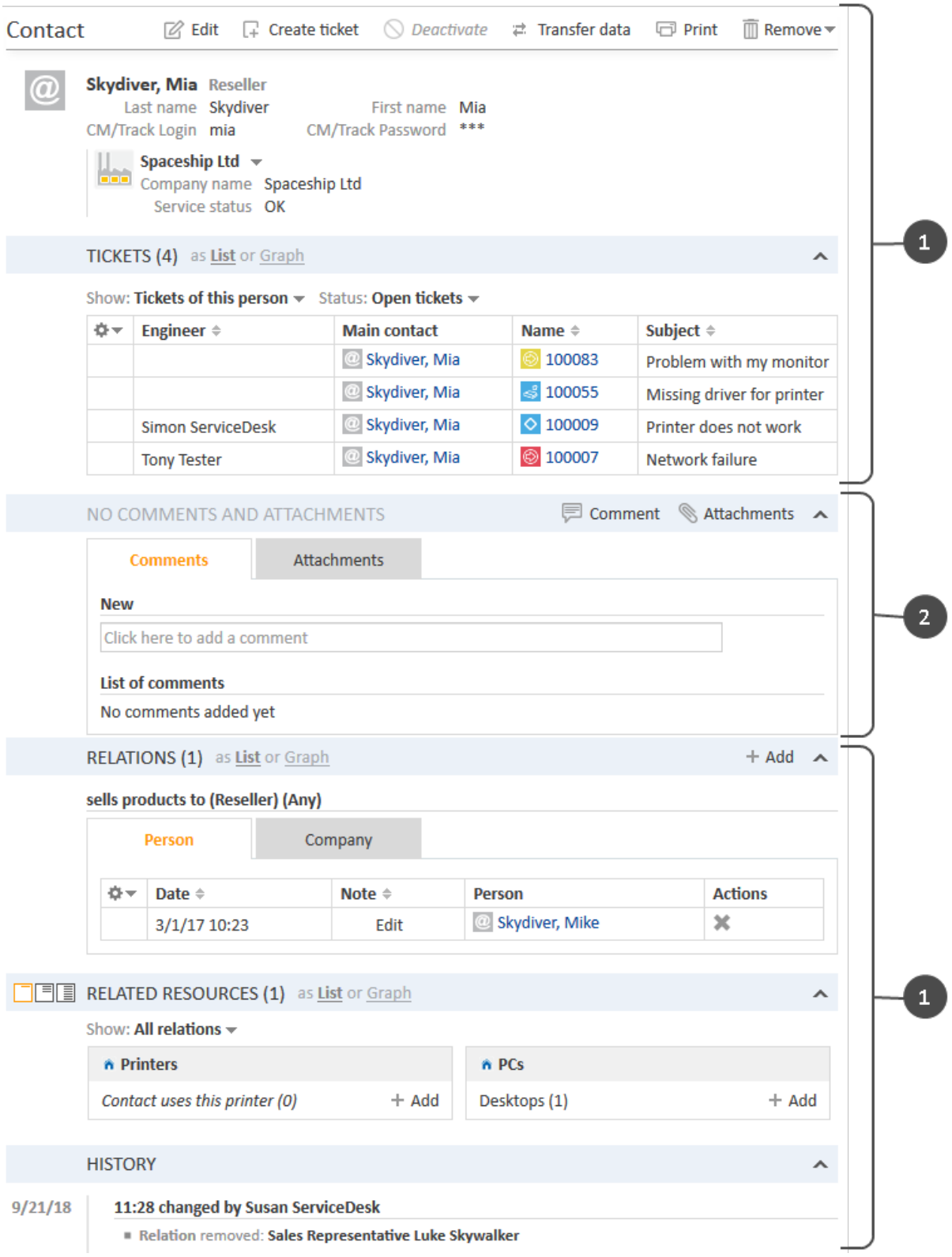


Figure 21: *ConSol CM Web Client - Sections of a contact page*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

The following access permissions can be granted:

- **Customer type**
  Refers to the tickets of the customer.
  - **Own**
    All (main or additional) customers of tickets which are currently assigned to the engineer or where the engineer is set as additional engineer.
  - **All**
    All customers.
- **General sections (1)**
  - **Read**
    Read the customer data.
  - **Write**
    Write/modify the customer data, and change the company of a contact on the contact page using the *Change* link.
  - **Delete**
    - Delete a customer data set. This refers to companies as well as to contacts. For contacts, two types of deletion are available (with or without related data).
    - Transfer all tickets associated with a customer of this customer group to another customer.
    - Anonymize a contact
  - **Act**
    Execute actions for this customer (see section Action Framework - Customer Actions for details about customer actions).
  - **Deactivate/activate**
    - Deactivate and (re-)activate the contact or company. It is not possible to create tickets for a deactivated customer.
    - Transfer all tickets associated with a customer of this customer group to another customer.

> **Information concerning transfer permissions for tickets and resources**
>
> Please note that starting with CM version 6.10.4.4, the permission *Transfer tickets* is linked to the permission *Delete* (customer data) as well as to the permission *Deactivate/activate* (customer data), i.e., an engineer can have either one of these permissions to be able to transfer data.

- **Comments and Attachments section (2)**
  - **Details read**
    Read customer data in the *Comments and Attachments* section.
  - **Details write**
    Write/modify customer data in the *Comments and Attachments* section.
  - **Details delete**
    Delete customer data in the *Comments and Attachments* section.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **General**
  - **Create**
    Create a customer data set. In a two-level customer data model this refers to contact as well as to company data sets.

> ⚠ Please keep in mind that an engineer must have at least read permissions for a customer group to open and/or create tickets for customers in this group!

Tab *Resource Types Permissions*

Resource types permissions control an engineer's access to resources, i.e., objects which are stored in the Resource Pool.

You reach this screen by clicking the *Resource Type Permissions* tab in the navigation item *Roles*, navigation group *Access and Roles*.

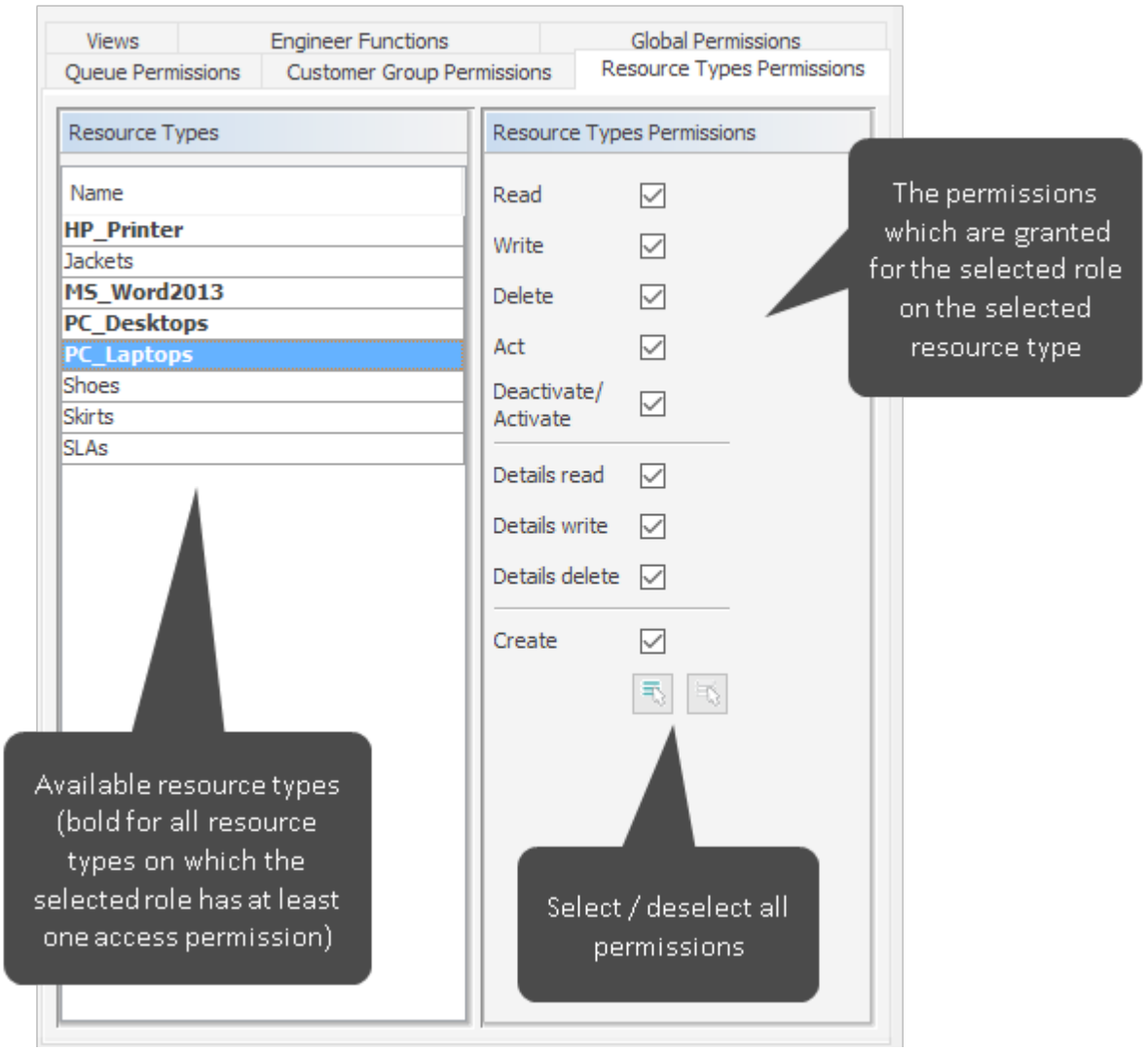


Figure 22: *ConSol CM Admin Tool - Access and Roles, Roles: Resource types permissions*

The following permissions can be granted:

- **Read**
  Load and display resources of the selected type in the Web Client.
- **Write**
  Change data fields of this type of resources.
- **Delete**
  Delete resources of the respective type from CM.
- **Act**
  Execute resource actions defined for this type of resources.
- **Deactivate/Activate**
  (De-) Activate resources of the selected type.
- **Details read**
  Load and display comments/attachments for resources of this type.
- **Details write**
  Add and change comments/attachments for resources of this type.
- **Details delete**
  Remove comments and attachments for resources of this type.
- **Create**
  Create new resource entries for the type of resources.

Tab *Views*

Views define which tickets engineers will see in the ticket list of the Web Client. This tab shows the assigned views on the left and the available views on the right (see also Views). The displayed views can be filtered by name and queue. Assigning views is optional.



We recommend to assign at least one view to a role. Otherwise an engineer with this role will see no tickets in the Web Client's ticket list.

You reach this screen by clicking the *Views* tab in the navigation item *Roles*, navigation group *Access and Roles*.

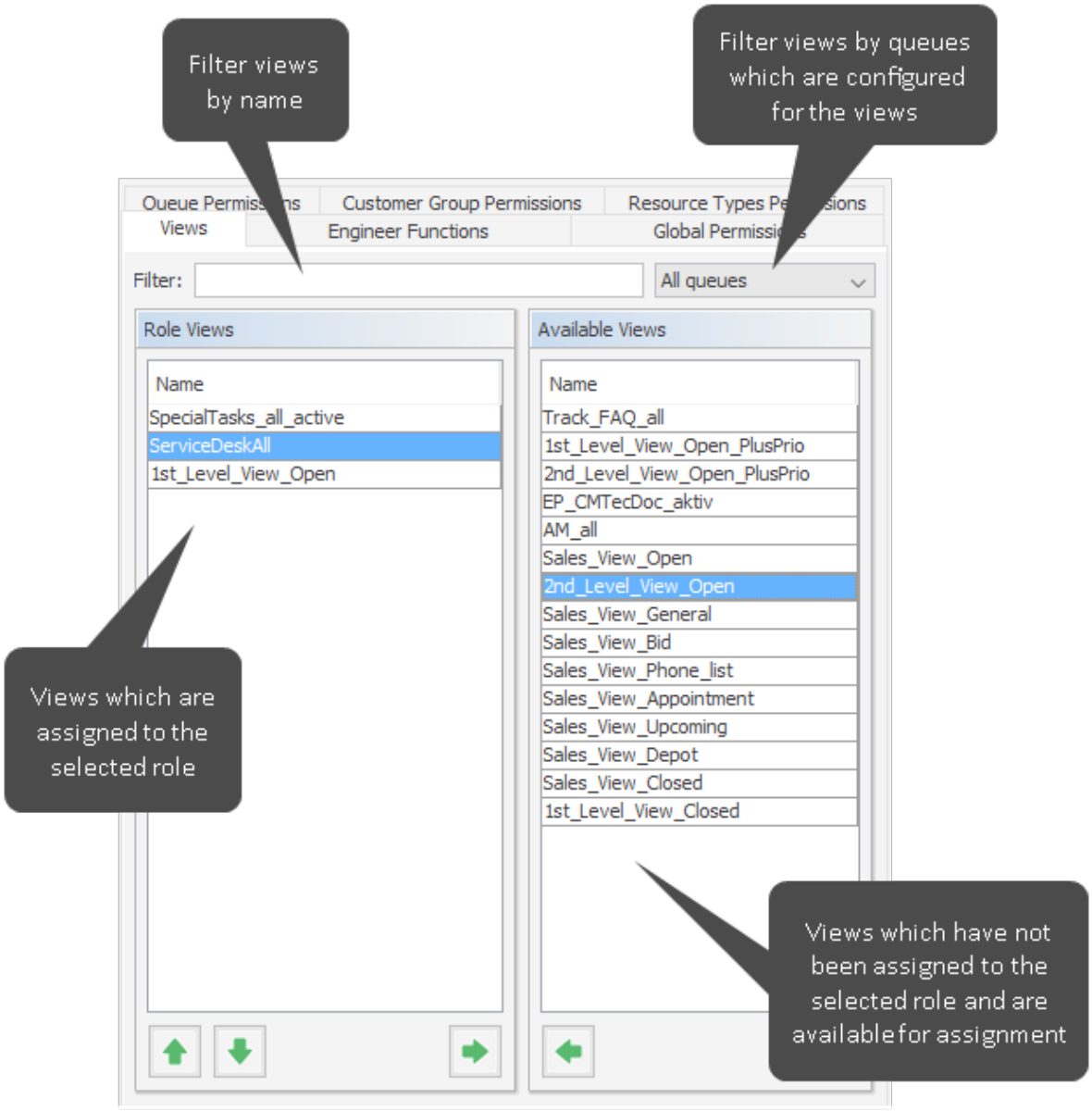


Figure 23: *ConSol CM Admin Tool - Access and Roles, Roles: Views*

Select a role on the left side of the page first and then the desired view(s) in the list of *available views*. Click the *Assign* button to move the selected view(s) to the list of *role views*. If you want to remove views from this list, select the respective views and click the *Unassign* button.

For regular roles, you cannot define the order of the views here. In the drop-down menu of the Web Client, the views will always be displayed in the order they have in the list of the view administration. Please see also section Views. When a role has been marked as *main role* for at least one engineer (and is thus marked with a red dot), the views can be sorted using the *Move upwards* and *Move downwards* buttons. The sorting affects the order in which the views are displayed in the Web Client.

Tab *Engineer Functions*

On this tab you can assign *engineer functions* to a role. Engineer functions are used if you need an additional engineer for a ticket, e.g., a supervisor who has to decide what to do before the ticket can

be moved on in the workflow. Thus you have to assign a role with the respective engineer function to this supervisor. In the Web Client engineer functions and associated engineers are shown when assigning an additional engineer.

You reach this screen by clicking the *Engineer Functions* tab in the navigation item *Roles*, navigation group *Access and Roles*.

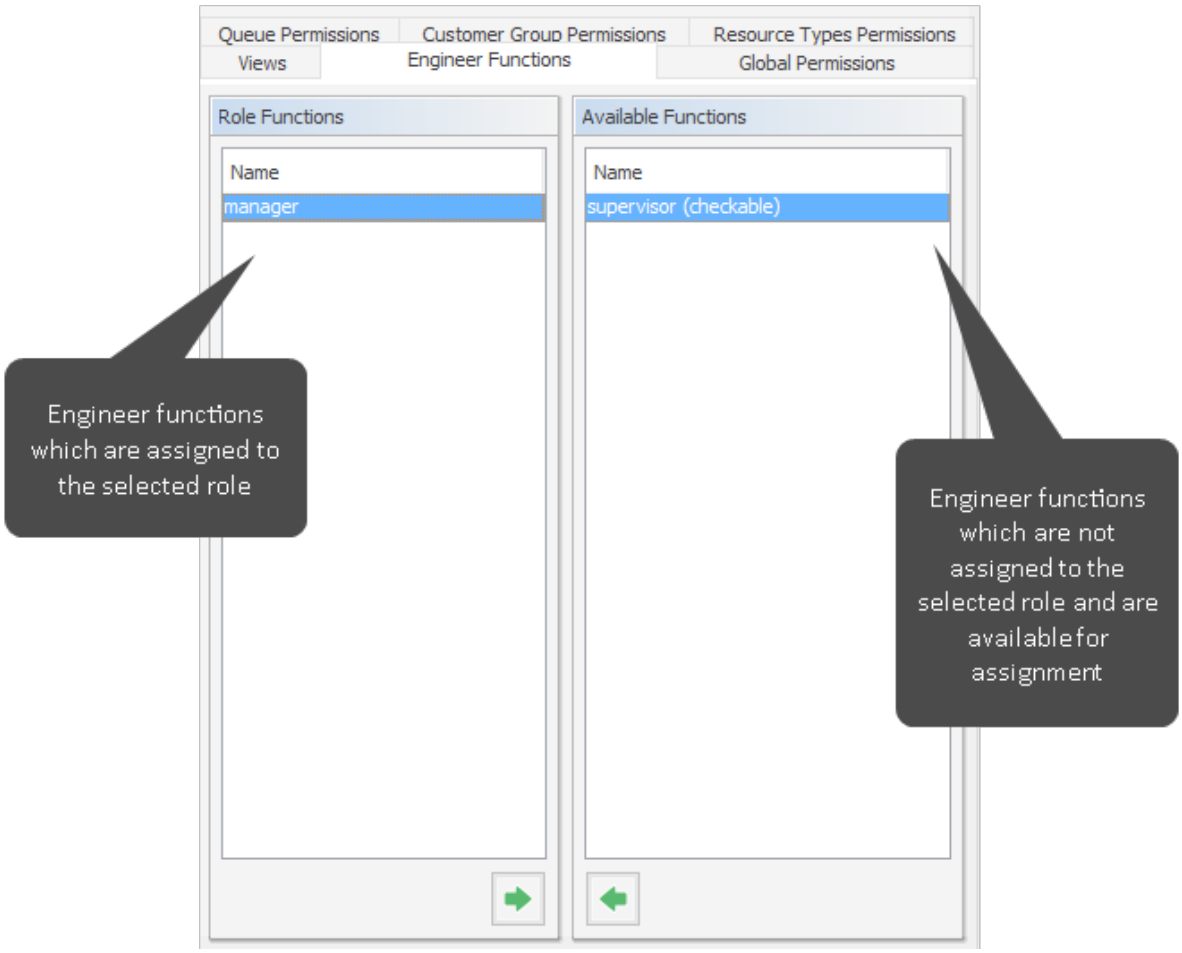


Figure 24: *ConSol CM Admin Tool - Access and Roles, Roles: Engineer functions*

Select a role on the left side of the page and then the desired engineer function(s) in the list of *available functions*. Click the *Assign* button to move the selected function(s) to the list of *role functions*. If you want to remove functions from this list, select the respective function(s) and click the *Unassign* button.

After you have defined the new role by setting permissions, views, and engineer functions in the tabs you can assign the role to the desired engineer accounts. Engineers obtain the rights of a role immediately after assignment (without an additional update of the system).

A.2.2.2 Delete a Role

Select the role you want to delete and click the *Delete* button below the role list. If you choose *Yes* in the following confirmation dialog, the role is removed from the list and the system.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH



If you delete a role, please consider that engineers with only this role will immediately lose all permissions in the system.

In case tickets, e.g., from a certain queue, are not covered by any role permission, engineers and/or administrators could get the impression that tickets are missing.

A.2.2.3 Copy a Role

If you want to create a new role and use an existing role as a template you can copy it. Select the existing role and click the *Copy* button below the role list. A pop-up window appears in which you can enter the name for the copy. Afterwards you can modify the copy according to your wishes.

A.2.2.4 Edit a Role

Select the role you want to edit in the list and modify the permissions in the respective tabs as desired. The changes are immediately effective for engineers with this role. The engineer just has to login again.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

A.2.3 Defining Additional Administrator Roles

There are three types of administrator permissions:

- **Global administrator**
  Access to all configuration and runtime data. Permission: *Administrate system (full access)*
- **Configuration administrator**
  Access to all configuration data, no access to runtime data. Permission: *Administrate system (Configuration only)*
- **Engineer administrator**
  Access to engineers, roles, views and engineer functions, no access to other configuration or runtime data. Permission: *Administrate access and roles*

There always needs to be at least one role with global administrator permissions and one user with this role. If required, you can create roles for additional administrators. This can be useful, for example, to create a user administrator role which can be assigned to the power users of the system. In this way, you can enable team leaders to create and delete engineers without giving them access to the rest of the configuration.



Roles with global administrator and configuration administrator permissions are displayed in a red color. Roles with engineer administrator permissions are displayed in a brown color.

A.2.3.1 Impact on Access and Role Management

Administrators cannot assign roles or permissions of a higher level or manage engineers who have roles with higher level permissions. Therefore, some restrictions regarding role and engineer management apply for the configuration and engineer administrator. These administrators cannot perform the following actions:

- Add or remove higher level permissions to/from roles
- Assign roles containing higher level permissions to engineers or unassign them from engineers
- Manage roles containing higher level permissions (create, copy, delete)
- Manage engineers who have roles containing higher level permissions (edit, enable, disable, delete)

These restrictions apply to the following permissions:

Configuration administrator:

- Administrate system (full access)

Engineer administrator:

- Administrate system (full access)
- Administrate system (configuration only)
- Workflow read
- Workflow write
- Workflow deploy

The permission which the current administrator is not allowed to assign are grayed out.

A.3 Views

This chapter discusses the following:

A.3.1 Introduction to View Administration 72

A.3.2 View Administration Using the Admin Tool 73

A.3.1 Introduction to View Administration

Views are used to filter tickets according to certain criteria (e.g. all active tickets in the Queue *Helpdesk)* and display the resulting tickets in the ticket list of the Web Client. Since views are associated with roles engineers obtain their view(s) via the roles which are assigned to them. Engineers can switch between their views in the Web Client.

Engineers need the appropriate permissions to see all tickets filtered by a view. Permissions are not automatically granted when a view is assigned, but they have to be assigned within the definition of roles (as queue and customer group permissions). One and the same view can result in varying subsets of tickets and information therein for engineers with different roles.

The creation of views is optional. However we recommend it in order to assure central features of the Web Client. Without a view engineers will not see any tickets in the ticket list. They can only access tickets by using the search function.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

A.3.2 View Administration Using the Admin Tool

To create, edit or delete views, open the navigation item *Views* in the navigation group *Access and Roles*, in the Admin Tool.

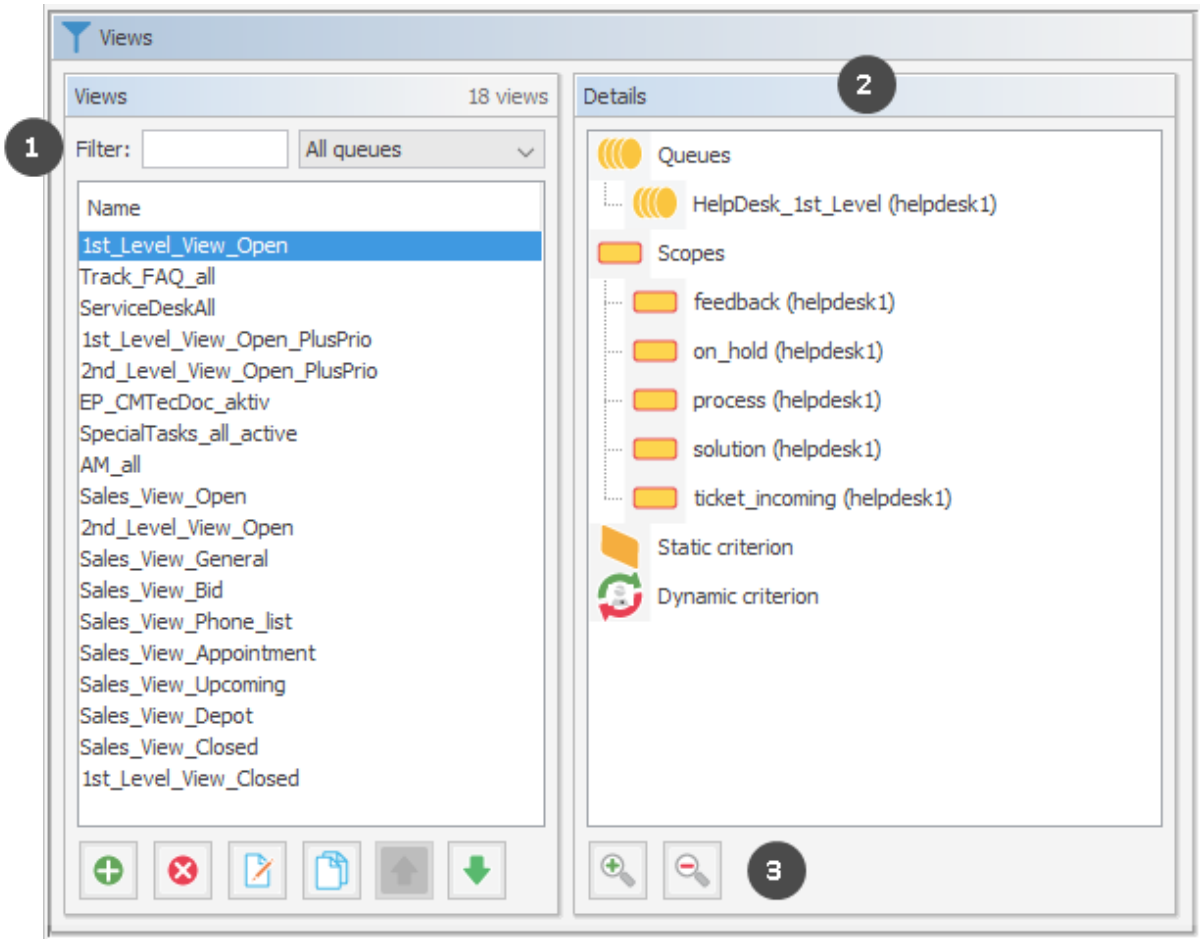


Figure 25: *ConSol CM Admin Tool - Access and Roles, Views: View administration*

The screen *Views* shows a list of all available views on the left. You can filter the list by entering the name of a view or selecting a queue (1). The details of the selected view, i.e., the queues and scopes which are configured n the view, are displayed on the right (2). You can expand or collapse the sub-trees using the buttons below the components (3).

A.3.2.1 Create a View

After clicking the *Add* button below the view list, the pop-up window *View Wizard* appears where you have to define the name for the new view first. You can also enter a description for it.

By clicking the *Localize* button you can localize view name and description. See section Localization of Objects in General, Type 1 for details.

Via *Next >* you can continue with the definition of view criteria:

- queue filter
- scope filter
- static criterion
- dynamic criterion

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Queue Filter

At first you choose the queues for the new view. Select the desired queues in the list *Unassigned* and move them to the list *Assigned* by clicking the *Assign* button. To remove an assigned queue, select it and click the *Unassign* button. Continue with the *Next >* button, in order to define scope filters, too.

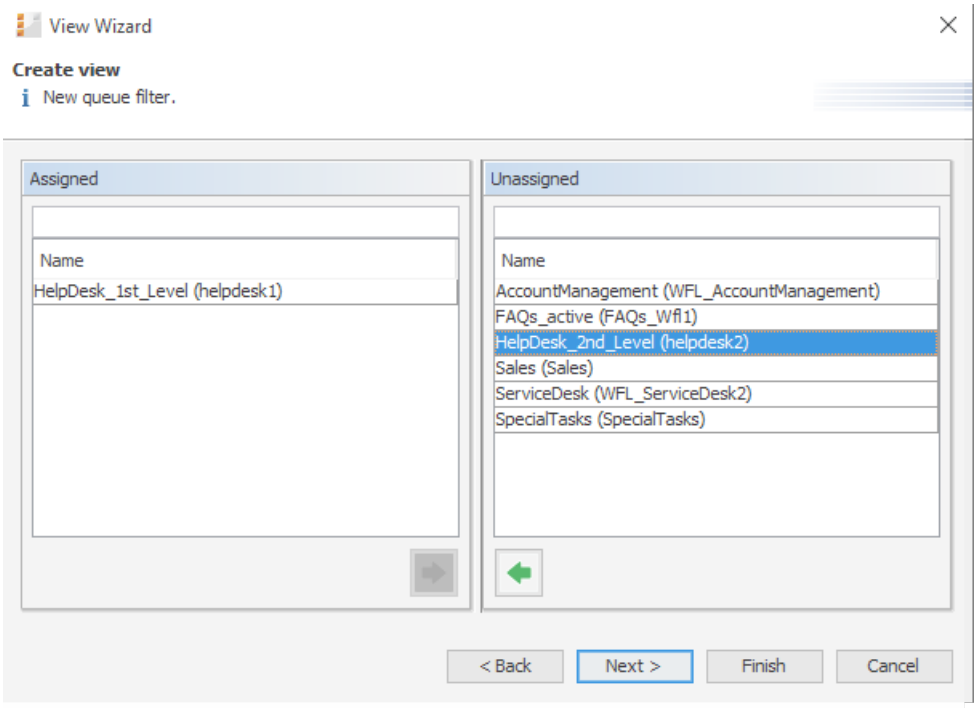


Figure 26: *ConSol CM Admin Tool - Access and Roles, Views: Selecting queues*

Scope Filter

Next you can limit the view to certain workflow scopes of the selected queue(s). Scopes group workflow activities that have a special topic in common, e.g., tickets with an appointment.

Select the desired scopes in the list *Unassigned* and move them to the list *Assigned* by clicking the *Assign* button. To remove assigned scopes, select them and click the *Unassign* button. Continue with the *Next >* button if you want to define further criteria, otherwise click *Finish* to create the view.

If you do not assign scopes in the View Wizard, the view exists by name but will not show tickets in the Web Client.

Since for the view definition you can only use scopes which have been defined during workflow development, please make sure that the workflows contain all required scopes. For example, if you want to have **active** and **inactive** tickets, there have to be separate scopes in the workflow, otherwise it will not be possible to define an **active** and an **inactive/waiting** view!

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

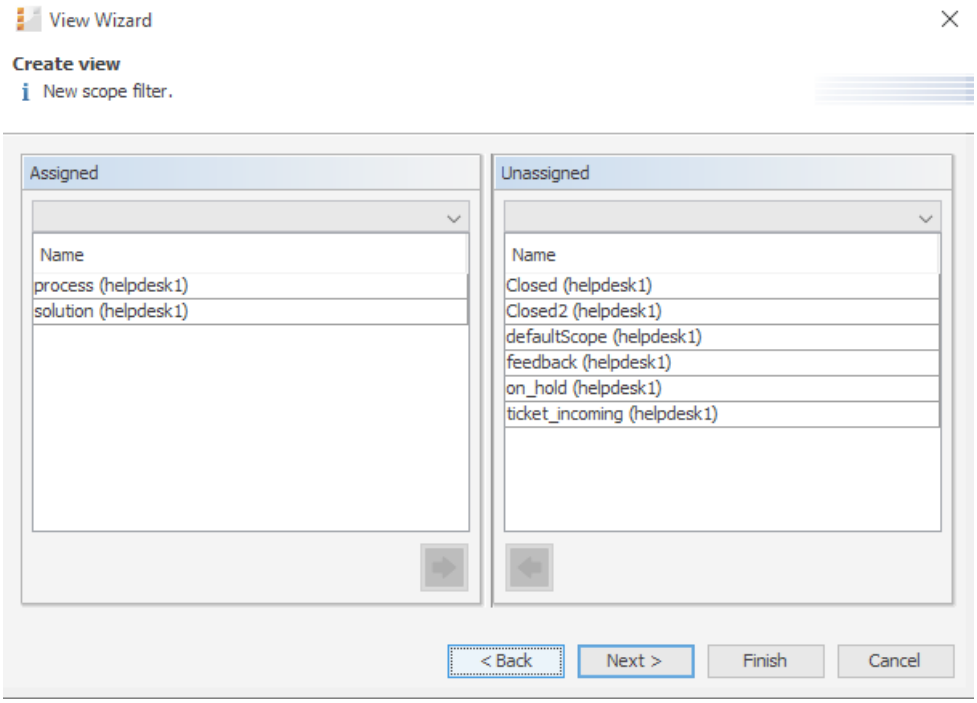


Figure 27: *ConSol CM Admin Tool - Access and Roles, Views: Selecting scopes*

Static Criterion

You can restrict the view further by a static criterion to show only tickets with a certain value in a defined data field, e.g., tickets concerning a special product or only tickets with high priority. The criterion is static because the engineer cannot change it in the Web Client. Please see the *ConSol CM User Manual* for a detailed description of working with views. Only data fields of type *enum* can be used as static criterion.

Choose the data field in the *Field* list (e.g. *product*) and select the desired value in the *Value* list below (e.g. *crm*). Continue with the *Next >* button if you want to define a dynamic criterion as well, otherwise click *Finish* to create the view.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

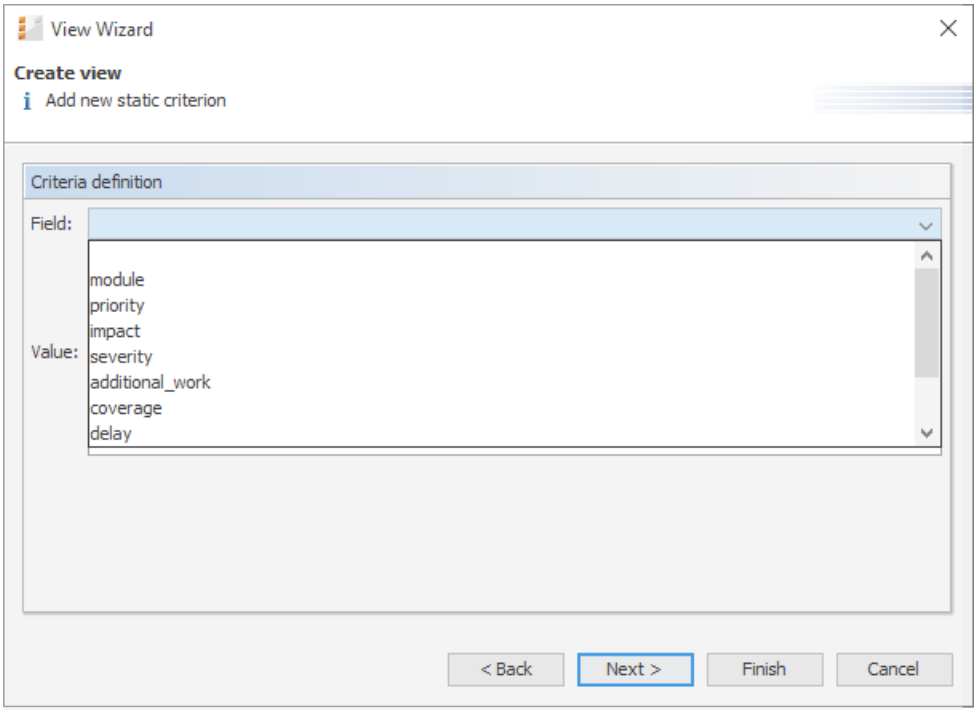


Figure 28: *ConSol CM Admin Tool - Access and Roles, Views: Selecting the static criterion*

Dynamic Criterion

Like a static criterion, a dynamic criterion is used to show only tickets with certain values in a defined data field, but in contrast to a static criterion, with a dynamic criterion engineers can choose the value (s) for the criterion themselves. This can be done in the Web Client by editing the *Engineer Profile*. Additionally, the administrator can adjust the value individually for each engineer on the *View criteria* tab of the engineer administration (see section Engineers). Please see the *ConSol CM User Manual* for a detailed description of working with views. Only data fields of type ENUM can be used as dynamic criterion.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

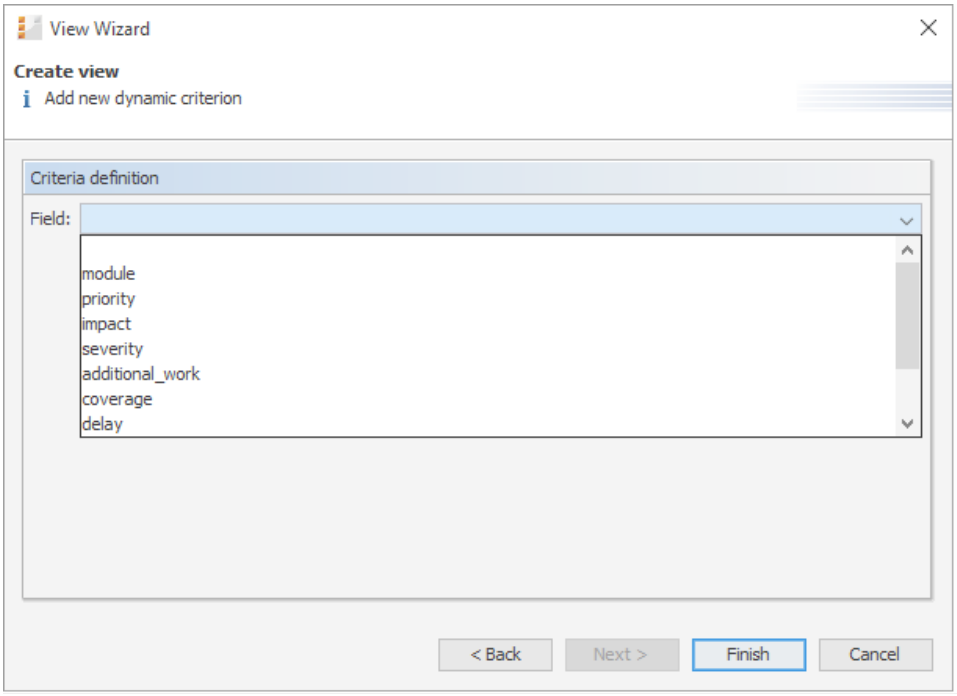


Figure 29: *ConSol CM Admin Tool - Access and Roles, Views: Selecting the dynamic criterion*

Click *Finish* to create the view. You can leave the window any time without storing by choosing *Cancel*. Via the *Back* button you can return to the previous step of the view definition.

Now you can see the new view in the view list on the left. The assigned criteria are shown in the *Details* area on the right side of the page.

⚠ Please note that in a view with a dynamic criterion, only the tickets are displayed which match this criterion. So if an engineer has not selected any criteria in his/her engineer profile, or if the administrator has removed all selections using the Admin Tool (*View criteria* in Engineers), the engineer's view will be empty! Make sure your users know about this fact and make sure you, as an administrator, are always aware of that fact.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

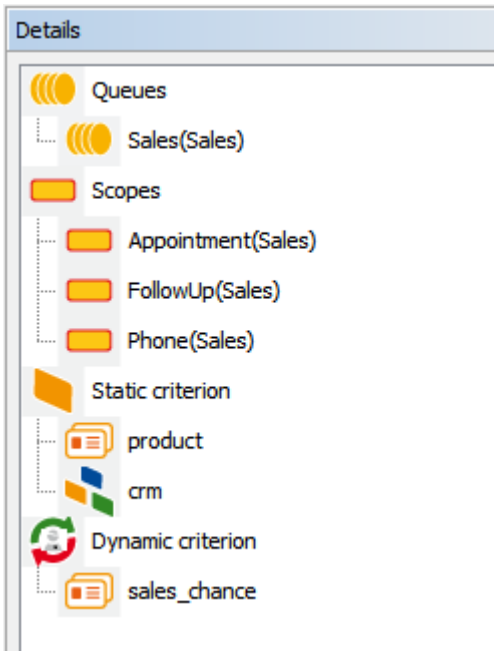


Figure 30: *ConSol CM Admin Tool - Access and Roles, Views: Viewing the details*

You can expand or collapse all details by clicking the *Plus* or *Minus* icon below the list.

> ⚠ We strongly recommend not to define views which contain closed tickets!
>
> The number of closed tickets will grow considerably during work with the application. Therefore, the view of closed tickets would always reach the maximum number of tickets allowed for a view (which can be defined using a system property). This can have negative influence on the Web Client performance and in most cases the desired tickets will not even be among the first 50 or 100 tickets.
>
> Conclusion: A view of closed tickets does not help and might decrease the speed of the system for the engineers. Only in test environments, a view for closed tickets might be an option.

A.3.2.2 Edit a View

Select the view you want to edit in the view list. The view details are shown on the right side of the page. To edit the selected view just click a filter criterion with the right mouse button. The following drop-down menu appears.

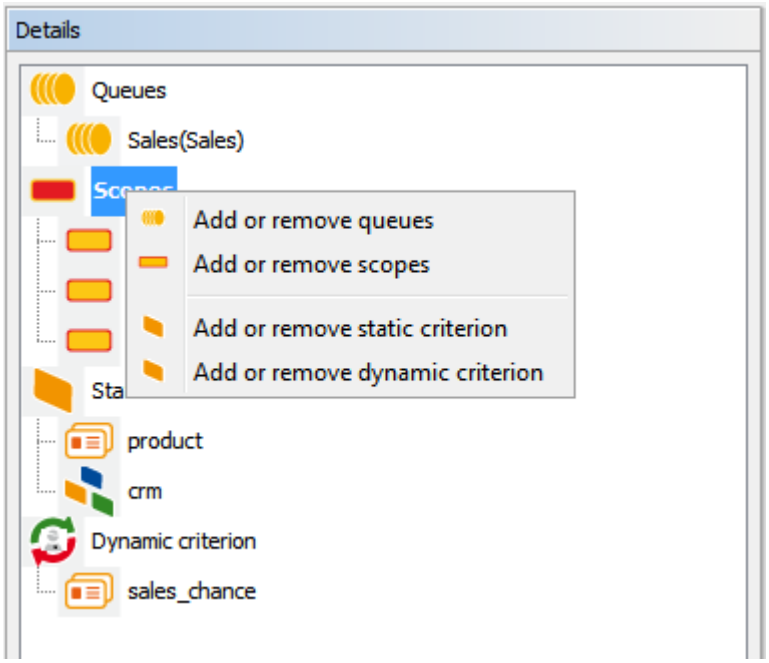


Figure 31: *ConSol CM Admin Tool - Access and Roles, Views: Editing a view*

The menu contains these options:

- Add or remove queues
- Add or remove scopes
- Add or remove static criterion
- Add or remove dynamic criterion

Just click the desired menu item. The respective window of the *View Wizard* appears where you can add or delete filter criteria as described in Create a View. Double-clicking on a filter criterion will also open the View Wizard.

> You cannot edit view criteria by clicking the *Edit* button. Here you can only modify the view's name and description.

A.3.2.3 Delete a View

Click the *Delete* button below the view list to delete the selected view. A pop-up window appears which asks whether you really want to delete the view. If you choose *Yes*, the view will not be available for any engineers. Engineer permissions are not affected by this operation.

A.3.2.4 Copy a View

The *Copy* button allows you to save time when creating a view. The selected view will be copied completely and you can edit the copy afterwards. The new view has the same name as the copied view. You can change it by double-clicking on the name or by clicking the *Edit* button.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

A.4 Engineer Functions

This chapter discusses the following:

A.4.1 Introduction 80
A.4.2 Create or Edit an Engineer Function 82
A.4.3 Delete an Engineer Function 83
A.4.4 Disable or Enable an Engineer Function 83
A.4.5 Engineer Permissions Concerning Engineer Functions 83

A.4.1 Introduction

Engineer functions are used if you need an additional engineer for a ticket, e.g., a supervisor who has to decide what to do before a ticket can be moved on in the workflow.

In the Admin Tool, engineer functions are managed using the navigation item *Engineer Functions* in the navigation group *Access and Roles*.

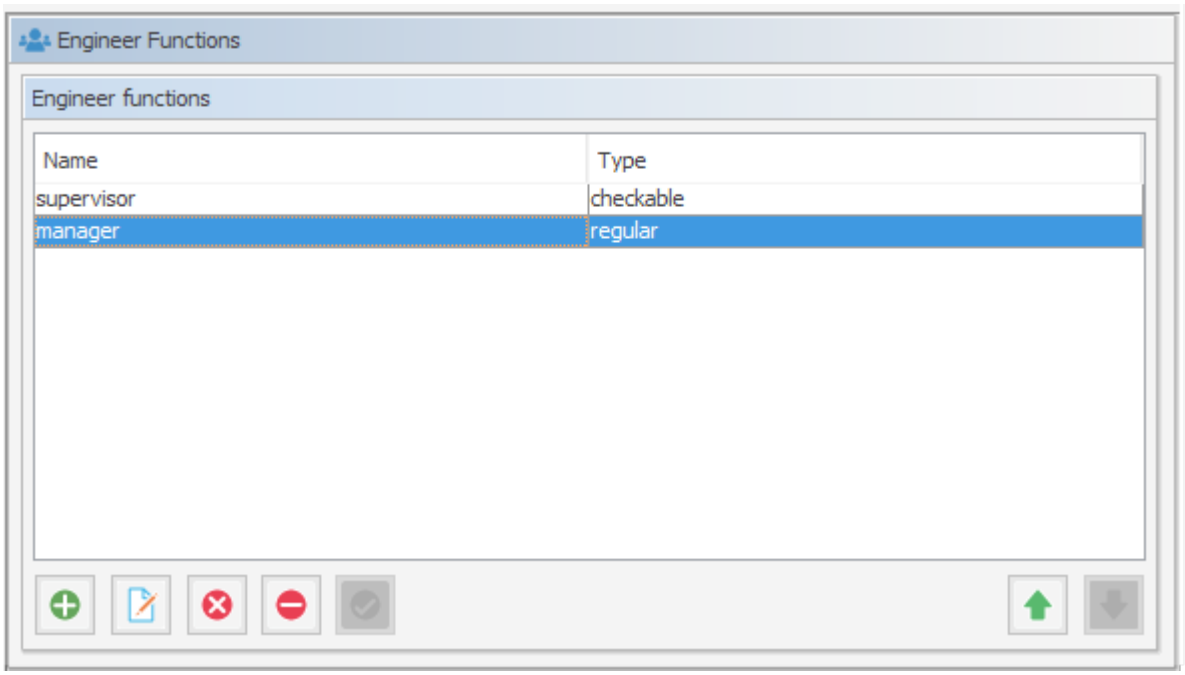


Figure 32: *ConSol CM Admin Tool - Access and Roles, Engineer Functions*

The corresponding activities for such a process have to be created in the workflow. Engineer functions are assigned to engineer roles which in turn need to be assigned to the respective engineers. In the Web Client you can choose a function and an appropriate engineer when adding an engineer to the ticket.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

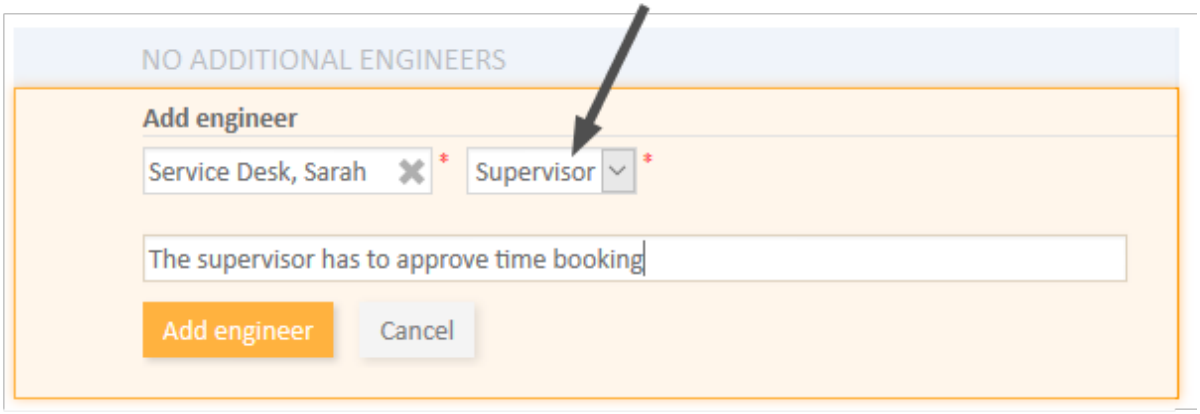


Figure 33: *ConSol CM Web Client - Assigning an additional engineer with an engineer function*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

A.4.2 Create or Edit an Engineer Function

An engineer function is defined by a name. By clicking the *Add* button a pop-up window appears where you can enter the name. You will get the same window when you click the *Edit* button in order to edit an engineer function. The checkbox *Checkable* has to be ticked if additional engineers shall have the permission to execute a certain activity, e.g., give their approval before the ticket can be moved on. The approval state is then displayed in the ticket.

> ⚠ After creation of an engineer function the checkbox *Checkable* cannot be de-selected anymore.

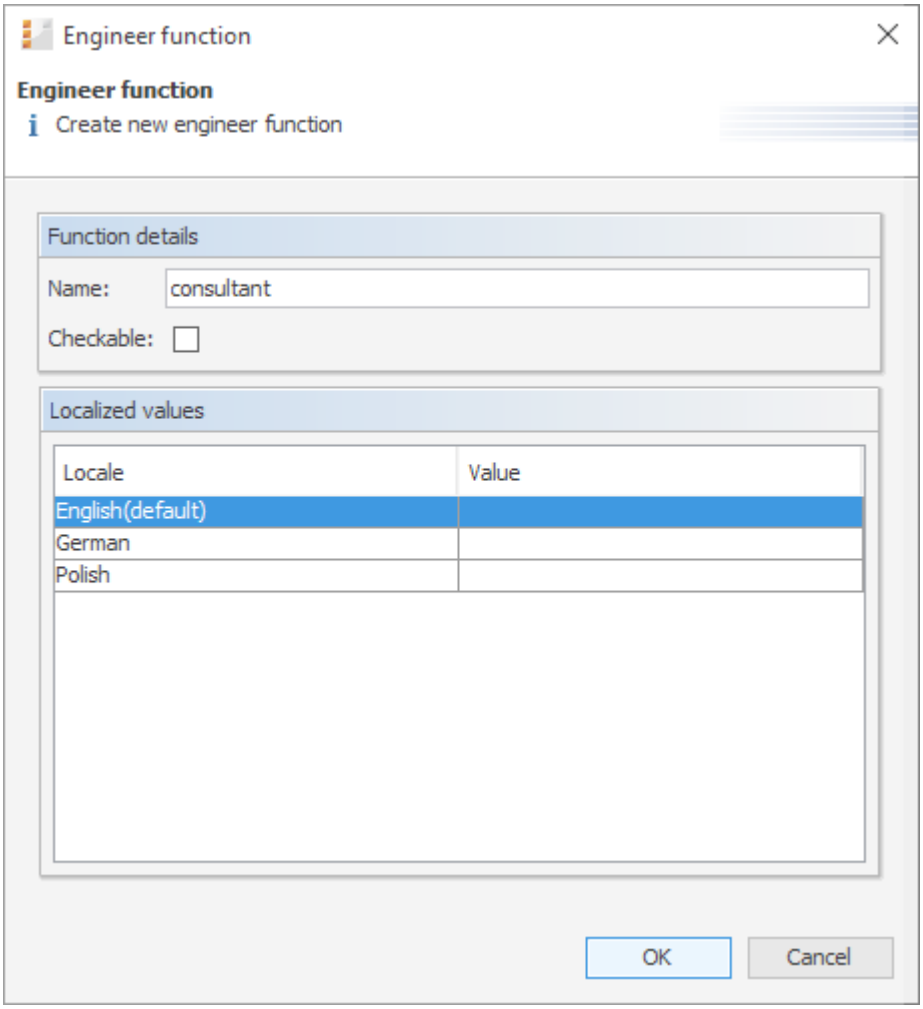


Figure 34: *ConSol CM Admin Tool - Access and Roles, Engineer Functions: Create or edit an engineer function*

You can also localize the name of an engineer function in this window. See section Localization of Objects in General, Type 2 for details.

After clicking *OK* the engineer function is created and the name will be displayed in the respective language of the engineer's locale.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

A.4.3 Delete an Engineer Function

An engineer function can only be deleted if it is not assigned to any roles. Otherwise you get a warning and you can only disable this engineer function (see below).

In order to delete an engineer function, select it in the list and click the *Delete* button. After choosing *Yes* in the confirmation dialog, the engineer function will be removed from the list and the system.

A.4.4 Disable or Enable an Engineer Function

If an engineer function is still assigned to a role but is not needed anymore you can disable it. To do this select the engineer function and click the *Deactivate* button. The entry in the list is shown in italics afterwards. The engineer function cannot be assigned anymore. Just click the *Activate* at the bottom of the page if you want to enable the function again.

A.4.5 Engineer Permissions Concerning Engineer Functions

For engineers, resp. for roles, specific permissions can be granted which concern only the tickets for which the engineer who has this role is assigned as engineer with a certain function. Those permissions are granted as queue permissions (see also section Roles). This principle provides a good basis for a very sophisticated management of processes like approvals or other processes where people with different roles and responsibilities have to work on a case at the same time.

Using the Role Administration (navigation group *Access and Roles*, navigation item *Roles*), you can grant/withdraw the following permissions (for a detailed explanation of the permissions see section Roles):

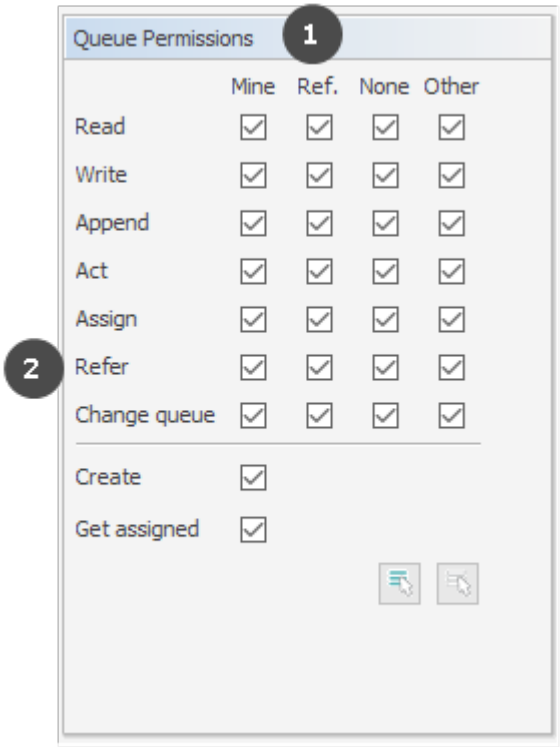
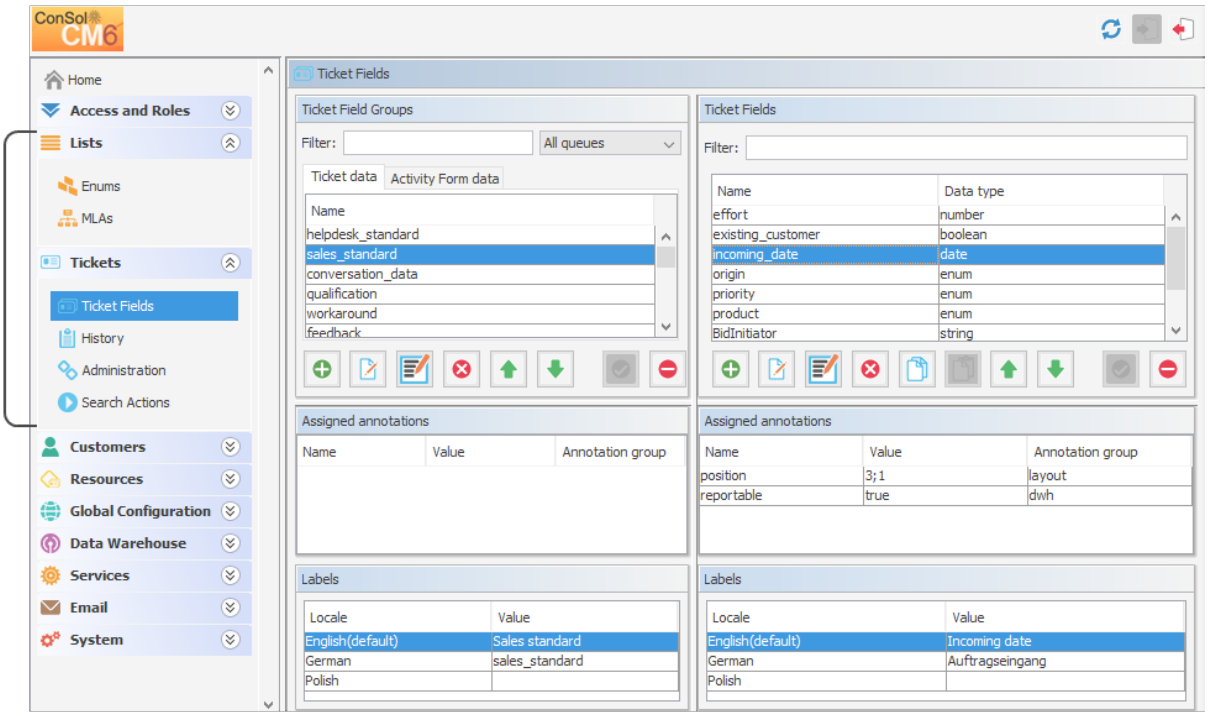


Figure 35: *ConSol CM Admin Tool - Access and Roles, Roles: Queue permissions concerning engineer functions*

- ***Ref.* column (1)**
  The permissions are valid for all tickets in this queue where a member of this role is set as additional engineer with a specific engineer function.
- ***Refer* row (2)**
  For the selected ticket types, the permission to set an additional engineer in those tickets is granted.

B - Ticket Data Model



In this section, you will learn how to define the data model for ticket data, thus this is about ticket field definition and the positioning of the ticket fields on the Web Client GUI. However, some of the data structures (enums, MLAs) might also be used for customer fields and resource fields as described in sections Customer Field Management and GUI Design for Customer Data and CM/Resource Pool - Setting Up the Basic Resource Model. In addition, the configuration options for the ticket history and ticket list are explained.

In this section, the following topics are covered:

- Ticket Fields (Setting Up the Ticket Data Model)
- Managing Sorted Lists: Enums
- MLAs
- Ticket History
- Ticket List

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

B.1 Ticket Fields (Setting Up the Ticket Data Model)

This chapter discusses the following:

B.1.1 Introduction 86
B.1.2 Ticket Field Administration Using the Admin Tool 88
B.1.3 Tab Ticket Data 89
B.1.4 Tab Activity Form Data 101
B.1.5 Frequently Used Annotations 105

B.1.1 Introduction

Ticket fields are defined for tickets. By default, all tickets contain information about the assigned engineer, the creation data, and the current queue and scope. These default fields are displayed in the upper part of the ticket next to the ticket icon (1). In addition to these fields, the CM administrator can define ticket fields adapted to the use case implemented in ConSol CM. Examples for ticket fields are *priority*, *software module*, *reaction time*, or *sales potential*. Ticket fields always belong to ticket field groups, which are then assigned to queues to make the ticket fields available. The ticket fields can be displayed in the ticket header (2) or in the Details section (3).

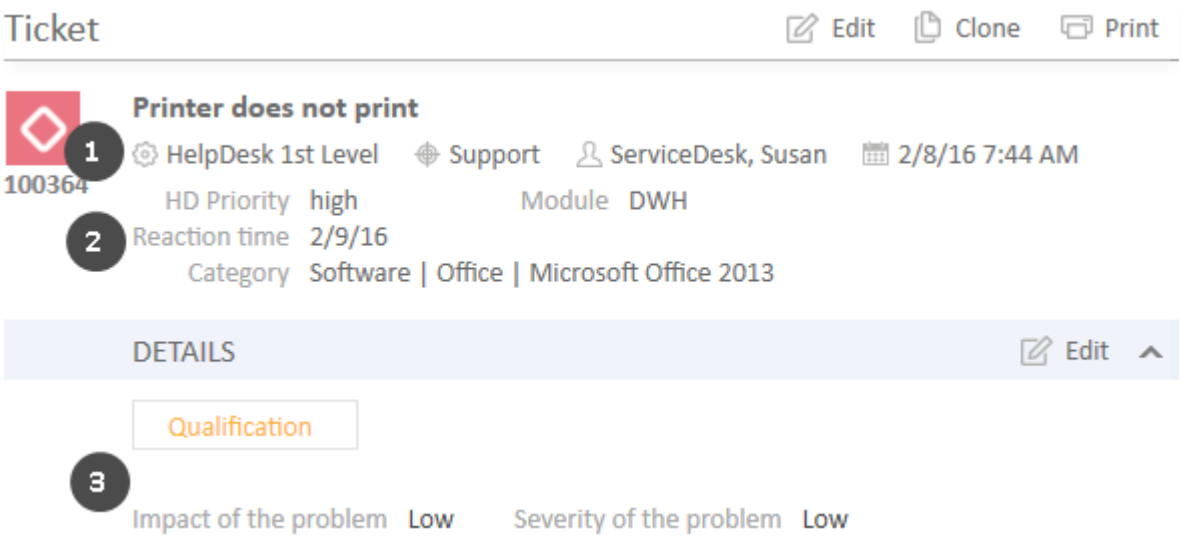


Figure 36: *ConSol CM Web Client - Ticket fields*

A **ticket field group**:

- can be assigned to a queue. For example, the ticket field group *helpdesk_fields* can be assigned to the queue *HelpDesk*.
- can be faded in and out in the Web Client GUI during the process. For example, you can fade in the ticket field group *solution* later in the process, so it is displayed at the point when the engineer found a solution to the issue reported by the customer. You cannot fade in or out single ticket fields.
- can be displayed below the default fields or as a tab in the *Details* section of a ticket. In the latter case, the title and mouse-over of the tab is the (localized) name of the ticket field group. If the ticket field group is displayed in the ticket header, the group name is not shown.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- is configured using group annotations. Annotations are set to define special parameters and characteristics of a ticket field group, e.g. the initial display mode on the user interface. Please see List of Group Annotations for further information about the available group annotations.
- is placed on the Web Client GUI based on its position in the list of ticket field groups, which defines the order of the ticket field groups in the ticket header and/or the order of the tabs.

A **ticket field**:

- is always defined within a ticket field group.
- is assigned to a queue as a part of its ticket field group.
- can be made invisible using annotations, but cannot be faded in or out as single field during the process.
- is configured using field annotations. Annotations are set to define special parameters and characteristics of a ticket field, e.g. the position on the user interface. Please see List of Field Annotations for further information about the available field annotations.
- is placed on the Web Client GUI based on the value of its `position` annotation or based on its position in the list (the first ticket field in the list is displayed first on the GUI) if `position` is not set.

To define new **ticket fields** for a queue, you have to perform the following steps:

1. Define a ticket field group and set the respective annotations.
2. Define all sorted lists which you will need for the ticket fields (see Managing Sorted Lists: Enums). For example, if the ticket field *priority* should contain a list of priorities, you have to define this list first.
3. Define all ticket fields within the new ticket field group.
4. Assign the new ticket field group to the queue where its ticket fields are required.
5. Test the results in the Web Client GUI. You do not need to log in again, it will be sufficient to refresh a page which shows a ticket in the respective queue.

All these steps are explained in detail in the following sections.

In case some data fields should be offered in a form during one or more processes, Activity Control Forms (ACFs) can be defined. This is explained in the section Tab Activity Form Data.

B.1.2 Ticket Field Administration Using the Admin Tool

In the Admin Tool, ticket fields are managed using the navigation item *Ticket fields* in the navigation group *Tickets*.

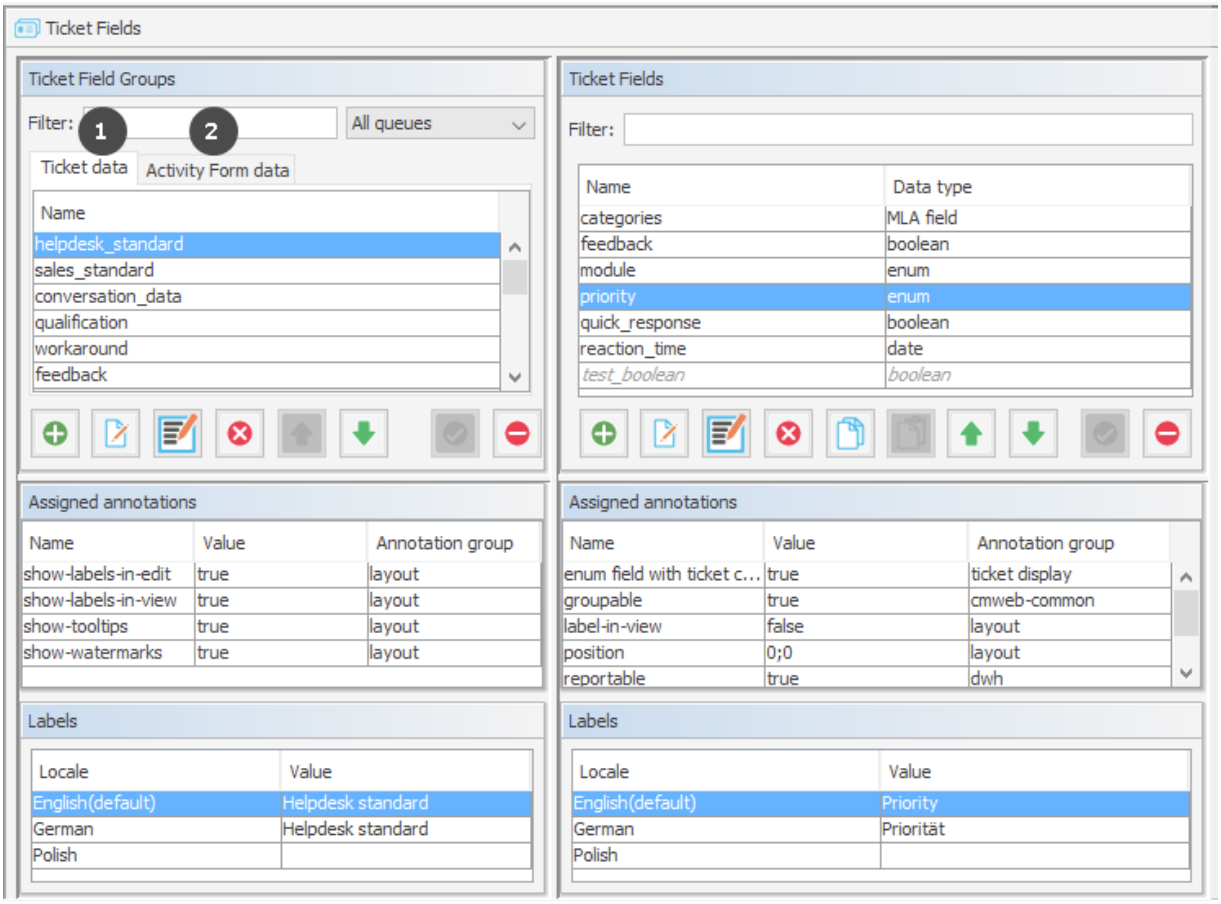


Figure 37: *ConSol CM Admin Tool - Tickets, Ticket Fields*

The navigation item *Ticket fields* has two tabs:

- Tab Ticket Data (1): This tab is used to define ticket field groups and ticket fields.
- Tab Activity Form Data (2): This tab is used to define Activity Control Forms.

Both tabs are explained in detail in the following sections. You can switch between the tabs by clicking the tab header.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

B.1.3 Tab *Ticket Data*

On this tab, you can define groups and fields for ticket data.

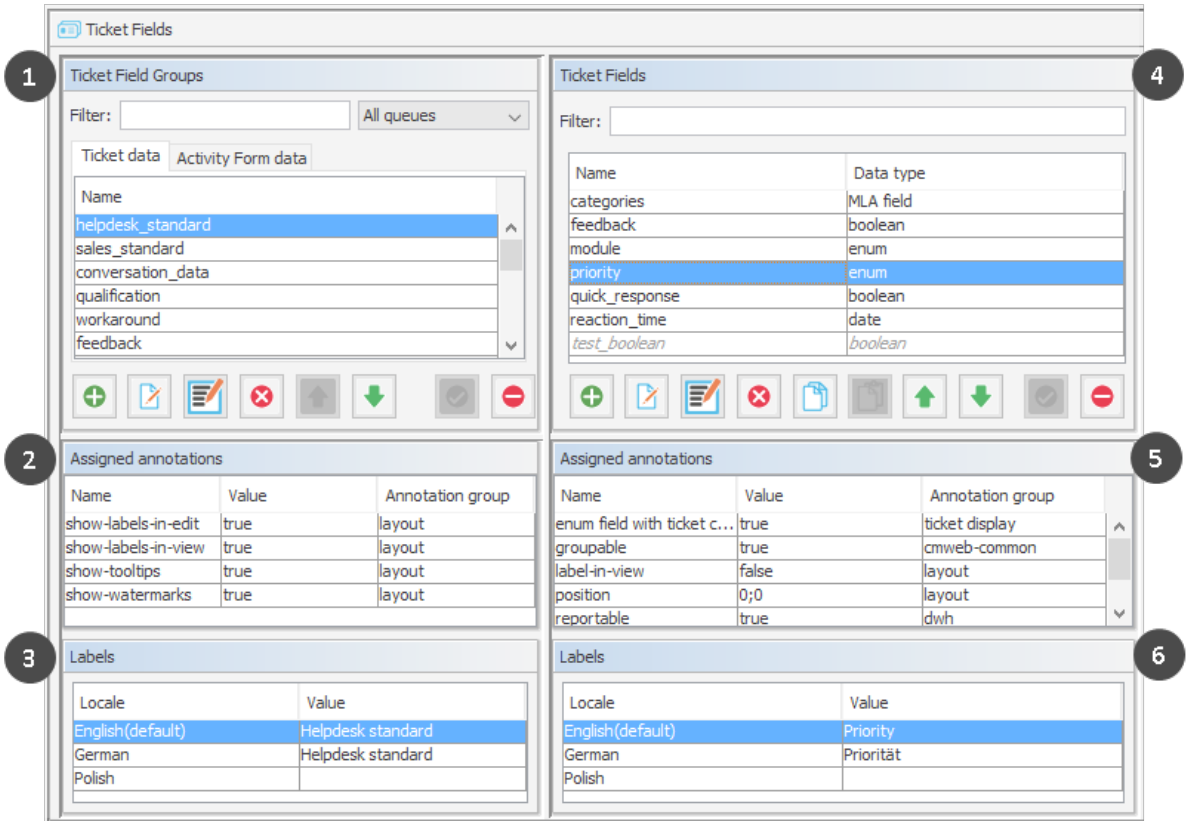


Figure 38: *ConSol CM Admin Tool - Tickets, Ticket Fields*

The *Ticket data* tab consists of six sections:

- Ticket field groups (1)
- Annotations of the selected ticket field group (2)
- Labels, i.e., localized values, of the selected ticket field group (3)
- The ticket fields of the selected ticket field group (4)
- Annotations of the selected ticket field (5)
- Labels, i.e., localized values, of the selected ticket field (6)

B.1.3.1 Create a Ticket Field Group

To create a new ticket field group just click the *Add* button below the list on the left side of the page. The following pop-up window appears.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

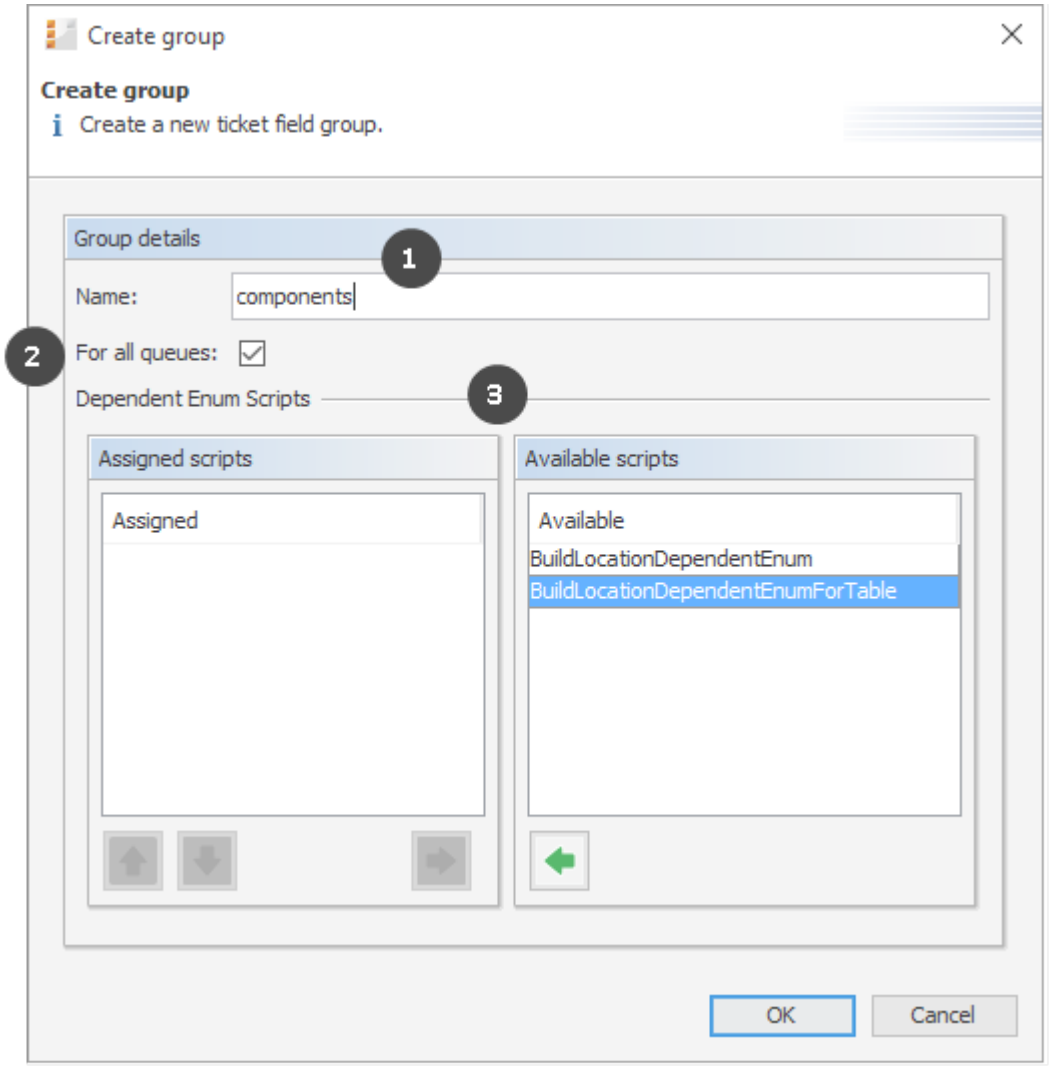


Figure 39: *ConSol CM Admin Tool - Tickets, Ticket Fields: Creating a ticket field group*

- **Name (1)**
  Enter a name for the ticket field group. The name must be unique. The localized names for a ticket field group are set in the main *Ticket Fields* panel. For details, please refer to section Localization of Data Fields.
- **For all queues (2)**
  If this check box is activated, this group's ticket fields are automatically assigned to all queues. A ticket field group which is marked as *for all queues* will also be assigned automatically to each newly created queue. Usually ticket field groups for ticket data are only valid in specific queues (see Queues).
- **Available scripts, i.e., Dependent Enum Scripts (3)**
  Dependent enum scripts define the structure of DEPENDENT ENUMS (hierarchical multi-level lists) used in ticket fields of this ticket field group. With DEPENDENT ENUMS you can limit the choices in multi-level lists. You select an element in a list and, based on this selection, only matching results will be shown in the next lower hierarchy level of the list. The ENUMS (single lists) for the ticket fields have to be created within the Managing Sorted Lists: Enums while the scripts that couple the lists to create the DEPENDENT ENUM are created using an Admin Tool

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

script, see section Admin Tool Scripts.
To assign dependent enum scripts to a ticket field group select the desired script(s) in the list *Available scripts* and move them to the list *Assigned scripts* by clicking the *Assign* button.

B.1.3.2 Edit a Ticket Field Group

If you want to edit a ticket field group, select it in the list and click the *Edit* button. The same window as described above for creating a ticket field group will appear. You can modify all fields and save your changes by clicking *OK*.

B.1.3.3 Annotate a Ticket Field Group

Ticket field groups are annotated to define their characteristics, e.g., where a group is displayed in the Web Client, if a group is indexed, or if it should be visible. You can define, e.g., whether a group is visible in the Web Client (annotation `group-visibility`) or whether it is shown in the *Details* section of the Web Client (annotation `show-in-group-section`). To assign annotations select a group and click the *Annotate* button. The following pop-up window appears:

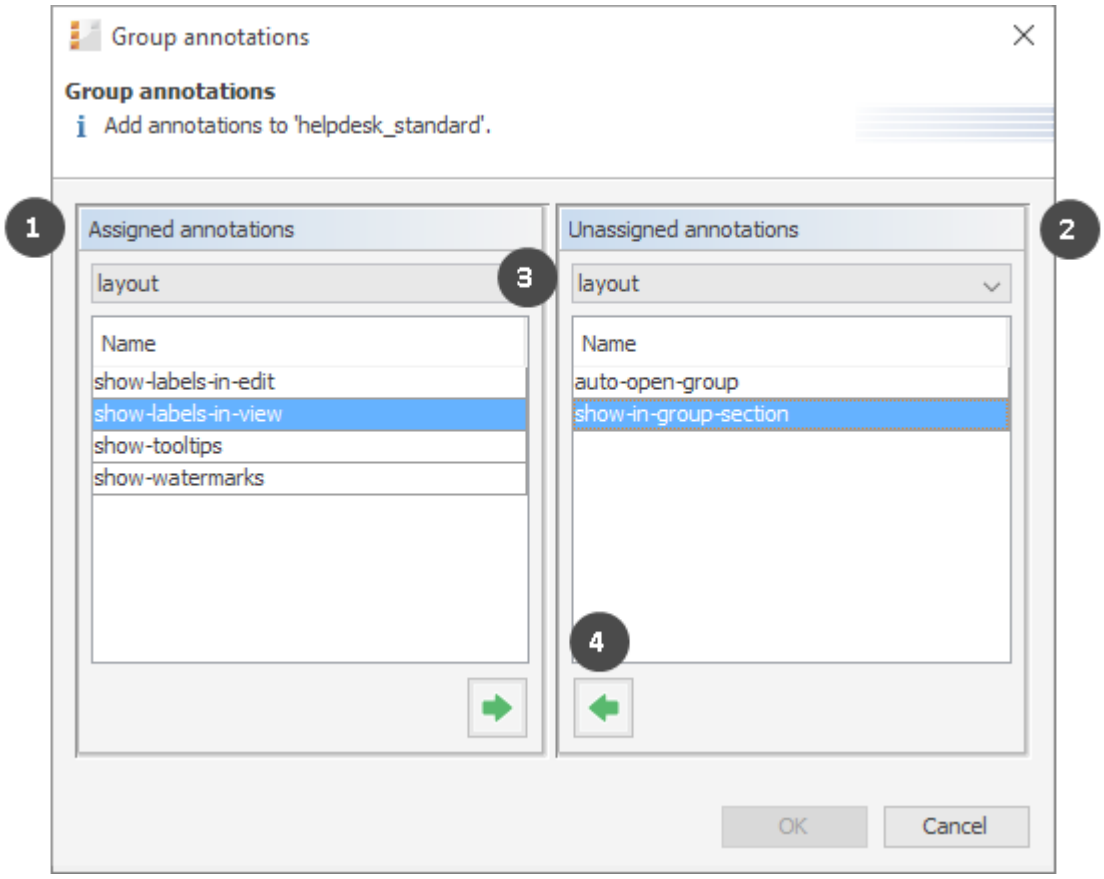


Figure 40: *ConSol CM Admin Tool - Tickets, Ticket Fields: Assigning annotations to a ticket field group*

The right part of the window contains the available annotations (2). Using the selection field above the list you can filter the display by annotation type, e.g., *common* or *layout* (3). Select the desired annotations and move them to the *Assigned annotations* list on the left (1) by clicking the *Assign* button (4). This list can also be filtered by annotation type. Click *OK* to assign the annotations to the ticket field group and to close the window. See Annotations, section *Group Annotations* for detailed information.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

The annotations are now shown with a default value (if available, e.g., “true” or “false”) in the bottom left-hand corner of the administration page. The value can be modified by double-clicking into the corresponding *Value* field and typing the desired value. Press the Enter key afterwards.

Ticket field groups will appear in the Web Client as they are ordered in the list. Select a group and use the *Move upwards* and *Move downwards* buttons and if you want to change the position of this group in the list.

B.1.3.4 Delete a Ticket Field Group

A ticket field group can only be deleted if it is not assigned to a queue or a ticket. Otherwise you get a warning stating you can only disable this group (see below). In order to delete a ticket field group select it in the list and click the *Delete* button. If you confirm the following dialog with *Yes*, the group with its corresponding fields is removed from the list and the system.

B.1.3.5 Enable or Disable a Ticket Field Group

If you cannot delete a ticket field group, or if you do not want to delete it because you might need it again, you can disable it. To do so select the group and click the *Deactivate* button. The entry in the list is shown in italics afterwards. Disabled ticket field groups are not displayed in the Web Client. Just click the *Activate* button below the group list if you want to enable the group again.

B.1.3.6 Create a Ticket Field

Ticket fields contain the data for tickets, e.g., priority, service level, deadline, or hardware module. The fields of a ticket field group are created in the right part of the page. To create a new ticket field, select the desired group first on the left and then click the *Add* button below the *Ticket Fields* area on the right. The following pop-up window appears:

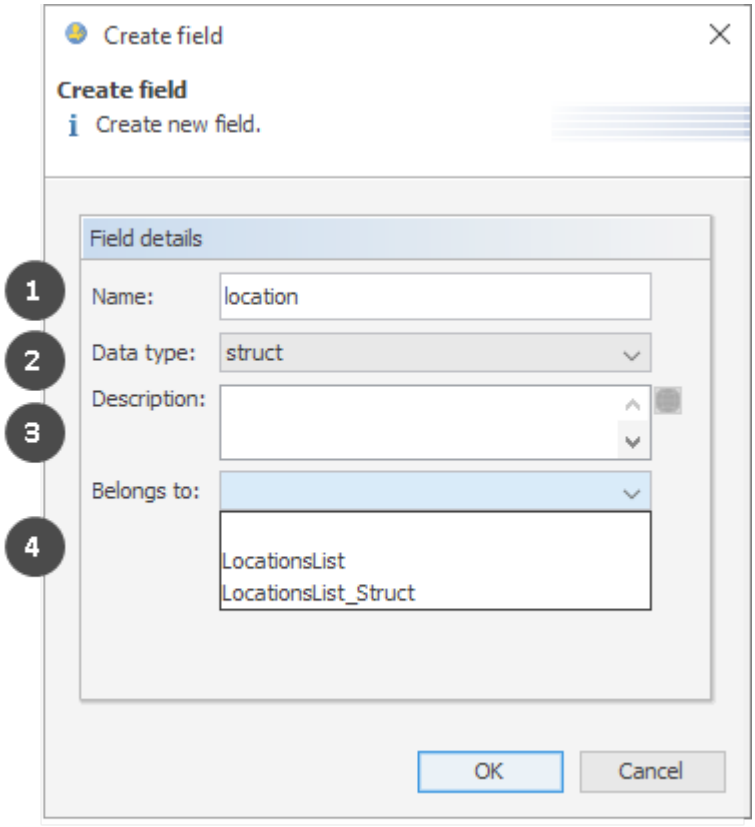


Figure 41: *ConSol CM Admin Tool - Tickets, Ticket Fields: Creating a ticket field*

Fill out the following information:

- **Name (1):**
  Enter a name for the ticket field. The name must be unique within the ticket field group. The localized names for a ticket field are set in the main *Ticket Fields* panel. For details, please refer to section Localization of Data Fields.
- **Data type (2):**
  Choose one of the available data types for the new ticket field. Please see Types of Data Fields.
- **Description (3):**
  Enter the description for the ticket field. If provided, the description will be used as a tooltip in the Web Client. You can enter a localized description using the globe icon. For details, please refer to section Localization of Objects in General, Type 1.
- **Belongs to (4):**
  This field shows the available ticket fields of the data types *list* and *struct* used to create lists or tables. Choose in the drop-down box to which list or structure the ticket field belongs (if applicable).

Types of Data Fields

The following data types are available for ticket fields, customer fields and resource fields.

- **autocomplete**
  A data field which contains a text autocomplete field. This is a dynamic list which is based on a script of type *Text Autocomplete*. A detailed explanation of text autocomplete fields is given in section Scripts of Type Text Autocomplete.
- **boolean**
  Values: true/false. Depending on the annotation `boolean-type`, the value is displayed as checkbox, radio buttons, or drop-down list.

  > If a boolean field has not been touched, its value is “NULL”. If it is checked, its value is “true”, and if it is unchecked again, its value is “false”.

- **date**
  Format and accuracy can be set by annotations.
- **enum**
  For sorted lists. The engineer can choose one of the enum values in the Web Client. Enums and values have to be created previously within the Managing Sorted Lists: Enums. Select the desired *Enum type* and *Enum group* in the fields below.
- **list**
  A data field of this data type is the first step to creating a list (one column) or a table (multiple columns) of input fields in the Web Client.
  - For a **table** the next step will be to create another field of type *struct* (see below) to contain the input of the individual list fields (which will become the columns of the table). So, if you want to create a table you have to define a field of the type *struct* first (see below) before you can add the fields for the table columns.
  - For a **simple list**, the next step will be to create fields which belong to the list. No *struct* is required.

  For all fields belonging to a list or table you have to set the dependencies in the field *Belongs to* (see below). For example, a table field (which is a regular data field) always belongs to a *struct*, a

struct always belongs to a *list.*

- **struct**
  A data field of this type defines a data structure (line of a table) which groups one or multiple fields. It is the second step to building a table after you have created a field of the type *list*. Add the fields for the columns of the table in the next step. The dependencies have to be set for each field in the *Belongs to* field (see below), i.e., a *struct* always belongs to a *list*.

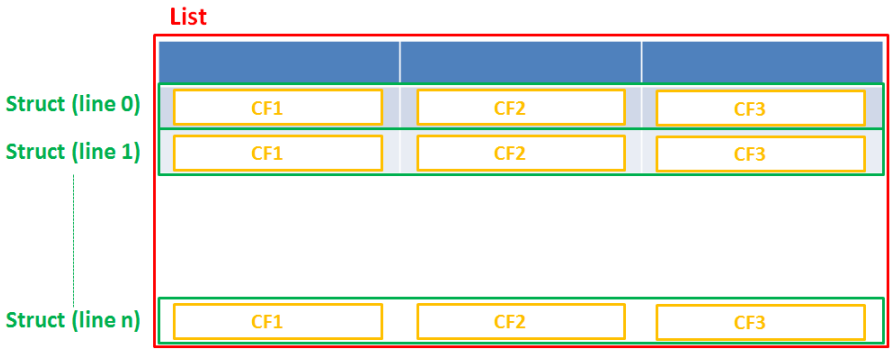


Figure 42: *Scheme: List of structs*

Technically spoken, the list is an array which contains a map (= key:value pairs) in each field.



Figure 43: *List of structs, technical principle*

- **number**
  For integer values.
- **fixed point number**
  For numbers with a fractional part, e.g., currencies. You have to enter the total number of digits (*Precision*) and the number of digits that fall to the right of the decimal point (*Scale*) in the respective fields below.
- **string**
  For up to 4000 alphanumeric characters.

> ⚠ Restriction when using an Oracle database: at most 4000 bytes can be saved in UTF encoding. Starting with Oracle12c.

- **long string**
  For large objects.

  > ⚠ For *long strings* the limit depends on the database system used for ConSol CM: MS SQL Server: 2 GByte; MySQL: 4 GByte; Oracle: 16 - 64 GByte (depending on page size of tablespace).

- **short string**
  For up to 255 alphanumeric characters.

  > ⓘ For *string* fields, you can use specific annotations to fine-tune the field definition. For example, a *string* field can be defined to contain a URL which will automatically be displayed as hyperlink or can be the hook for an autocomplete list. Please read the following section.

- **contact data reference**
  Special data type used internally for referencing the contacts associated with a ticket. This data type is not displayed but only used internally in the CM system.
- **MLA field**
  This data type is used for fields that contain hierarchical lists with a tree structure called *MLA* (Multi Level Attributes). The name of the field is the name of the new MLA that has to be defined within the MLAs. The group of the field has to be referenced when the MLA is created.

> ⚠ The data type you choose on creating a data field cannot be changed afterwards!

Using Annotations to Fine-Tune Data Fields

String fields can be used for various purposes, from simple input fields to text boxes, URLs or passwords. The configuration is done using annotations, see Annotations. The most commonly used annotation are listed in the following section.

How can I ...

... insert a **text box** instead of a single line?

Value for the annotation `text-type`: “textarea”

The size of the text box can be adjusted, displayed as standard text box depending on web browser. Use the `field-size` annotation in case a specific size of the text box is required.

... hide the input of the fields for **passwords**?

Value for the annotation `text-type`: “password”

Only dots are displayed. This annotation does **not** define the field to contain a password. It only defines the display mode. Use the `password` annotation to define a string field to contain the CM/Track password.

... display a **hyperlink**?

Value for the annotation `text-type`: “url”

The input is displayed as a hyperlink in view mode. The string has to match a specific URL pattern:

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

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

The first part of the string is the link (url), the second part is the name which should be displayed.

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

... display a **file link**?

Value for the annotation `text-type`: “file-url”

The input is displayed as a link to a file on the file system. The web browser has to allow/support those links.

Example: Enabling file:// URLs in a Firefox browser

Add the following lines to either the configuration file `prefs.js` or to `user.js` in the user profile. On a Windows system usually in a folder like `C:\Users\<USERNAME>\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");

Alternatively a Firefox browser add-on like ***Local Filesystem Links*** can be installed for better access to the referenced files and folders.

The link is also displayed as a tooltip.

The URL is correctly formed if the following conditions are met:

- It starts with `file:` followed by regular slashes:
  - three slashes "///" for files on the same computer as the browser (alternatively "//localhost/")
  - two slashes followed by the server name followed by another slash for files on file servers accessible from the computer running the browser
- These are followed by the full path to the file ending with the file name.
- The path on Microsoft Windows systems is also written with forward slashes instead of backslashes.
- The drive letter of a local path on Microsoft Windows systems is noted as usual, for example `C:`.
- Paths with spaces and special characters like "{, }, ^, #, ?" need to be percent encoded ("%20" for a space for example) for Microsoft Windows systems.

Example URLs:

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

See also the explanation about `file-url` in the section text-type

... define a **label**?

Value for the annotation `text-type`: “label”

This is a read-only field which is displayed in label styling. Use the `label-group` annotation to link the label and the input fields which belong together. Please take a look at the annotations for labels (`show-label-in-edit`, `show-label-in-view`) before implementing special label fields.

... define a **title**?

Value for the annotation `text-type`: “title”

This is a read-only field which is displayed in title styling, i.e. in a bold font which is larger than regular labels.

... define a field for the **valid email addresses**?

Value for the annotation `email`: “true”

The field may only contain valid email addresses. The input is validated according to the standard email format <name>@<domain>.

... define a scripted autocomplete list?

Value for the annotation `text-type` = “autocomplete”

Optional: value for the annotation `autocomplete-script` = <name of the respective script>

A scripted autocomplete list is used to show a drop-down list which is filled dynamically using the input the engineer has provided so far. For example, when the user types "Mil", the possible values "Miller", "Milberg", and "Milhouse" are displayed as a list and the engineer can select the one required for the field. The values are based on a result set which can contain strings, engineers, tickets, customers or resources.

The logic to create the result set needs to be implemented in a script, see Scripts of Type Text Autocomplete.

... define a field containing personal data?

Value for annotation `personal-data`: “true”

This annotation can be assigned to ticket and contact fields. Contact fields with this annotation will be deleted when a contact is anonymized. Ticket fields with this annotation will be deleted when the main customer of the ticket is anonymized. See Example 8: Removing Customer Data for information about how to anonymize a contact.

> ⚠ When defining a field to contain personal data, please take into account that the deletion of the field during the anonymization process is treated as a regular update. Therefore, business event triggers reacting on changes to ticket fields fire, and the contact update action script is executed.
>
> This might lead to unwanted side-effects.

... define a **rich text** field?

Create a field of the type *long string*.

Value for the annotation `string-content-type`: “html-inline” (field content is directly displayed on the GUI in view mode) or “html-link” (field content is displayed in a pop-up window after clicking the *Show content* link in view mode)

A rich text field can contain images and formatted text. In the edit mode, it features a basic editor.

... define a tooltip for a data field?

Value of the annotation `show-tooltip` = “true”

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

When creating or editing a data field, you can enter a text in the *Description* field. This text is used as a tooltip in the Web Client. You can enter a localized description using the globe icon. For details, please refer to section Localization of Objects in General, Type 1.

> This does not apply to date fields and fields containing phone numbers and URLs, as they have special tooltips.

... optimize the data field layout?

Use three columns to display the data fields (two columns are also possible, but do not use the available space as efficiently). This is set in the system property cmweb-server-adaper, defaultNumberOfCustomFieldsColumns. In ACFs, you can determine the number of columns using the *Column limit* setting of the ACF.

Display data fields which most likely have a value on the left in order to avoid gaps.

Add the annotations `position` = 0;0 and `colspan` = 3 to all tables. Remove the `colspan` annotations from other types of data fields.

Show the labels for all data fields (annotation `show-label-in-view` should be `true`).

B.1.3.7 Copy a Ticket Field

You do not necessarily have to create every data field using the pop-up menu. You can also copy an existing data field. This might be a ticket field, customer field, or resource field. Select the respective data field in the data field group and click the *Copy* button. Then navigate to the data field group where the copied field should be placed. This might be a ticket field group, a customer field group, or a resource field group. You can copy a field from one object type (e.g. ticket) to another (e.g. customer). Click into the list of data fields of the target field group and click the *Paste* button.

Every type of field can be copied, from simple string fields over enums to entire lists of structs. To copy a list of structs or a list of fields, mark and copy only the list, the child objects (e.g., structs with fields) will be copied automatically.

These things will be copied:

- almost all annotations with their values
- all localized values

These things will not be copied:

- the following annotations
  - *ldapid*: This annotation can be only once in a data model. Duplication would cause an invalid model.
  - *username*: This annotation can be only once in a data model. Duplication would cause an invalid model.
  - *password*: This annotation can be only once in a data model. Duplication would cause an invalid model.
  - *position*: The value for this annotation must be unique, so the annotation is duplicated, but the value is removed from the copy.
  - *ticket-list-position*: This annotation should be used very specifically and, thus, it should be present only for very few fields. Duplicating it would potentially multiply it although it should not be used for the majority of copies.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Naming of the copied field(s):

- Copy to other field group: field name is used
- Copy to same field group: <field name>_copy is used

Besides the Copy/Paste buttons you can also use the keyboard shortcuts:

- Copy: CTRL-C
- Paste: CTRL-V

B.1.3.8 Edit a Ticket Field

If you want to edit a ticket field, select it in the list and click the *Edit* button. The same window as described above for creating a ticket field will appear. Except for *data type*, *enum type*, and *enum group* you can modify all fields and save your changes by clicking *OK*.

B.1.3.9 Annotate a Ticket Field

Just like ticket field groups, ticket fields are annotated to define the properties of the field, e.g., is it read-only, should it be indexed, and where it should be displayed on the Web Client GUI (please see section Annotations for a list of all available annotations). Select a field and click the *Annotate* button below the list. The following pop-up window appears:

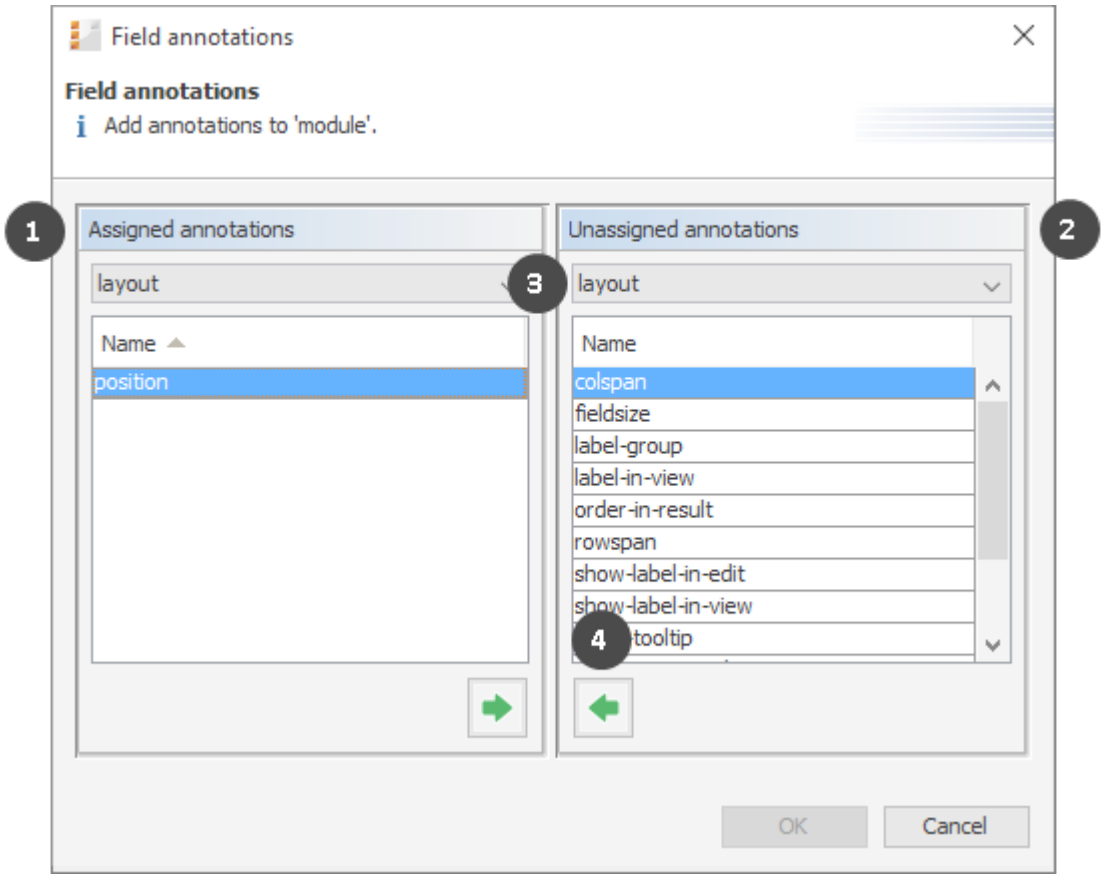


Figure 44: *ConSol CM Admin Tool - Tickets, Ticket Fields: Assigning annotations for a ticket field*

The right part of the window contains the available annotations (2). Using the selection field above the list you can filter the display according to annotation type (3). Select the desired annotations and

move them to the *Assigned annotations* list on the left (1) by clicking the *Assign* button (4). This list can also be filtered according to annotation type. Click *OK* to assign the annotations to the ticket field and to close the window.

The annotations for the selected field are now shown with a default value (if available, e.g., “true” or “false”) in the bottom right-hand corner of the administration page. The value can be modified by double-clicking the corresponding *Value* field and typing the desired value. Press the Enter key afterwards.

Ticket fields will appear in the Web Client as they are ordered in the list unless you have assigned a layout via the `position` annotation. You can change the position of a field in the list by using the *Move upwards* and *Move downwards* buttons below.

Note on the layout of the ticket data:

You can define several columns of data fields on each line, e.g., the position annotation can end with 0, 1 or 2.

0;0	0;1	0;2
1;0	1;1	1;2

B.1.3.10 Delete a Ticket Field

A ticket field can only be deleted if it is not assigned to a queue or a ticket, otherwise you get a warning stating you can only disable this field (see below). In order to delete a ticket field, select it in the list and click the *Delete* button. If you confirm the following dialog with *Yes*, the ticket field will be removed from the list and the system.

B.1.3.11 Enable or Disable a Ticket Field

If you cannot delete a ticket field, or if you do not want to delete it because you might need it again, you can disable it. To do so select the field and click the *Deactivate* button. The entry in the list is shown in italics afterwards. A disabled ticket field is not displayed in the Web Client. Just click the *Activate* button below the ticket field list, if you want to enable the field again.

B.1.3.12 Visibility of Ticket Data in CM/Track

If CM/Track, a ConSol CM Add-on which provides a customer portal, is active in your CM system, you will have to configure ticket fields which are specific for CM/Track. Furthermore, the visibility of the ticket fields in the customer portal has to be configured. This is explained in detail in section CM/Track: Data Availability For Customers.

B.1.3.13 Using Scripted Field Visualization for Ticket Fields

Using Scripted field visualization, you can enhance the display of data in ticket fields. Please see section Scripts of Type Field Visualization for details.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

B.1.4 Tab *Activity Form Data*

B.1.4.1 Introduction

An Activity Control Form is a web form which is displayed during the process, i.e., when the engineer works on the ticket. If the engineer clicks a workflow activity, the ACF is displayed first. The engineer has to fill out certain data fields and save the changes in order to proceed and execute the workflow activity.

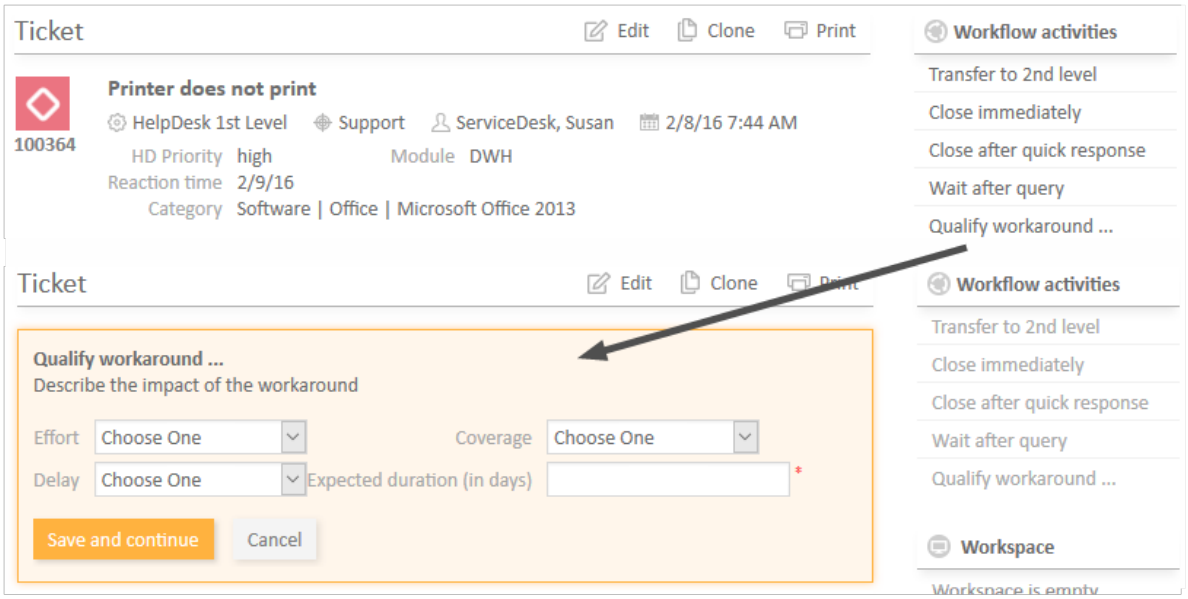


Figure 45: *ConSol CM Web Client - Example for ACF*

The ticket fields which are used in an ACF have to be defined first, as regular ticket fields in the ConSol CM Admin Tool. ACFs can include ticket fields from one or more ticket field groups. The ACFs is integrated into the process (workflow) using the ConSol CM Process Designer. There, you can also define which ticket fields should be required and which fields are optional. The display of an ACF can depend on a condition, e.g., the ACF *Reasons for dismissal of request* is only displayed if the customer has *Gold* or *Platinum* status. ACF dependencies are configured using scripts. This is explained in detail in the *ConSol CM Process Designer Manual*.

B.1.4.2 Definition of Activity Control Forms (ACFs)

In this tab, you can create activity control forms (ACF) which can be assigned to activities in the Process Designer. They are used to gather input in the Web Client if a manual workflow activity needs more information for the next step, e.g., if a ticket has to be qualified before it can be moved on or if you want feedback for a ticket. ACFs are basically a set of ticket fields already created in the tab *Ticket data*. An ACF can contain ticket fields of more than one ticket field group. However, all ticket field groups have to be assigned to the queue to which the workflow using the ACF is assigned. Please read the chapter on ACFs in the *ConSol CM Process Designer Manual* for detailed information about the process flow with ACFs and the features provided using programming ACF scripts.

The following figure shows the available ACFs. You reach this screen by opening the navigation item *Ticket Fields* in the navigation group *Tickets* and selecting the *Activity Form data* tab.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

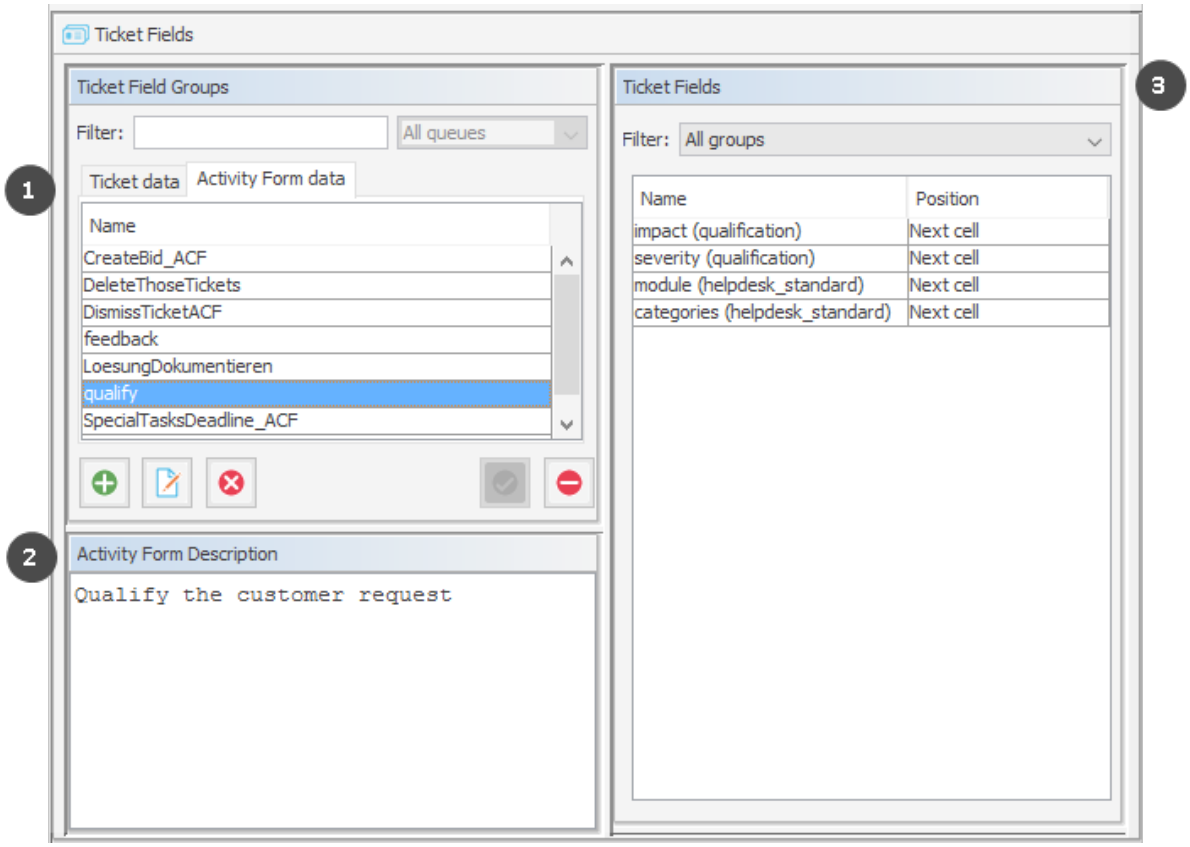


Figure 46: *ConSol CM Admin Tool - Tickets, Ticket Fields: Activity Form data*

The *Activity Form data* tab consists of three sections:

- **Activity Form data (1):**
  Contains the available Activity Forms
- **Activity Form Description (2):**
  Contains the description of the selected Activity Form
- **Ticket Fields (3):**
  Contains the ticket fields of the selected Activity Form

B.1.4.3 Create an Activity Control Form

To create an ACF just click the *Add* button below the list on the left side of the page. The following pop-up window appears (this is the same window for creating and for editing an ACF).

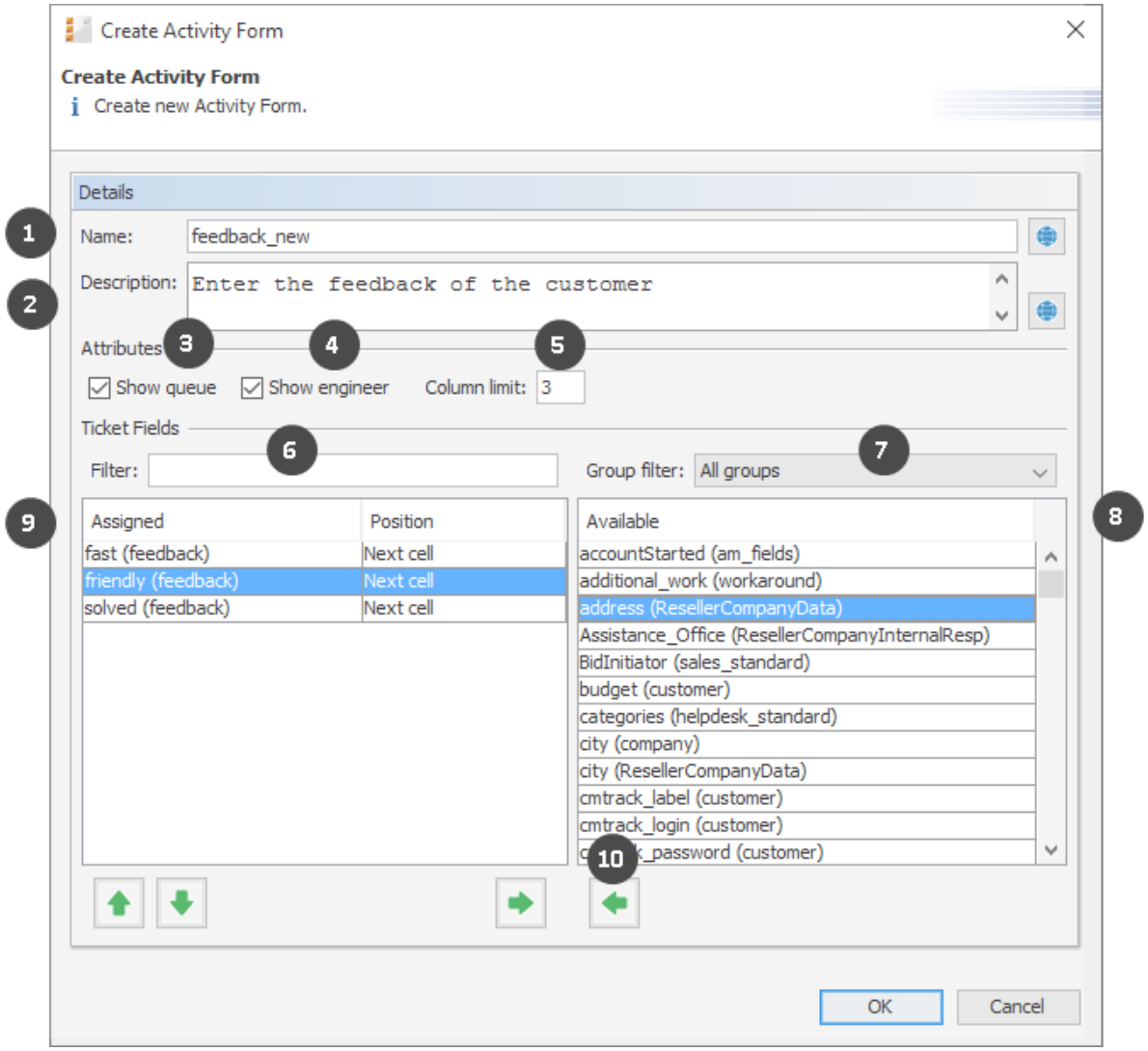


Figure 47: *ConSol CM Admin Tool - Tickets, Ticket Fields: Creating an Activity Control Form*

Please enter or select the following data:

- **Name (1)**
  Enter the name of the ACF in this field. You can localize the name by clicking the *Localize* button. For details, see section Localization of Objects in General, Type 1.
- **Description (2)**
  Enter a description for the ACF in this field. The description is shown as the title of the ACF in the Web Client. You can localize the description by clicking the *Localize* button. For details, see section Localization of Objects in General, Type 1.
- **Show queue (3)**
  Check this box if the queue of the ticket should be displayed with the ACF in the Web Client.
- **Show engineer (4)**
  Check this box if the engineer of the ticket should be displayed with the ACF in the Web Client.
- **Column limit (5)**
  Number field. Defines the number of columns for the display of ticket fields in the ACF. *0* means no column limit, i.e., all ticket fields in a single row. You can also work with the parameter *Display in new row* for each ticket field, see below.

- **Filter (6)**
  You can enter a string of characters into this field to filter the assigned ticket fields by name.
- **Group filter (7)**
  Select a group of ticket fields from this list if you want to display only ticket fields belonging to this group in the list of available ticket fields below.
- **Ticket field lists (8/9)**
  The list on the right (8) shows the available ticket fields with their respective ticket field group. You can sort the entries in ascending or descending order by clicking into the title field of the list. The small up and down arrow icons show the sort order. Select the ticket fields for the ACF in this list and move them to the list *Assigned* on the left (9) by clicking the *Assign* button (10). For each assigned ticket field you can define the position. The following values are possible:
  - **Next row**
    A new row will be displayed, starting with this field.
  - **Next cell** (default)
    The field will be displayed next to the previous field, no new row.
  - **New row (full width)**
    The field is displayed in a new row and spans to whole available width. This option should be used for data structures which need a lot of space, e.g. tables.

The assigned ticket fields will appear in the Web Client as they are ordered in the list. You can rearrange the list by selecting an item and clicking the *Move upwards* or *Move downwards* button. To remove assigned ticket fields, select them and click the *Unassign* button.

Click *OK* afterwards to store your entries and to close the window.

B.1.4.4 Edit an Activity Control Form

If you want to edit an ACF, select it in the list and click the *Edit* button or just double-click the name of the ACF. The same pop-up window as for creating an ACF will appear, where you can modify the details. Store your changes by clicking *OK*.

B.1.4.5 Delete an Activity Control Form

An ACF can only be deleted if it is not assigned to a workflow activity, otherwise you get a warning stating that you can only disable this ACF (see below). In order to delete an ACF, select it in the list and click the *Delete* button. If you confirm the following dialog with *Yes*, the ACF will be removed from the list and the system.

B.1.4.6 Enable or Disable an Activity Control Form

If you cannot delete an ACF, or if you do not want to delete it because you might need it again, you can disable it. To do so select the ACF and click the *Deactivate* button. The entry in the list is shown in italics afterwards. A disabled ACF is not available in the Process Designer. ACFs which are in use cannot be disabled. Just click the *Activate* button below the ACF list if you want to enable the form again.

B.1.4.7 Localize an Activity Control Form

By clicking the *Localize* button in the create or edit window you can localize name and description of an ACF. For details, please refer to the section Localization of Objects in General, Type 1.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

B.1.5 Frequently Used Annotations

Here are some frequently used annotations on field level. You can find a complete list of group and field annotations in Annotations.

groupable 106
sortable 106
readonly 106
visibility 106
text-type 106
reportable 107
field indexed 108
position 108
colspan 108
rowspan 108
field-group 109
fieldsize 109
enum field with ticket color 109
accuracy 109
format 110
maxLength 110
minLength 110
required 110

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

groupable

- **Type**: cmweb-common
- **Description**: Enables grouping of the ticket list by this field. Please see section Grouping for a detailed explanation.
- **Values**:
  - *true*: Used only with ENUM data fields. Remove the annotation if you want to disable grouping.

sortable

- **Type**: cmweb-common
- **Description**: Used to enable sorting of the ticket list by this field. Please see also the detailed explanation in section Frequently Used Annotations
- **Values**:
  - *true*: Used for data fields of type DATE or of type ENUM. Remove the annotation if you want to disable sorting.
    For ENUM fields: Works only if order index is set for all values of the ENUM field.

readonly

- **Type**: common
- **Description**: Used to indicate that the ticket field cannot be modified.
- **Values**:
  - **all**: The field is always read-only.
  - **edit**: The field is read-only when creating or editing a ticket, customer or resource. It can be edited in forms.
  - **form**: The field is read-only in forms. It can be edited when creating or editing a ticket, customer or resource.
  - **unset**: The field is always editable. This is the default behavior if the annotation is not assigned.

visibility

- **Type**: common
- **Description**: Defines when the field is visible.
- **Values**:
  - *edit*: Field will be displayed in edit mode.
  - *view*: Field will be displayed in view mode.
  - *none*: Field is not visible.
  - If any other, or no value, is set then the field will always be visible.

text-type

- **Type**: component-type
- **Description**: Defines the possible types of a STRING field.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Values**:
  - *text* (default): Single-line input field
  - *textarea*: Multi-line input field
  - *password*: Input field for passwords.
    Password will be displayed as ******* in view mode.
  - *label*: Read-only label, no input is possible.
  - *title*: Read-only label styled as a headline, no input is possible.
  - *autocomplete*: The field will be used as autocomplete list. Please see the detailed explanation in section Scripts of Type Text Autocomplete.
  - *url*: The input will be displayed as a hyperlink in view mode. If no protocol is provided, *http://* is added automatically in front of the entered string. If a protocol, e.g. “http”, “https”, “mailto”, “file”, or “ftp” is used, the URL is rendered as is. The display text for the URL can be entered after a whitespace.
    Example: “http://consol.de ConSol”
  - *file-url*: Input will be displayed as a link to a file on the file system. The web browser has to allow/support those links! See section Using Annotations to Fine-Tune Data Fields on how to achieve this. The link will also be displayed as tooltip.

    The URL is correctly formed if the following conditions are met:

    It starts with `file:` followed by regular slashes:

    - three slashes “///” for files on the same computer as the browser (alternatively “//localhost/”) or
    - two slashes followed by the server name followed by another slash for files on file servers accessible from the computer running the browser.

    These are followed by the full path to the file ending with the file name. The path on Microsoft Windows systems is also written with forward slashes instead of backslashes.

    The drive letter of a local path on Microsoft Windows systems is noted as usual, for example C:. Paths with spaces and special characters like “{, }, ^, #, ?” need to be percent encoded (“%20” for a space for example) for Microsoft Windows systems.

    Example URLs:

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

    See also the explanation in the section Using Annotations to Fine-Tune Data Fields.

reportable

- **Type**: dwh
- **Description**: Indicates that the field is reportable and that it should be transferred to the DWH.
- **Values**:
  - *true / false*: Field is reportable if value is set to “true”.

field indexed

- **Type**: indexing
- **Description**: Indicates that a database index will be created for this field. This is necessary if the field should be used to search for entities or to sort entries in a table.
- **Values**:
  - *transitive*: Only relevant for customers. An index is created for the field, so the customer can be searched by this field. For contacts, the tickets of the contact are also found. For companies, the tickets of the company and the contacts of the company are found as well.
  - *unit*: Only relevant for customers. An index is created for the field, so the customer can be searched by this field. For contact, only the contact itself is found. For companies, the contacts of the company are found as well. This is the default value for customers.
  - *local*: An index is created for the field, so the object (ticket, customer or resource) can be searched by this field. Only the object itself is found. This is the default value for tickets and resources.
  - *<annotation not set>*: No index is created for the field, so it it cannot be searched.

position

- **Type**: layout
- **Description**: Defines the position of a field within a grid layout or defines the position of a field within a list (STRUCT).
- **Values**:
  - **<number>;<number>**: Values define row and column (row;column), numbering starts at 0;0. If no values are set, the data field will take the next free grid cell.
  - **0;<number>**: Only the column value is used, the row value is ignored.

colspan

- **Type**: layout
- **Description**: Defines how many columns are reserved for the field in the layout.
- **Values**:
  - **<number>**: Number of columns.

⚠ This annotation only works if the annotation `position` is also set for the field.

rowspan

- **Type**: layout
- **Description**: Indicates how many rows within the layout are occupied by this field.
- **Values**:
  - **<number>**: Number of rows.

⚠ This annotation only works if the annotation `position` is also set for the field.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

⚠ This annotation does not work in version 6.13.0 and higher.

field-group

- **Type**: layout
- **Description**: Allows grouping of fields in view mode. Annotation is ignored in edit mode.
- **Values**:
  - **<string>**: To group fields the same string value has to be set in the annotation of each field. Two or more data fields are bound when they share the same value for this annotation. The group of coupled data fields is shown only if all of them have values set.
- Removed in ConSol CM version 6.11.0.1.

fieldsize

- **Type**: layout
- **Description**: Displayed field size within the ticket layout.
- **Values**:
  - **<rows>;<cols>**: Displayed field size for textareas

    For STRING fields with `text-type` = `textarea`: rows;cols (corresponds to <textarea rows="" cols="">).
  - **<number>**: Displayed field size for strings and numbers

    For STRING fields with `text-type` other than `textarea` and NUMBER fields: n indicates the number of characters in the fields; for string fields this is the number of monospaced capital M characters.

    For ENUM data fields: Defines how many values are directly visible in the list box. Used only for layout purposes. This annotation is not relevant for ENUM autocomplete fields.

ⓘ This is only a layout configuration, for validation use `maxlength` of type `validation`.

enum field with ticket color

- **Type**: ticket display
- **Description**: Defines the background color of the ticket icon for ticket list and ticket.
- **Values**:
  - *true / false*: The field has to exist within Enum Administration where lists, values, and colors are defined.

accuracy

- **Type**: validation
- **Description**: For ticket, customer and resource fields of type DATE. To define the level of detail displayed

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Values**:
  - *date (default)*: Show date without time.
  - *date-time*: Show date with time.
  - *only-time*: Show only time, no date.

format

- **Type**: validation
- **Description**: Used for validating the format of date fields.
- **Values**:
  - *<date format>*: The pattern for the date is based on `SimpleDateFormat`, e.g., dd.MM.yyyy.
    Remember to set the proper colspan when including hours/minutes in the format. See http://docs.oracle.com/javase/6/docs/api/java/text/SimpleDateFormat.html for the format reference.



This annotation is applied to date fields both in the Web Client and in CM/Track. If it is not set, the standard date format of the browser locale is used in the Web Client, and the German standard date format (dd.MM.yyyy) is used in CM/Track.

maxLength

- **Type**: validation
- **Description**: Defines the maximum length of input for STRING data fields.
- **Values**:
  - **<number>**: May be used with STRING data fields.

minLength

- **Type**: validation
- **Description**: Defines the minimum length of input for STRING data fields.
- **Values**:
  - **<number>**: May be used with STRING data fields.

required

- **Type**: validation
- **Description**: Indicates that this is a required field.
- **Values**:
  - *true / false*: Field is required if value is set to “true”. The user cannot save the ticket without having entered a value in a required field. In the Web Client, required fields are marked by a red asterisk.

B.2 Managing Sorted Lists: Enums

This chapter discusses the following:

B.2.1 Introduction 111

B.2.2 Enum Administration Using the Admin Tool 113

B.2.1 Introduction

Enums, also called sorted lists, are lists with predefined list values. You define an enum in the navigation item *Enums* from the navigation group *Tickets*. Enums are defined once and can be used in several places:

- as a selection list (in drop-down menus) for ticket fields, customer fields, or resource fields of type ENUM
- as hierarchical lists for ticket fields, customer fields, or resource fields of type *MLA field* (Multi Level Attributes, see section MLAs)
- as dependent enums, i.e., as enums that form a hierarchy, a data construct implemented by Scripts of Type Dependent Enum



You only define the lists, i.e., the structures with various list values, in the enum administration. To display the enum in the Web Client (as ticket field values, customer field values, or resource field values), you have to complete one of the following steps:

- Create a ticket field of type *enum* and assign the respective enum there.
- Create a customer field of type *enum* and assign the respective enum there.
- Create a resource field of type *enum* and assign the respective enum there.
- Create an MLA which is linked automatically as ticket field to the ticket field group, as customer field to the customer field group, or as resource field to the resource field group that has been indicated during MLA set-up.
- Create a dependent enum script and assign it to a ticket field group, customer field group, or resource field group.

Examples:

A list of country names (Germany, Italy, France, etc.) is used in the customer field *Country* belonging to an address data set. The same list can also be used in the ticket field *Machine location* and in other data fields. Priority lists (high, normal, low, etc.) are other typical examples.

Depending on the value of the `enum-type` annotation, an enum field is displayed in the Web Client as follows:

- **drop-down menu**
  enum-type not set or enum-type = select
- **radio buttons**
  enum-type = radio
- **self-completing (autocomplete) list**
  enum-type = autocomplete

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

If the annotation `enum-type` is not set, a drop-down menu is displayed by default (see example in the picture below).

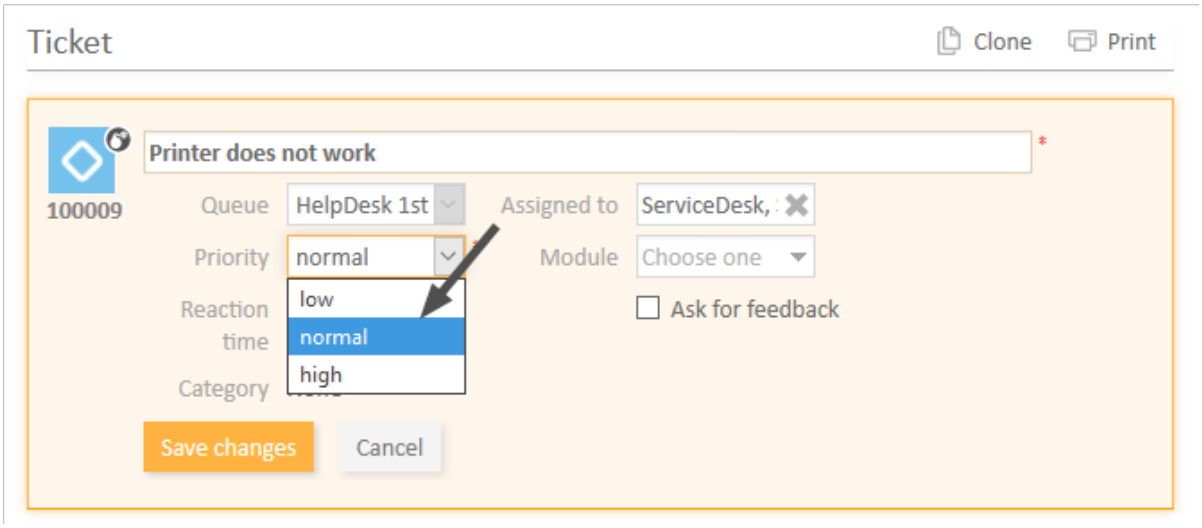


Figure 48: *ConSol CM Web Client - Enum for Priority (localized enum values displayed as list items)*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

B.2.2 Enum Administration Using the Admin Tool

The following figure shows the available sorted lists. You reach this screen by opening the navigation item *Enums* in the navigation group *Lists*.

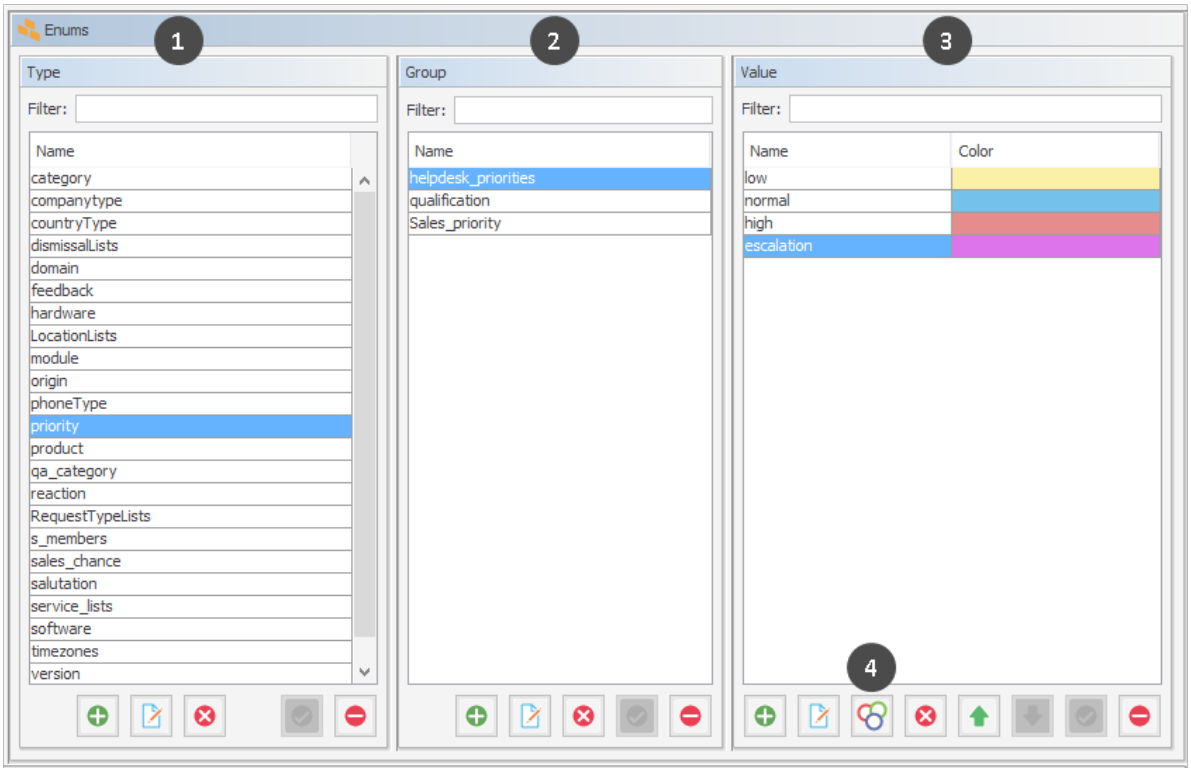


Figure 49: *ConSol CM Admin Tool - Lists, Enums*

Enums are organized in three levels:

- **Type (1)**
  The *type* helps to organize your lists within the Admin Tool. Its name is never displayed in the Web Client and does not have any other implications.
- **Group (2)**
  The *group* represents a group of enum values, i.e., the list.
- **Value (3)**
  The *value* represents one value within a list. You can pick a color for each value (4), which will be used as a background color for the ticket icon in the defined queue. You can use the up and down arrows to determine the value's position within the list in the Web Client.

B.2.2.1 Enum Types

Create an Enum Type

To create a new enum type just click the *Add* button below the list in the *Type* area on the left of the window. The following pop-up window appears.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

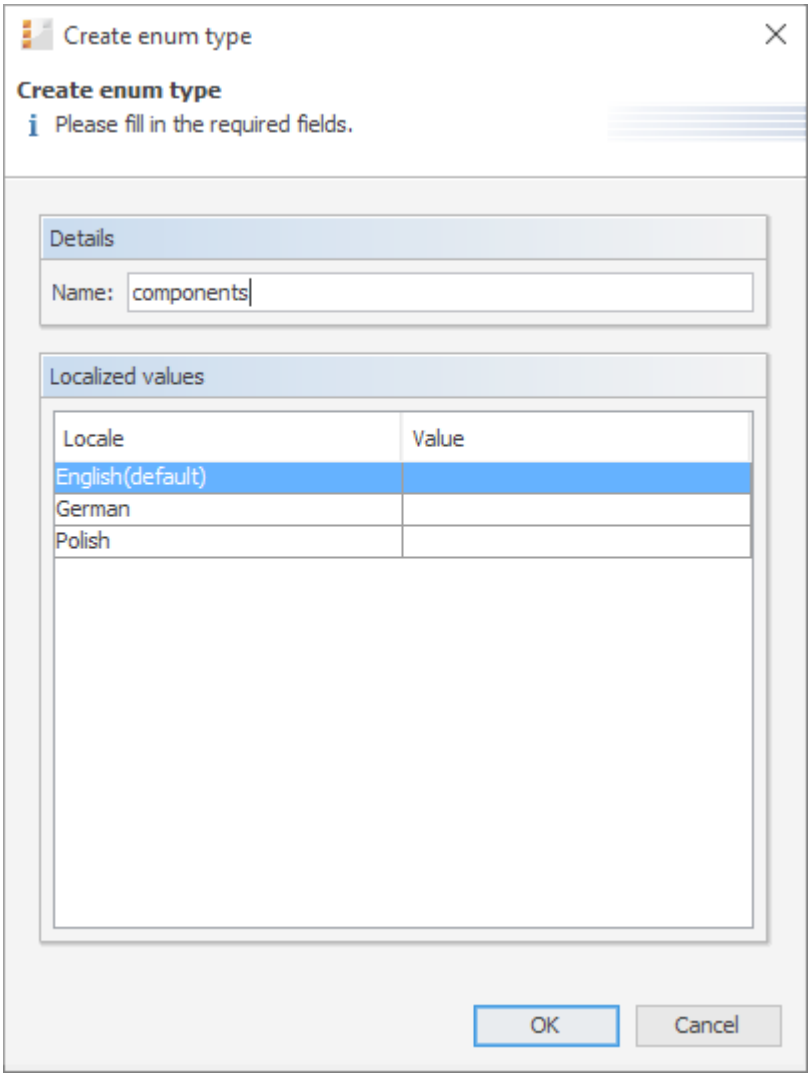


Figure 50: *ConSol CM Admin Tool - Lists, Enums: Creating an enum type*

- **Name:**
  Enter a name for the new enum type. The name must be unique.
- **Localized values:**
  Enter the corresponding type name in the *Value* field for each additional language. For details, please refer to the section Localization of Objects in General, Type 2.

Click *OK* to create the enum type and to close the window.

Edit an Enum Type

If you want to edit an enum type, select it in the list and click the *Edit* button. The same window as described above for creating an enum type will appear. You can modify all fields and save your changes by clicking *OK*.

Delete an Enum Type

An enum type can only be deleted if there are no enum groups for it anymore. Either you have to delete all groups belonging to this type first or you have to assign them to another type. In order to delete an enum type, select it in the list and click the *Delete* button. If you confirm the following dialog with *Yes*, the type will be removed from the list and the system.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Enable or Disable an Enum Type

If you do not want to delete an enum type because you might need it again, you can disable it. To do so select the type and click the *Deactivate* button. The entry in the list is shown in italics afterwards. Just click the *Activate* button below the type list, if you want to enable the type again.

If you disable an enum type, it cannot be selected for data fields of the type ENUM anymore. You can only disable enum types which are not in use, i.e., if the enum type is used in a data field, you have to disable the field before disabling the enum type.

B.2.2.2 Enum Groups

Create an Enum Group

An enum group represents a list, i.e., the enum group is a collection of list (enum) values. All groups of an enum type (i.e., all lists which belong to this type) are created and managed in the middle part of the *Enum Administration* window. To create a new enum group select the desired type on the left, then click the *Add* button below the *Group* area. The following pop-up window appears.

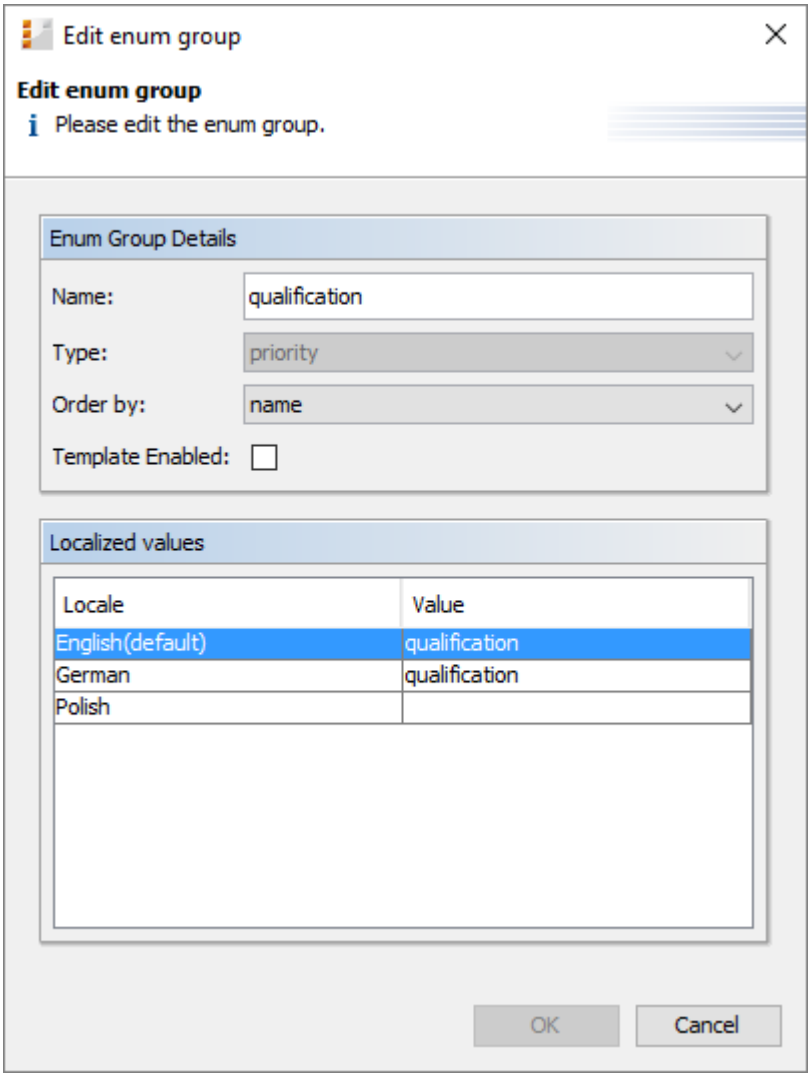


Figure 51: *ConSol CM Admin Tool - Lists, Enums: Creating an enum group*

- **Name:**
  Enter a name for the enum group. The name must be unique.

- **Type:**
  This field shows the selected enum type for the group. You can also choose any other type from the selection list, e.g., if you want to assign the group to a different type.
- **Order by:**
  Here you can define the way the values of a group shall be ordered:
  - **user-defined**
    You can specify the sort order by means of arrow icons below the value list.
  - **name**
    The values will be ordered alphabetically.
- **Template Enabled:**
  When this value is set to *true* (checkbox ticked), the enum is offered as *enum parameter* in the Text Template Manager. This is the only module where this value is relevant. If you do not know what to do with the checkbox, leave it empty. Please see section The ConSol CM Text Template Manager, String and Enum Parameters.
- **Localized values:**
  Enter the corresponding group name in the *Value* field for each additional language. For details, please refer to the section Localization of Objects in General, Type 2.

Click *OK* to create the enum group and to close the window.

Edit an Enum Group

If you want to edit an enum group, select it in the list and click the *Edit* button. The same window as described above for creating an enum group will appear. You can modify all fields and save your changes by clicking *OK*.

Delete an Enum Group

An enum group can only be deleted if it is not used in a ticket or an MLA. In order to delete a group, select it in the list and click the *Delete* button. If you confirm the following dialog with *Yes*, the group will be removed from the list and the system.

Enable or Disable an Enum Group

An enum group cannot be disabled if it is still used in an MLA.

If you cannot delete an enum group or if you do not want to delete it, because you might need it again, you can disable it. To do so select the group and click the *Deactivate* button. The entry in the list is shown in italics afterwards. Just click the *Activate* button below the group list, if you want to enable the group again.

If you disable an enum group, it cannot be selected for data fields of the type ENUM anymore. You can only disable enum groups which are not in use, i.e., if the enum group is used in a data field, you have to disable the field before disabling the enum group.

B.2.2.3 Enum Values

Create an Enum Value

The individual values of an enum group (i.e., the list values) are created in the right part of the window. Select the desired group and click the *Add* button below the *Value* area. The following pop-up window appears.

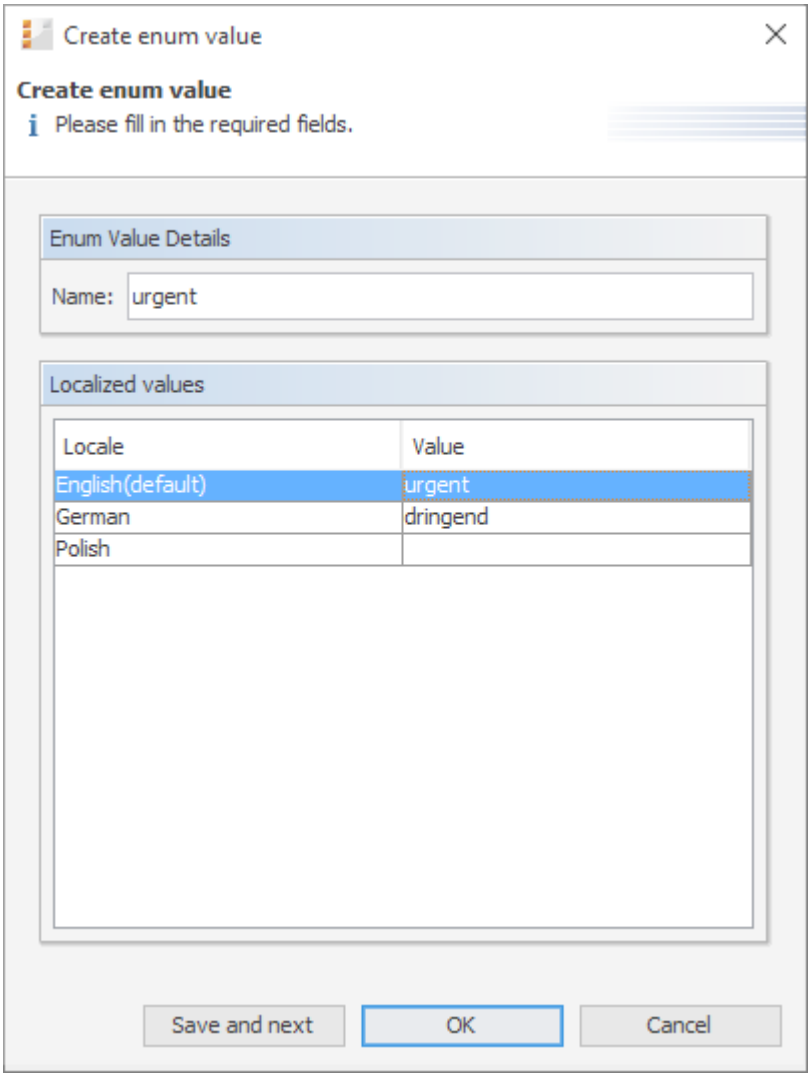


Figure 52: *ConSol CM Admin Tool - Lists, Enums: Creating an enum value*

- **Name:**
  Enter a value which will be displayed in the sorted list on the Web Client.
- **Localized values:**
  Enter the corresponding value name in the *Value* field for each available language. For details, please refer to the section Localization of Objects in General, Type 2.

Click *Save and next* if you want to continue to create values for this enum group. To finish the creation of the list click *OK*.

Edit an Enum Value

If you want to edit an enum value, select it in the list and click the *Edit* button. A pop-up window will appear where you can modify the name and the localized values. Click *OK* to save your changes.

Set a Background Color

You can assign a color to a selected enum value by clicking the *Color* button. This can be used, for example, for priorities. The priority of a ticket in the Web Client can be recognized immediately by the background color of the ticket icon. This will be effective when the respective annotation is set, see the following info box.

Please note that only one enum can determine the color of the ticket icon for a queue. You have to assign the annotation `enum field with ticket color` to the respective ticket field of type ENUM in the Ticket Fields (Setting Up the Ticket Data Model). For example, you can use the ticket field *priority* to determine the ticket icon color in the *helpdesk* queue and the ticket field (enum) *likelihood of closing a deal* in the *sales* queue.

The pop-up window contains a range of colors from which you can choose the desired background color. Click the desired color to set it for the marked list value. You can check the selected color in the *Preview* area. Click *OK* to save your choice. Click *Reset* if you want to restore the last saved color.

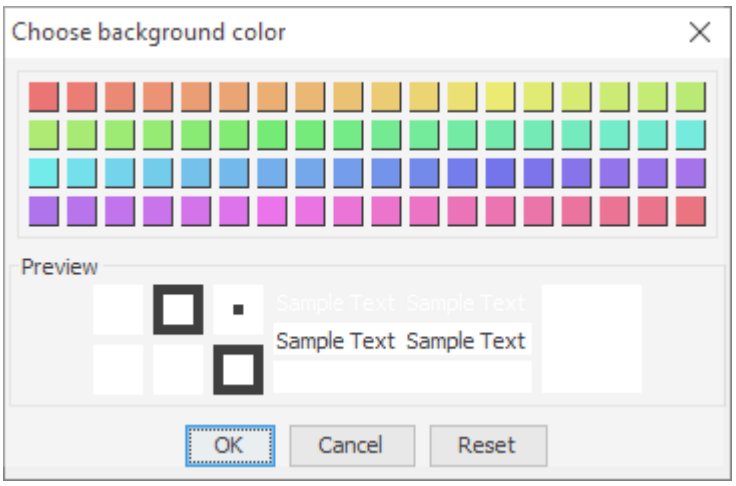


Figure 53: *ConSol CM Admin Tool - Lists, Enums: Setting a background color for an enum value*

Delete an Enum Value

An enum value can only be deleted if it is not used in an MLA. To delete a value, select it in the list and click the *Delete* button. If you confirm the following dialog with *Yes*, the value will be removed from the list and the system.

Before you delete an enum value, make sure there are no references to it in workflow scripts! This is not checked in the Admin Tool!

Change the Order of the Value List

If you have chosen *user-defined* in the *Order by* field of an enum group, you can arrange the enum values by using the arrow icons below the list. Click the *Move upwards* button to move the selected value one line up resp. click the *Move downwards* button to move it one line down. If you change the value for *Order by* from *user-defined* to *name*, the enum values will automatically be ordered by name in alphabetical order.

Enable or Disable an Enum Value

An enum value cannot be disabled if it is still used in an MLA.

If you do not want to delete an enum value because you might need it again, you can disable it. To do so, select the value and click the *Deactivate* button. The entry in the list is shown in italics afterwards and the value is not available in the Web Client any longer. Just click the *Activate* button below the value list if you want to enable the value again.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

B.2.2.4 Placing an Enum in the Data Model

Enums can be used in ticket fields (i.e. for ticket data), customer fields (i.e. for customer data) and resource fields (i.e. for resource data). The example below shows an enum for ticket data.

Enums for Ticket Data

In order to place an enum in the ticket data model, i.e., to make it available in queues and visible in the Web Client, a ticket field of type ENUM has to be defined. Select ENUM as a data type (1) and select the enum type and group which the enum belongs to (2). Please see section Ticket Fields (Setting Up the Ticket Data Model) for a detailed explanation of the work with ticket fields.

Enums for Customer Data

In order to place an enum in the customer data model, i.e., to make it available for company or contact data in the Web Client, a customer field of type ENUM has to be defined. Select ENUM as a data type (1) and select the enum type and group which the enum belongs to (2). Please see section Customer Field Management and GUI Design for Customer Data for a detailed explanation of the work with customer fields.

Enums for Resource Data

In order to place an enum in the resource data model, i.e., to make it available in resources in the Web Client, a resource field of type ENUM has to be defined. Select ENUM as a data type (1) and select the enum type and group which the enum belongs to (2). Please see section CM/Resource Pool - Setting Up the Basic Resource Model for a detailed explanation of the work with resource fields.

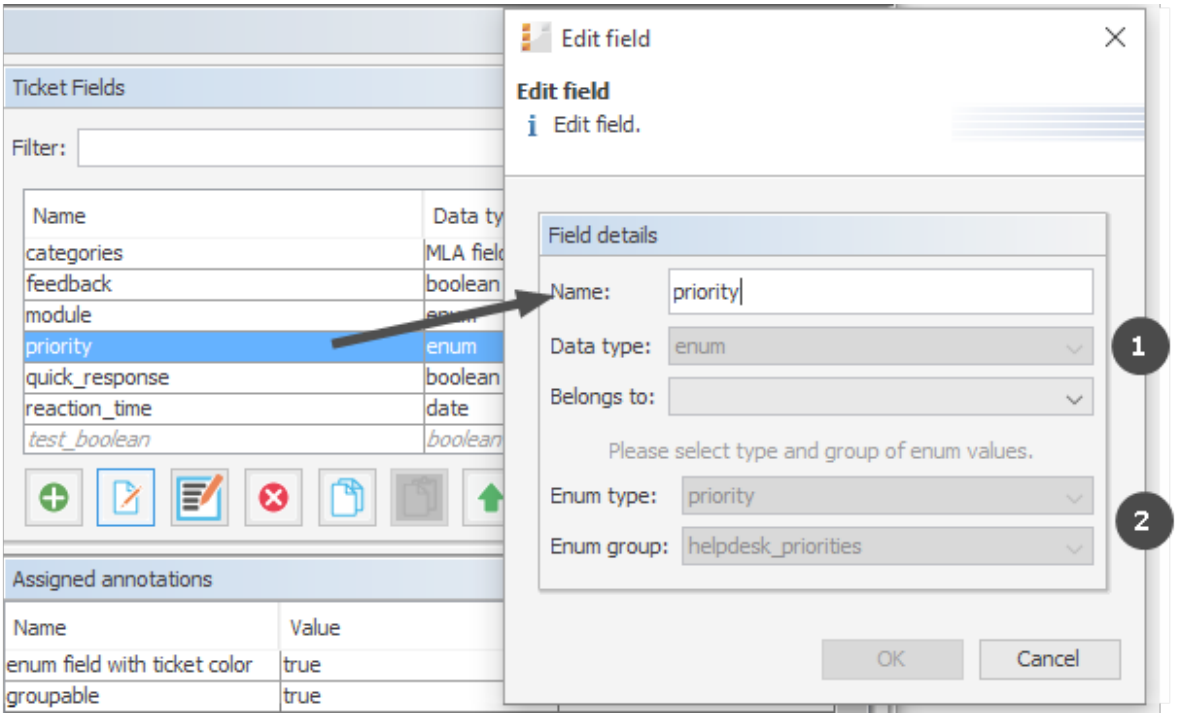


Figure 54: *ConSol CM Admin Tool - Tickets, Ticket Fields: Defining a ticket field of type enum*

B.3 MLAs

This chapter discusses the following:

B.3.1 Introduction 120

B.3.2 MLA Administration Using the Admin Tool 122

B.3.1 Introduction

MLA is the abbreviation for Multi Level Attribute. An MLA is used to represent a hierarchical data set and consists of several lists which form a tree structure. Each item of a list can lead to a list of the next level with the item name being the name of the subordinate list. An MLA can be used for ticket data, customer data or resource data.

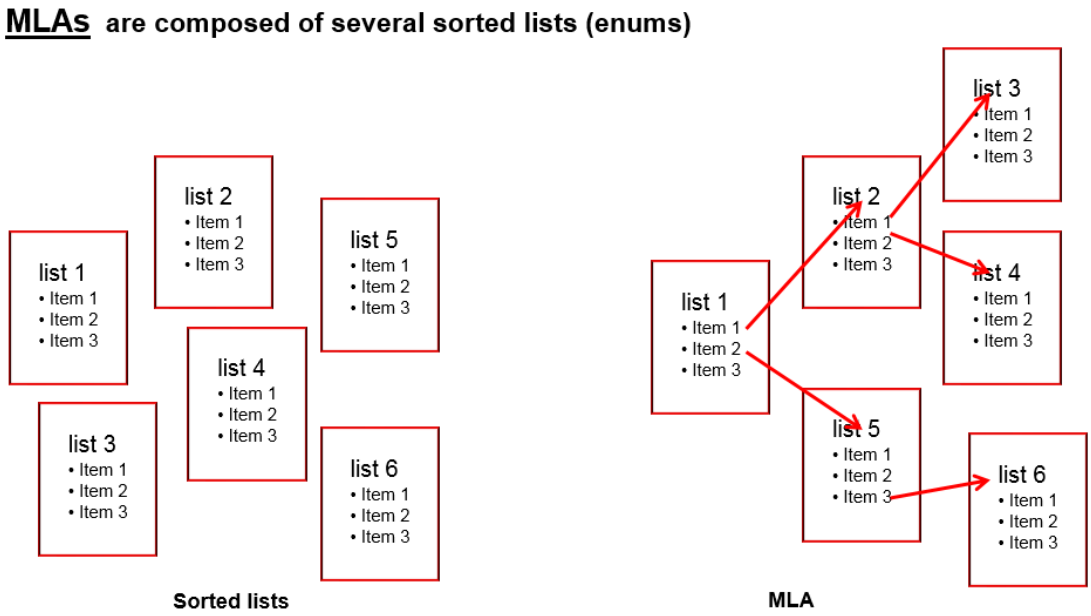


Figure 55: *ConSol CM Admin Tool - MLA construction principle*

Example:

For quality management you need to specify hardware or software products in a ticket. For this purpose you can create an MLA with the name *QA_MLA*. The next step will be to create the first level with the items *Hardware* and *Software*. For each item of a level you can create further levels, e.g. *Graphics Card*, *Monitor*, and *Motherboard* for item *Hardware* and so on. The picture below shows such an MLA in the Web Client.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

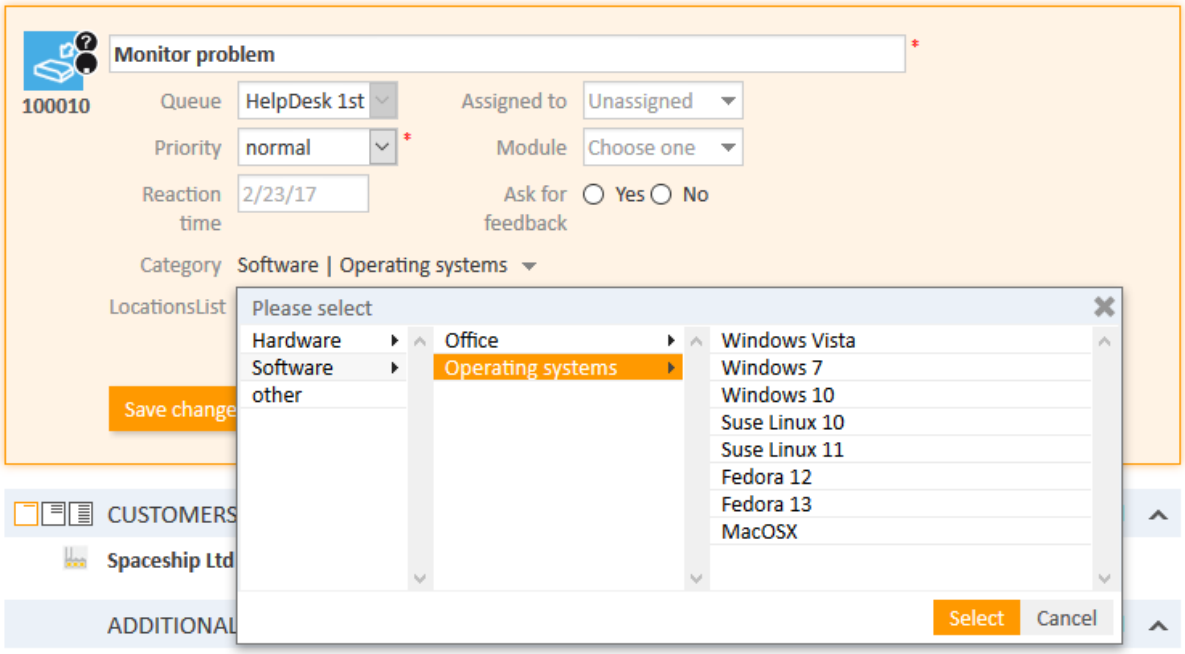


Figure 56: *ConSol CM Web Client - MLA for hardware and software selection*

The sorted lists (enums) for each level of an MLA ...

- can be created within the Managing Sorted Lists: Enums and are only referenced when a new MLA level is defined.
- can be completely created within the MLA administration during the set-up of a new MLA.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

B.3.2 MLA Administration Using the Admin Tool

The following figure shows the available MLAs. You reach this screen by opening the navigation item *MLAs* in the navigation group *Lists*.

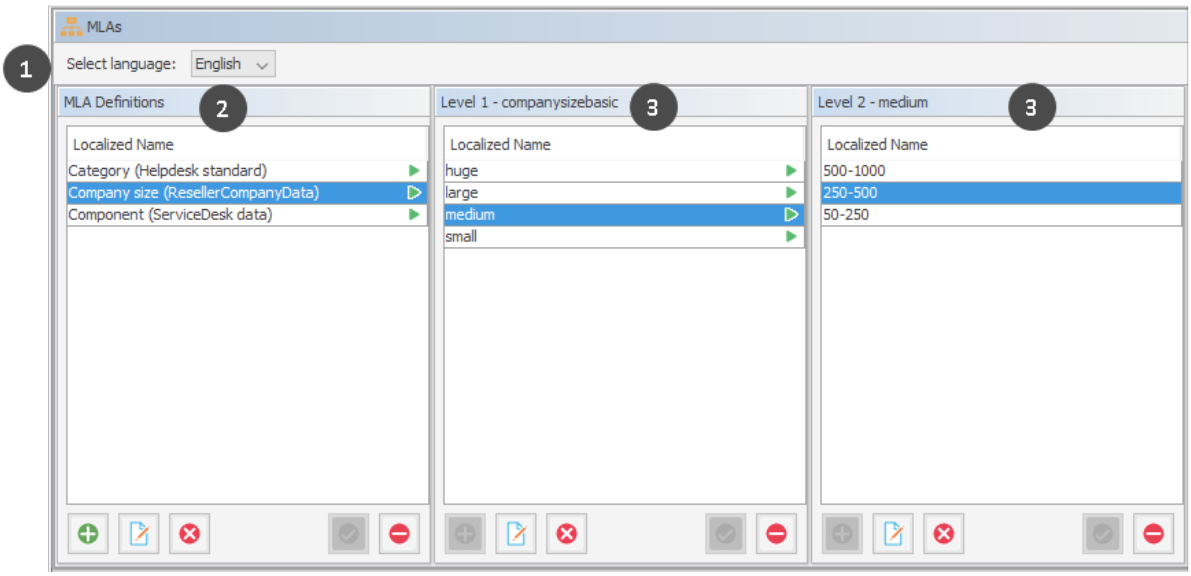


Figure 57: *ConSol CM Admin Tool - Lists, MLAs*

All entries are shown with their localized names (i.e., how they are displayed on the Web Client) in the selected language. You can change the display language of this page by choosing a different locale in the *Select language* field (1) above the list. The first column of the list contains the available MLAs (2). The following columns contain the ENUM values for the respective levels (3).

B.3.2.1 Create an MLA

To create an MLA click the *Add* button below the MLA list in the bottom left corner of the page. The following pop-up window appears.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

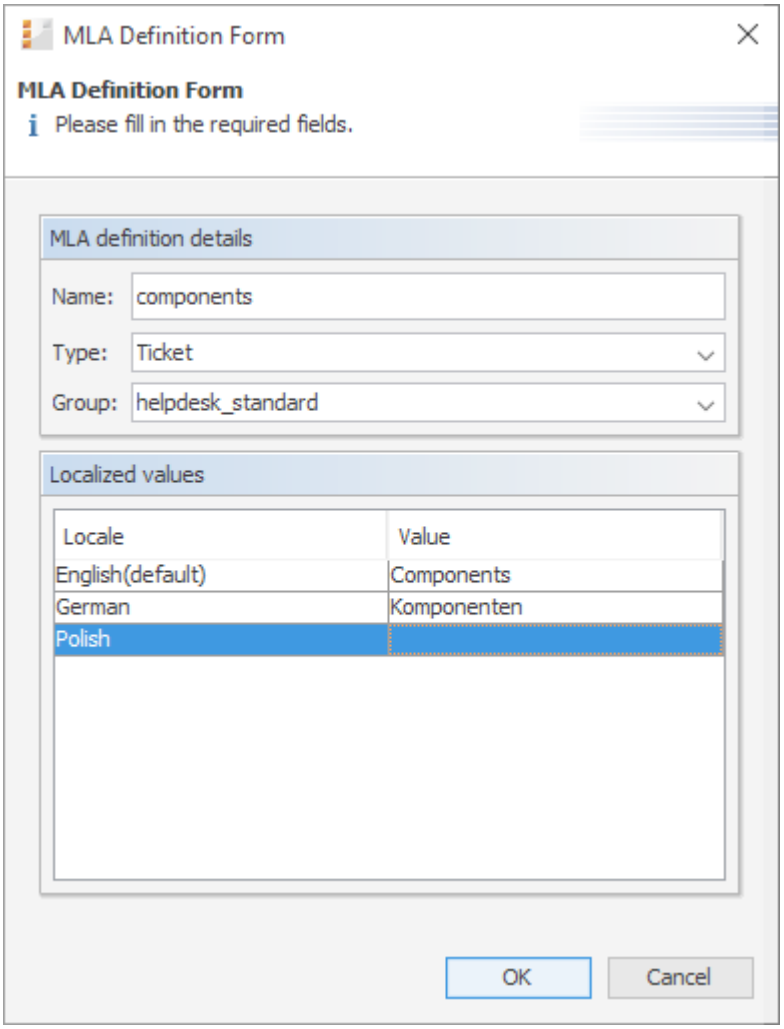


Figure 58: *ConSol CM Admin Tool - Lists, MLAs: Creating an MLA*

- **Name:**
  Enter a name for the new MLA. The name must be unique.
- **Type:**
  - **Ticket**
    MLA will be used in ticket data, i.e., in a ticket field.
  - **Customer object**
    MLA will be used in customer data, i.e., in a customer field.
  - **Resource**
    MLA will be used in resource data, i.e., in a resource field.
- **Group:**
  Choose the required ticket field group (ticket data), customer field group (customer data) or resource field group (resource data) in the list box. For the new MLA a ticket field, customer field or resource field of type *MLA field* will be created automatically in this group. This is necessary to display the MLA in the Web Client. The ticket field, customer field or resource field can be annotated as described in sections Ticket Fields (Setting Up the Ticket Data Model), Customer Field Management and GUI Design for Customer Data, and CM/Resource Pool - Setting Up the Basic Resource Model.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Localized values:**
  Enter the corresponding MLA name in the *Value* field for each additional language. For details, please refer to section Localization of Objects in General, Type 2.

Click *OK* to save the details of the new MLA.

> You can also create the ticket field, customer field, or resource field for the MLA first. In this case you will find the localized name of the ticket field, customer field, or resource field already in the list of available MLAs.

Create an MLA Level

Having created a name and a ticket field, customer field, or resource field for the MLA you can go on with the definition of levels. Select the MLA in the list and click the *Add* button below *Level 1*. You will get the *Enum level form* where you can specify an enum for this level.

This menu is only available during the process of creating a new MLA, not when editing an existing MLA.

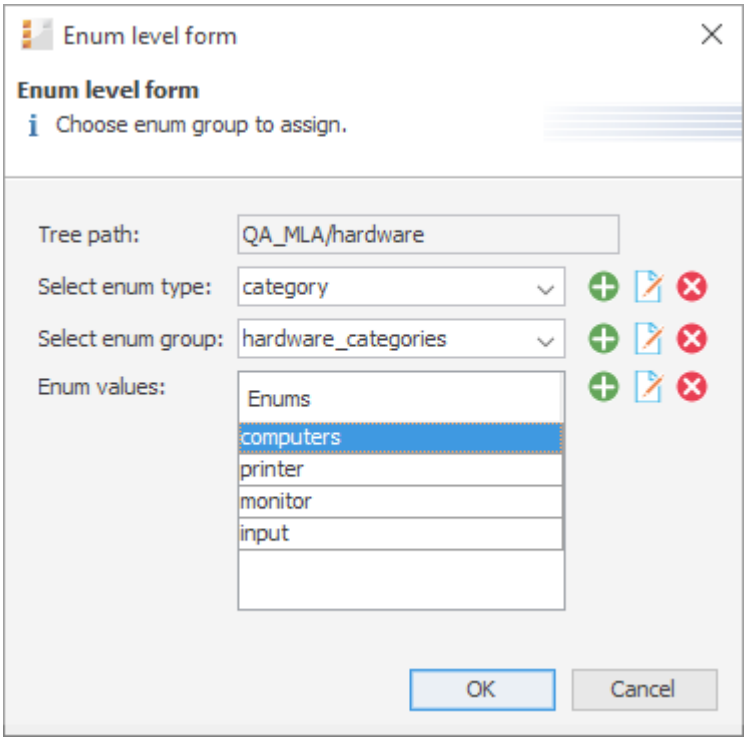


Figure 59: *ConSol CM Admin Tool - Lists, MLAs: Creating an MLA level*

- **Tree path:**
  This field shows the tree path of the new MLA level. Thus you can always see the position of the level within the MLA. The field is read-only.
- **Select enum type:**
  Choose an enum type from the list to use the corresponding enum groups (which have been created in enum administration first) or create a new enum type directly here in the MLA administration. The new type will also be visible in enum administration afterwards.

- **Select enum group:**
  Choose the desired enum group for this level from the lists in enum administration which are located within the selected type. If you have created a new enum type in the previous step, you also have to create a new enum group. The new enum group will also be visible in enum administration afterwards.
- **Enum values:**
  These are the list values of the new level which will be displayed in the Web Client. You can either take the list as-is or you can enter/add or delete values. The changes will be immediately visible in enum administration. If you have created a new enum group, you also have to create one or more enum values for the new group. You can either create all the enums you need before you start to define an MLA, or create an enum during the definition of a level in the MLA Administration by clicking the *Add* button next to the respective fields in the window. By clicking the *Edit* button or the *Delete* button, you can also edit or delete enum types, groups, and values here, but please consider that changes will affect other MLAs using the same enum. You cannot delete an enum if it is used in another MLA.

Click *OK* to create the new MLA level and to close the window.

For each value of a level you can create further levels as previously described. Just select the value in the list and click the *Add* button below the next level area to the right.

If you have finished your MLA definition and see that you need an additional value for one of the levels, you have to create that value in the respective enum group within Managing Sorted Lists: Enums.

Please note that an ENUM can only be used once within an MLA. It can be re-used in other MLAs though.

Edit a Level Value

If you want to edit a value of the level, select it in the list and click the *Edit* button. You can change the object name and the localized values but please consider that changes will affect other MLAs using the same enum.

Delete a Level

A level can only be deleted if it is not used in a ticket. In order to delete it click the *Delete* button below the respective level. If you confirm the following dialog with *Yes*, the level and all its dependent levels will be removed from the list and the system.

Enable or Disable a Level

If you cannot delete a level, or if you do not want to delete it because you might need it again, you can disable it. Just click the *Deactivate* button below the respective level. The level values (including the values of dependent levels) are shown in italics afterwards. Click the *Activate* button if you want to enable the level again.

B.3.2.2 Edit an MLA

If you want to edit an MLA, select it in the list and click the *Edit* button. The same window as described above for creating an MLA will appear. You can modify all fields except the ticket field group, customer field group, or resource field group. Click *OK* to save your changes.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

B.3.2.3 Delete an MLA

You can only delete an MLA if it has not been used. If it has been used, you get a warning stating you can only disable this MLA (see below). In order to delete an MLA select it in the list and click the *Delete* button. If you confirm the following dialog with *Yes*, the MLA (and the ticket field within *Ticket Field Administration*, customer field within *Customer Data Model*, or resource field within *Resource Model*) will be removed from the list and the system.

B.3.2.4 Enable or Disable an MLA

If you cannot delete an MLA, or if you do not want to delete it because you might need it again, you can disable it. To do so select the MLA and click the *Deactivate* button. The entry in the list is shown in italics afterwards. A disabled MLA will not be displayed in the Web Client. Just click the *Activate* button below the MLA list if you want to enable the MLA again.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

B.4 Ticket History

This chapter discusses the following:

B.4.1 Introduction 127
B.4.2 Editing Comments in the Ticket History 128
B.4.3 Display Modes of Ticket History in the Web Client 129
B.4.4 General Information about the Visibility of Ticket History Entries in the Web Client 135
B.4.5 Ticket History Storage and Transfer to the Data Warehouse (DWH) 138

B.4.1 Introduction

For each ticket, the ticket history is stored in the ConSol CM database, i.e., the entire life cycle of a ticket can be traced. This comprises actions like workflow activities or incoming emails as well as automatic and system actions like the change of the responsible engineer or to check if a time-based trigger has to fire. For those of you familiar with ConSol CM version 5: this is the equivalent to the ticket protocol.

The ticket history is displayed in the *History* section on the ticket page.

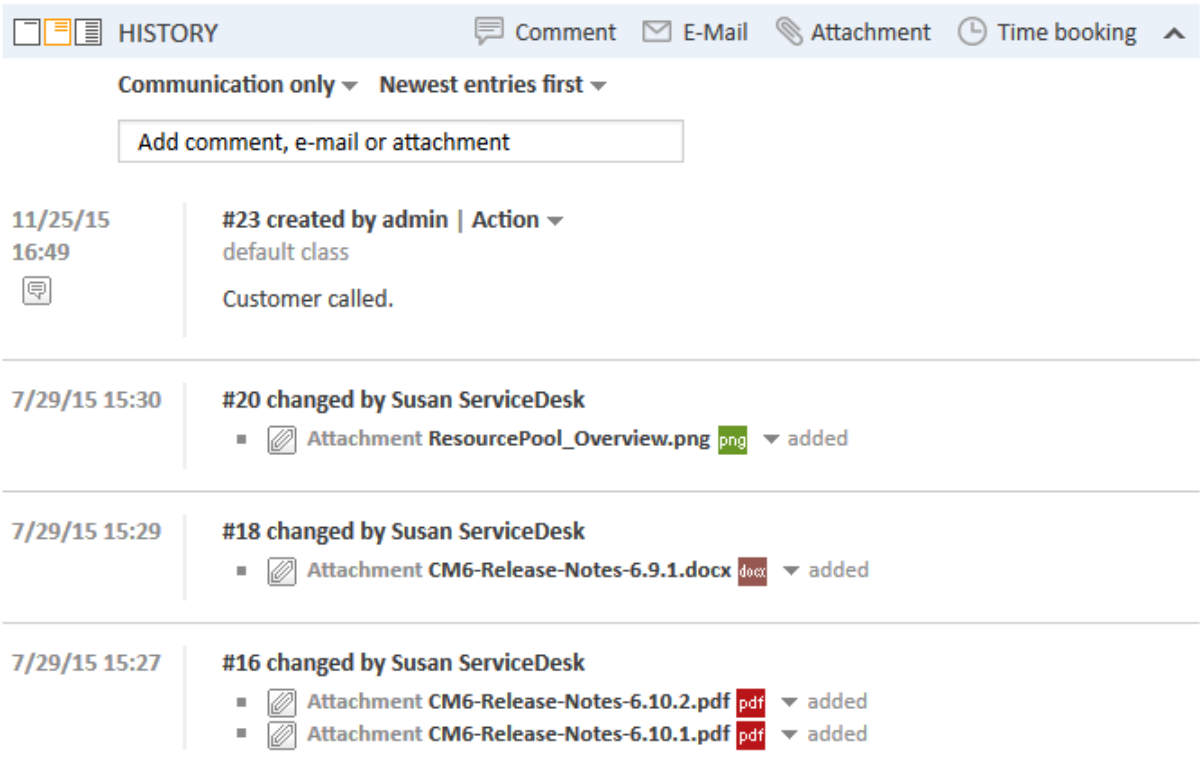


Figure 60: *ConSol CM Web Client - History section*

In case the DWH is in use (see section Data Warehouse (DWH) Management), you can configure if the history data should be transferred to the DWH, see section Ticket History Storage and Transfer to the Data Warehouse (DWH)on this page.

B.4.2 Editing Comments in the Ticket History

The ticket history is read-only, i.e., ticket history entries cannot be modified. Nevertheless, in the case of comment entries, the engineer can edit his own comments during a determined period of time. The time period is configured using the page customization attribute allowedTimeToEditTextEntry of the scope *acimSection*. The functionality to edit comments can be deactivated by setting the time period to “0”.

The ticket history always shows the newest version of the comment, and includes a link to open the comment history.



Engineers can only edit comments created by themselves. They can never edit comments created by other engineers.

Comments can be edited using scripts. A script can change comments of any user. In this case the comment author and the author of all modifications to the comment is set to the user who executes the script.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

B.4.3 Display Modes of Ticket History in the Web Client

The display mode for the ticket history can be configured on the navigation item *History* in the navigation group *Tickets*. On this item you can configure the visibility level for each action or event that has taken place concerning a ticket. The entries of the indicated type(s) will be visible in the ticket history if the user has selected the respective visibility level. This is of importance if the display mode *Display all entries* is used.

> The visibility of emails, comments and attachments (i.e., their visibility level and if the text entries are displayed full or short) can be controlled by Classes of Text. The visibility of entries with no class of text assigned to them is determined by the default class of text (in a default installation of ConSol CM, the default class for emails and comments is named *default_class* and the default class for attachments is named *default_attachment_class*).

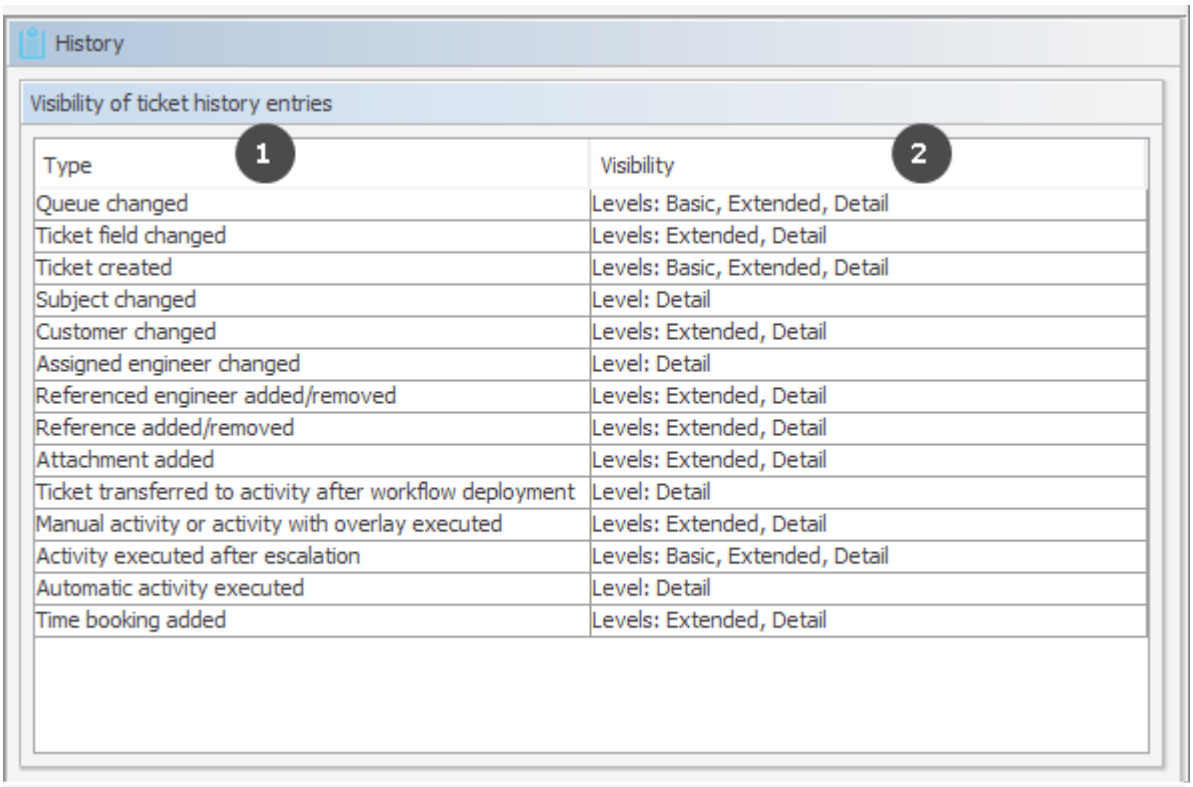


Figure 61: *ConSol CM Admin Tool - Tickets, History*

The editing panel for the ticket history shows a list of all configured values, each with:

- **Type (1)**
  The type of action that has been performed.
- **Visibility (2)**
  The visibility level in the Web Client. There are three levels:
  - Basic (1st level)
  - Extended (2nd level)
  - Detail (3rd level)

The following figures show the action type *time booking added* configured for *Basic* and *Extended*.

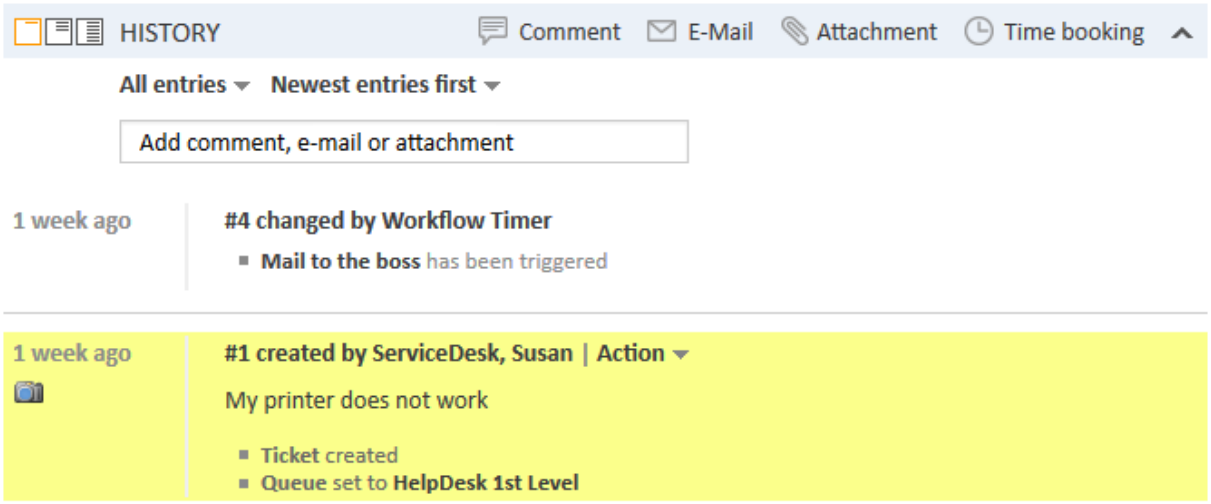


Figure 62: *ConSol CM Web Client - Time booking entry not visible on Basic level*

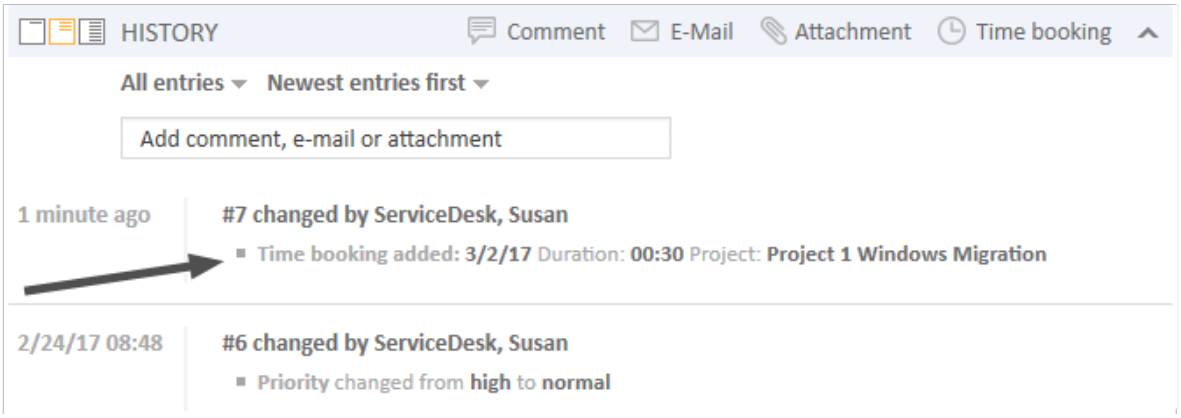


Figure 63: *ConSol CM Web Client - Time booking entry visible on Extended level*

It is not possible to add new action types to the list. To edit the visibility for an existing entry, double-click the visibility value you would like to modify and select the desired option from the drop-down menu.

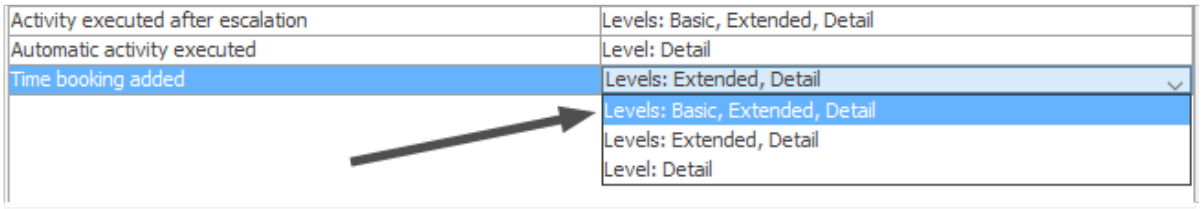


Figure 64: *ConSol CM Admin Tool - Tickets, History: Changing the visibility level for an action type*

The next picture shows the visibility for the action type *time booking added* after the setting has been modified on the navigation item *History* in the navigation group *Tickets*:

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

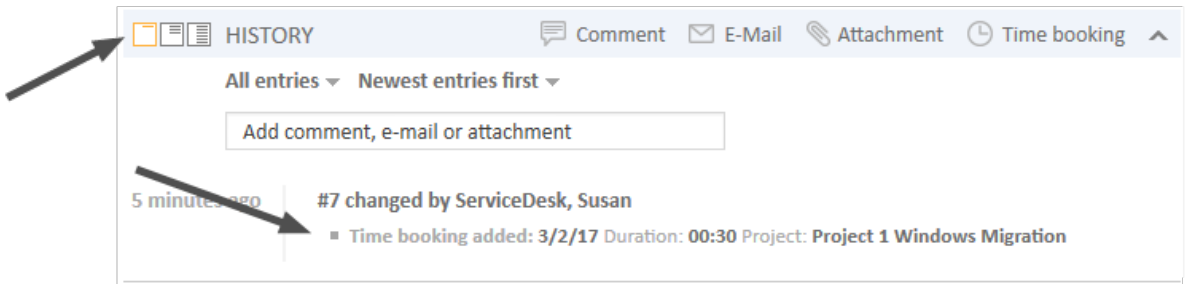


Figure 65: *ConSol CM Web Client - Time booking entry visible on Basic level*

B.4.3.1 Setting the Mode for the Display of Ticket Field Changes in the Ticket History

Please note that the visibility level in the history can be set

- globally, using the visibility parameter *Ticket field changed*
- specifically for one or more ticket fields using the annotation `visibility configuration`

> Please note that the visibility of attachments is configured only here (*Attachment added*). Text classes (see section Classes of Text) are not taken into consideration for attachments.

In the following example, changing a ticket field (i.e. setting a new value) will be displayed in the *Extended* and *Detail* levels of the ticket history. This is the system-wide setting done in the navigation item *History*, navigation group *Tickets*.

History

Visibility of ticket history entries

Type	Visibility
Queue changed	Levels: Basic, Extended, Detail
Ticket field changed	Levels: Extended, Detail
Ticket created	Levels: Basic, Extended, Detail
Subject changed	Level: Detail
Customer changed	Levels: Extended, Detail
Assigned engineer changed	Level: Detail
Referenced engineer added/removed	Levels: Extended, Detail
Reference added/removed	Levels: Extended, Detail
Attachment added	Levels: Extended, Detail
Ticket transferred to activity after workflow deployment	Level: Detail
Manual activity or activity with overlay executed	Levels: Extended, Detail
Activity executed after escalation	Levels: Basic, Extended, Detail
Automatic activity executed	Level: Detail
Time booking added	Levels: Extended, Detail

Figure 66: *ConSol CM Admin Tool - Tickets, History: Visibility configuration for ticket field changed*

For the ticket field *volume_product*, the visibility configuration is set to *on every level*. In this way, the change of this ticket field would always be visible in the ticket history, no matter which visibility level the engineer has selected (*Basic*, *Extended*, or *Detail*). This setting is done in the navigation item *Ticket Fields*, navigation group *Tickets*.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

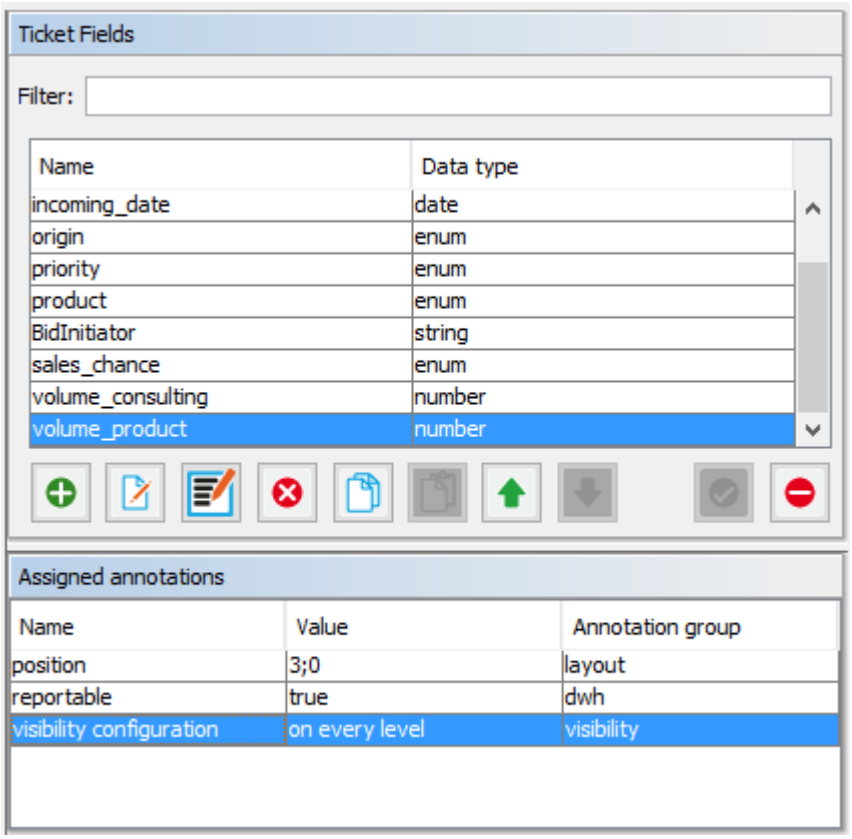


Figure 67: *ConSol CM Admin Tool - Tickets, Ticket Fields: Setting the visibility configuration for one specific ticket field*

B.4.3.2 Modifying the Display Behavior for the Basic and Extended Level to Optimize Ticket History Display

Tickets with a great number of history entries, e.g. comments, might be too large to get a quick overview of all entries. You can use the *lazy loading* feature (see Page Customization attributes *headHistoryElementsCount* and *tailHistoryElementsCount*) to reduce the required space of a ticket.

You can also modify the number of lines which are displayed for comments in the basic and extended visibility level. This can be done for ticket history entries which are marked with a class of text which applies to the first two "short" levels, i.e. for classes of text with the visibility *Basic (short)* and *Extended (short)*.

The configuration is done using the two page customization attributes of type *acimSection*, scope *ticketEditPage/acimSection*, see acimSection (Type or Subscope).

- **basicViewCharactersLimit** (formerly **standardViewCharactersLimit**)
  for Basic (short) level. Defines the number of characters displayed, default 150.
- **extendedViewCharactersLimit**
  for Extended (short) level. Defines the number of characters displayed, default 350.

In the following example, both view limits have been set, but only when the level is short (according to the class of text, here: *default_class*), the limit is applied.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

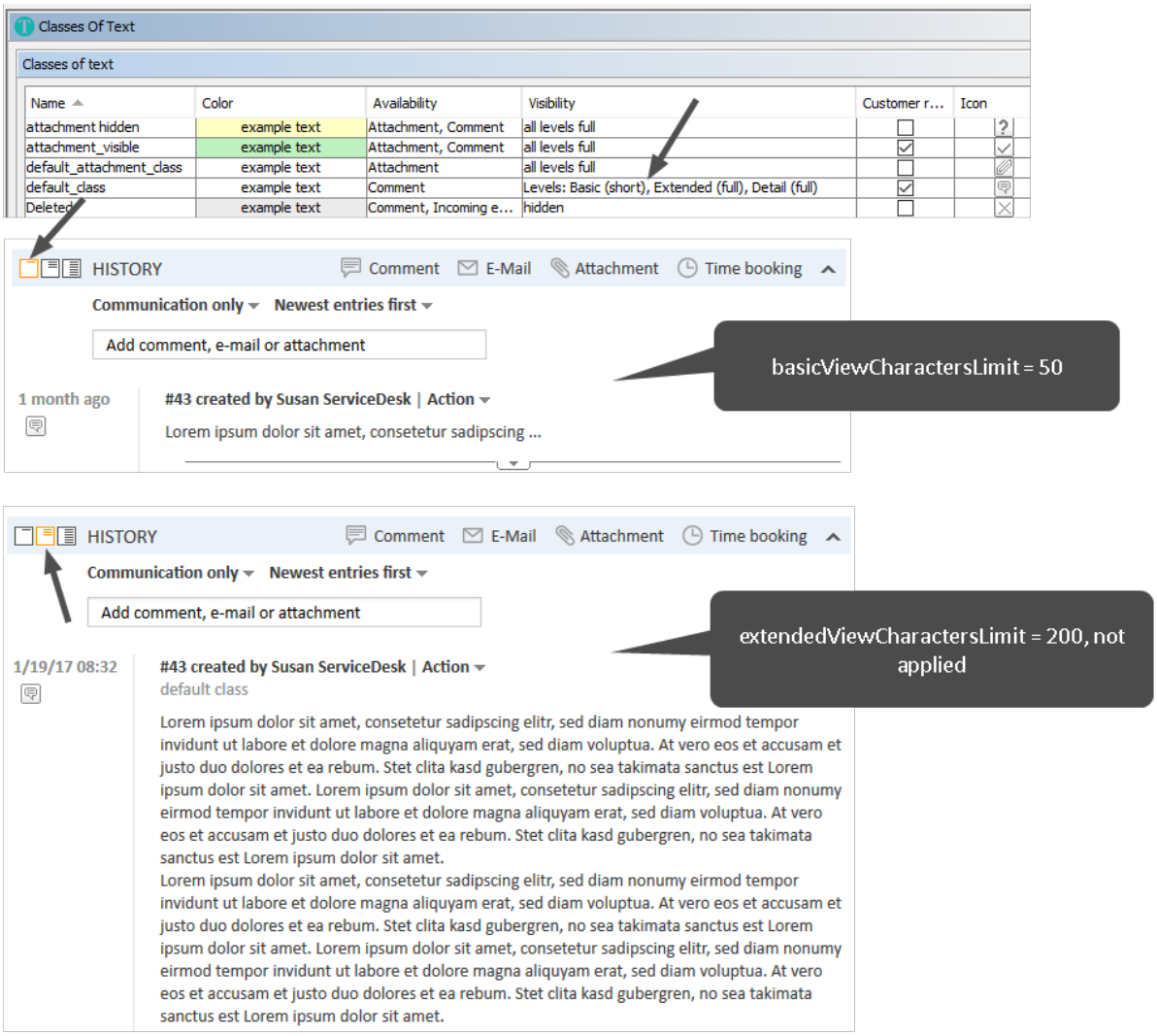


Figure 68: *Configuration of comment display using a class of text plus page customization (ViewCharactersLimit), 1*

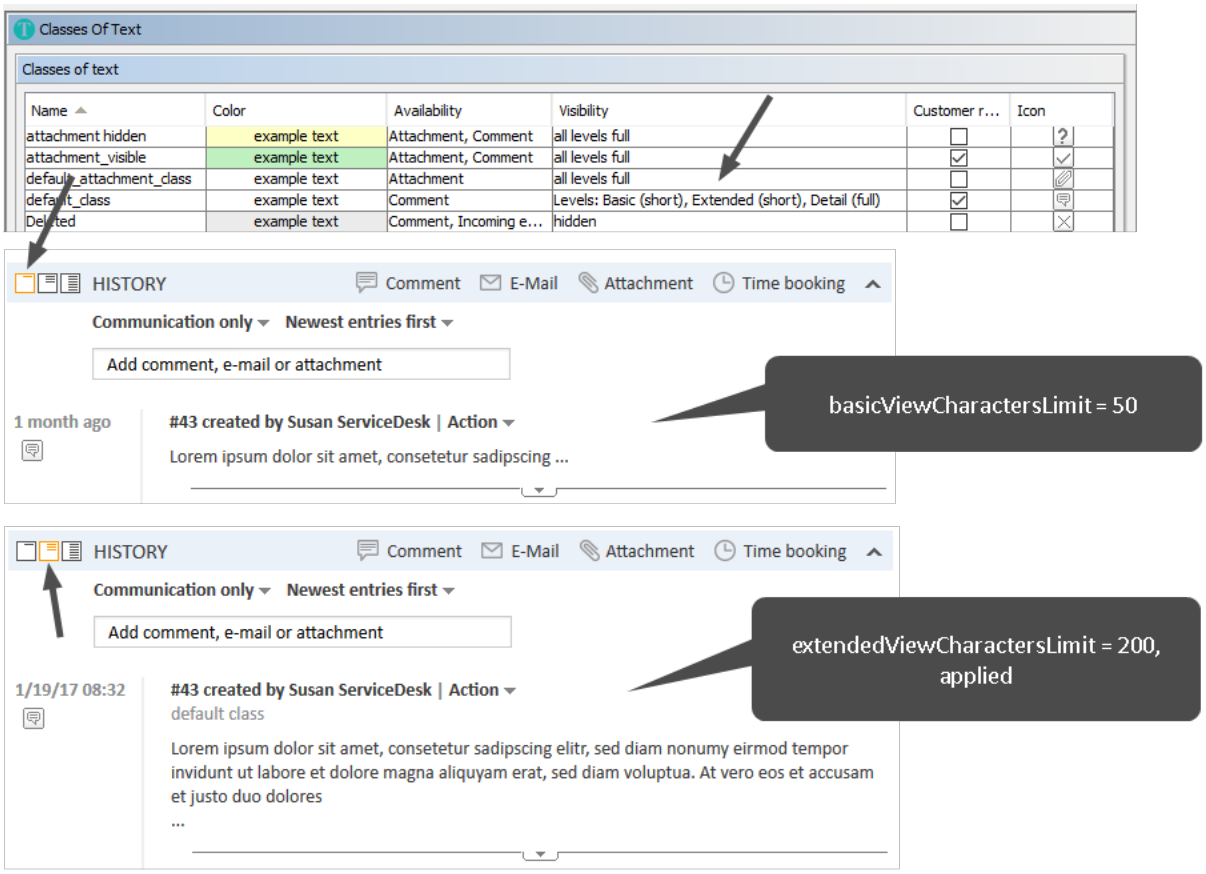


Figure 69: *Configuration of comment display using a class of text plus page customization (ViewCharactersLimit), 2*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

B.4.4 General Information about the Visibility of Ticket History Entries in the Web Client

There a **two visibility settings** for ticket history entries:

- The chosen display mode
- The chosen visibility level (including the additional information, whether entries should be displayed *full* or *short* within the chosen visibility level).

There are **two kinds of ticket history entries** for which you can define the visibility settings:

- **Comments/Communication**
  Refers to comments, emails and attachments. Their *visibility level*, and whether the text entries are displayed *full* or *short*, can be controlled by classes of text. The visibility of entries with no class of text assigned to them is determined by the *default class of text* (in a default installation of ConSol CM, the default class for emails and comments is named *default_class* and the default class for attachments is named *default_attachment_class*).
- **Other entries**
  For example, entries about changes to a ticket's queue or scope, assigning and unassigning engineers, or the execution of automatic activities. The full list of all possible types of *other entries* can be found in the Admin Tool in the navigation item *Ticket History*, where you can also set the *visibility level* for this entries.

B.4.4.1 The Display Mode in the Web Client

The display mode determines which kinds of ticket history entries are generally shown in the ticket history. Like the two kinds of ticket history entries (*comments/communication* and *other entries)*, there are two display modes for the ticket history:

- Display mode *Display communication* (shows communication entries only)
- Display mode *Display all entries* (shows communication and all other entries)

Display mode *Display communication*

Shows only comments, emails, and attachments (not the full attachment, but an entry that an attachment has been added and a link to open the attachment). Whether a communication entry is shown, and how much of its contents are shown, depends an the selected visibility level (including the additional information, e.g., whether entries shall be displayed in *full* or *short* forms within the chosen visibility level).

Display mode *Display all entries*

Shows all other entries in addition the communication entries (e.g., entries about changes to a ticket's queue or scope). The full list of all possible types of *other entries* can be found in the Admin Tool in the navigation item *Ticket History*, where you can also set the *visibility level* for these entries.
These *other entries* are visible only if the display mode *Display all entries* is chosen. After this requirement is met, the selected visibility level determines whether the different kinds of entries are shown. In contrast to communication entries, there is no differentiation between *full* and *short* visibility for these *other entries*.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

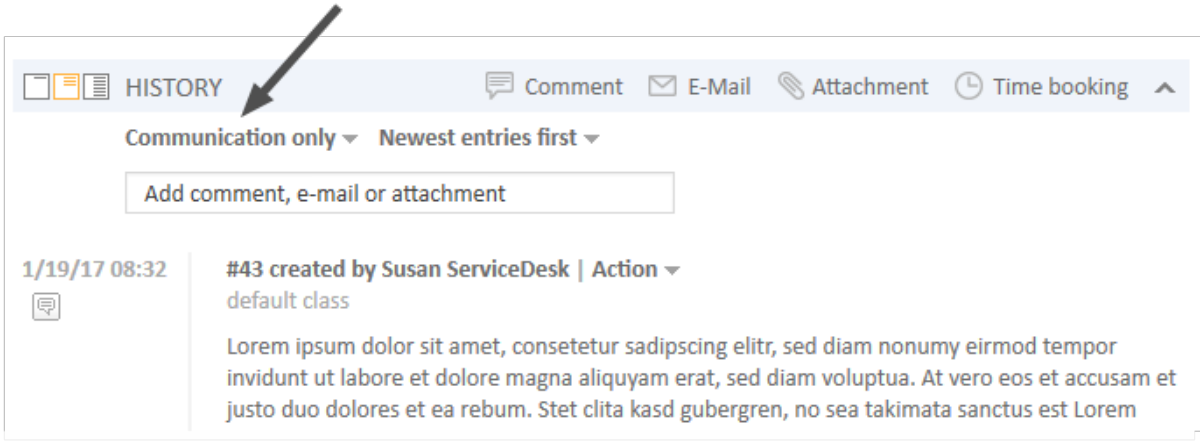


Figure 70: *ConSol CM Web Client - Setting the display mode*

B.4.4.2 The Visibility Level in the Web Client

There are three ticket history levels in the Web Client:

- Basic (1st level)
- Extended (2nd level)
- Detail (3rd level)

Visibility level for communication entries

In which visibility level (Basic, Extended or Detail) communication entries will be shown or hidden in the ticket history is determined by classes of text. You can create and manage them in the navigation item *Classes of Text*. The visibility of entries with no class of text assigned to them is determined by the *default class of text* (in a default installation of ConSol CM, the default class for emails and comments is named *default_class* and the default class for attachments is named *default_attachment_class*).

In addition to the general visibility of a communication entry, you can also determine if an entry of this class should be displayed in full length (*full*) or shortened after a certain number of characters (*short*).

Visibility level for other entries

In which visibility level (Basic, Extended or Detail) other entries will be shown or hidden in the ticket history can be configured in the navigation item Ticket History. In contrast to communication entries, there is no differentiation between *full* and *short* visibility for these other entries.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

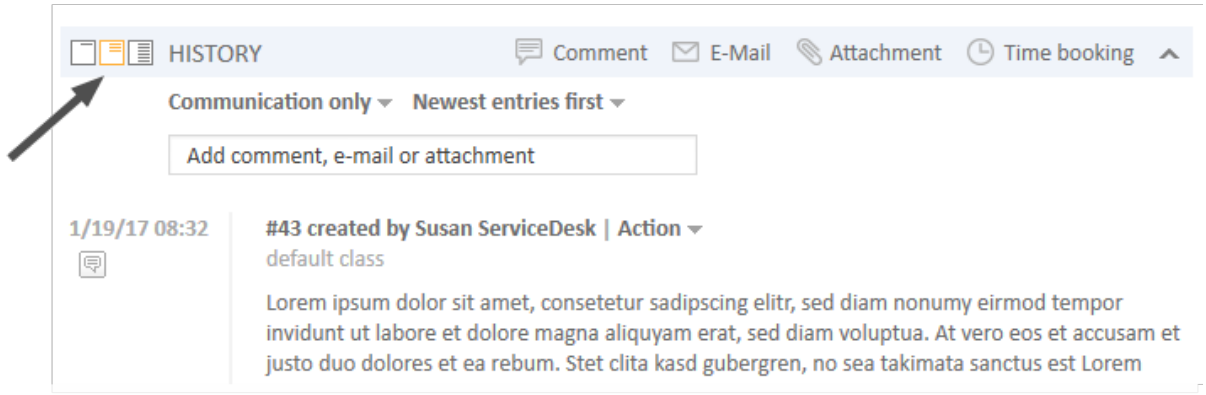


Figure 71: *ConSol CM Web Client - Setting the visibility level*

B.4.5 Ticket History Storage and Transfer to the Data Warehouse (DWH)

In a standard CM installation which uses the Data Warehouse, the ticket history is stored in the CM database and transferred to the DWH.

You can use annotations to prevent the writing of the ticket history into the CM database and to prevent the transfer of ticket history data to the DWH:

Available in CM versions	Group annotation	Field annotation	Triggered behavior
6.6.0 and up	no-history	no-history-field	For the ticket field groups or single ticket fields with this annotation, no history data will be written into the CM database. So no history data will be available at all - the `dwh-no-history` annotation does not have to be set for the respective ticket fields or ticket field groups.
6.10.2 and up	dwh-no-history	dwh-no-history-field	For the ticket field groups or single ticket fields with this annotation, no history data will be transferred to the DWH.

Thus, if you want to see the history in the Web Client, but there are good reasons not to transfer all the data to the DWH, use `dwh-no-history` and `dwh-no-history-field`. In case you do not want to write history data at all, e.g., if some internal programming data is written into ticket fields, use the `no-history` and `no-history-field` combination. Of course you can combine the two variants and use different values for each field or field group.

Please note that there might have been changes concerning the history transfer configuration during the life of a CM system. This might result in a CM database which contains (old) history data even though, currently, the annotation `no-history` is set.



Please note the CM behavior during an update to versions 6.10.2 and higher:

To keep old system configurations valid, for all ticket fields and ticket field groups which are annotated with `no-history-field` and `no-history`, the new annotations `dwh-no-history-field` and `dwh-no-history` are added automatically.

B.5 Ticket List

This chapter discusses the following:

B.5.1 Introduction 139
B.5.2 Configuration of Grouping and Sorting of the Ticket List 141
B.5.3 Configuration of Parameters Which Are Displayed for One Ticket 146
B.5.4 The Definition of Customer Templates 147
B.5.5 The Annotation of Ticket Fields 147
B.5.6 The Page Customization for the Ticket List-Specific Attributes 148

B.5.1 Introduction

In the Web Client, the ticket list is displayed on the left hand side of the user interface. The list serves as to-do list for the engineer.



Figure 72: *ConSol CM Web Client - Ticket list*

The configuration of views and the assignment of views to certain roles define which tickets will be displayed in the ticket list. Please see section Views for a detailed introduction to the configuration of views performed by the administrator.

B.5.2 Configuration of Grouping and Sorting of the Ticket List

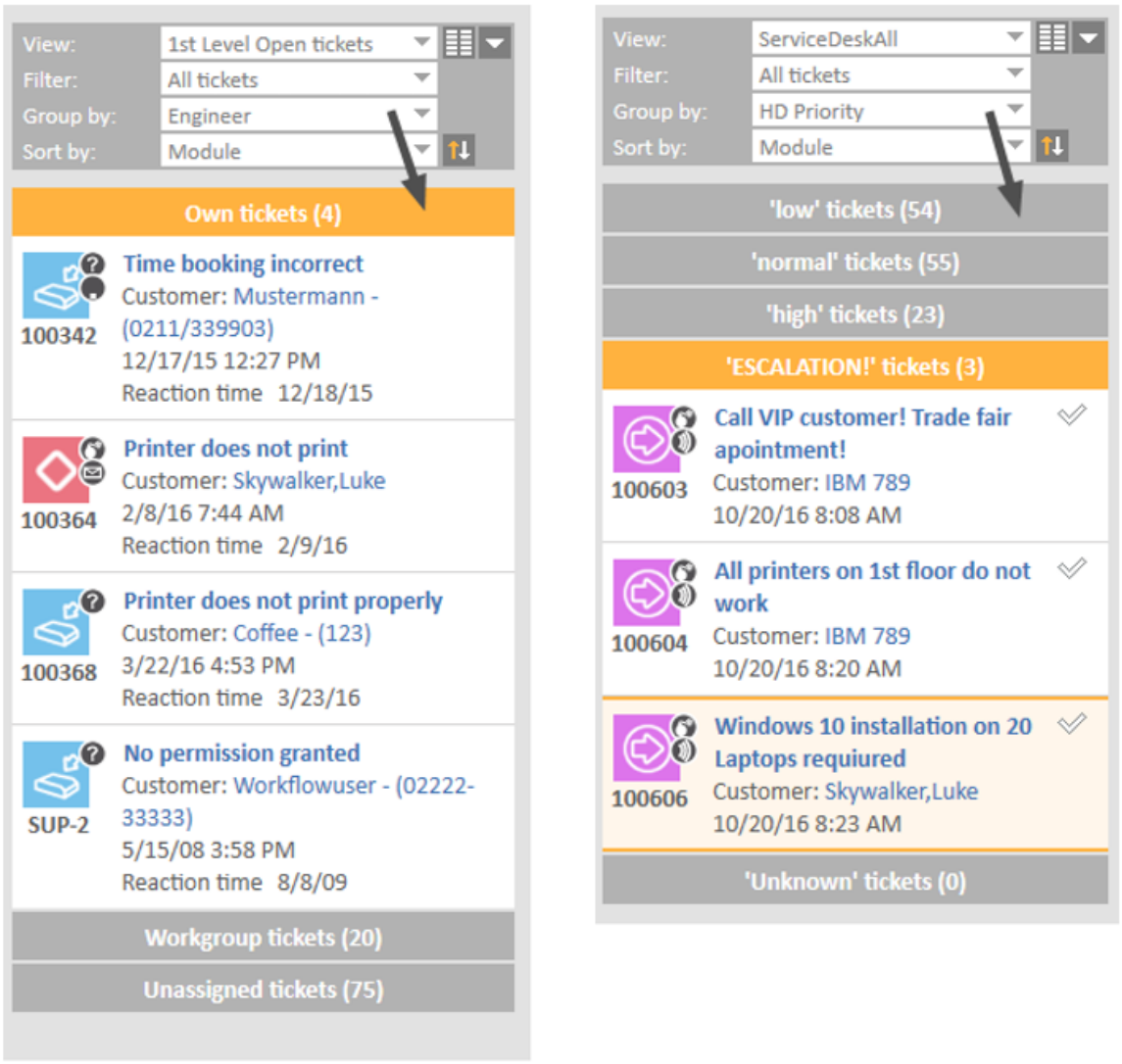
B.5.2.1 Grouping

The ticket list can be displayed in groups. In the image above, the groups are based on which engineer is assigned to the ticket resulting in three groups: *Own tickets*, *Workgroup tickets* and *Unassigned tickets*. The grouping can be changed by the engineer who is working with the Web Client by using the drop-down menu *Group by* in the ticket list configuration.

Three standard values are always offered:

- **No grouping**
  All tickets are shown in one group named *All tickets*
- **Engineer**
  Tickets are divided into three groups. The names of the groups depend on your individual CM configuration, but the groups always serve the same purpose:
  - Tickets assigned to the currently logged in engineer (e.g., Own tickets)
  - Tickets assigned to any other engineer but the currently logged in engineer (e.g., Workgroup tickets)
  - Tickets not assigned to any engineer (e.g., Unassigned)
- **Queue**
  Tickets are divided into as many groups as there are queues in your CM system, but you can only see the groups for queues for which you have at least read permission. The queues are sorted alphabetically by their localized queue names.

In addition to these values, you, as an administrator, can define further parameters by which the ticket list can be grouped. Every ticket field of type *enum* can be configured to be used for this purpose. All you have to do is setting the field annotation `groupable` for the respective ENUM value. In the following example, the annotation has been set for the ticket field *priority* which has the English localization *HD priority*.



Ticket list grouped by engineer

Ticket list grouped by HD priority

Figure 73: *ConSol CM Web Client - Two different groupings of the ticket list (engineer: standard, HD priority: customized)*

In the Admin Tool, the annotation `groupable` is set for the ticket field in the navigation group *Tickets*, navigation item *Ticket fields*.

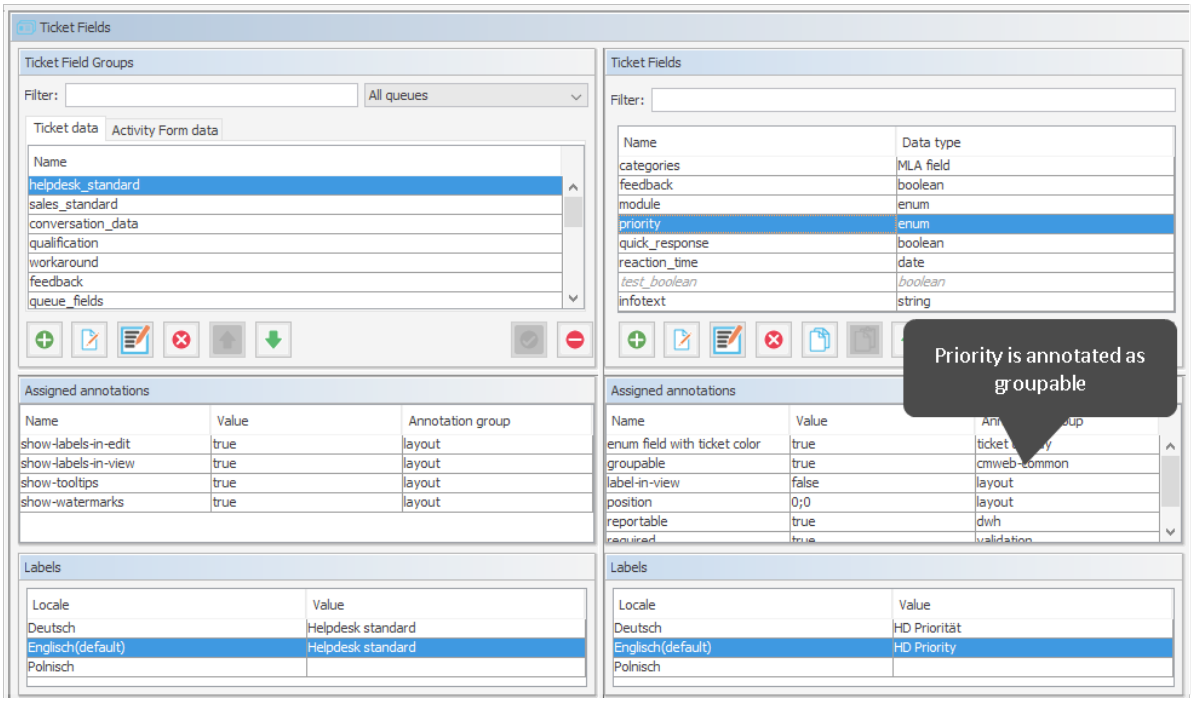


Figure 74: *ConSol CM Admin Tool - Tickets, Ticket Fields: Annotation groupable set for a ticket field*

B.5.2.2 Sorting

Within a group of the ticket list, the tickets can be sorted. The sorting can be changed by the engineer who is working with the Web Client by using the drop-down menu *Sort by* in the ticket list configuration. There are three standard values by which the tickets can be sorted:

- **Scope**
  Sorts the tickets within the groups by the logical order of the scopes they are currently in. The logical order of the scopes is the order of the process steps in your business processes, e.g., Open Ticket - Ticket in progress - Query the technical department - Give solution to customer - Close Ticket.
- **Creation date**
  Sorts the tickets within the groups by their creation date.
- **Modification date**
  Sorts the tickets within the group by the date of the last modification. A modification is one of the following types of event whereby the event can be triggered manually (via Web Client) or automatically (via workflow):
  - ticket field edition
  - details edition
  - addition of additional customer
  - changing a role of customer
  - setting a customer as main customer
  - adding a relation
  - deleting a relation
  - adding an additional engineer

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- deleting an additional engineer
- adding a resource relation
- deleting a resource relation
- adding a comment
- sending an email
- reply, forward, retry an email saved on history
- adding an attachment
- deleting an attachment
- adding time booking

In addition to these values, you, as an administrator, can define further parameters by which the tickets within a group can be sorted. Every ticket field of type *enum* can be configured to be used for this purpose. All you have to do is setting the field annotation `sortable` for the respective ENUM value. In the following example, the annotation has been set for the ticket field *priority* which has the English localization *HD priority*.



Figure 75: *ConSol CM Web Client - Two different sortings of the ticket list, grouped by engineer, group Workgroup tickets (creation date: standard, HD priority: customized)*

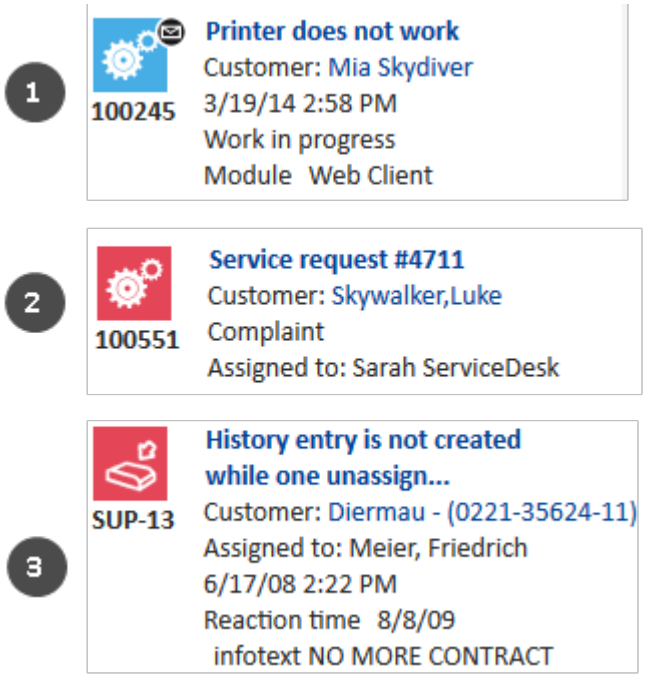
In the Admin Tool, the annotation `sortable` is set for the ticket field in the navigation group *Tickets*, navigation item *Ticket fields*.

B.5.3 Configuration of Parameters Which Are Displayed for One Ticket

In the current section, you will learn how to configure the parameters which are displayed for one ticket. A ticket in the ticket list can look different in different systems, depending on the following parameters:

- the definition of customer templates
- the annotation of ticket fields
- the page customization for the ticket list-specific attributes

Please see the following examples for different ticket appearances:



In the example displayed above, the following data is displayed:

(1) Service Desk ticket example

- Ticket subject (standard)
- Customer (formatted by customer template)
- Creation date (standard for unassigned tickets)
- Scope (formatted by page customization)
- Ticket field (here: Module, an enum; formatted by annotation)

(2) Customer Service ticket example

- Ticket subject (standard)
- Customer (formatted by customer template)
- Ticket field (here: Ticket type, an enum)
- Assigned to (standard for assigned tickets)

(3) Support ticket example

- Ticket subject (standard)
- Customer (formatted by customer template, here with phone number)

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- Assigned to (standard for assigned tickets)
- Creation date
- Ticket field (here: Reaction time deadline, a DATE field)
- Ticket field (here: a String field, infotext)

B.5.4 The Definition of Customer Templates

The appearance of customer data (e.g. name only or name and phone number) is defined by the template which is assigned in the customer object. This can be the *Ticket list* template or, if the *Ticket list* template is not defined, the *Default* template. Please see section Templates for Customer Data for a detailed explanation of templates for customer data.

B.5.5 The Annotation of Ticket Fields

B.5.5.1 Ticket List-Specific Annotations

There are three annotations which are specific for the display of ticket field data in the ticket list:

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

ticket-list-position

This annotation can be set for ticket fields. It defines the position of the given ticket field in the ticket list. Please be aware that all fields which are annotated this way have to fit in a common matrix. The principle is the same as for the positioning of customer fields on the ticket GUI, see section Ticket Fields (Setting Up the Ticket Data Model).

ticket-list-colspan

This annotation can be set for ticket fields. It defines the number of columns which should be used by the respective field.

ticket-list-rowspan

This annotation can be set for ticket fields. It defines the number of rows which should be used by the respective field.

B.5.5.2 General Annotations Which Help Designing the Ticket List

The following annotations help designing the ticket list, but they are not ticket-list-specific, i.e., they can be used for any other ticket field as well:

- show-label-in-view

show-label-in-view

This annotation can be set for ticket fields, for customer field and for resource fields. Regarding the ticket list configuration, only ticket fields are relevant.

If the annotation is set to “true”, the label (name of the ticket field, technical or localized) is displayed in the view mode (here: in the ticket list). If you do not want the label to be displayed, set the value to “false”.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

B.5.5.3 Example

The following example shows a ticket in the ticket list, configured using several parameters.

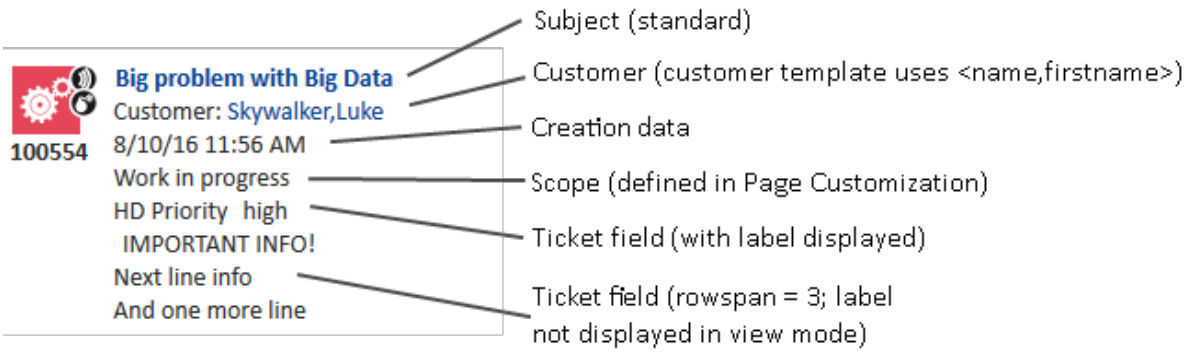


Figure 76: *ConSol CM Web Client - Ticket in ticket list, formatted using several parameters*

B.5.6 The Page Customization for the Ticket List-Specific Attributes

Four page customization attributes play role for configuring the appearance of a ticket in the ticket list. They are set for the page customization type *accordionTicketList*.

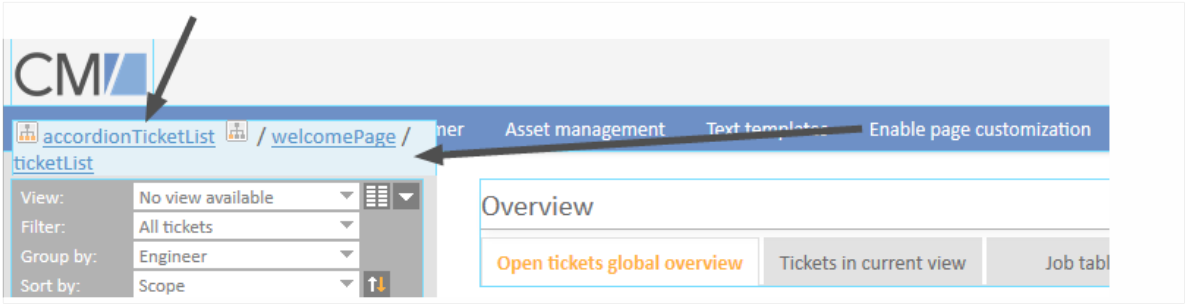


Figure 77: *ConSol CM Web Client - Page Customization for the ticket list*

The following attributes can be used:

- hideEmptyGroups
- ticketDataConfigQueueGroupingScript
- ticketDataConfigEngineerGroupingScript
- ticketDataConfigCustomGroupingScript
- ticketDataConfigNoGroupingScript

See also section accordionTicketList (Type).

In order to define the layout of each ticket in the ticket list, the following keywords can be used (for a detailed explanation of the required syntax, please see section Formatting Principle and Syntax):

- CREATION_DATE
- QUEUE
- SCOPE

- ENGINEER
- CUSTOMER

Additionally, each type of grouping (see following sections) uses an additional type-specific set of keywords.

B.5.6.1 ticketDataConfigQueueGroupingScript

Defines the ticket information display when grouping by queue is selected. The following additional keywords can be used (for a detailed explanation of the required syntax, please see section Formatting Principle and Syntax):

- Queue names

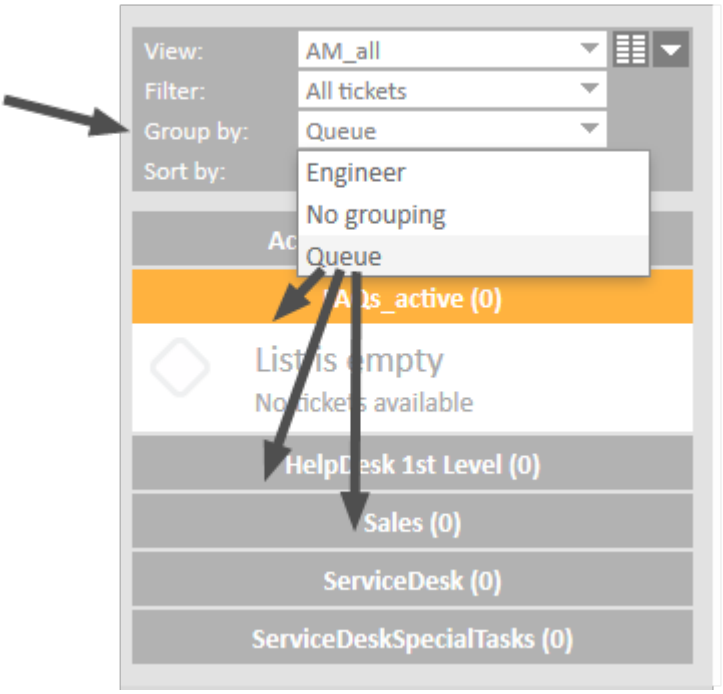


Figure 78: *ConSol CM Web Client - Ticket list grouped by queue*

B.5.6.2 ticketDataConfigEngineerGroupingScript

Defines the ticket information display when grouping by engineer is selected. The following additional keywords can be used (for a detailed explanation of the required syntax, please see section Formatting Principle and Syntax):

- MINE
- GROUP
- UNASSIGNED

B.5.6.3 ticketDataConfigCustomGroupingScript

Defines the ticket information display when grouping by a ticket field is selected. The following additional keywords can be used (for a detailed explanation of the required syntax, please see section Formatting Principle and Syntax):

- enum_grp_name.enum_value_name

B.5.6.4 ticketDataConfigNoGroupingScript

Defines the ticket information display when no grouping criterion is selected. No additional keywords can be used.

- Only the DEFAULT setting can be used.

B.5.6.5 Formatting Principle and Syntax

The formatting principle and syntax are identical for all four attributes.

The configuration String can contain 2048 characters. In case you have to use a longer construct, use an Admin Tool script (of type page customization).

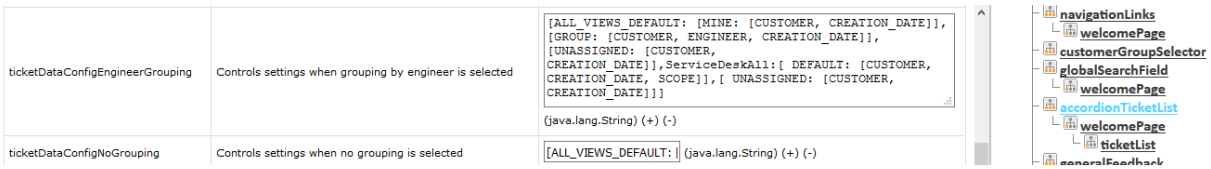


Figure 79: *ConSol CM Web Client - Page Customization attributes for accordionTickeList (two examples displayed)*

To define the ticket list display with page customization (for the attributes mentioned above), a hierarchical configuration is used.

The **first** level is represented by the view definition

- ALL_VIEWS_DEFAULT
- View names like, e.g., *2nd_Level_View_Open*

On the **second** level, you can define for which subset the parameter should be displayed.

For queue-based grouping (i.e. in `ticketDataConfigQueueGroupingScript`), the following parameters can be used:

- Queue names

For engineer-based grouping (i.e. in `ticketDataConfigEngineerGroupingScript`), the following parameters can be used:

- MINE
- GROUP
- UNASSIGNED

For custom-defined grouping (i.e. in `ticketDataConfigCustomGroupingScript`), the following parameters can be used:

- enum_grp_name.enum_value_name, e.g. helpdesk_standard.module.misc.

For each type of grouping, a DEFAULT value can be set which will be used for all constellations which are not defined specifically.

The **third** level is represented by the parameters to display. This can be

- CUSTOMER
- SCOPE
- ENGINEER

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- CREATION_DATE
- QUEUE

Please see the following example (for an engineer-specific grouping) for a first impression. To improve readability, we have inserted line breaks. In CM, please do not set any line break in the statement.

```
[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]]]
```

The meaning of the template above:

For all views, it is set that the engineer's assigned tickets contain customer and creation date, that the group tickets contain customer, the assigned engineer and the creation date, and the unassigned tickets contain customer and creation date. For tickets in the queue ServiceDesk, a specific configuration is provided: for all types of tickets (i.e. tickets assigned to the engineer and group tickets), the customer data, creation date and scope should be displayed. For unassigned tickets, only customer data and creation date are displayed.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

The ticket list, in the example, is displayed as shown in the following two figures.

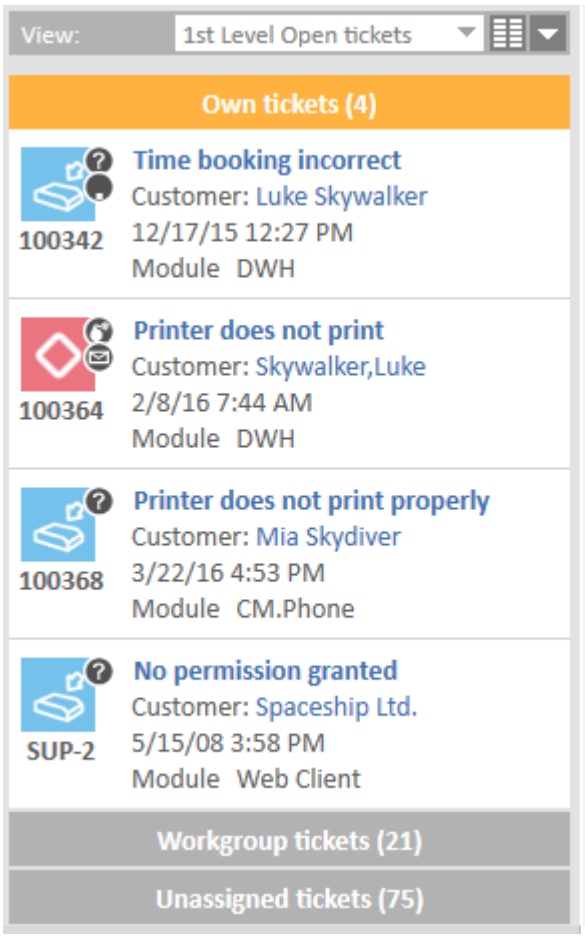


Figure 80: *ConSol CM Web Client - Ticket list for view HelpDesk 1st Level, defined by ALL_VIEWS_DEFAULT*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

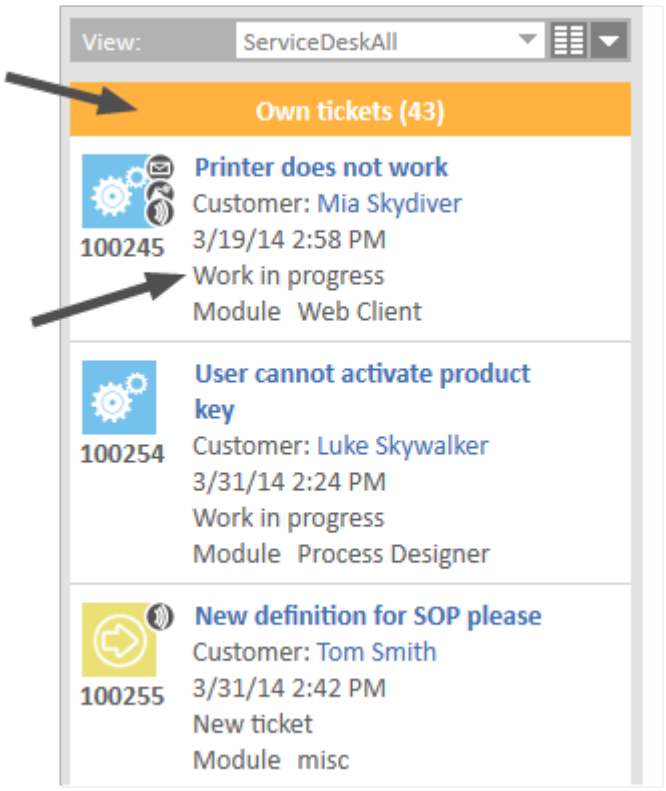


Figure 81: *ConSol CM Web Client - Ticket list for view ServiceDeskAll, defined view-specific, scope displayed for the engineer's tickets*

Since two of the definitions are identical, you could work with the DEFAULT value here. The following JSON statement represents a configuration identical to the example shown above:

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

Restore default setting for a configuration:

- DEFAULT_CONFIGURATION_SCRIPT

B.5.6.6 Default System Settings

If no values are changed, or if the default configuration is restored (using the parameter DEFAULT_CONFIGURATION_SCRIPT), the following values are set:

- **Queues** (ticketDataConfigQueueGrouping):

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

- **Engineers** (ticketDataConfigEngineerGrouping)

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

- **Ticket fields** (ticketDataConfigCustomGrouping):

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

- No grouping (ticketDataConfigNoGrouping):

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

B.5.6.7 Configuring Page Customization for the Ticket List Using a Script

As for all page customization attributes, instead of setting the values for each single attribute using the Web Client, you can set the required attributes using an Admin Tool script. For a detailed explanation, please refer to the section about Page Customization.

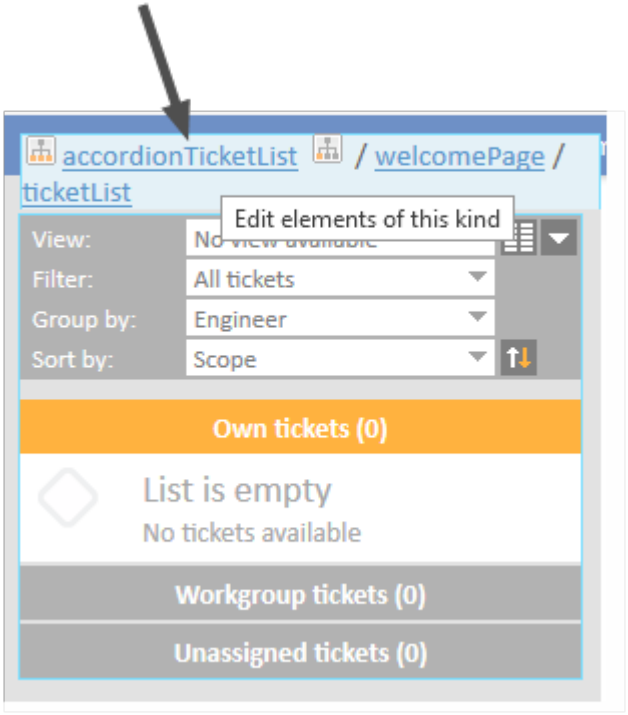


Figure 82: *ConSol CM Web Client - Page Customization for the ticket list*

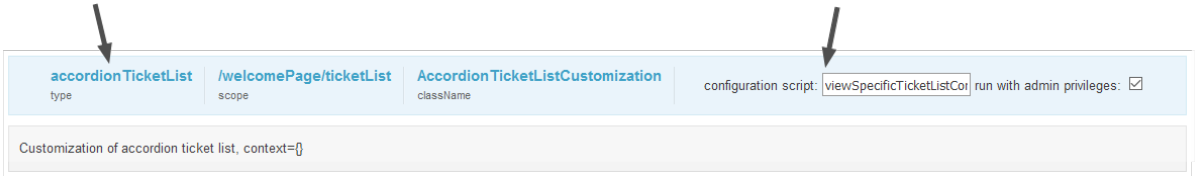


Figure 83: *ConSol CM Web Client - Configuring a script for the Page Customization for the ticket list*

The attributes for the ticket list have to be set using a script for the type *accordionTicketList*. The script can either return only one of the attributes for the type or more, or even values for all attributes. The following example shows a script which returns values for the two attributes *ticketDataConfigNoGrouping* and *ticketDataConfigQueueGrouping*.

```
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 example 1: *Example script for ticket list configuration (viewSpecificTicketListConfig.groovy)*

The following ticket list configurations will be displayed in the Web Client, see following figure.

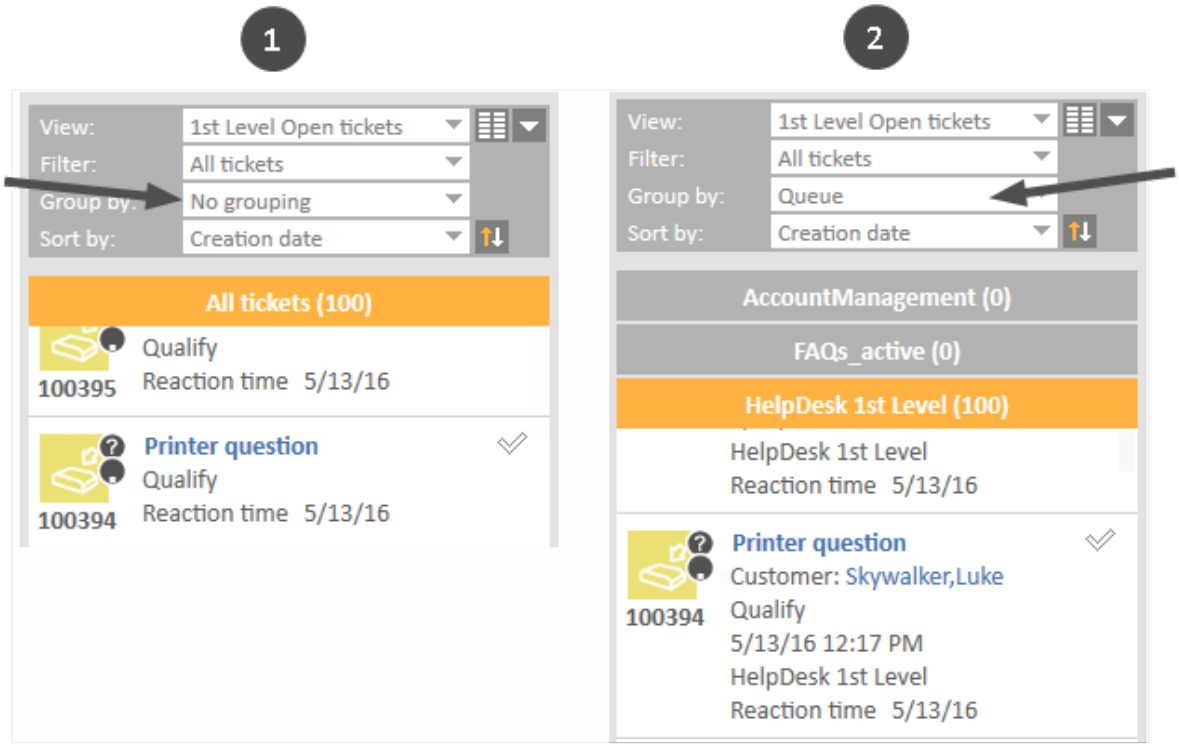
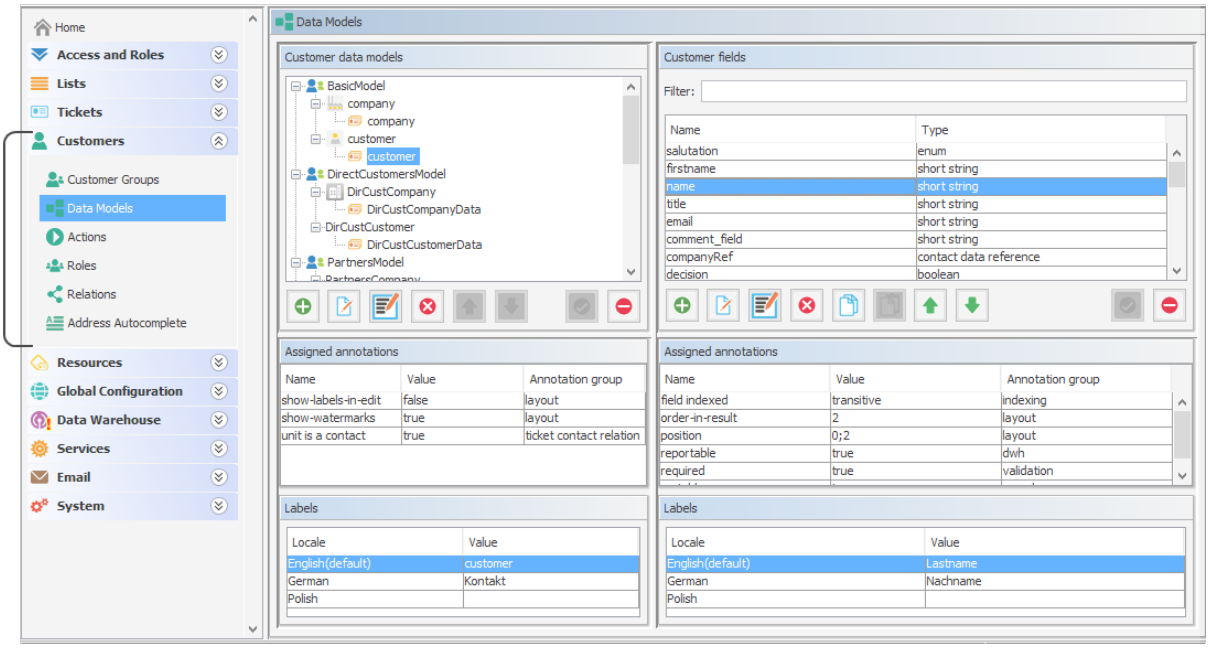


Figure 84: *Parts of two ticket lists according to the configuration in the example script*

- (1) No grouping. HelpDesk_1st_Level:
  DEFAULT: SCOPE
  (Reaction time stems from annotation!)
- (2) Queue grouping. HelpDesk_1st_Level:
  CUSTOMER, SCOPE, CREATION_DATE, QUEUE displayed
  (Reaction time stems from annotation!)

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C - Customer Data Model



ConSol CM offers a very flexible and powerful customer administration using different customer data models.

In this section, you learn how to set up and manage the customer data models, and all related topics. The following subjects are explained:

- Introduction to Customer Data Models
- A Short Introduction to Customer-Specific Web Client Functionalities
- Setting Up the Customer Data Model
- Customer Field Management and GUI Design for Customer Data
- Templates for Customer Data
- Managing Customer Groups
- Customer Roles
- Customer Relations
- Action Framework - Customer Actions
- Address Autocomplete

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.1 Introduction to Customer Data Models

This chapter discusses the following:

C.1.1 The Flexible Customer Data Model at a Glance 158
C.1.2 Introduction to the Objects in the Customer Data Model 161
C.1.3 Important Terms Used for Objects in the Customer Data Model 162
C.1.4 Management of Customer Objects Using the Admin Tool 163

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.1.1 The Flexible Customer Data Model at a Glance

The ConSol CM customer data model, also called FlexCDM, offers a very high degree of flexibility. Various **customer groups** can be defined, each with its particular data model.

In ConSol CM, we talk about **customers** to describe the general CM object. This can be either a company or a contact. A **company** represents an object on company level which will, in most cases, be a real company, a subsidiary, a division or some other organizational unit on a higher level. It can also be a collection of products, a machine pool, or any other object which comprises sub-objects. A **contact** represents an object on contact level, i.e., on the lower level of the customer model. A contact will often be a real person but can also be a product, a machine or some other object.

Contacts as well as companies can be set as customer for a ticket.

There are different ways to configure customer data models for customer groups. Within a customer group, there might be ...

- a *contact* and a *company* level:
  then we talk about a **two-level customer data model** (where a company can contain several contacts)
- only a *contact* or only a *company* level:
  then we talk about a **one-level customer data model**

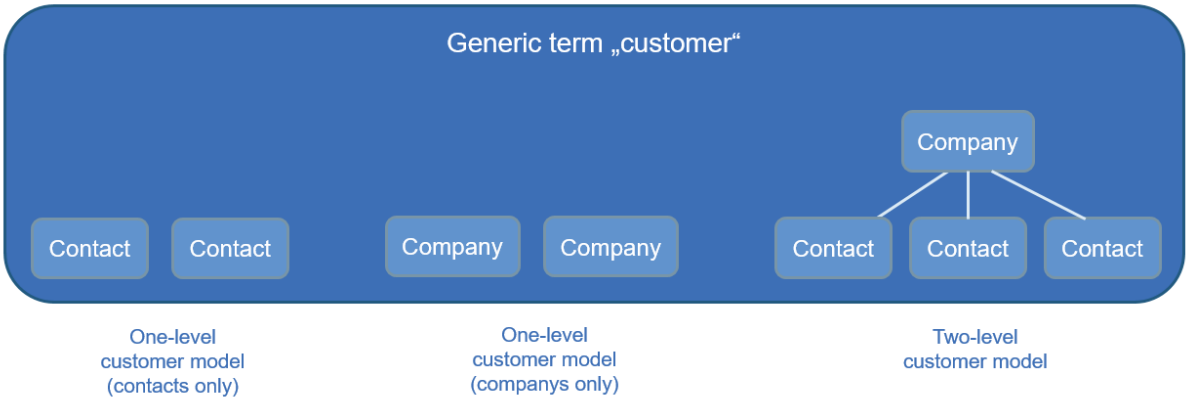


Figure 85: *Types of customer data models in ConSol CM*

For example, you could classify your customers in two customer groups:

1. **Resellers**
   With contact and company level.
2. **End customers**
   With contact level only.

You can configure as many customer data models as required. Every customer data model can be used for one or more customer groups.

A customer data model comprises the general model, i.e. the levels (contact and company or contact/company only) and all data fields for all components (e.g. name, address, and phone for a company or name, email, and room number for a contact).

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

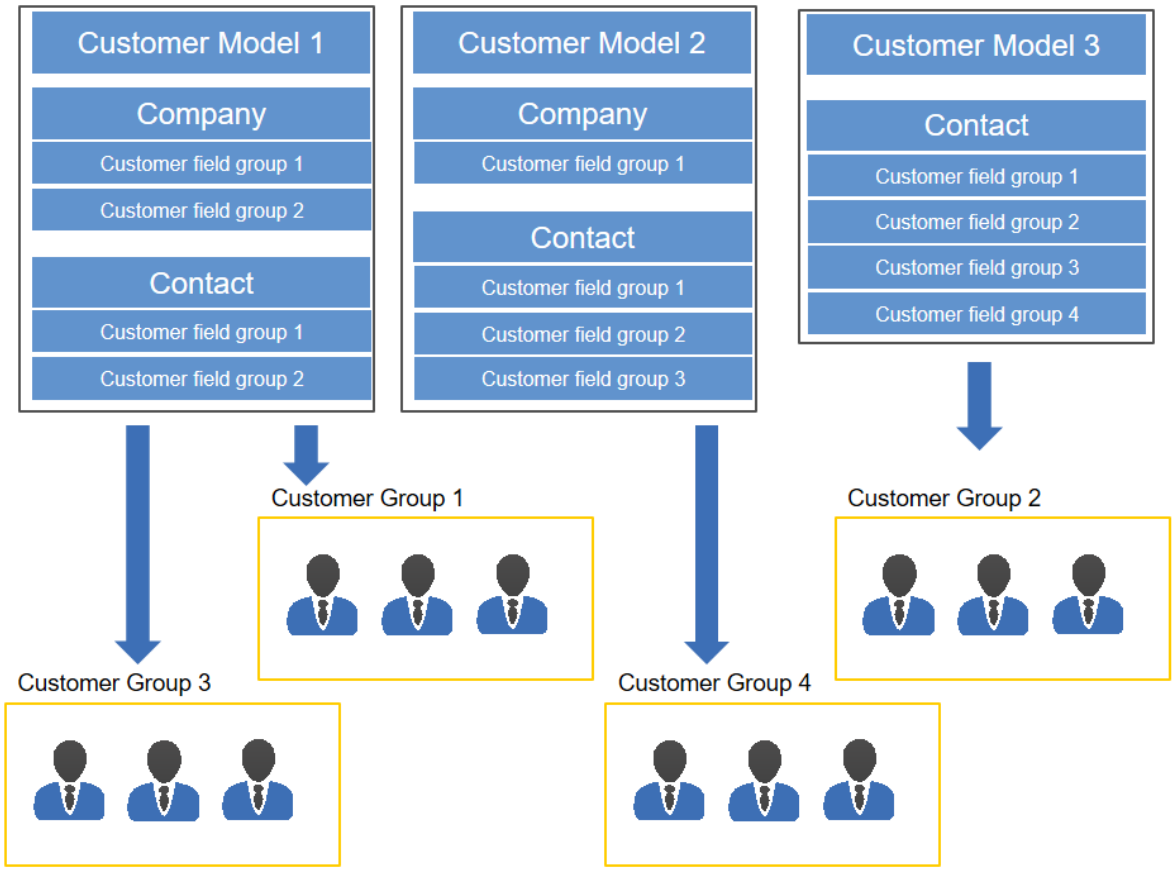


Figure 86: *ConSol CM customer data models - General principle*

For a two-level customer data model:

The terms *company* and *contact* are used to indicate the hierarchical level of an object within the customer data model. An object of type *company* does not necessarily have to be a real company, it can also be a town with several machines (contacts) located in this town, an organization with several subsidiaries (contacts), or even a technical unit (e.g. a ship) with several contacts in the unit. Similarly, an object of type *contact* does not necessarily have to be a person, it can also be a location, a machine, or anything else which should represent the contact level.

For a one-level customer data model:

The customer objects in a one-level customer model are either of type *contact* or of type *company*.

For the customers which are managed by your ConSol CM system, the levels and names of all components entirely depend on the configuration of the customer data model.

The possibility to create different customer data models within your ConSol CM system allows you to build different realms where each includes a specific customer group and the respective data and processes.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

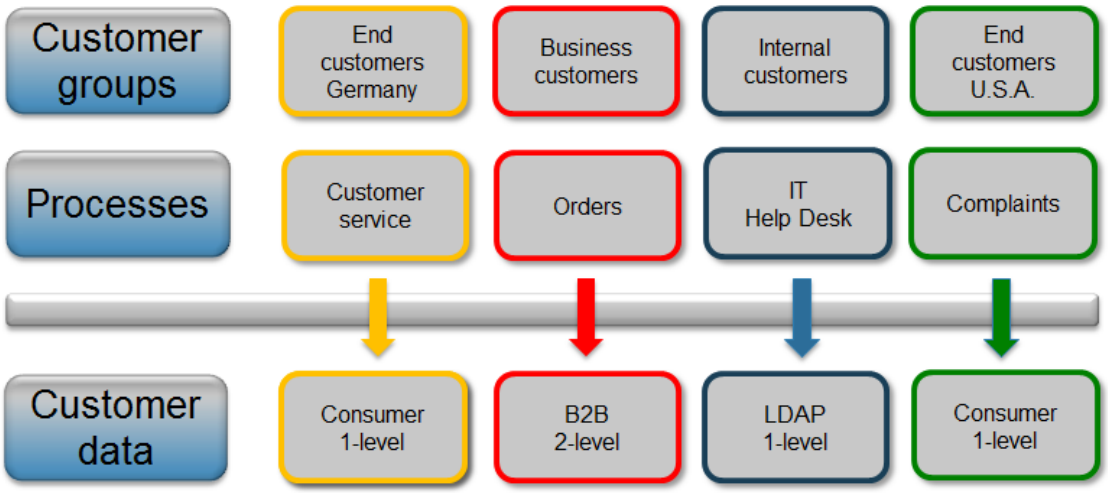


Figure 87: *ConSol CM - Customer data model*

Please see section Setting Up the Customer Data Model for a detailed description of the customer management.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.1.2 Introduction to the Objects in the Customer Data Model

In this section, we will give you an overview of all objects which are relevant for the customer data model.

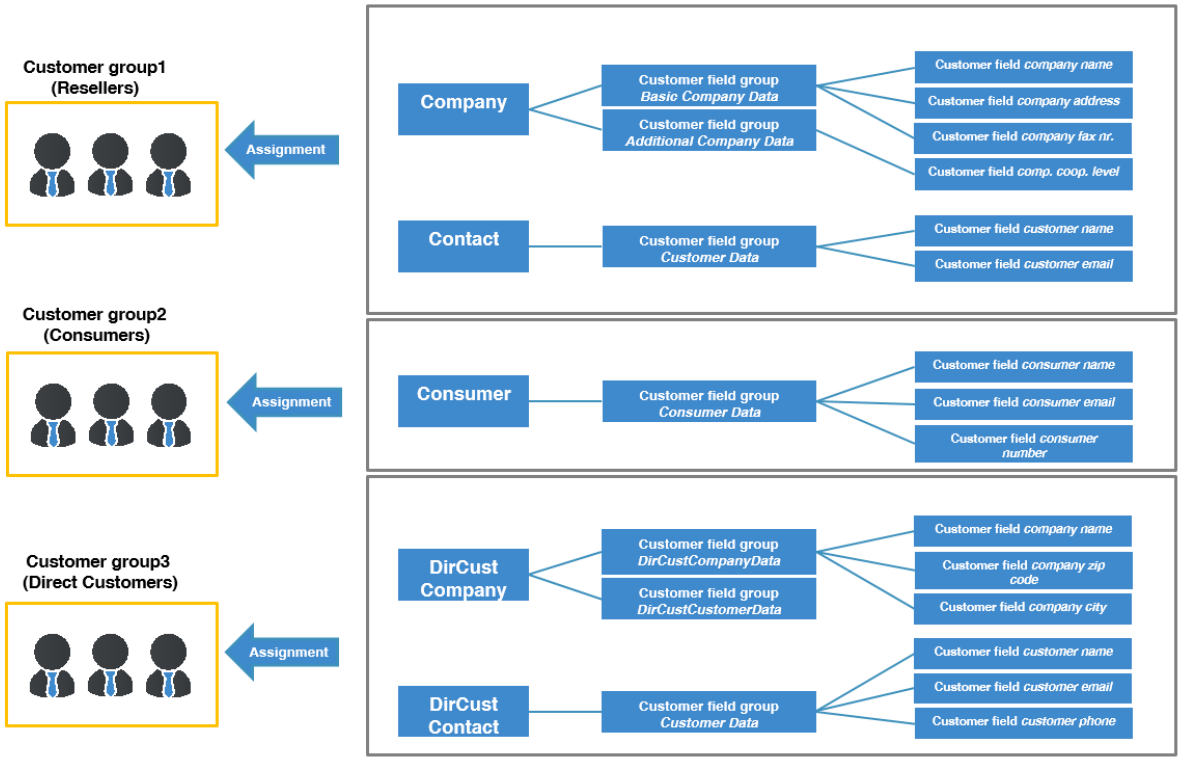


Figure 88: *ConSol CM - Example configuration with three customer data models*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.1.3 Important Terms Used for Objects in the Customer Data Model

The above description uses the Admin Tool terms for the objects in the customer data model. The terms used in the Web Client, i.e., displayed to the end users, can be configured using Labels.

The following table contains an overview of the definitions, the terms used in the Admin Tool, and the default values for the Web Client after an update to the latest ConSol CM version.

Definition	Admin Tool term	Default Web Client term
general term for customer objects, can be of type *contact* or of type *company*	customer	contact
customer object of type *company*, company level	company	company
customer object of type *contact*, contact level	contact	person
group of customers with a specific customer data model, basis for the assignment of permissions	customer group	customer group
object within the customer data model, can be a contact or a company	customer object (technically called a unit)	(not used)
all definitions pertaining to the customer object, i.e., customer field groups, assigned annotations, assigned templates	customer object definition	(not used)
a group comprising one or more data fields (customer fields)	customer field group	contact field group
a single data field that can contain customer data	customer field	contact field
whole data model that can be assigned to a customer group, can have: • one level (only contact or only company) • two levels (company and contact)	customer data model	customer data model
relations between a company and contacts or between companies or between contacts	customer relations	contact relations
application of the Action Framework used for customers	customer actions	contact actions

Please note that the default terms for the Web Client depend on the type of setup. For new installations of ConSol CM, the Web Client terms from the above table are used. For update installations from a previous version of ConSol CM, the default values match the Admin Tool terms from the above table.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.1.4 Management of Customer Objects Using the Admin Tool

In the Admin Tool, you reach the screen with the customer configuration options by opening the navigation group *Customers*.

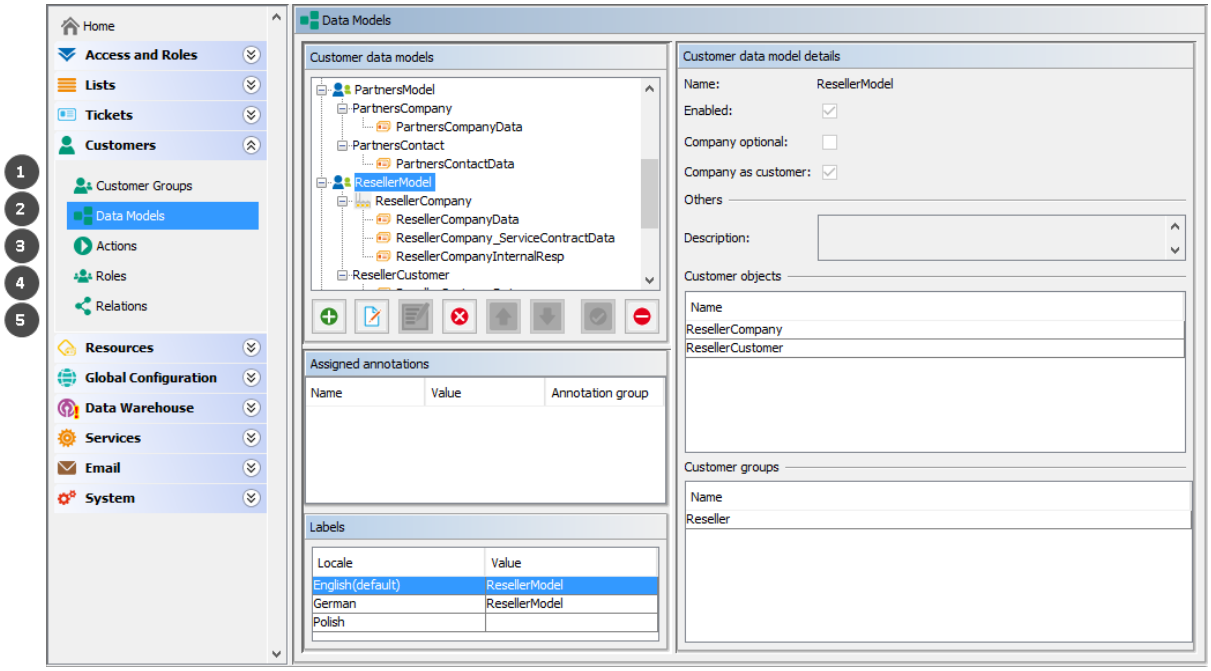


Figure 89: *ConSol CM Admin Tool - Customer navigation items*

The following navigation items are relevant for the definition and management of the customer data model:

- **Customer Groups (1)**
  Assignment of the following components (which have to be defined beforehand) to a customer group (see section Managing Customer Groups for details).
  - Customer data model
  - Customer actions
  - Search actions
  - CM/Phone configuration, if CM/Phone is active
- **Data Models (2)**
  Definition of the data models, i.e., definition of the data fields for customers (i.e., contacts and companies) and the GUI design (i.e., placing the data fields on the Web Client GUI). Please see sections Setting Up the Customer Data Model and Customer Field Management and GUI Design for Customer Data for details.
- **Actions (3)**
  Definition of customer actions, i.e., contact actions and company actions, please see section Action Framework - Customer Actions for details.
- **Roles (4)**
  Definition of customer roles, see section Customer Roles for details.
- **Relations (5)**
  Definition of customer relations, which represent references between customer (i.e., contact

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

and company) objects, please see section Customer Relations for details.
(Relations from resources to customers have to be defined in the Resource Pool data model, they cannot be configured here.)

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.2 A Short Introduction to Customer-Specific Web Client Functionalities

This chapter discusses the following:

C.2.1 Introduction 165

C.2.2 Working with Customers in the ConSol CM Web Client 165

C.2.1 Introduction

When you work with the CM customer data model, you have to know the effects of all administration actions. And of course, those actions are visible in the Web Client. Therefore in this section, we will take the role of an engineer and show several examples of working with the customer data model.

All configuration details which are required to understand the system's behavior are explained in the corresponding sections of the manual.

> Please bear in mind that the terms used for the customer objects depend on the values which are set in the corresponding Labels. In this manual, the terms *customer*, *company*, and *contact* will be used for your convenience.

C.2.2 Working with Customers in the ConSol CM Web Client

C.2.2.1 Example 1: Selecting the Customer Group

Provided that the engineers have access permissions for more than one customer group, they can **select the customer group** which should be used for certain operations using the *Customer Groups Filter*, a drop-down list in the main menu. The name which is displayed is the localized name of the customer group.

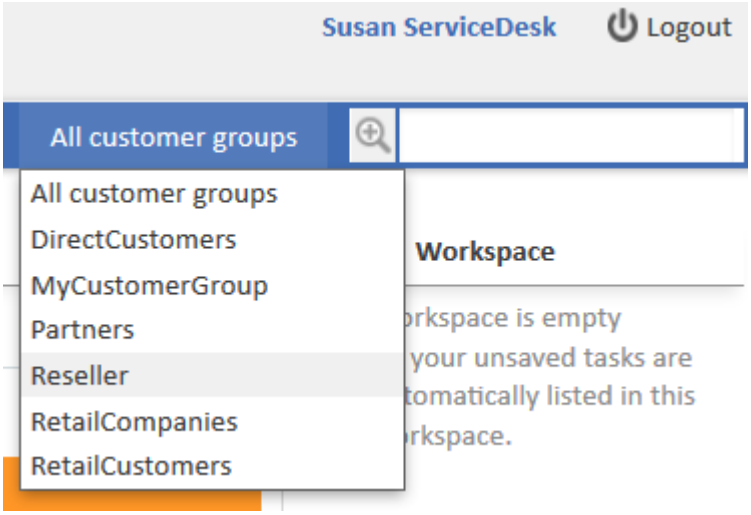


Figure 90: *ConSol CM Web Client - Selecting a customer group*

The selection influences the following actions:

- The Quick Search is performed only within the selected customer group.
- In the Detailed Search, the criterion *customer group* is only offered when *All customer groups* has been selected in the drop-down menu, otherwise the search implicitly uses only data from the selected customer group.
- In the Detailed Search, only the search fields from the selected customer group are offered.
- When a ticket is created, only the selected customer group is offered (implicitly) when a company and/or contact should be created in-line.
- A ticket can be created only in queues to which the selected customer group has been assigned.
- In the ticket list, views are only available if they contain tickets from queues to which the selected customer group has been assigned.



Please note that the Page Customization attribute *hiddenCustomerGroups* can influence the list of customer groups in the Customer Groups Filter. See section customerGroupSelector of the Page Customization for details.

C.2.2.2 Example 2: Creating a New Company and Contact

If engineers have access to several customer groups (and have selected *All customer groups* in the customer groups filter in the main menu, see example 1), they can **select the customer group when a new ticket is created** and a contact/company should be created in-line. This also depends on the selected queue. Only the customer groups which are assigned to the selected queue are available. If the option *All customer groups* has been selected in the drop-down menu, one tab is visible for the customer data of each customer group and the engineer can select the desired group.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

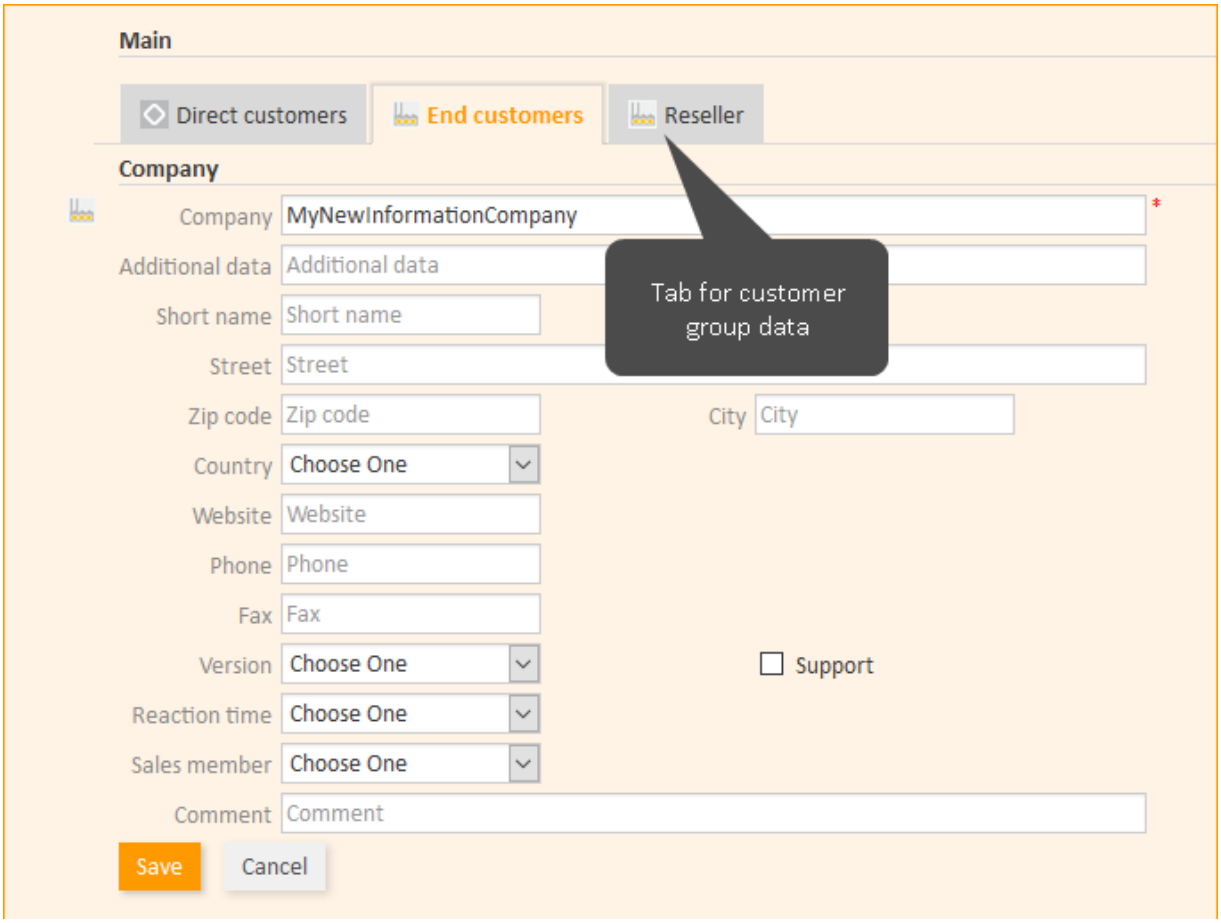


Figure 91: *ConSol CM Web Client - Creating a new company within a customer group*

If an engineer works primarily in a certain customer group but does not want to set the customer groups filter to this group, he can set his default customer group in his engineer profile. Then, the tab of this customer group will be opened / displayed when the ticket create page is opened.

C.2.2.3 Example 3: Using Company and Contact Page

Provided there is a two-level customer data model (company and contact) there are **separate company and contact pages**. On both pages, you can add comments and attach files. Those operations are then also visible in the history of the company (resp. contact) page.

For the company and for the contact object, icons can be defined for each customer data model, improving usability. The header of the contact and company page is defined by the localized value of the name of the respective customer object, see section Setting Up the Customer Data Mode, Name of the Customer Object.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Company Page

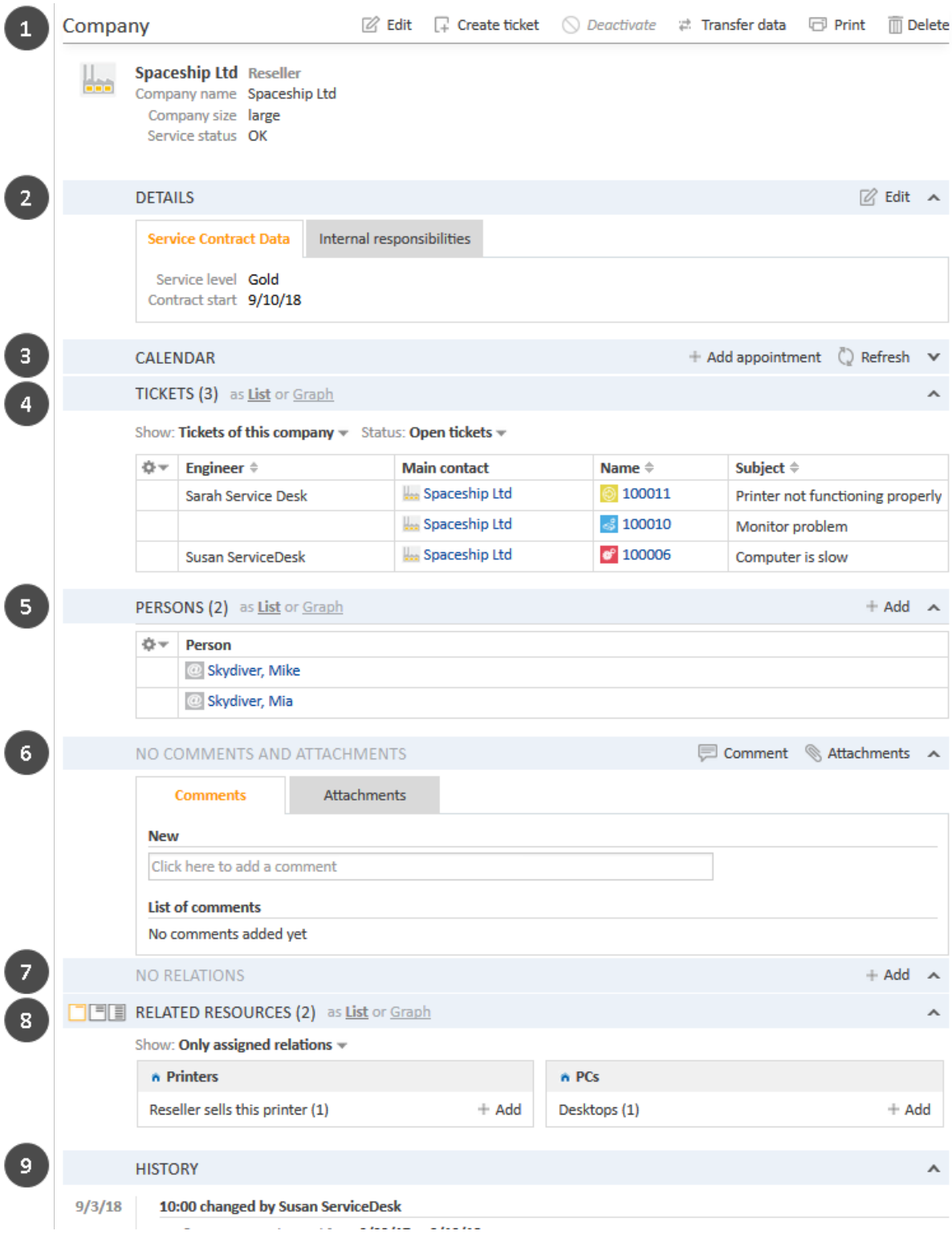


Figure 92: *ConSol CM Web Client - Company page*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

The *Company* page contains the following sections:

- **Company data (1, 2)**
  Company data like address, phone number, service data (i.e., the customer fields you have defined in the customer data model). All company data might be placed in the header section (1) or there might be one or more tabs in the *Details* section (2).
  The name of the company is rendered using the Ticket Page template. If this is not defined, the Default template will be used. For details on customer templates, please refer to section Templates for Customer Data, especially subsection Ticket Page.
- **Calendar (3)**
  This section is optional. It can contain a Microsoft Exchange calendar. Please see Microsoft Exchange Calendar Integration for further information.
- **Tickets (4)**
  All tickets of the company and/or its contacts are listed in this section (also see Example 9: Using the Ticket Filters on Company or Contact Pages).
- **Persons (5)**
  This section shows a list of all contacts belonging to this company. Click on a contact name to open the contact page.
- **Comments and Attachments (6)**
  There are two tabs:
  - **Comments**
    All comments concerning this company are listed here.
  - **Attachments**
    All attachments of the company are listed here. The list of attachments can be filtered or sorted based on file type, name, description, date, or engineer.
- **Relations (7)**
  All relations established between this company and other customers (companies or contacts) are listed here.
- **Related Resources (8)**
  This section is only displayed if CM/Resource Pool is activated in the CM system and if at least one resource relation to companies of the respective customer group is configured in the Admin Tool. It shows all resources linked to this company.
- **History (9)**
  All actions which have been performed with this company object are listed here, e.g., the change of a name or any other value of one of the customer fields.

Please refer to the *ConSol CM User Manual* for a detailed introduction of how to work with companies.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Contact Page

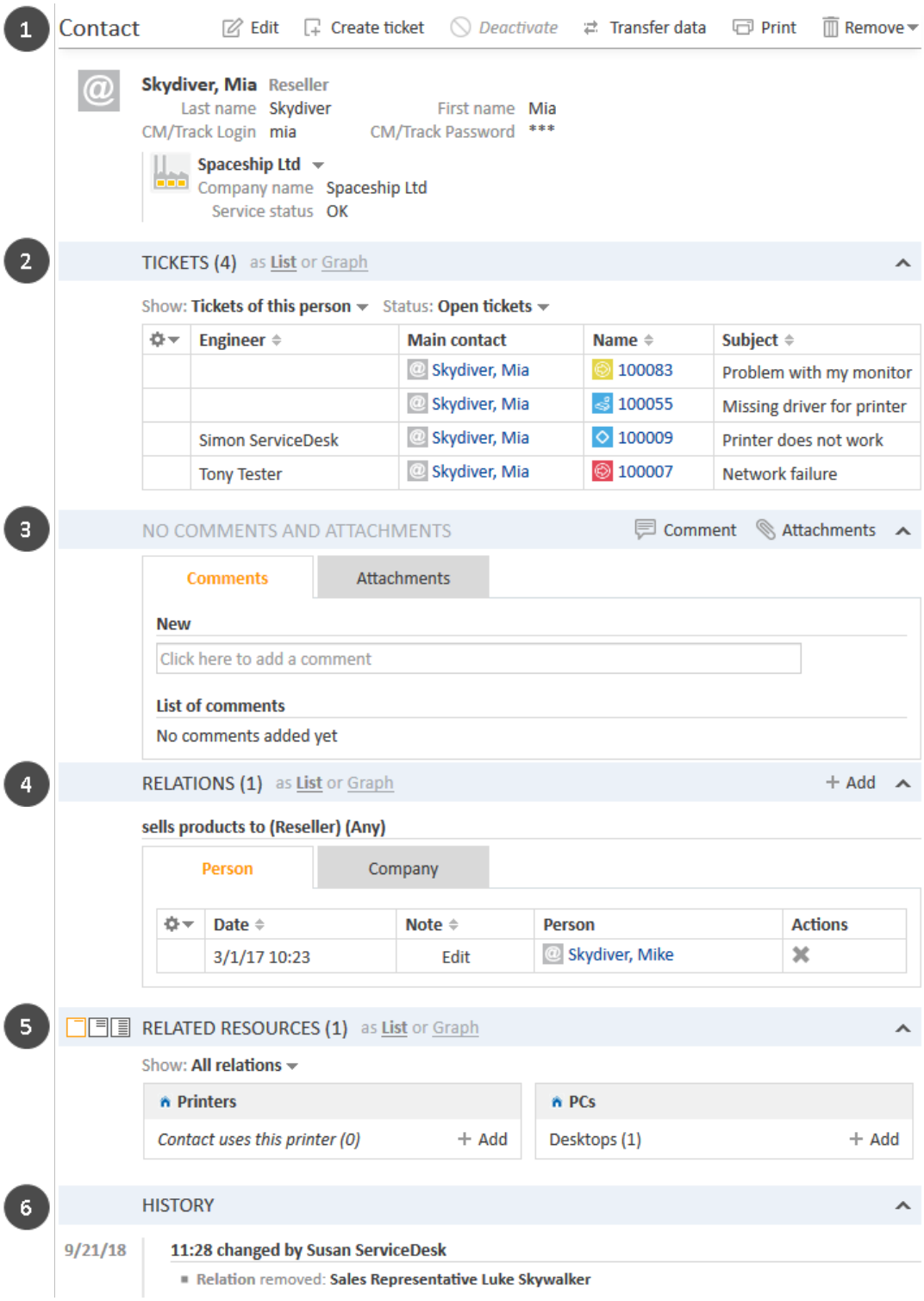


Figure 93: *ConSol CM Web Client - Contact page*

These are the sections on the *Contact* page:

- **Contact data (1)**
  Contact data like address, phone number, service data (i.e., the customer fields you have defined in the customer data model). All contact data might be placed in the header section or

there might be one or more tabs in the *Details* section (not shown in figure above).
The name of the contact is rendered using the Ticket Page template. If this is not defined, the Default template will be used. For details on customer templates, please refer to section Templates for Customer Data, especially subsection Ticket Page.

- **Tickets (2)**
  All tickets for the contact and/or its company are listed in this section (also see Example 9: Using the Ticket Filters on Company or Contact Pages).
- **Comments and Attachments (3)**
  There are two tabs:
  - **Comments**
    All comments concerning this contact are listed here.
  - **Attachments**
    All attachments of the contact are listed here. The list of attachments can be filtered or sorted based on file type, name, description, date, or engineer.
- **Relations (4)**
  All relations established between this contact and other customers (companies or contacts) are listed here.
- **Related Resources (5)**
  This section is only displayed if CM/Resource Pool is activated in the CM system and if at least one resource relation to contacts of the respective customer group is configured in the Admin Tool. It shows all resources linked to this contact.
- **History (6)**
  All actions which have been performed with this contact object are listed here, e.g., the change of a name or any other value of one of the customer fields, or adding/removing relations, comments or attachments.

Please refer to the *ConSol CM User Manual* for a detailed introduction of how to work with contacts.



Please keep in mind that only engineers who have at least one role with the following access permissions for the respective customer group are allowed to access the Comments and Attachments section of the customer page:

- Details read
- Details write
- Details delete

Also note that only engineers who have at least one role with write permissions for the respective customer group can use the *Change* link in the context menu of the company (on the contact page) to assign the contact to another company.

C.2.2.4 Example 4: Setting a Company as Main Customer of a Ticket

If the configuration option *Company as customer* has been set for a customer data model, a **company can be used as main customer** for a ticket.

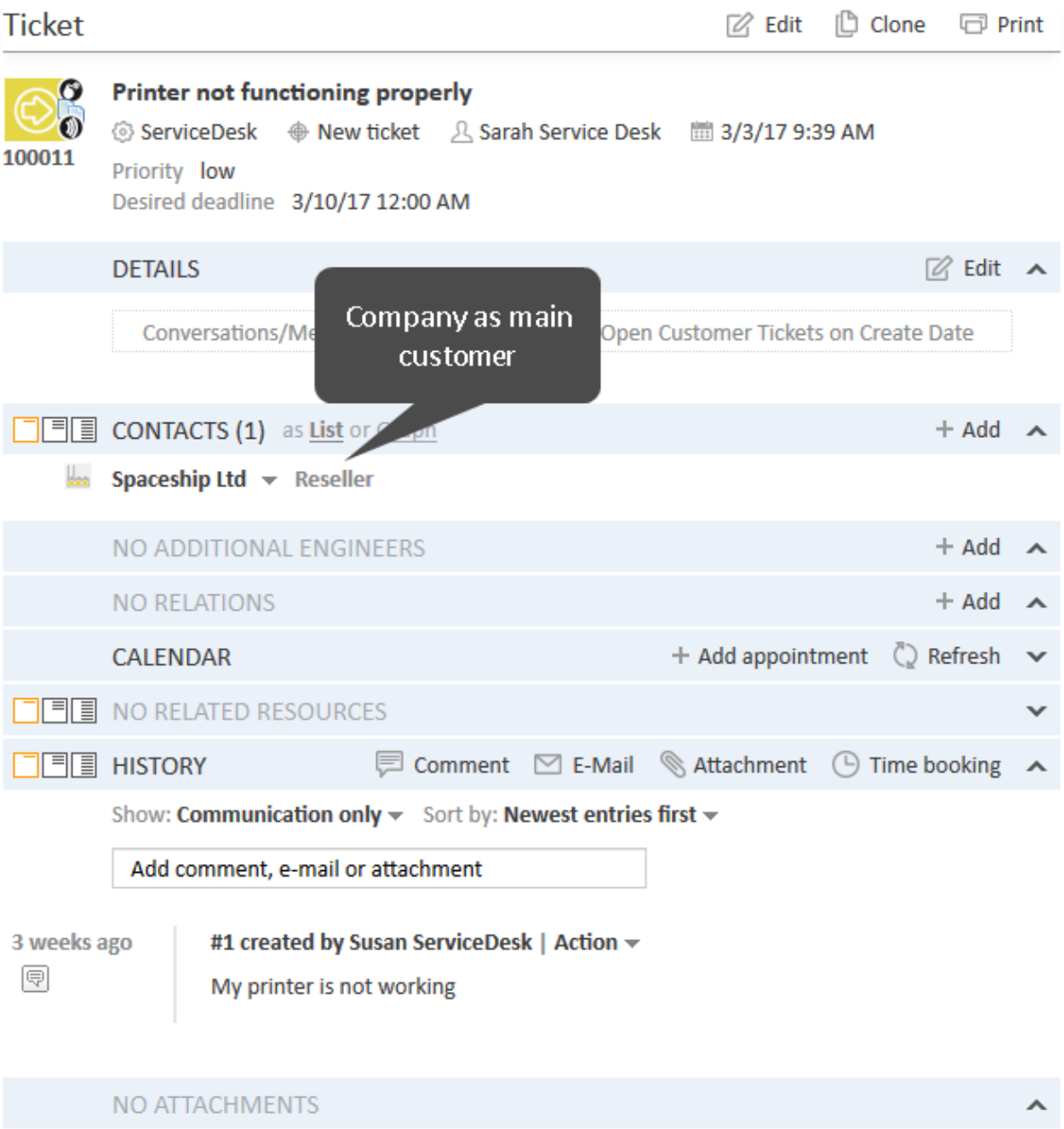


Figure 94: *ConSol CM Web Client - Using the company as main customer for a ticket*

C.2.2.5 Example 5: Using Company and Contact Actions

For companies and contacts, manual and automatic actions can be defined.

Manual actions are triggered using links in the Web Client, very similar to workflow actions (activities) for tickets. In this way, actions concerning the company or the contact data can be performed which are independent of ticket data. For example, an engineer can load the KPIs of the last month for the company, create a new contact within the company (see the following figure), update the contact data from another database, or create a ticket for the contact (see the figure after next).

Automatic actions can be performed when a system action takes place (create/update/delete a customer). The company and contact actions are part of the *Action Framework*.

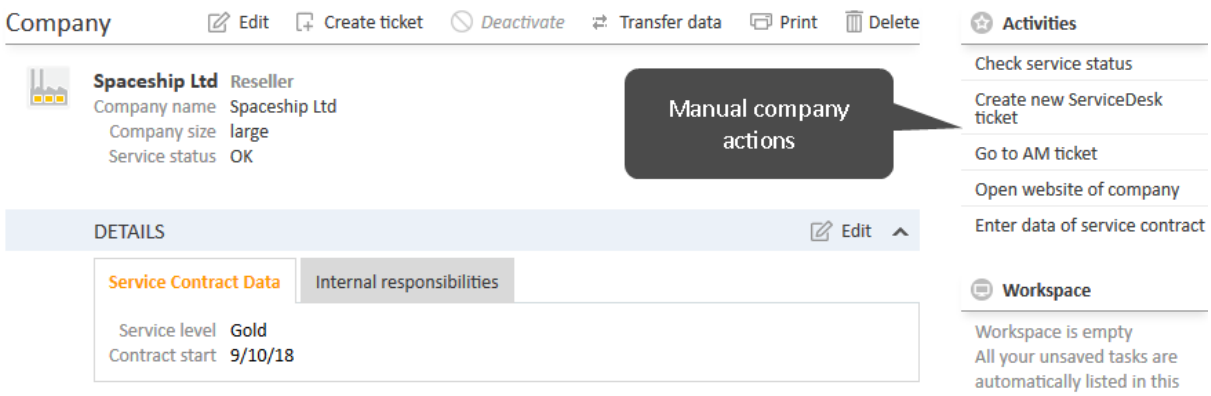


Figure 95: *ConSol CM Web Client - Manual company actions*

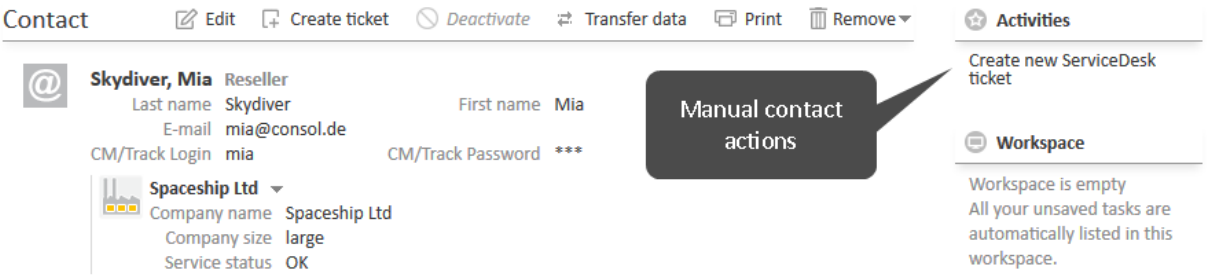


Figure 96: *ConSol CM Web Client - Manual contact actions*



Please keep in mind that only engineers who have at least one role with the following access permissions for the respective customer group are allowed to use the customer actions, i.e., only then the activities will be displayed in the Web Client:

- Act

C.2.2.6 Example 6: Setting Relations between Contacts and Companies

Particularly when you work with several customer groups it can be important to establish **relations between contacts and/or companies**. For example, your ConSol CM system can then represent a reference *sells products to ...* between a company and a contact. Or a relation *is supervisor of ...* between two contacts. In this way, you can create a network of your companies and contacts and use CM for *Customer Relationship Management* (CRM).

In the Web Client, relations between companies and/or contacts are established and displayed similarly to ticket relations. In the example, *MyNewSpaceCompany* sells products to the end customer *Mr. Sample*.

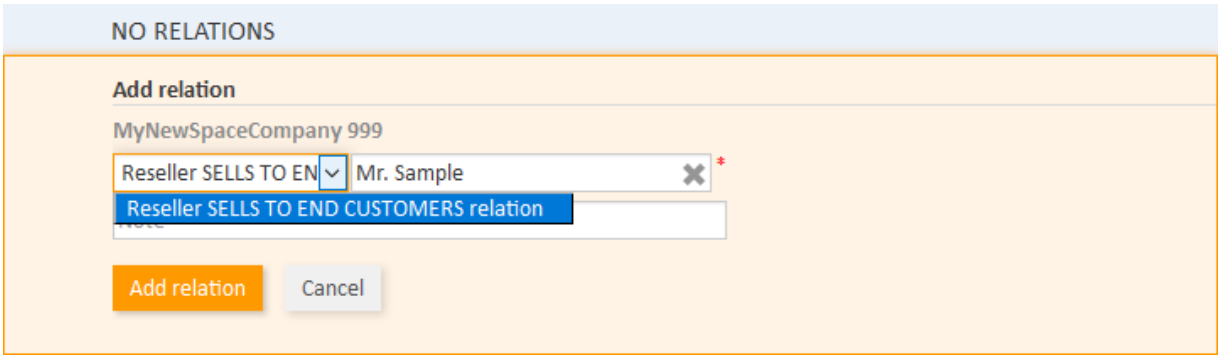


Figure 97: *ConSol CM Web Client - Establishing a company-contact relation*

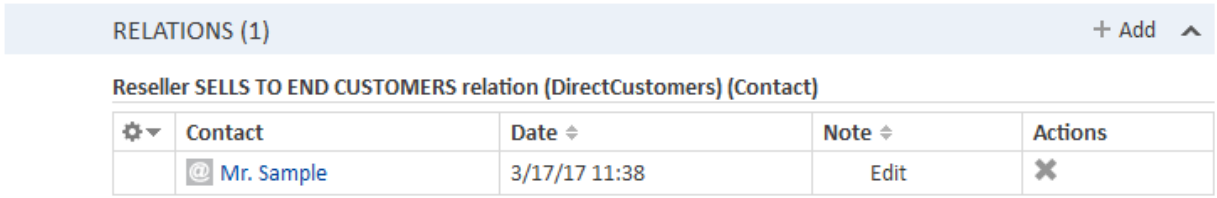


Figure 98: *ConSol CM Web Client - Display of company-contact relation*

C.2.2.7 Example 7: Deactivate a Customer (i.e., a Company or a Contact)

A customer, i.e., **a company or a contact**, **can be deactivated**. This feature might be useful when a contract with a company is no longer valid or when an employee (= contact) has left the company. In this way, the tickets can be kept and retrieved under the **old** contact/company name, but it is not possible to create new tickets for this customer. If the customer has to be deleted, all of their tickets (open and closed) have to be moved. In that case, the former contact-ticket or company-ticket relation is not as easy to find.

The contact or company can only be deactivated if no **open** tickets are assigned to this contact or company.

⚠ Please keep in mind that only engineers who have at least one role with the following access permissions for the respective customer group are allowed to deactivate (and reactivate) companies and contacts, i.e., only then will the *Deactivate/Activate* menu items be displayed in the Web Client:

- Deactivate/activate

The following actions **can** be performed for a deactivated customer:

- Edit the customer data (e.g., name, address, phone).
- Delete the customer.
- Transfer the closed tickets to another customer.
- Only if the checkbox *include deactivated customers* is ticked: search the system for the customer using the Detailed Search.

The following actions **cannot** be performed for a deactivated customer:

- Create a new ticket.
- Assign a ticket to this customer.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- Assign a deactivated contact to another company.
- Assign contacts to a deactivated company.
- Search for the customer (deactivated contacts and companies are not shown in search results, except in the Detailed Search when the engineer specifically activates the checkbox *include deactivated customers*).

Deactivation in a two-level customer data model
When a company (or more generally spoken: an object at the company level) is deactivated, all assigned contacts are deactivated automatically.

There are two use cases:

1. **All** contacts of the company **can** be deactivated (no open ticket assigned).
   In this case the company **and** all assigned contacts will be deactivated. Afterwards the company page will be reloaded, company and contact data are marked as deactivated.

2. The company has still contacts which **cannot** be deactivated because of open tickets. Here, the deactivation of a company is **not** allowed. The deactivate option is not selectable.

Reactivation in a two-level customer data model
If a company is reactivated, the assigned contacts will not be reactivated automatically. They have to be reactivated manually.

C.2.2.8 Example 8: Removing Customer Data

There are three options to delete a customer and/or remove customer data from the CM system.

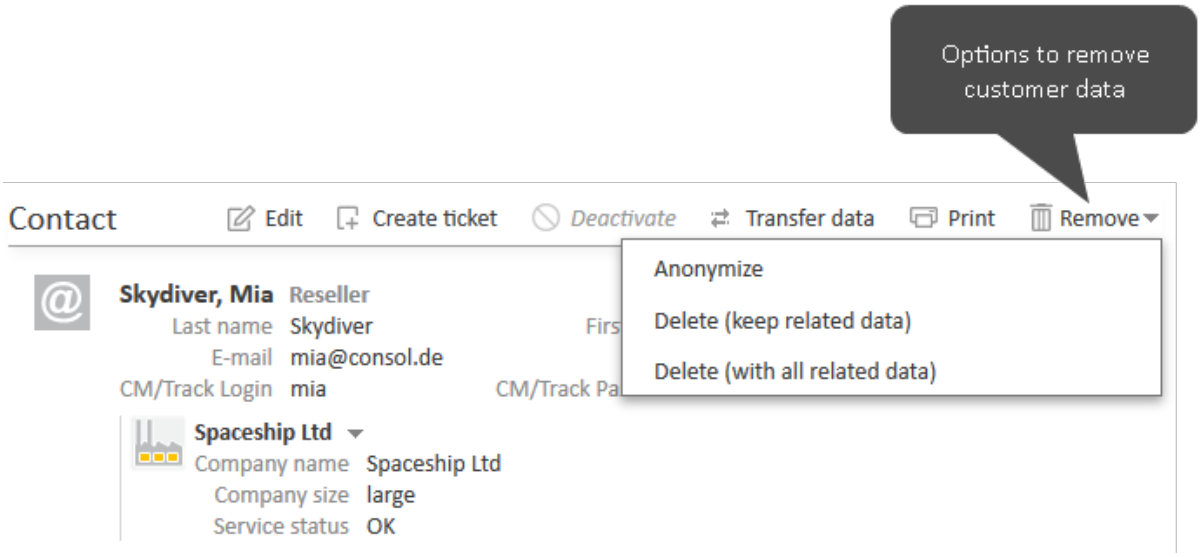


Figure 99: *Contact page with context menu showing delete options*

- **Delete (keep related data) / Delete**

  The customer is deleted, but some references to the customer might remain in the system in history entries of other objects. Before deleting a customer, all his tickets have to be transferred to another customer.

  > The option is called *Delete (keep related data)* for contacts and *Delete* for companies.

- **Delete (with all related data)**

  This option is only available for contacts. The contact and all the tickets where the contact is the main customer are deleted. All references to the contact in history entries of other objects are anonymized.

  > This option can be used for GDPR-compliant removal of contact data if the contact and his tickets are not needed for reports.

- **Anonymize**

  The personal data within the contact and within the tickets where the contact is the main customer are removed. Please see ... define a field containing personal data for how personal data is defined. All existing history entries, comments, and attachments of the contact are removed, and an entry stating that the contact has been anonymized is added to the history. The complete history of the contact's tickets is removed and an entry stating that the data has been anonymized is added to the history. Relations to other objects are removed and the contact is anonymized in the history of these objects.

  > This option can be used for GDPR-compliant removal of contact data if the contact and his tickets are still needed for reports.

C.2.2.9 Example 9: Using the Ticket Filters on Company or Contact Pages

On the company and contact pages, ticket filters are available, i.e., **filter options** can be used to display selected tickets for the company or contact.

Ticket Filters on Company Page

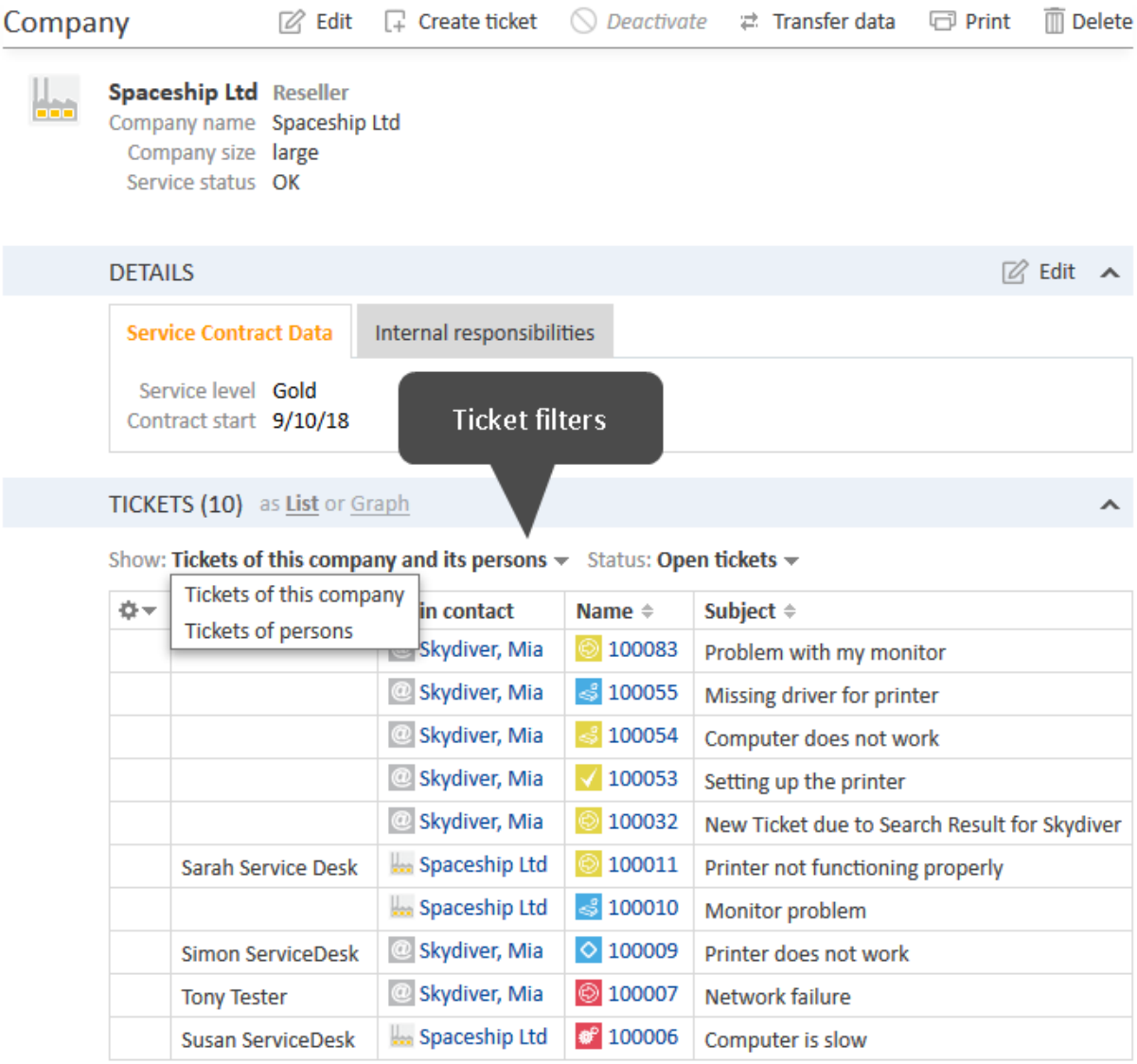


Figure 100: *ConSol CM Web Client - Ticket filters on company page*

The displayed tickets depend on your selection in the two ticket filters:

- **Show**
  - **Tickets of this company:** Tickets where the company is either the main or an additional customer.
  - **Tickets of this company (main customer only):** Tickets where the company is the main customer. This is only possible in customer groups which use a customer data model where the option *Company as customer* is enabled. This filter is only shown if the company is selected as an additional contact in at least one ticket.
  - **Tickets of persons:** Tickets where a contact of the company is the main customer or an additional customer.
  - **Tickets of this company and its persons:** Tickets where the company or a contact of the company is the main customer or an additional customer.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Status**
  - Open tickets
  - Closed tickets
  - Open and closed tickets

Ticket Filter on Contact Page

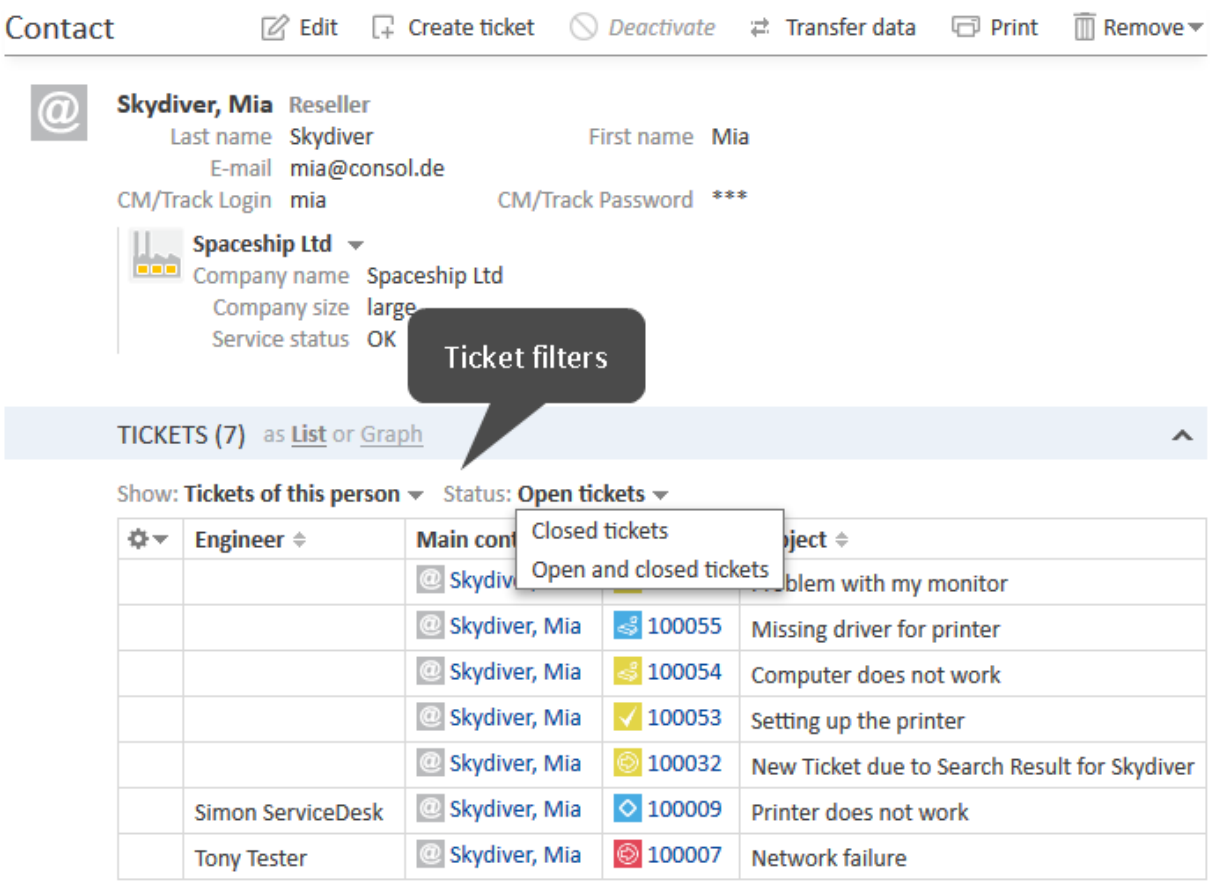


Figure 101: *ConSol CM Web Client - Ticket filters on contact page*

The displayed tickets depend on your selection in the two ticket filters:

- **Show**
  - **Tickets of this person:** Tickets where the contact is either the main or an additional customer.
  - **Tickets of this person (main customer only):** Tickets where the contact is the main customer.
  - **All tickets of the company:** Tickets where the company of the contact, the contact itself or another contact of the company is the main or additional customer.
- **Status**
  - Open tickets
  - Closed tickets
  - Open and closed tickets

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.2.2.10 Example 10: Customer Group Displayed in Quick Search

The **customer group is displayed for all search results** in the list. The following notation is used:

```
<Localized name of customer field group> (<localized name of customer group>)
```

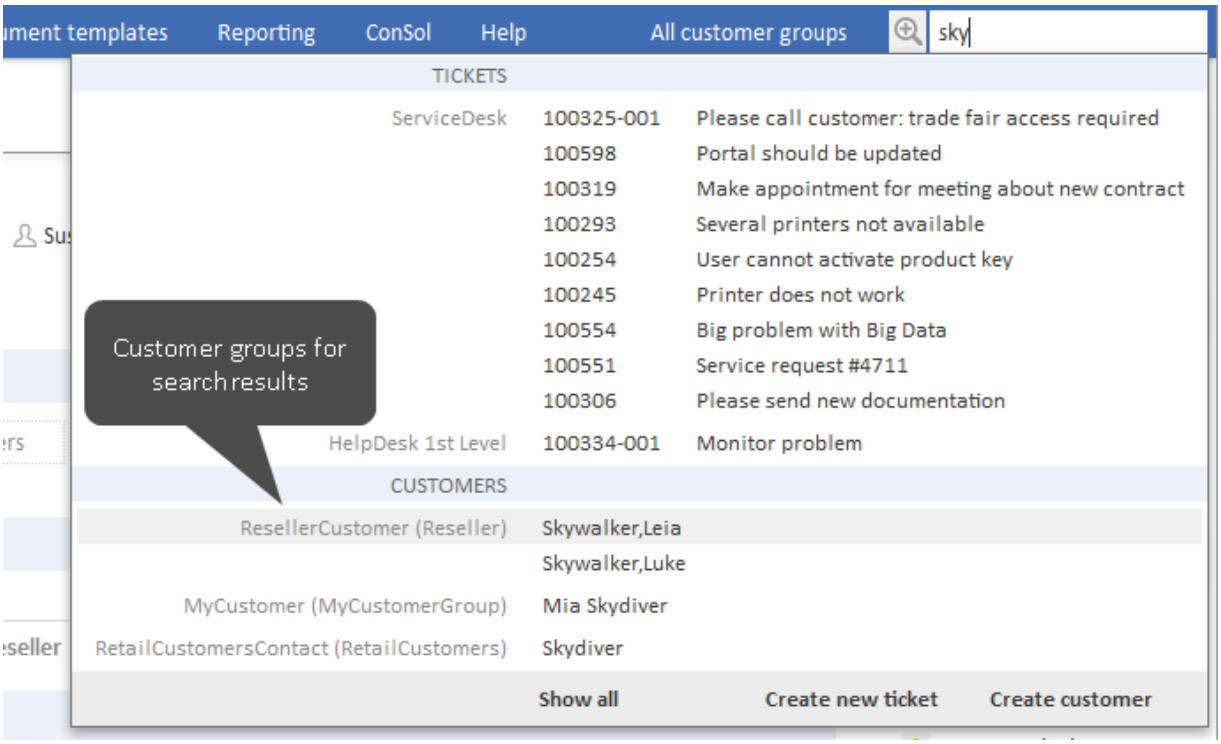


Figure 102: *ConSol CM Web Client - Search results for Quick Search*

C.2.2.11 Example 11: Using Search Actions for Company or Contact Data

See section Action Framework - Search Actions.

C.3 Setting Up the Customer Data Model

This chapter discusses the following:

C.3.1 Introduction to Setting Up the Customer Data Model 180

C.3.2 Managing Contacts and Companies Using the Admin Tool 181

C.3.1 Introduction to Setting Up the Customer Data Model

You can implement various customer data models. Please refer to section Introduction to Customer Data Models for a detailed introduction. To work with a new customer data model within a certain customer group (or in several customer groups), the following steps have to be performed:

1. Create a customer data model.
   (This implies you have already decided whether this should be a one- or a two-level data model. In this example, we will create a two-level model.)
2. Create a new customer group.
3. Assign the customer data model to the group.

A customer data model comprises objects on three model levels:

1. **The customer data model definition**
2. **The customer objects within this model**
   A customer object can be of one of two types:
   a. **Company**
      E.g., an institution, but can also be a machine, a ship, or anything else which represents the company level.
   b. **Contact**
      E.g., a person, but can also be a machine, a hardware device, a product, or anything else which represents the contact level.
      If a company level is present, the contact is a sub-level of the company. For a simple customer data model, use only the contact object or only the company object.
3. **The customer fields**
   These are the data fields for the customer objects, i.e., either the customer fields for company data (e.g., ZIP, address, phone) or the customer fields for contact data (e.g., surname, name, email address).

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.3.2 Managing Contacts and Companies Using the Admin Tool

To manage components of the customer data model, use the items in the navigation group *Customers* in the Admin Tool. Open the navigation item *Data Models* to set up a new data model or to edit existing models.

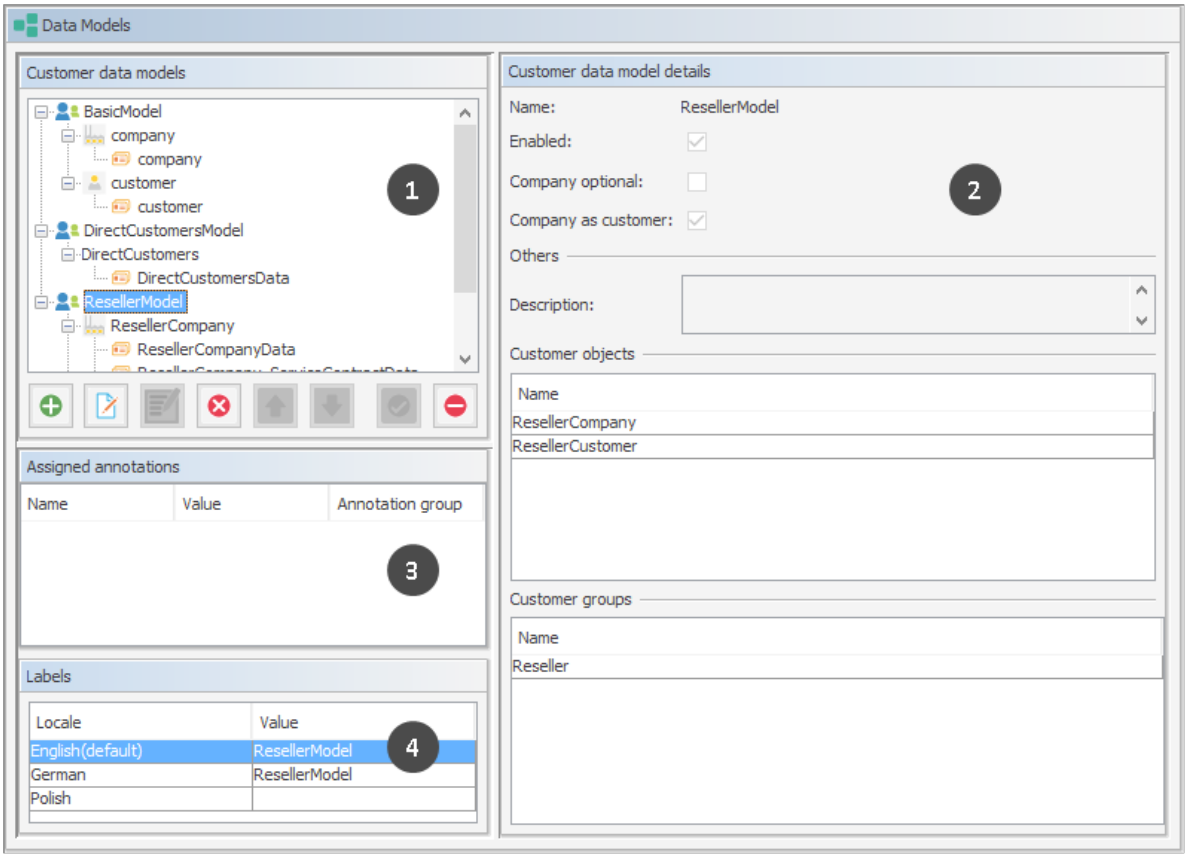


Figure 103: *ConSol CM Admin Tool - Customers, Data Models: Customer data model definition*

The figure shows the following sections:

- tree of all components of all customer data models (1)
- details of the selected component (2)
- annotations for customer field groups of a data model, empty for data models or if no group is selected (3)
- labels for the selected component (4)

To explain how to work with the customer data model, we will walk you through an example in the next sections.

C.3.2.1 Creating a New Two-Level Customer Data Model

To create a new customer data model you have to create the objects on all levels of the data model. In the following example, we will build a customer data model for partner data. We will create a customer data model with a company and a contact object, i.e., we will have to create the following objects:

- the customer data model itself
- the company object (1st level)

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- the customer fields for the company
- the contact object (2nd level)
- the customer fields for the contact

After having defined an object, the parameters for this object can be (or rather should be) configured.

Step 1: Create the Customer Data Model with the First Customer Object

When you create a new customer data model, you have to add a customer object and the respective customer fields in one step.

To create a new customer data model, mark another customer data model (that way you select the level on which you want to work) and use the *Add* button to open the pop-up window.

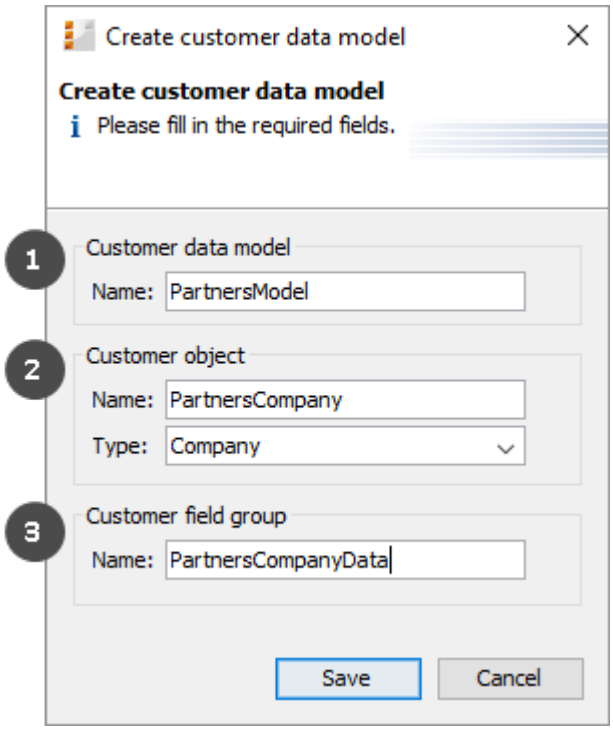


Figure 104: *ConSol CM Admin Tool - Customers, Data Models: Creating a new customer data model*

You have to fill in the following fields:

- **Customer data model (1)**
  - **Name**
    The name of the new customer data model. The localized names for the data model are set in the *Labels* section of the main *Data Models* panel. For details, please refer to section Localization of Data Fields.
- **Customer object (2)**
  - **Name**
    The unique technical name of the company/contact object. The localized names for the customer object are set in the *Labels* section of the main *Data Models* panel. For details, please refer to section Localization of Data Fields
  - **Type**
    Select *Contact* or *Company*. There can be only one company object and one contact object within one customer data model.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Customer field group (3)**
  - **Name**
    The unique technical name of the first customer field group for company data within the defined customer object. More customer field groups can be added later on. The localized names for the customer field group are set in the *Labels* section of the main *Data Models* panel. For details, please refer to section Localization of Data Fields

Step 2: Create Another Customer Object

In the next step, you have to add the contact object. Select the object *PartnersCompany* (to set the correct level for the following *Add* operation) and click the *Add* button to open the pop-up window.

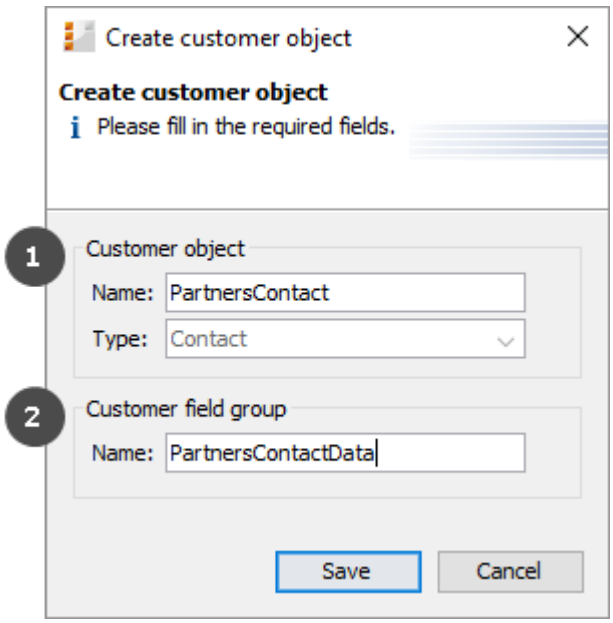


Figure 105: *ConSol CM Admin Tool - Customers, Data Models: Adding a new customer object*

You have to fill in the following fields:

- **Customer object (1)**
  - **Name**
    The unique technical name of the contact object. The localized names for the customer object are set in the *Labels* section of the main *Data Models* panel. For details, please refer to section Localization of Data Fields. The localized name will also be used as header of the customer page (i.e., contact or company page, see section Example 3: Using Company and Contact Page).
  - **Type**
    Here, *Contact* is pre-selected and cannot be modified, because a company object is already present in the customer data model.
- **Customer field group (2)**
  - **Name**
    The unique technical name of the first customer field group for contact data within the defined customer object. More customer field groups can be added later on. The localized names for the customer field group are set in the *Labels* section of the main *Data Models* panel. For details, please refer to section Localization of Data Fields

Step 3: Configuring the Parameters for the Defined Objects

Parameters for the Customer Data Model

Double-click on the name of the customer data model (*PartnersModel* in our example) or mark the customer data model in the list and click the *Edit* button to open the pop-up window where you can define the parameters for the model.

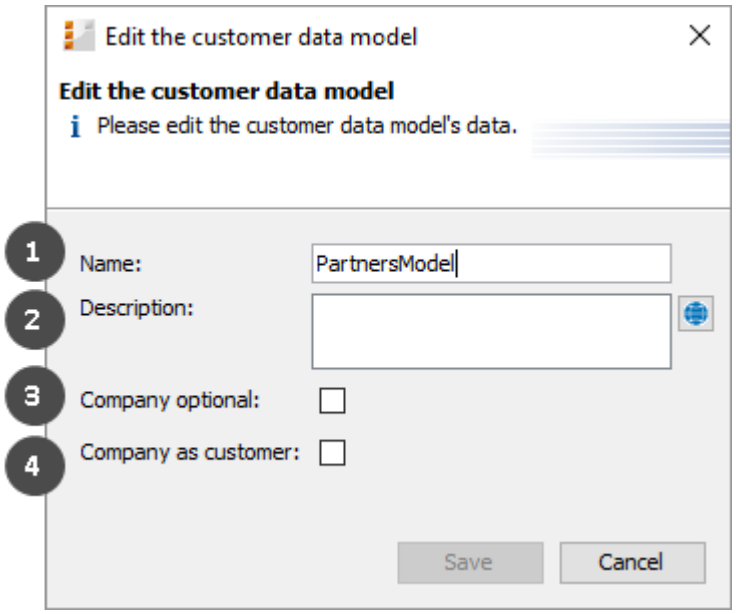


Figure 106: *ConSol CM Admin Tool - Customers, Data Models: Parameters for a customer data model*

You can fill in the following fields:

- **Name (1)**
  Check or modify the existing name of the model.
- **Description (2)**
  Optional description. Can be localized. Please see section Localization of Objects in General, Type 1 for details.
- **Company optional (3)**
  If this checkbox is checked, it is possible to add a contact to a ticket without the contact being part of a company. The company might be set later but it is not required. So, here you can enable working with single contacts, even within a two-level customer data model.
- **Company as customer (4)**
  Check this checkbox if it should be allowed to create tickets not only for contacts, but also for companies within the model.

Parameters for the Customer Object

Double-click on the name of a customer object, e.g., the *PartnersCompany* or select the object and click the *Edit* button to open the pop-up menu where you can configure the parameters for this object.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

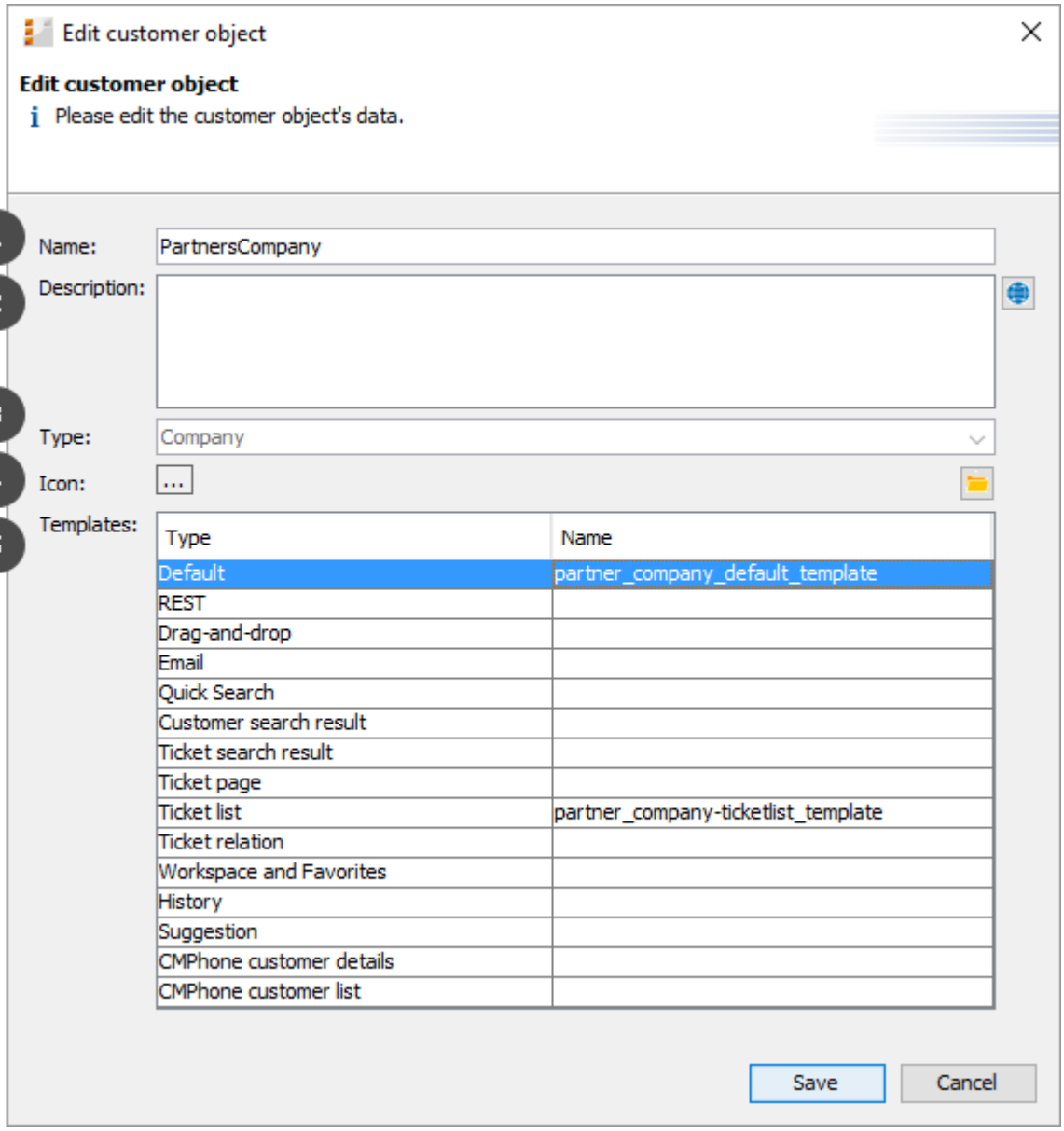


Figure 107: *ConSol CM Admin Tool - Customers, Data Models: Parameters for a customer object*

You can fill in the following fields:

- **Name (1)**
  The unique technical name of the customer object with technical name and localized name(s). The localized name will also be used as header of the customer page (i.e., contact or company page, see section Example 3: Using Company and Contact Page).
- **Description (2)**
  The description of the customer object. Will be used in future ConSol CM versions. Can be localized. Please see section Localization of Objects in General, Type 1 for details.
- **Type (3)**
  A read-only field which displays the type (*contact* or *company*) of the customer object.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Icon (4)**
  The icon for all companies within this model. It will be displayed in the Web Client. You can either use one of the standard CM icons using the button (...) or upload an icon using the *File explorer* button.
- **Templates (5)**
  The templates which are used to render the data of the customer object, i.e., the templates which define the data fields that are displayed in the Web Client for objects of this type. There are various positions in the GUI for which the layout of the company or contact data can be defined. The templates are stored in the *Script and Template* section of the Admin Tool. Please see section Templates for Customer Data for a detailed explanation.

Parameters for the Customer Field Group

Double-click on the name of the customer field group to change the technical name (only possible when respective fields in the customer data sets are empty) or to assign/unassign Dependent Enum Scripts for the group.

To define the customer fields, use the GUI elements on the right-hand side. In the following figure, an example for the definition of customer fields for the *PartnersCompany* is shown. Here, only one customer field group (*PartnersCompanyData*) is used. You can use as many customer field groups as you need in one customer object.

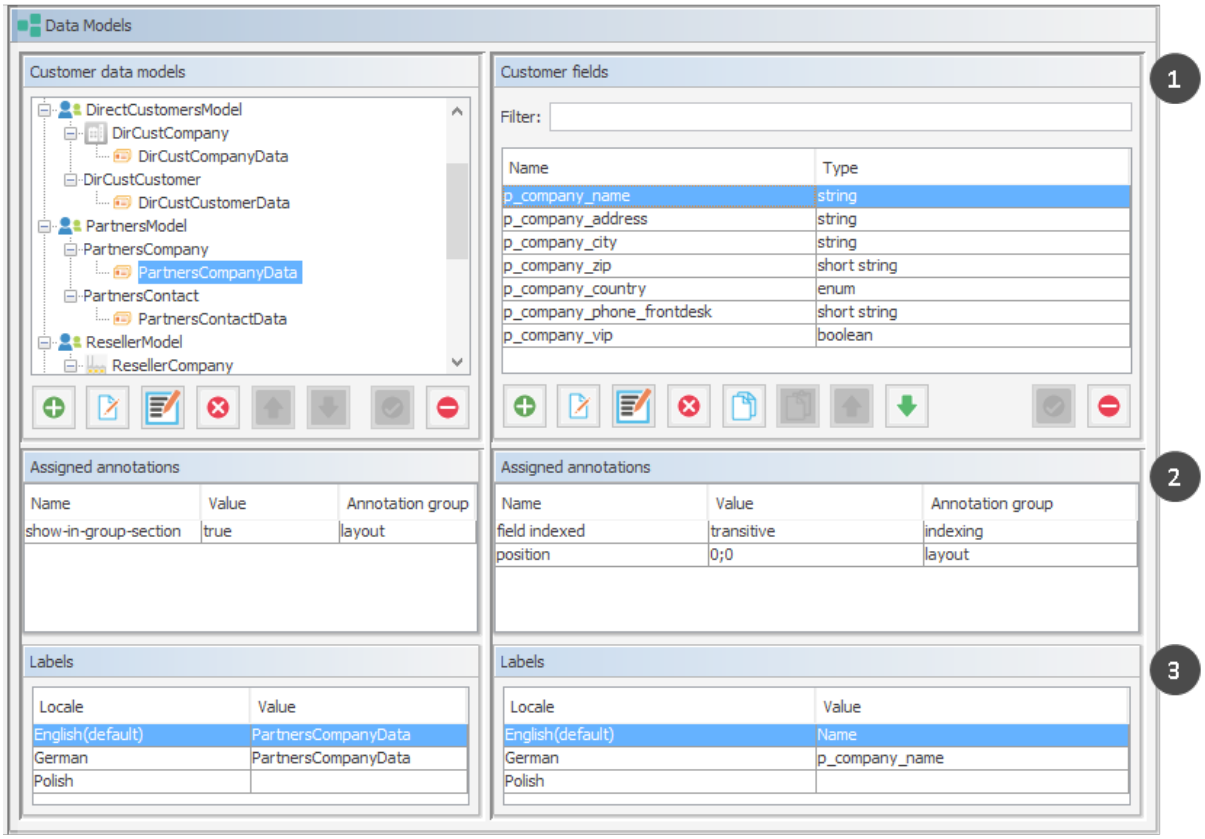


Figure 108: *ConSol CM Admin Tool - Customers, Data Models: Parameters for the customer field group*

The figure shows the following sections:

- list of customer fields in the selected customer field group (1)
- field annotations for the selected customer field (2)
- labels for the selected customer field (3), i.e., the localized values for the name of the customer field

The definition of customer fields within customer data models is based on the same principles as the definition of ticket fields for ticket data. For a detailed introduction to the definition and management of ticket fields, please refer to sections Ticket Fields (Setting Up the Ticket Data Model) and Customer Field Management and GUI Design for Customer Data.

The available customer field annotations are listed in section Annotations. The annotation `unit is a contact` is no longer in use because the level of a unit (i.e., the company or contact) is defined by its unit type (*company* or *contact*).

> ⚠ Please make sure that the annotation `field indexed` is set for all fields which should be searchable. This affects the Quick Search, Detailed Search, and all auto-complete operations! See also section Search Configuration.

Congratulations! When you have completed all the steps in the previous sections, you have created a new ConSol CM customer data model and can now go on to assign the model to one or more customer group(s).

C.3.2.2 Creating a New Customer Group Using the New Customer Data Model

When the customer data model has been defined, it can now be assigned to one or more customer groups. In the example, we will create the new customer group *OurPartnerCompanies* which will use the new *PartnersModel*.

Use the navigation item *Customer Groups* in the navigation group *Customers* of the Admin Tool to create a new customer group and to assign the desired customer data model.

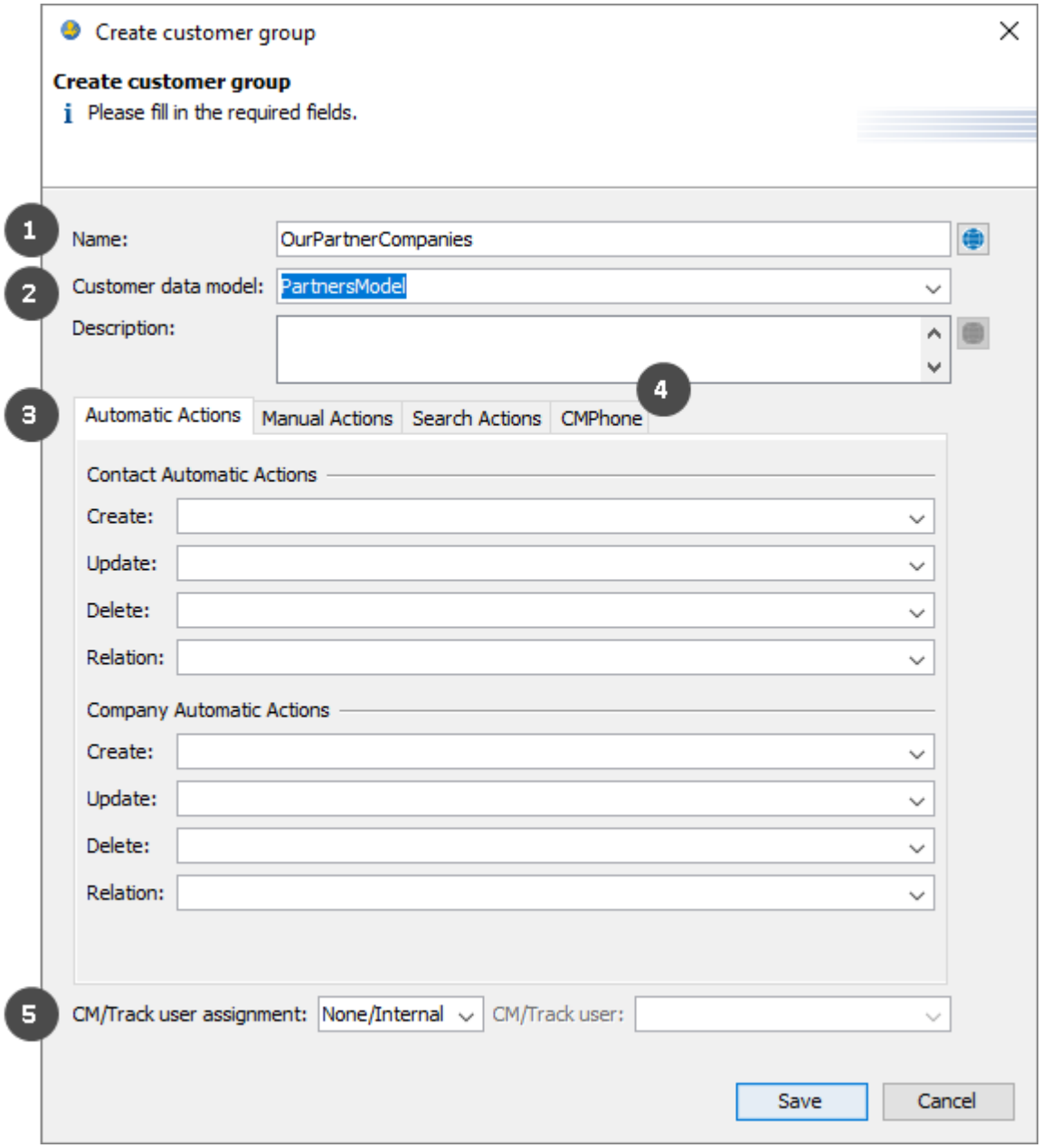


Figure 109: *ConSol CM Admin Tool - Customers, Customer Groups: Definition of a new customer group*

You can fill in the following fields:

- **Name (1)**
  The unique technical name (and localized name) of the new customer group. Can be localized. Please see section Localization of Objects in General, Type 1 for details.
- **Customer data model (2)**
  Select the desired data model from the drop-down menu.
- **Automatic/Manual/Search Actions (3)**
  On those tabs you can assign customer actions. This is explained in detail in section Action Framework - Customer Actions.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **CMPhone (4)**
  This tab will only be displayed if CM/Phone is installed. Please see section CM/Phone: CTI with ConSol CM for details.
- **CM/Track user assignment (5)**
  Select the assignment mode of the CM/Track user profile. Please see section Defining the User Assignment Mode for further information.

An engineer who has access permissions for six customer groups will see the following in the Web Client.

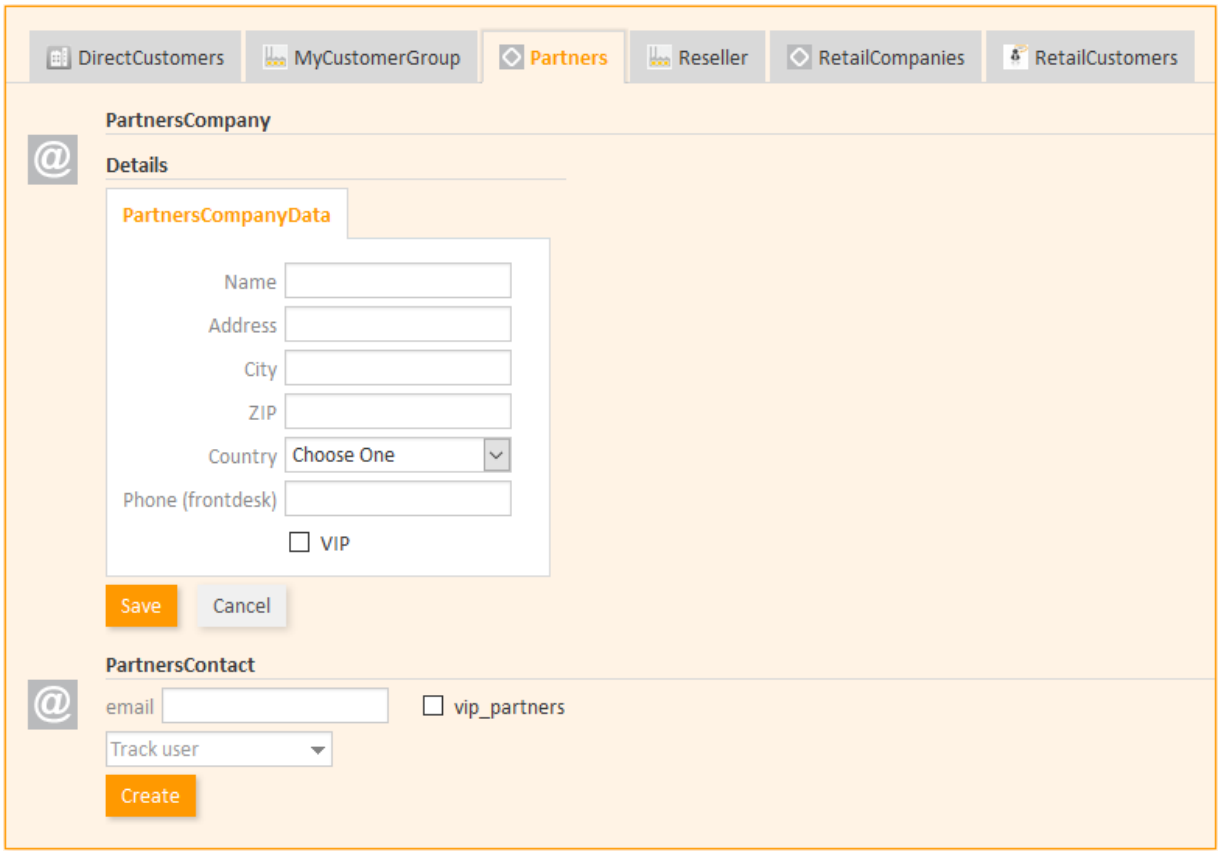


Figure 110: *ConSol CM Web Client - Creating a new company and contact*

C.3.2.3 Assigning Access Rights for Customer Groups with the New Model to Roles

In order to let engineers work with customer data from the new customer group, i.e., to create new partner data sets or to modify existing sets, you have to grant access permissions for the customer groups to one or more roles.

See section Tab Customer Group Permissions.

C.3.2.4 Assign the New Customer Groups to Queues

Please keep in mind that you have to assign the new customer group to all queues where tickets should be created for customers of this group. See section Queues for details.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.3.2.5 Deactivate Objects in the Customer Data Model

You can only deactivate an entire customer data model. It is not possible to deactivate company or contact objects or customer field groups. The button to deactivate these objects is present, but takes no effect.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.4 Customer Field Management and GUI Design for Customer Data

This chapter discusses the following:

C.4.1 Introduction 191
C.4.2 Defining Customer Fields for Customer Data Using the Admin Tool 191
C.4.3 Scripting Using Customer Objects 202
C.4.4 Using Scripted Field Visualization for Customer Fields 205

C.4.1 Introduction

One feature of ConSol CM is the great flexibility as far as customer data model and GUI design are concerned. You, as an administrator, can define any data field which is required and place it in the user interface wherever it is suitable. The basic principle is now the same as the one you know for ticket fields: full flexibility.

The management of the ticket data model and GUI design is explained in section Ticket Fields (Setting Up the Ticket Data Model). The management of objects within the customer data model is explained in section Setting Up the Customer Data Model. Please refer to those sections for a detailed explanation. In this chapter, we assume that you have a working knowledge of those topics.

C.4.2 Defining Customer Fields for Customer Data Using the Admin Tool

C.4.2.1 Admin Tool GUI

The data field definition for customer data is part of the definition of the entire customer data model, see section Setting Up the Customer Data Model. The data model is defined on the navigation item *Data Models* in the navigation group *Customers*, in the Admin Tool.

Data fields for customer objects within the customer data model are called *customer fields*. Customer fields are based on the same principles as ticket data fields (ticket fields): data fields are managed in groups and the groups, as well as the single fields, can be annotated.

You reach this screen by opening the navigation item *Data Models* in the navigation group *Customers*.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

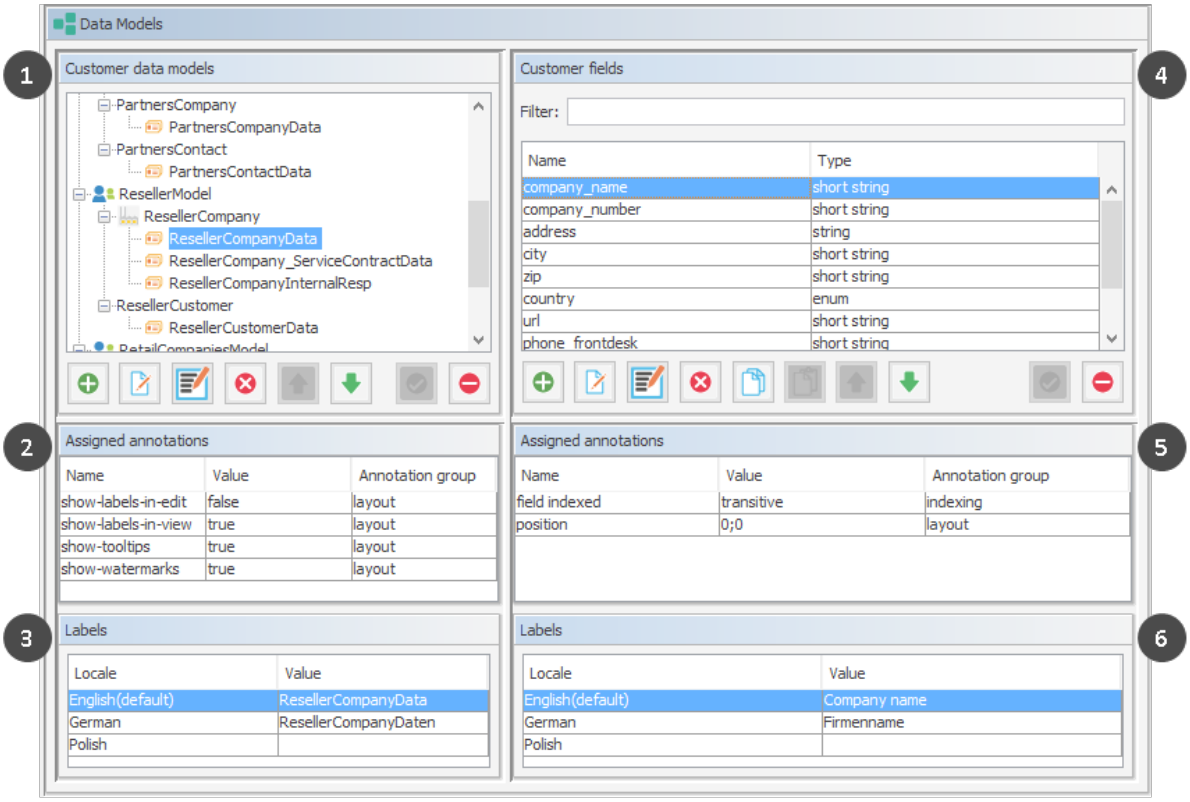


Figure 111: *ConSol CM Admin Tool - Customers, Data Models: Definition of customer fields*

The *Data Models* screen consists of six sections:

- Customer data models (1)
- Annotations for the selected customer field group (2)
- Labels for the selected customer field group (3), i.e., the localized values for the name of the customer field group
- Customer fields of the selected customer field group (4)
- Annotations for the selected customer field (5)
- Labels for the selected customer field (6), i.e., the localized values for the name of the customer field

C.4.2.2 Customer Field Groups

Like the ticket fields which you are already familiar with from previous CM versions, customer fields are placed in groups, the *customer field groups*. Each customer object within a customer data model can have as many customer field groups as required. For example, for a reseller company there can be a customer field group for the general data, one for the contract data, and one for the persons who are responsible for this reseller. For contacts within the reseller data model, one customer field group with general data is defined.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

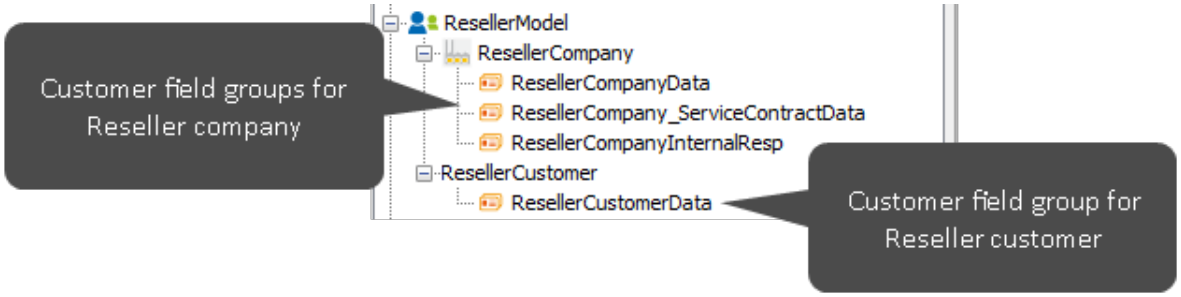


Figure 112: *ConSol CM Admin Tool - Customers, Data Models: Customer data model with several customer field groups*

The organization of data fields in groups has several implications. Please keep them in mind to make sure your data model design meets the users' needs.

A customer field group ...

- can be faded in and out in the GUI during the process, but only the whole group, not single fields (= customer fields).
- can be displayed as a tab or in the customer data section. The title (and mouse-over) of the tab is the (localized) name of the customer field group. For details about localization of customer field groups, please refer to section Localization of Data Fields.
- is configured by group annotations.
- is placed in the GUI based on its position in the customer field group list (defines, e.g., the order of tabs).

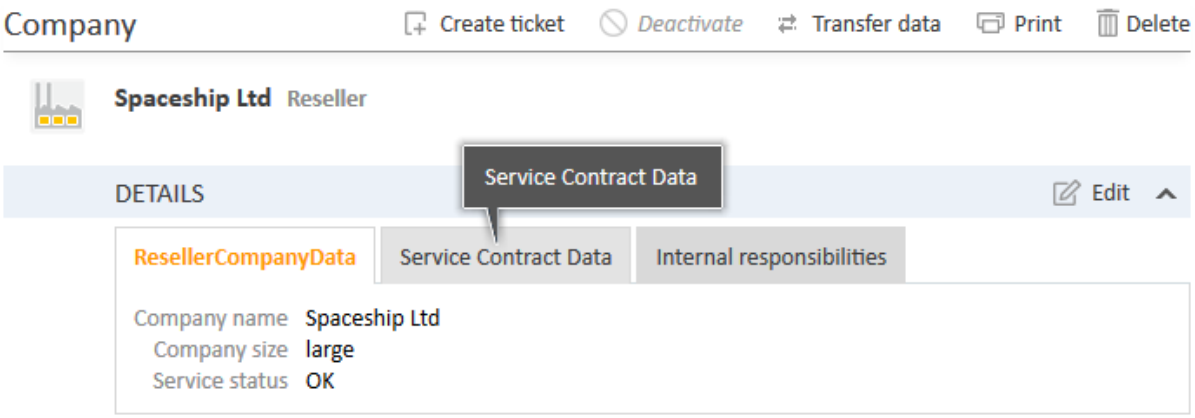


Figure 113: *ConSol CM Web Client - Company data organized in tabs (based on customer field groups)*

C.4.2.3 Customer Field Definition

The definition of customer fields (i.e., data fields like *name*, *address*, or *phone number*) is based on the proven concepts which have been used for ticket fields since the first CM6 version.

A customer field ...

- is defined by a data type.
- is configured using field annotations (e.g., `position` or `field-indexed`), as described in Annotations.

Types of Data Fields

The following data types are available for ticket fields, customer fields and resource fields.

- **autocomplete**
  A data field which contains a text autocomplete field. This is a dynamic list which is based on a script of type *Text Autocomplete*. A detailed explanation of text autocomplete fields is given in section Scripts of Type Text Autocomplete.
- **boolean**
  Values: true/false. Depending on the annotation `boolean-type`, the value is displayed as check-box, radio buttons, or drop-down list.

  > If a boolean field has not been touched, its value is “NULL”. If it is checked, its value is “true”, and if it is unchecked again, its value is “false”.

- **date**
  Format and accuracy can be set by annotations.
- **enum**
  For sorted lists. The engineer can choose one of the enum values in the Web Client. Enums and values have to be created previously within the Managing Sorted Lists: Enums. Select the desired *Enum type* and *Enum group* in the fields below.
- **list**
  A data field of this data type is the first step to creating a list (one column) or a table (multiple columns) of input fields in the Web Client.
  - For a **table** the next step will be to create another field of type *struct* (see below) to contain the input of the individual list fields (which will become the columns of the table). So, if you want to create a table you have to define a field of the type *struct* first (see below) before you can add the fields for the table columns.
  - For a **simple list**, the next step will be to create fields which belong to the list. No *struct* is required.

  For all fields belonging to a list or table you have to set the dependencies in the field *Belongs to* (see below). For example, a table field (which is a regular data field) always belongs to a *struct*, a struct always belongs to a *list.*
- **struct**
  A data field of this type defines a data structure (line of a table) which groups one or multiple fields. It is the second step to building a table after you have created a field of the type *list*. Add the fields for the columns of the table in the next step. The dependencies have to be set for each field in the *Belongs to* field (see below), i.e., a *struct* always belongs to a *list*.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

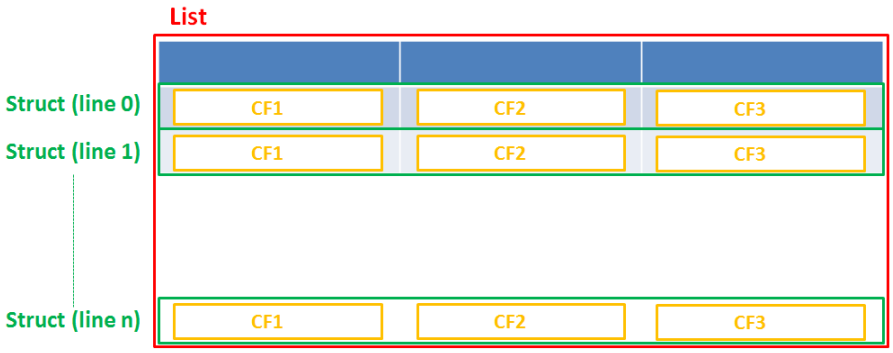


Figure 114: *Scheme: List of structs*

Technically spoken, the list is an array which contains a map (= key:value pairs) in each field.

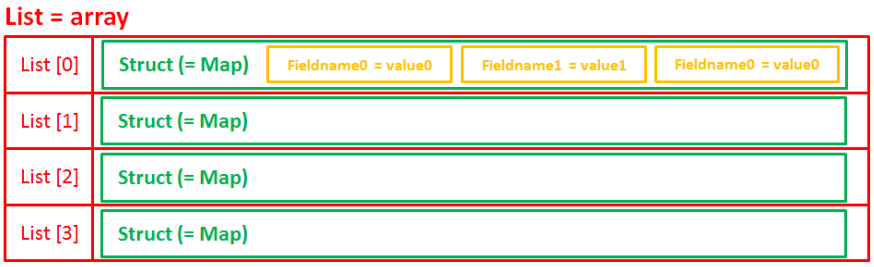


Figure 115: *List of structs, technical principle*

- **number**
  For integer values.
- **fixed point number**
  For numbers with a fractional part, e.g., currencies. You have to enter the total number of digits (*Precision*) and the number of digits that fall to the right of the decimal point (*Scale*) in the respective fields below.
- **string**
  For up to 4000 alphanumeric characters.

  ⚠ Restriction when using an Oracle database: at most 4000 bytes can be saved in UTF encoding. Starting with Oracle12c.

- **long string**
  For large objects.

  ⚠ For *long strings* the limit depends on the database system used for ConSol CM: MS SQL Server: 2 GByte; MySQL: 4 GByte; Oracle: 16 - 64 GByte (depending on page size of tablespace).

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **short string**
  For up to 255 alphanumeric characters.

  > For *string* fields, you can use specific annotations to fine-tune the field definition. For example, a *string* field can be defined to contain a URL which will automatically be displayed as hyperlink or can be the hook for an autocomplete list. Please read the following section.

- **contact data reference**
  Special data type used internally for referencing the contacts associated with a ticket. This data type is not displayed but only used internally in the CM system.
- **MLA field**
  This data type is used for fields that contain hierarchical lists with a tree structure called *MLA* (Multi Level Attributes). The name of the field is the name of the new MLA that has to be defined within the MLAs. The group of the field has to be referenced when the MLA is created.

> ⚠ The data type you choose on creating a data field cannot be changed afterwards!

Using Annotations to Fine-Tune Data Fields

String fields can be used for various purposes, from simple input fields to text boxes, URLs or passwords. The configuration is done using annotations, see Annotations. The most commonly used annotation are listed in the following section.

How can I ...

... insert a **text box** instead of a single line?

Value for the annotation `text-type`: “textarea”

The size of the text box can be adjusted, displayed as standard text box depending on web browser. Use the `field-size` annotation in case a specific size of the text box is required.

... hide the input of the fields for **passwords**?

Value for the annotation `text-type`: “password”

Only dots are displayed. This annotation does **not** define the field to contain a password. It only defines the display mode. Use the `password` annotation to define a string field to contain the CM/Track password.

... display a **hyperlink**?

Value for the annotation `text-type`: “url”

The input is displayed as a hyperlink in view mode. The string has to match a specific URL pattern:

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

The first part of the string is the link (url), the second part is the name which should be displayed.

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

... display a **file link**?

Value for the annotation `text-type`: “file-url”

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

The input is displayed as a link to a file on the file system. The web browser has to allow/support those links.

Example: Enabling file:// URLs in a Firefox browser

Add the following lines to either the configuration file `prefs.js` or to `user.js` in the user profile. On a Windows system usually in a folder like `C:\Users\<USERNAME>\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");

Alternatively a Firefox browser add-on like ***Local Filesystem Links*** can be installed for better access to the referenced files and folders.

The link is also displayed as a tooltip.

The URL is correctly formed if the following conditions are met:

- It starts with `file:` followed by regular slashes:
  - three slashes "///" for files on the same computer as the browser (alternatively "//localhost/")
  - two slashes followed by the server name followed by another slash for files on file servers accessible from the computer running the browser
- These are followed by the full path to the file ending with the file name.
- The path on Microsoft Windows systems is also written with forward slashes instead of backslashes.
- The drive letter of a local path on Microsoft Windows systems is noted as usual, for example `C:`.
- Paths with spaces and special characters like "{, }, ^, #, ?" need to be percent encoded ("%20" for a space for example) for Microsoft Windows systems.

Example URLs:

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

See also the explanation about `file-url` in the section text-type

... define a **label**?

Value for the annotation `text-type`: “label”

This is a read-only field which is displayed in label styling. Use the `label-group` annotation to link the label and the input fields which belong together. Please take a look at the annotations for labels (`show-label-in-edit`, `show-label-in-view`) before implementing special label fields.

... define a **title**?

Value for the annotation `text-type`: “title”

This is a read-only field which is displayed in title styling, i.e. in a bold font which is larger than regular labels.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

... define a field for the **valid email addresses**?

Value for the annotation `email`: “true”

The field may only contain valid email addresses. The input is validated according to the standard email format <name>@<domain>.

... define a scripted autocomplete list?

Value for the annotation `text-type` = “autocomplete”

Optional: value for the annotation `autocomplete-script` = <name of the respective script>

A scripted autocomplete list is used to show a drop-down list which is filled dynamically using the input the engineer has provided so far. For example, when the user types "Mil", the possible values "Miller", "Milberg", and "Milhouse" are displayed as a list and the engineer can select the one required for the field. The values are based on a result set which can contain strings, engineers, tickets, customers or resources.

The logic to create the result set needs to be implemented in a script, see Scripts of Type Text Autocomplete.

... define a field for the **CM/Track login**?

Value for annotation `username`: “true”

Will be used for authentication against CM/Track server. Only for customer fields in a contact object.

... define a field for the **CM/Track password**?

Value for annotation `password`: “true”

Will be used for authentication against CM/Track server (in DATABASE mode). Only for customer fields in a contact object.

... define a field containing personal data?

Value for annotation `personal-data`: “true”

This annotation can be assigned to ticket and contact fields. Contact fields with this annotation will be deleted when a contact is anonymized. Ticket fields with this annotation will be deleted when the main customer of the ticket is anonymized. See Example 8: Removing Customer Data for information about how to anonymize a contact.

> ⚠ When defining a field to contain personal data, please take into account that the deletion of the field during the anonymization process is treated as a regular update. Therefore, business event triggers reacting on changes to ticket fields fire, and the contact update action script is executed.
>
> This might lead to unwanted side-effects.

... define a **rich text** field?

Create a field of the type *long string*.

Value for the annotation `string-content-type`: “html-inline” (field content is directly displayed on the GUI in view mode) or “html-link” (field content is displayed in a pop-up window after clicking the *Show content* link in view mode)

A rich text field can contain images and formatted text. In the edit mode, it features a basic editor.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

... define a tooltip for a data field?

Value of the annotation `show-tooltip` = “true”

When creating or editing a data field, you can enter a text in the *Description* field. This text is used as a tooltip in the Web Client. You can enter a localized description using the globe icon. For details, please refer to section Localization of Objects in General, Type 1.

> This does not apply to date fields and fields containing phone numbers and URLs, as they have special tooltips.

... optimize the data field layout?

Use three columns to display the data fields.

Display data fields which most likely have a value on the left in order to avoid gaps.

Add the annotations `position` = 0;0 and `colspan` = 3 to all tables. Remove the `colspan` annotations from other types of data fields.

Show the labels for all data fields (annotation `show-label-in-view` should be `true`).

Labels for Customer Fields

You can decide whether watermarks or labels (or both) should be used. Labels are implemented similar to the labels for ticket fields. Furthermore, tooltips can be added. The display mode of labels for customer fields is controlled by annotations. Annotations can be set on customer field group level and on customer field level. The field annotations overwrite the group annotations.

If no annotations are set, the customer field groups are rendered with the default configuration. This means:

- standard labels
- watermarks disabled
- tooltips enabled

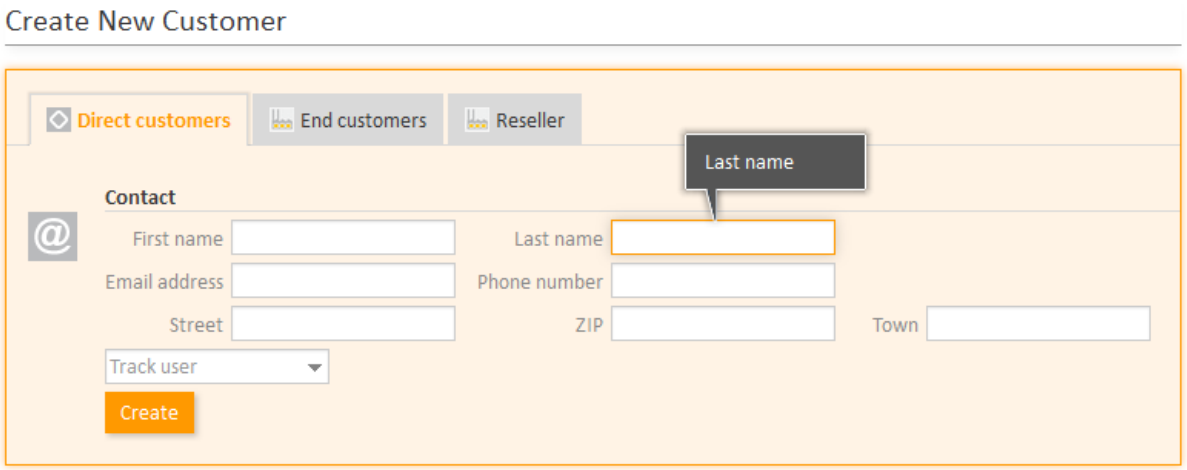


Figure 116: *ConSol CM Web Client - Default display mode for customer fields*

You can set annotations to modify the behavior of a field group or one or more single fields, as demonstrated in the example below.

Annotations for customer field groups:

- layout:show-labels-in-view (“true” if not set)
- layout:show-labels-in-edit (“true” if not set)
- layout:show-watermarks (“false” if not set)
- layout:show-tooltips (“true” if not set)

Annotations for customer fields:

- layout:show-label-in-view
- layout:show-label-in-edit
- layout:show-watermark
- layout:show-tooltip

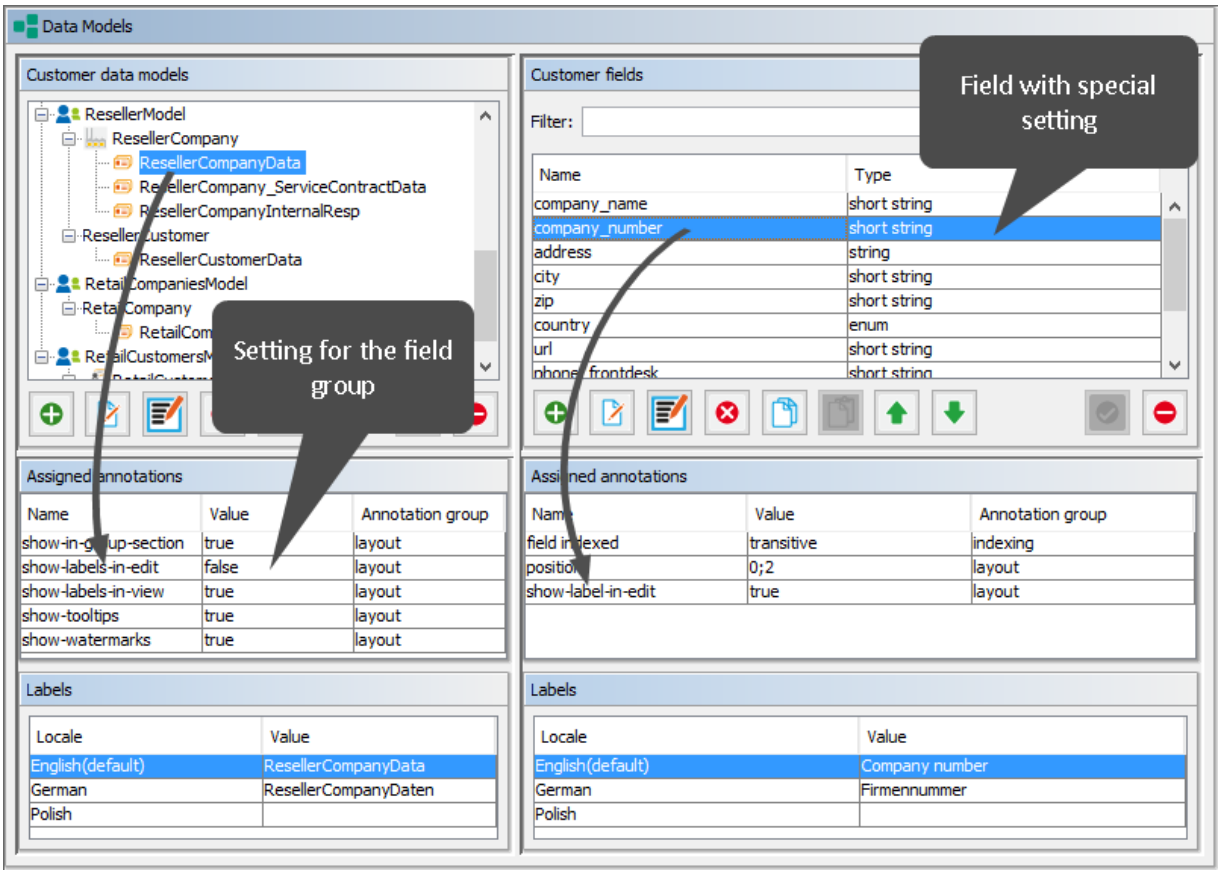


Figure 117: *ConSol CM Admin Tool - Customers, Data Models: Configuration for Reseller customer field group, one field with special configuration*



Figure 118: *ConSol CM Web Client - View for Reseller customer field group, one field (company_number) annotated field-specific (edit mode)*



Figure 119: *ConSol CM Web Client - View for Reseller customer field group, one field (company_number) with special configuration (Example: No influence on view mode)*

The labels are inserted automatically, so in the end there might be, at most, six columns. In the following example, two columns are used.

- `Field_1: position 0;0`
- `Field_2: position 0;1`
- `Field_3: position 1;0`
- `Field_4: position 1;1`
- `Field_5: position 2;0`
- `Field_6: position 2;1`

Label_1	Field_1	Label_2	Field_2
Label_3	Field_3	Label_4	Field_4
Label_5	Field_5	Label_6	Field_6

Figure 120: *Example for customer field positions in the grid*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.4.3 Scripting Using Customer Objects



In this book we will use the terms *customer* and *customer definition*. In previous versions of ConSol CM, the term *data object* was used to refer to customers. However, the names of the corresponding Java classes are *Unit* and *UnitDefinition*. All other Java classes which deal with customer objects are also still named *Unit*... Please keep that in mind if you work as both a ConSol CM administrator and programmer. Please refer to the *ConSol CM Java API Doc* for details.

Use the following notation to retrieve unit field (customer field) data:

- **For one field:**
  unit.get("group1:name")
- **For numerous fields:**
  unit.get("group1:field1.group2:field2")

```
unit.get("contactFields:companyReference.companyFields:name")
```

Code example 2: *Example to get the name of the company's contact*

You can access the customer data model objects from different scripts:

- **Workflow:**
  - Scripts in workflow activities
  - Scripts in workflow conditions
- **Admin Tool (AT) scripts of type:**
  - Dependent enum
  - Email
  - Clone
  - Default values
  - Customer action
  - Customer condition
  - Workflow

C.4.3.1 Convenience Methods

Examples of convenience methods:

Object.Method	Explanation
`def contacts = unit.get("contacts()")` `List contacts = company.getContacts()`	Using CFEL ("contacts()") a list of all contacts is retrieved for the company (unit).

Object.Method	Explanation
`Unit company = mainContact.getCompany()`	For a contact, the company can be retrieved easily.
`newContact.set("company()", newCompany)`	For a (new) contact, the company is assigned the CFEL expression "company()", provides easy access to the company object.
`List tickets = company.get ("tickets()")`	For a company, all tickets are retrieved.

`Ticket ticket = getTicket();`
`Unit mainContact = ticket.getMainContact()`
`List tickets = mainContact.get ("tickets()")`
	For a contact, all tickets are retrieved.

`Integer count = contact.get ("company().contacts()[0].tickets ()[count]");`	A chain of expressions is used to get the number of tickets for a specific contact.

```
TicketCriteria ticketCriteria = new TicketCriteria();
Unit patternContact = new Unit("contact", customerGroup);
mdcmCriteriaBuilder.setReferencedContactCriteria(ticketCriteria, patternContact);
```

Code example 3: *Example 1: Search for the tickets of a contact or of a company*

```
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 example 4: *Search for the tickets of the contact who is member of a certain company*

```
UnitCriteria unitCriteria = new UnitCriteria();
Unit companyPattern = new Unit("company", customerGroup);
mdcmCriteriaBuilder.setReferencedCompanyCriteria(unitCriteria, companyPattern);
```

Code example 5: *Search for contacts of a certain company*

For detailed information about the methods, including input parameters and output data type (method signatures), please refer to the *ConSol CM Java API Doc*.

C.4.3.2 Important Objects

The objects which are available in the script obviously depend on the script's context. The following examples demonstrate some of the possible use cases:

Startig point	Script	Objects	Example
Company page	customer action script	`unit`, represents the company	`def contacts = unit.get("contacts()")`
Contact page	customer action script	`unit`, represents the contact	`List tickets = unit.get("tickets()")`
Workflow activity	workflow action or condition script	`ticket`	`def id = ticket.getId()`
Workflow activity with script in AT	workflow action or condition script	`ticket` not present implicitly!	`import com.consol.cmas.common.model.ticket.Ticket` `def id = ticket.getId()`

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.4.4 Using Scripted Field Visualization for Customer Fields

Using scripted field visualization, you can enhance the display of data in customer fields. Please see section Scripts of Type Field Visualization for details.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.5 Templates for Customer Data

This chapter discusses the following:

C.5.1 Introduction to Using Templates for the Display of Customer Data 206

C.5.2 Coding Templates 207

C.5.3 Template Types 210

C.5.1 Introduction to Using Templates for the Display of Customer Data

In the ***ConSol CM*** Web Client, customer data sets are displayed in a short form, the so called display name of the customer, at various locations. The display name is based on the template for the respective location. In this way, it is possible to use a different display name for each location, for example to adapt to the available screen space.

The configuration of the display name is based on the following principle:

- Templates are assigned to a customer object, i.e., to a contact definition or company definition, in the navigation group *Customers*, navigation item *Data Models* in the Admin Tool. Several template types are available and control the display in certain areas of the Web Client. A template of type *Default* is always required and serves as a fallback for all other template types.
- The referenced template must be stored in the Admin Tool, navigation group *System*, navigation item *Scripts and Templates*, tab *Templates*. The name of a template is user-defined, but it needs to match the referenced template name in the customer object definition.

In the following paragraphs, the syntax and coding for templates and all possible template types are explained.

Please see Admin Tool Templates for a more generic description of how to use templates.

The following figure shows an example of a customer template to define the display name in the ticket list.

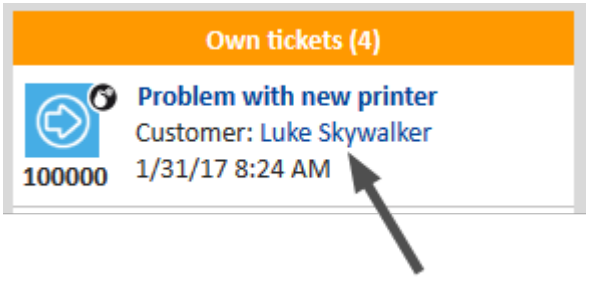


Figure 121: *ConSol CM Web Client - Example of a customer template (ticket list)*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.5.2 Coding Templates

C.5.2.1 General Principle

The templates are written using ***FreeMarker*** notation. For detailed information, please refer to the FreeMarker web site.

Within the templates, you work with three object types:

- customer object, i.e., the technical name of a company or contact object
- technical name of the customer field group
- technical name of the customer fields

⚠ The customer templates must be single-row! They must not contain line breaks!

If you need the display name of a customer as determined by a template in scripts, you can use the method `getDisplayName()` from the class `UnitService`. The following example shows how to print the customer name as determined by the default template in German to the log files. It can be used for example in a customer action, where the `unit` object is available.

```
log.info unitService.getDisplayName(unit, UnitTemplateType.DEFAULT, new Locale
 ("de"));
```

C.5.2.2 Examples of Templates

The following example shows a template for a *Reseller* company which includes company name and number.

```
${ResellerCompany.getFieldValue("ResellerCompanyData","company_name")!} -
 ${ResellerCompany.getFieldValue("ResellerCompanyData","company_number")!}
```

How to Handle Missing Values

You can use if statements in combination with the `??` operator or the `?has_content` operator to check if an object exists or a field has a value for the given customer or resource.

```
<#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 example 6: *Example of a customer template with if statement (has to be written in one line!)*

Alternatively, you can specify a default value using the `!` operator to avoid exceptions due to missing values. The default value can be empty.

```
${ResellerCompany.getFieldValue("ResellerCompanyData","company_name")!} -
 ${ResellerCompany.getFieldValue("ResellerCompanyData","company_number")!}
```

Code example 7: *Example of a customer template with empty default value (has to be written in one line!)*

How to Set the Format of Numbers

Freemarker allows to set a number format pattern.

```
<#setting number_format="0.#"/>${customerModelCompany.getFieldValue("groupName",
 "numberValueField")!}
```

Code example 8: *Setting a number format, so only one decimal place is displayed*

How to Use Company Data in a Contact Template

In a two-level customer data model, you can also use data of the company in the template of the contact. The following example shows how to display the company name in parenthesis after the name of the contact.

```
${ResellerCustomer.getFieldValue("ResellerCustomerData","customer_name")!},
 ${ResellerCustomer.getFieldValue("ResellerCustomerData","forename")!}
 (${ResellerCompany.getFieldValue("ResellerCompanyData","company_name")!})
```

C.5.2.3 Localizing Enum Values in Templates

It is possible to display localized enum values in templates by using the `localize()` method. This method retrieves the localized enum value as defined in the Admin Tool to display it in the Web Client. Otherwise, the technical enum value would be displayed.

The following example shows a template for a resource. The resource field *SLA_country* is an enum containing a list of countries. The `localize()` method is used to display the country name in the respective browser locale.

```
${resource.getFieldValue("SLA_Fields_basic","SLA_Name")!} (${localize
 (resource.getFieldValue("SLA_Fields_basic","SLA_country"))!})
```

C.5.2.4 Abbreviating Values in Templates

It is possible to abbreviate the values retrieved by a template using the `abbreviate()` method. In this way, long values are shortened, so all fields are displayed even if the first field contains a long value.

The `abbreviate()` method needs two parameters:

- The value which should be abbreviated.
- The number of characters to be displayed. Please note that this is the number of all characters which are displayed, i.e., including the three dots. Thus, for example, to display five letters and three dots, use “8” as a parameter. The minimum value is “4”.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

The following example shows the template for a company and how the abbreviated company name is displayed in the ticket list.

```
${abbreviate(ResellerCompany.getFieldValue("ResellerCompanyData","company_
 name"),8)!} - ${ResellerCompany.getFieldValue("ResellerCompanyData","company_
 number")!}
```



Figure 122: *ConSol CM Web Client - Abbreviated value in ticket list*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.5.3 Template Types

The following template types can be defined for the different locations where customers are displayed in the Web Client:

- Standard (Default)
- REST
- Drag-and-drop
- Email
- Quick Search
- Customer Search Result
- Ticket Search Result
- Ticket Page
- Ticket List
- Ticket Relation
- Workspace and Favorites
- History
- Suggestion
- CM/Phone Customer Details
- CM/Phone Customer List

C.5.3.1 Standard (Default)

This template must always be defined. In a two-level customer data model, this has to be done for the company and for the contact level. If it is not, there will be an error in the log file and in the Web Client (-- *unknown* -- is displayed as name of the customers). The template is used for all locations in the Web Client for which no other templates have been defined and as REST template for CM/Track in case there is no dedicated REST template. The templates which are described in the subsequent sections override the default template for the specific locations.

C.5.3.2 REST

This template defines the display name of a customer when accessed using the ***REST API***. This includes the display of the customer name in ***CM/Track***. When retrieving a customer data set using the ***REST API***, the display name is included in the `mark` tag of the output.

C.5.3.3 Drag-and-drop

This templates defines the display name of a customer when the data set is dragged, e.g., from the *Customers* section to the *Favorites* section.

C.5.3.4 Email

When selecting recipients for an email in a ticket, an autocomplete search is performed and the names of the matching customers are displayed in a drop-down list next to their email addresses. This template defines the name of the customers in this list and the sent email. Therefore, you need to make sure that the template returns an RFC-compliant name for the customer. Otherwise, problems with email processing might occur.

The following example shows an RFC-compliant email template:

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

```
${ResellerCustomer.getFieldValue("ResellerCustomerData","forename")!}
 ${ResellerCustomer.getFieldValue("ResellerCustomerData","customer_name")!}
```

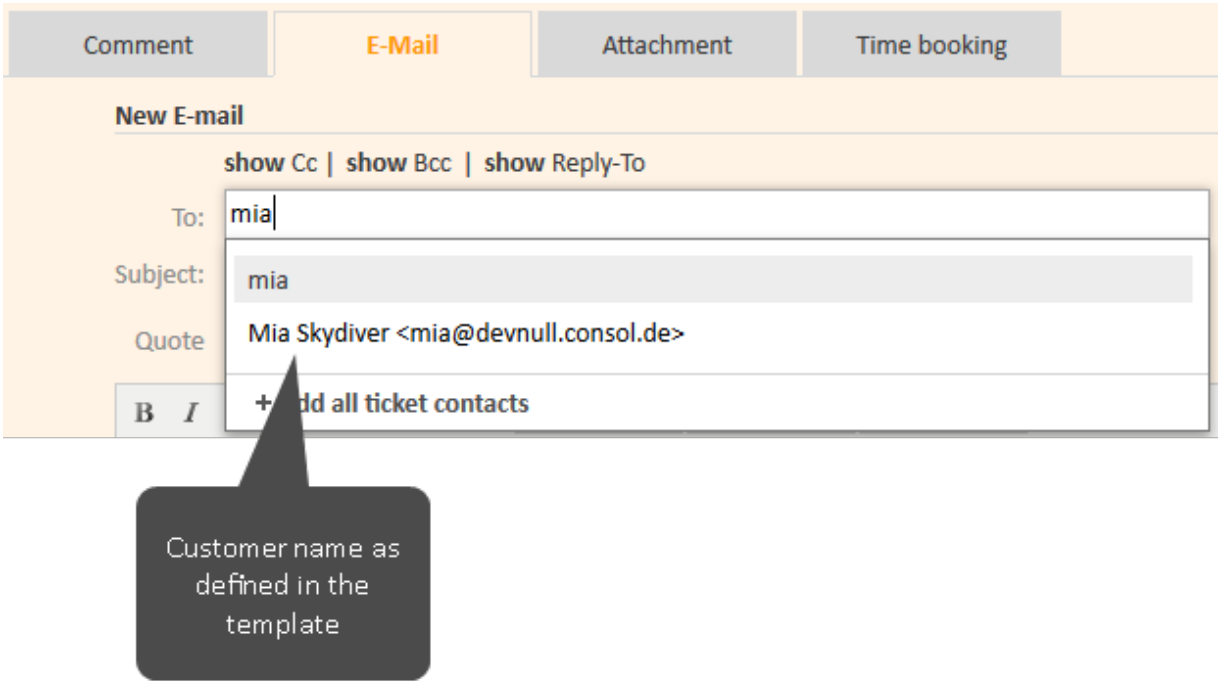


Figure 123: *ConSol CM Web Client - Email template*

> Please contact the ConSol CM support or your ConSol CM consultant for assistance regarding RFC-compliant email addresses.

C.5.3.5 Quick Search

This template defines the display name of the customers in the results of the Quick Search. The template is restricted to a single line of output.

C.5.3.6 Customer Search Result

This template defines the display name of the customers in search results in autocomplete fields. It is used, for example, when selecting a company for a ticket.

C.5.3.7 Ticket Search Result

On the results page of the Detailed Search, the tickets found by the search are displayed as a list. One column of this list contains the main customer of the ticket. This template defines the display name of the customers in this column.

C.5.3.8 Ticket Page

This template defines the display name of a customer on the customer page and in the *Customers* section of tickets. The following figure shows the locations affected by the ticket page template.

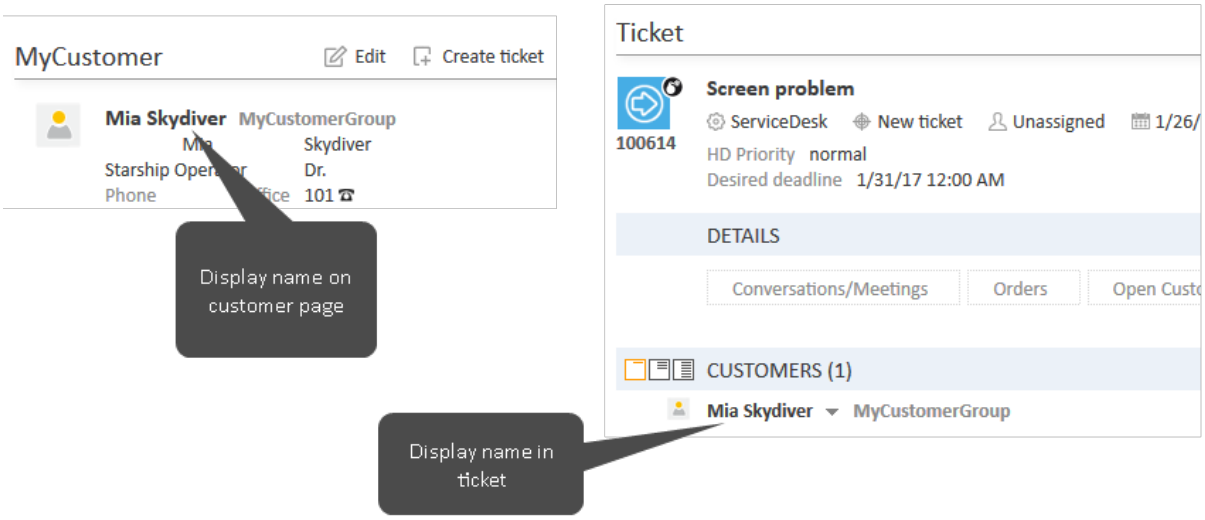


Figure 124: *ConSol CM Web Client - Locations for ticket page template*

> In versions prior to 6.11, the first line of customer fields, i.e., fields with position (0,x), was used as display name on the customer page. Since CM version 6.11, the ticket page template is used, so you can determine the display name regardless of the fields' position.

C.5.3.9 Ticket List

This template defines the display name of the customers in the ticket list.

> The customer name is only displayed in the ticket list if the page customization attribute *accordionTicketList.mainCustomerDescriptionVisible* is set to "true".

C.5.3.10 Ticket Relation

This template defines the display name of a customer in related tickets in the *Relations* section of a ticket. Please keep in mind that the customers of related tickets are only displayed in the visibility level "extended".

C.5.3.11 Workspace and Favorites

This template defines the display name of a customer in the *Favorites* and the *Workspace* sections.

C.5.3.12 History

This template defines the display name of a customer in the ticket history. It is used for example in the history entries which are created when the customer of a ticket changes.

C.5.3.13 Suggestion

This template defines the display name of the customers in suggestions which are displayed in a small pop-up window where a contact can be selected for a ticket.

C.5.3.14 CM/Phone Customer Details

See section CM/Phone: CTI with ConSol CM.

C.5.3.15 CM/Phone Customer List

See section CM/Phone: CTI with ConSol CM.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.6 Managing Customer Groups

This chapter discusses the following:

C.6.1 Basic Principles of Customer Data Models and Customer Groups 214
C.6.2 Managing Customer Groups Using the Admin Tool 214
C.6.3 Assigning Access Rights for Customer Groups 218

C.6.1 Basic Principles of Customer Data Models and Customer Groups

In a ConSol CM system multiple customer groups can be used.

> **Principles:**
>
> There can be any number of customer groups and any number of customer data models.
>
> Each customer group has exactly one customer data model.
>
> Each customer data model can be assigned to any number of customer groups.

In the following example, the system contains four customer groups, each with its specific customer data model.

C.6.2 Managing Customer Groups Using the Admin Tool

In the Admin Tool, customer groups are managed using the navigation item *Customer Groups* in the navigation group *Customers*.

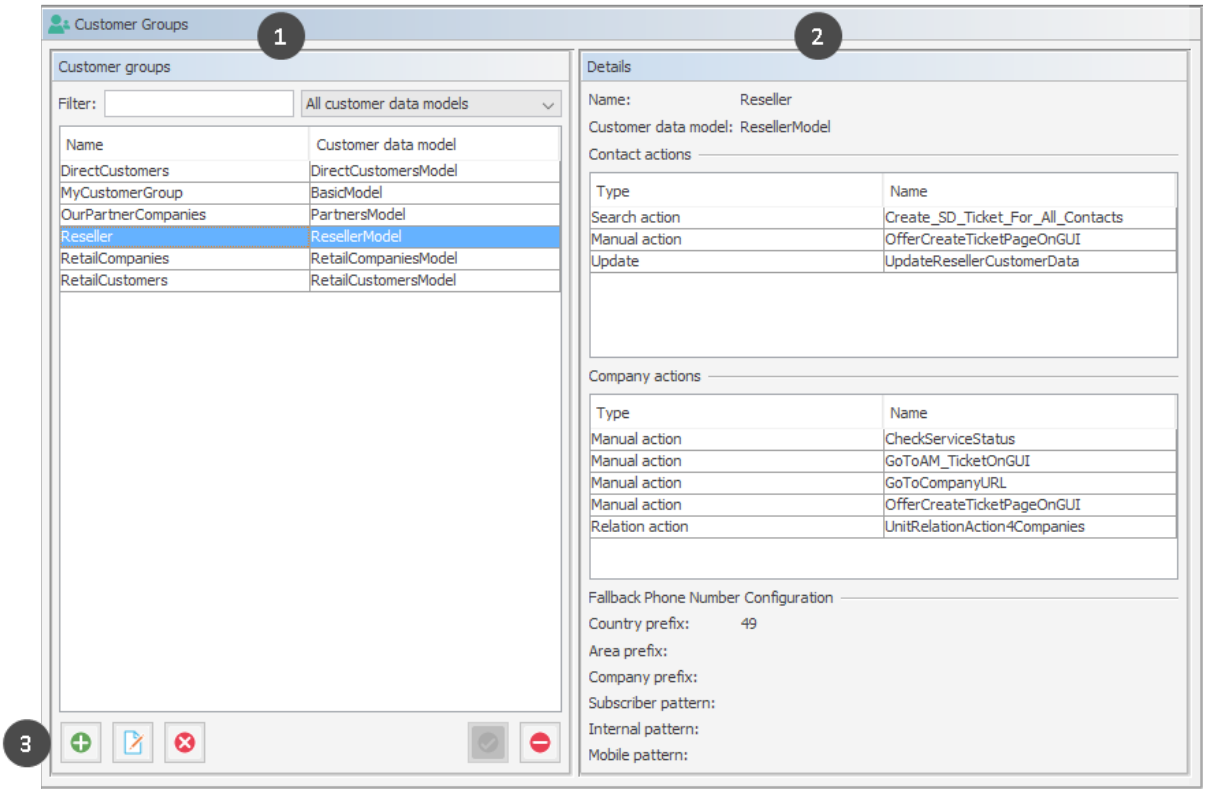


Figure 125: *ConSol CM Admin Tool - Customers, Customer Groups: Managing customer groups*

- List of all customer groups (1). Can be sorted by clicking the column header, and can be filtered.
- Details of the selected customer group (2)
- Buttons to add, edit, delete, activate or deactivate a customer group (3)

C.6.2.1 Customer Groups List

On the left side, all customer groups are listed:

- **Name**
  The technical name of the customer group.
- **Customer data model**
  The name of the customer data model which has been assigned to the customer group.

You can apply two sorts of filters:

- **Name filter**
  Enter a text or some characters in the field *Filter*. Only the customer groups where the name contains the text/characters will be displayed in the list.
- **Customer data model filter**
  Select a customer data model from the drop-down list. Only customer groups with the selected data model will be displayed in the list.

C.6.2.2 Customer Group Details

On the right side, the details of the selected customer group will be displayed. An explanation of all parameters is given in the following section.

C.6.2.3 Creating a New Customer Group

You create a new customer group by clicking the *Add* button below the group list. A pop-up window is opened where you have to enter the customer group parameters.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

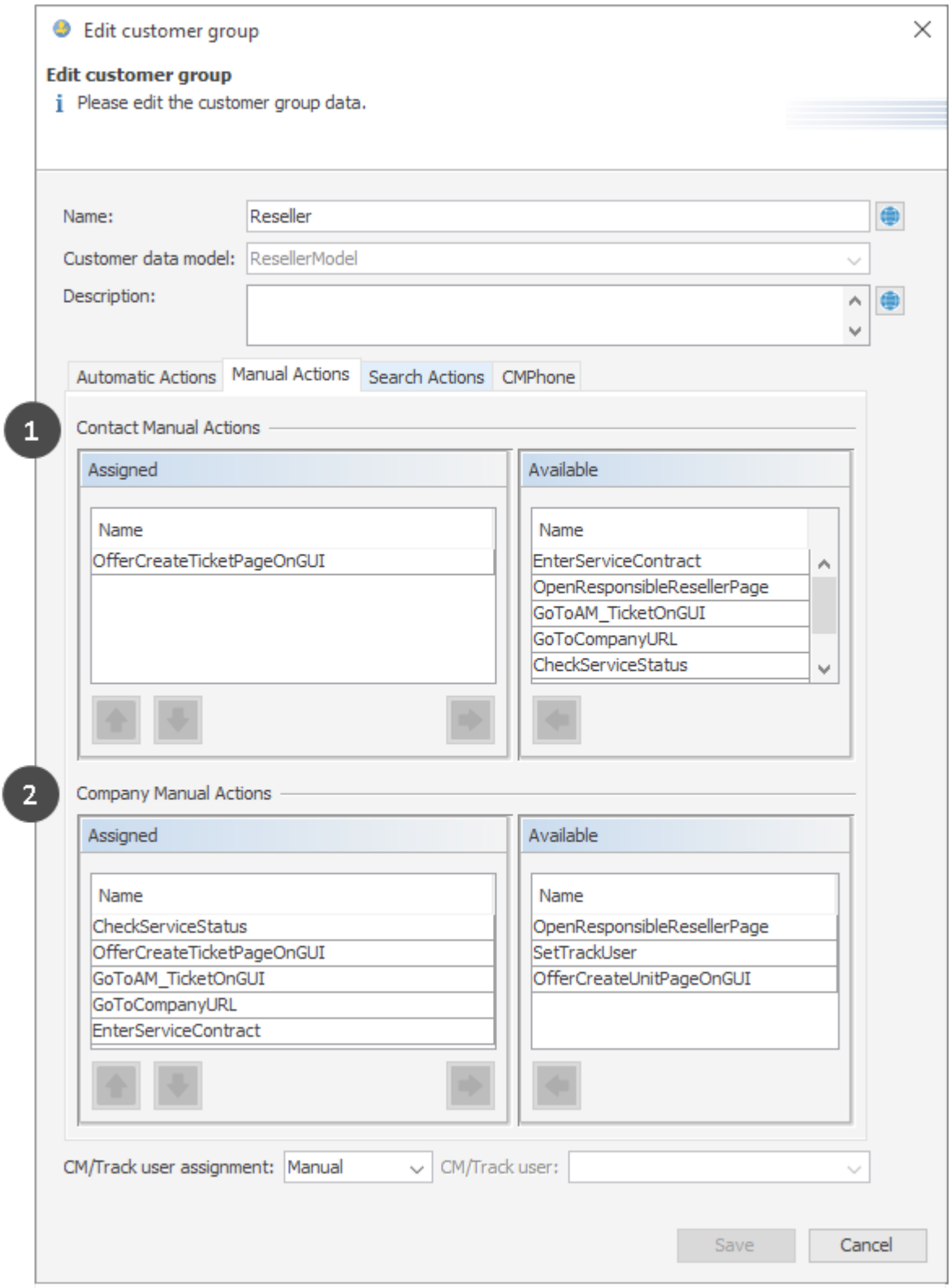


Figure 126: *ConSol CM Admin Tool - Customers, Customer Groups: Parameters for a customer group*

- **Name**
  The unique technical name of the customer group. Can be localized using the *Localize* button. For details, please refer to section Localization of Objects in General, Type 1.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Customer data model**
  Select the customer data model from the drop-down list. All customer data models which have been defined (see section Setting Up the Customer Data Model) are available.
- **Automatic Actions**
  Here, automatic customer actions can be assigned to a customer group. All customer actions which have already been defined will be in the lists. You can assign those actions to contacts and/or companies of the customer group. Automatic actions will be triggered when a customer is created, edited or deleted. The entire configuration is explained in detail in section Action Framework - Customer Actions.
- **Manual Actions**
  Manual actions can also be assigned to contacts and/or to companies. All customer actions which have already been defined will appear in the lists. The actions will be listed as activities on the contacts or company page and have to be triggered manually by an engineer, very similar to workflow activities for tickets. You can use the arrow buttons below the lists of assigned actions to define the order of the activities in the Web Client. Details about manual customer actions are provided in section Action Framework - Customer Actions.
- **Search Actions**
  Search actions are offered as activities for result lists of Detailed Searches and have to be assigned to contacts and/or to companies of a customer group. For example, a company search action for the customer group *Reseller* will always be offered in search result lists which contain lists of Reseller companies. All search actions for customers which have already been defined will be offered in the lists. Search actions are part of the Action Framework and are explained in detail in section Action Framework - Search Actions.
- **CMPhone**
  Tab for all CM/Phone parameters. Only available if CM/Phone is active, see section CM/Phone: CTI with ConSol CM.
- **CM/Track user assignment**
  Decide how the CM/Track user profile should be assigned. Possible values are:
  - **Fixed**
    Select a CM/Track user profile (i.e., an engineer which has been defined as CM/Track user profile). This CM/Track user profile is used for all new contacts created in this customer group. The *Track user* field is not displayed in the Web Client.

    > ⚠ As soon as the customer group has contacts, the assignment mode *Fixed* cannot be changed anymore. If you select the *Fixed* mode for a customer group which already has some contacts, the CM/Track user profile of these contacts is **not** changed automatically. They need to be migrated to the new user profile using a task script.

  - **Manual**
    Default. The assignment of a CM/Track user profile is done manually by an engineer in the Web Client as described in section Granting Access to CM/Track for Customers. The user profile can also be set using the REST API.
  - **None/internal**
    The assignment of a CM/Track user profile can only be done via script, e.g., in a contact action. The *Track user* field is not displayed in the Web Client and the assignment cannot be done via REST API either. You can use the following methods to set a CM/Track user

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

in a contact action script:

```
Engineer trackUser = engineerService.getByName("mytrackuser");
unitEngineerRelationServiceImpl.updateEngineer(unit, trackUser);
```

Actions can be assigned to the contacts (1) and/or to the companies (2) of the customer group.

C.6.2.4 Editing a Customer Group

If you want to edit a customer group, select it in the list and click the *Edit* button or just double-click the name of the customer group. Modify the customer group parameters and click *Save* to store your modifications.

C.6.2.5 Deleting a Customer Group

Select the customer group you want to delete in the list and click the *Delete* button. If you confirm the following dialog with *Yes*, the customer group will be deleted and will no longer be available in the system. A customer group can only be deleted if it is not assigned to a queue and if there are no tickets for customers of the group. In a system which has been in operation for a while, it will usually not be possible to delete a customer group.

C.6.2.6 Disabling and (Re-)Enabling a Customer Group

To disable a customer group, select the customer group in the list and click the *Deactivate* button. The entry in the list is now shown in italics. Just click the *Activate* button at the bottom of the page to enable the customer group again. If a customer group is disabled, it is not possible to create new tickets for companies or contacts of the group. Tickets of the group are still visible.

C.6.3 Assigning Access Rights for Customer Groups

In order to let engineers work with customer data of a customer group, e.g., to create new reseller data sets or to modify them, you have to grant access permissions for the user groups to one or more roles.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

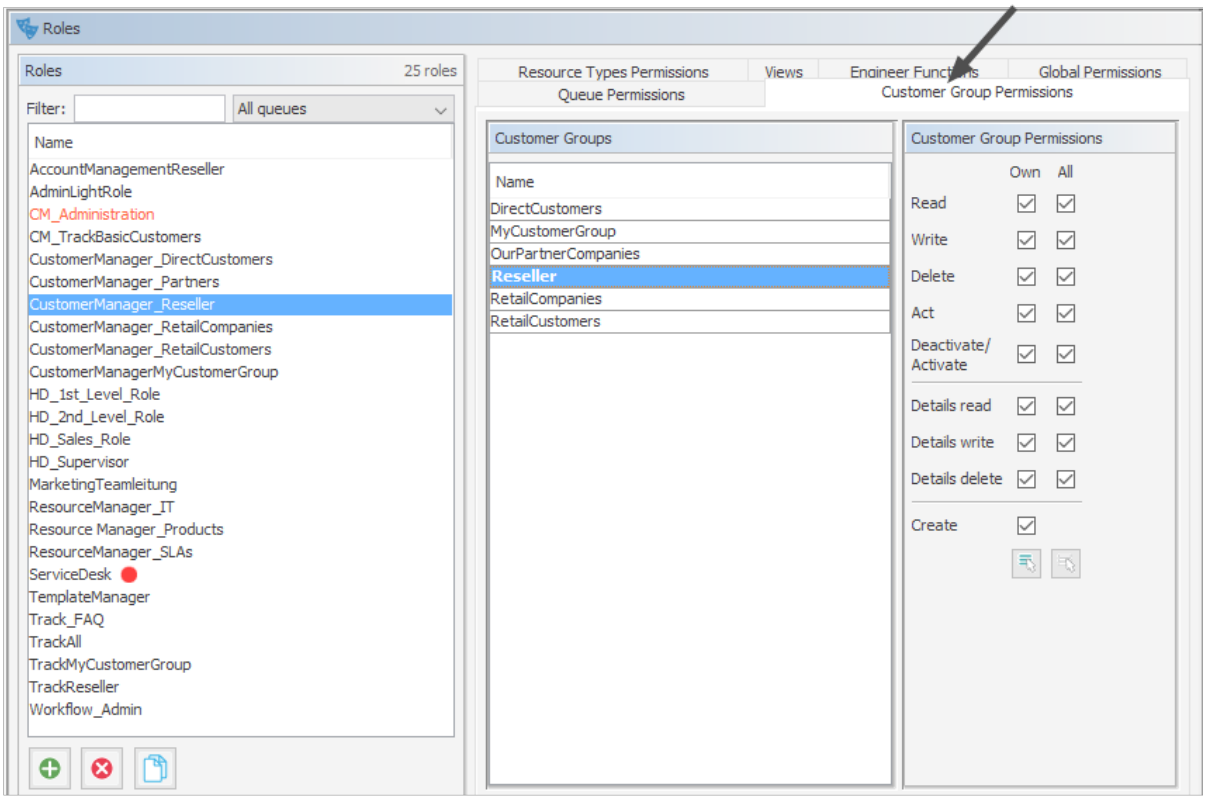


Figure 127: *ConSol CM Admin Tool - Access and Roles, Roles: Assigning permissions on customer groups to a role*

The access rights which can be granted for customer data also comprise the *Comments and Attachments* section of the customer page, i.e., of the company page and/or the contact page.

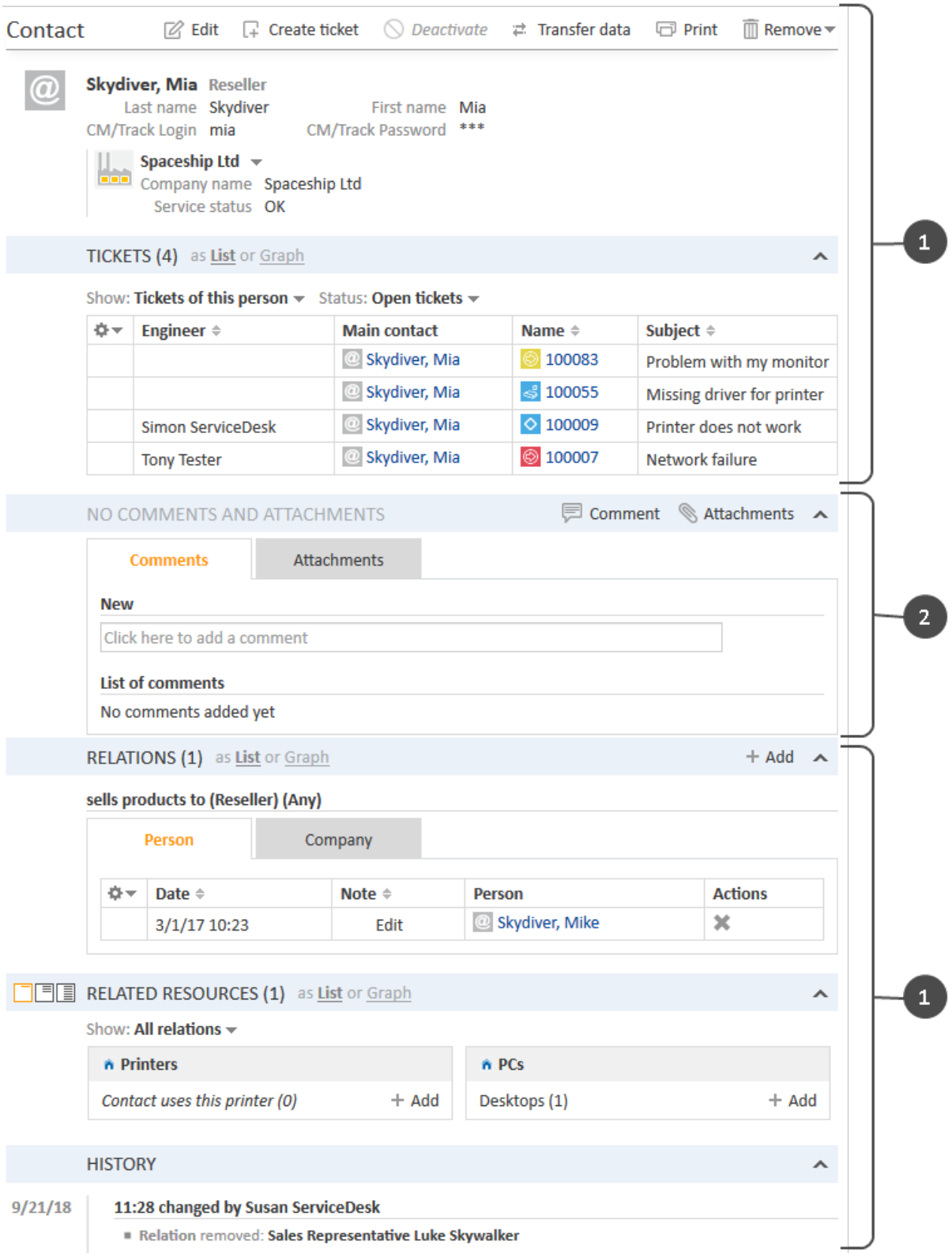


Figure 128: *ConSol CM Web Client - Contact page: Comments and Attachments section*

The following access permissions can be granted:

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Customer type**
  Refers to the tickets of the customer.
  - **Own**
    All (main or additional) customers of tickets which are currently assigned to the engineer or where the engineer is set as an additional engineer.
  - **All**
    All customers.
- **General sections (1)**
  - **Read**
    Read the customer data.
  - **Write**
    Write/modify the customer data.
  - **Delete**
    Delete a customer data set.
  - **Act**
    Execute actions for this customer (see section Action Framework - Customer Actions for details about customer actions).
  - **Deactivate/activate**
    Deactivate and (re-)activate the customer. It is not possible to create tickets for a deactivated customer.
- **Comments and Attachments section (2)**
  - **Details read**
    Read customer data in the *Comments and Attachments* section.
  - **Details write**
    Write/modify customer data in the *Comments and Attachments* section.
  - **Details delete**
    Delete customer data in the *Comments and Attachments* section.
- **General**
  - **Create**
    Create a customer data set. In a two-level customer data model this refers to contact as well as to company data sets.



Please keep in mind that an engineer must have at least read permissions for a customer group to open and/or create tickets for customers in this group!

C.7 Customer Roles

This chapter discusses the following:

C.7.1 Introduction 222

C.7.2 Defining Customer Roles Using the Admin Tool 223

C.7.1 Introduction

Customer roles help you distinguish different additional customers who are linked to a ticket. For example, it will help a great deal if the engineer finds an incident ticket and instantly knows which of the customers is the manager, who is the technical contact person and who he might have to contact for questions concerning billing.

You can define any number of customer roles. Technically, it is a simple list, and the engineer, working with the Web Client, can (but does not have to) assign a customer role to any additional customer of the ticket. An additional customer can only have zero or one roles, not more than one.

Besides helping engineers in their everyday work, customer roles can also be helpful for designing processes and writing scripts, e.g., to write an email to all managers of all open Service Desk tickets of a company.

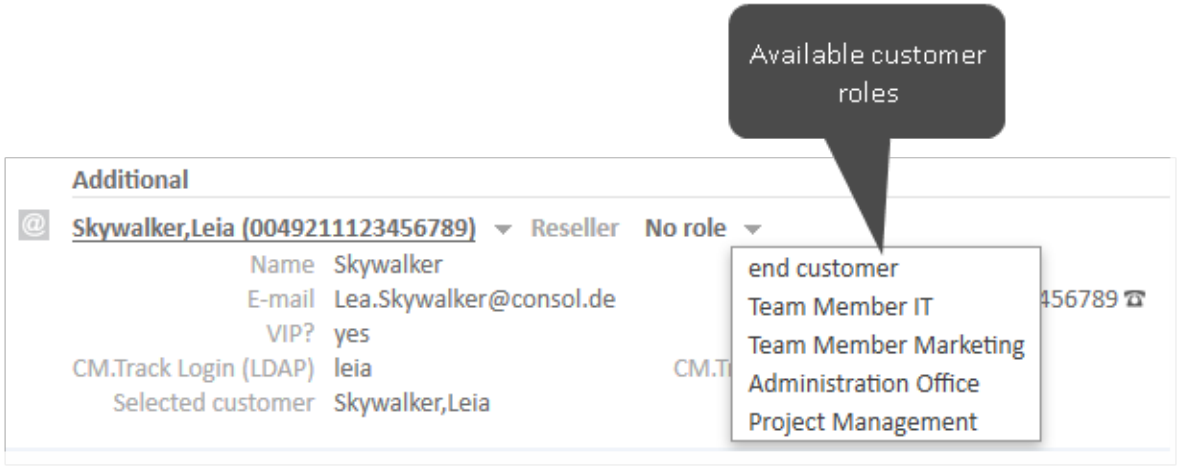


Figure 129: *Customer roles for an additional customer*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.7.2 Defining Customer Roles Using the Admin Tool

Open the navigation item *Roles* in the navigation group *Customers* to add (create), edit, or delete customer roles.

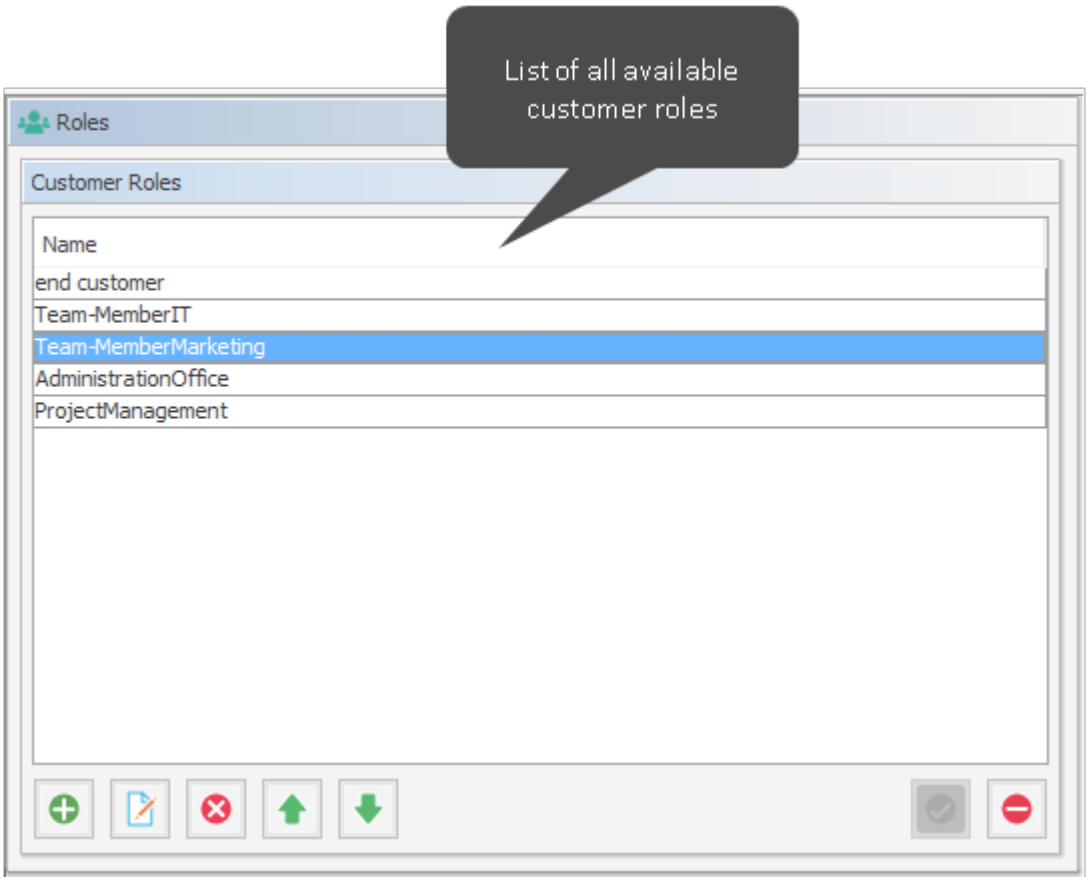


Figure 130: *ConSol CM Admin Tool - Customers, Roles: Customer Roles*

In the Web Client these roles can be assigned to additional customers of a ticket to show the function of these customers, e.g., project manager or end customer.

There are two implications of the assignment of a customer role to an additional customer:

1. It provides information in the Web Client (e.g., you would not want to send a log file to a *manager* but to the *Team Member IT*).
2. The customer role can be used in workflow programming to control the process flow (e.g., send an email to all *Team Members IT* but not to contacts with other roles).

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

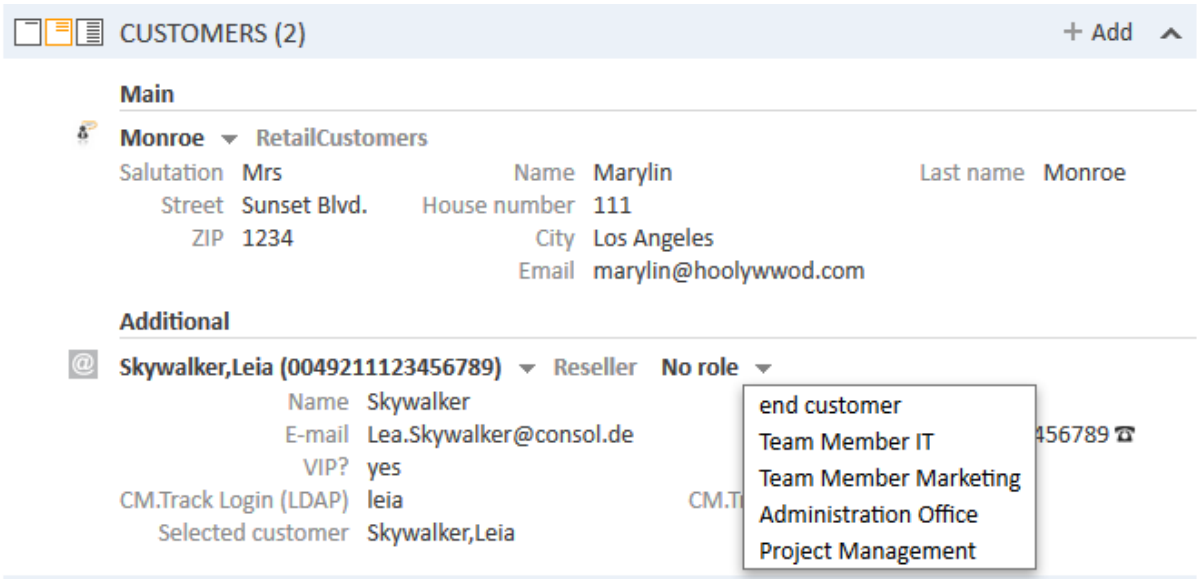


Figure 131: *ConSol CM Web Client - Setting a customer role for an additional customer*

C.7.2.1 Create or Edit a Customer Role

A customer role is defined by its name. By clicking the *Add* button a pop-up window appears where you can enter the name. Using the *Localize* button next to the name field you can localize the name (see below). For details about localization, please refer to section Localization of Objects in General, Type 1. The checkbox *Enabled* is already selected to set the customer role active in the system (see also Disable or Enable a Customer Role). You will see the same window when you click the *Edit* button in order to edit a customer role.

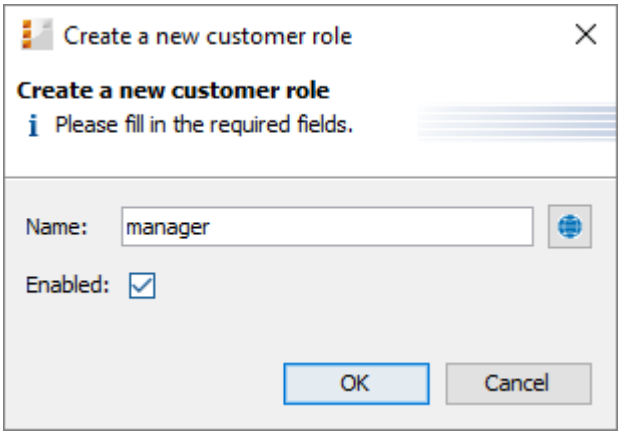


Figure 132: *ConSol CM Admin Tool - Customers, Roles: Create or edit a customer role*

C.7.2.2 Delete a Customer Role

A customer role can only be deleted if it is not assigned to any customers, otherwise you get a warning and can only disable this customer role (see below).

In order to delete a customer role, select it in the list and click the *Delete* button. After choosing *Yes* in the confirmation dialog the customer role will be removed from the list and the system.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.7.2.3 Disable or Enable a Customer Role

If a customer role is still assigned to a customer but is not needed anymore you can disable it. To do this select the customer role and click the *Deactivate* button. The entry in the list is shown in italics afterwards. The customer role cannot be assigned anymore. Just click the *Activate* button at the bottom of the page if you want to enable the role again.

You can also enable or disable a customer role in the window used for editing customer roles by selecting or de-selecting the *Enabled* checkbox. When you create a customer role, this check box is automatically selected.

C.7.2.4 Localize a Customer Role

Click the *Localize* button in the create or edit window to enter the localized name(s) of a customer role. For details about localization, please refer to section Localization of Objects in General, Type 1.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.8 Customer Relations

This chapter discusses the following:

C.8.1 Introduction 226
C.8.2 Management of Customer Relations Using the Admin Tool 228
C.8.3 Creating Customer Relations Using the Web Client 231
C.8.4 Scripting Using Relations 232
C.8.5 Customer Relations to Resources 233

C.8.1 Introduction

Customer relations represent relations between customers, i.e., companies and contacts. They can be one of two types:

- **directional** (different levels in a hierarchy)
- **reference** (same level, no hierarchy)

A relation is of one of the following types:

- **company - company**
  e.g., ... *has a cooperation with* ... (company X cooperates with company Y)
  - The companies can belong to the same or to different customer groups.
  - The involved customer groups can have the same or different customer data models.
- **company - contact**
  e.g., ... *is customer of* ... (contact X is customer of company Y)
  - The company and the contact can belong to the same or to different customer groups.
  - The involved customer groups can have the same or different customer data models.
- **contact - contact**
  e.g., ... *is serviced by* ... (contact X from company X is serviced by contact Y from company Y)
  - The companies and contacts can belong to the same or to different customer groups.
  - The involved customer groups can have the same or different customer data models.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

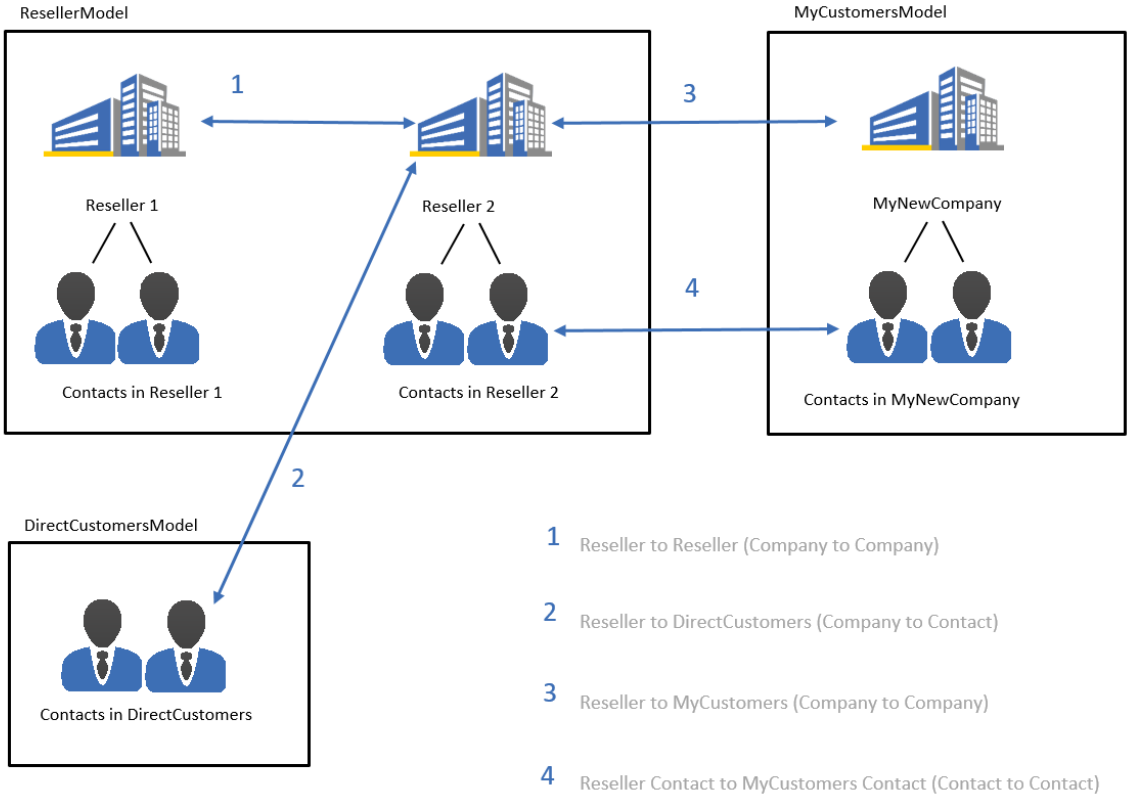


Figure 133: *ConSol CM customer data model - Examples of customer relations*

C.8.2 Management of Customer Relations Using the Admin Tool

To make customer relations available to the engineers, the relations have to be defined in the Admin Tool. Open the navigation item *Relations* in the navigation group *Customers*. All relations are listed, new relations can be added, and old ones can be deleted.

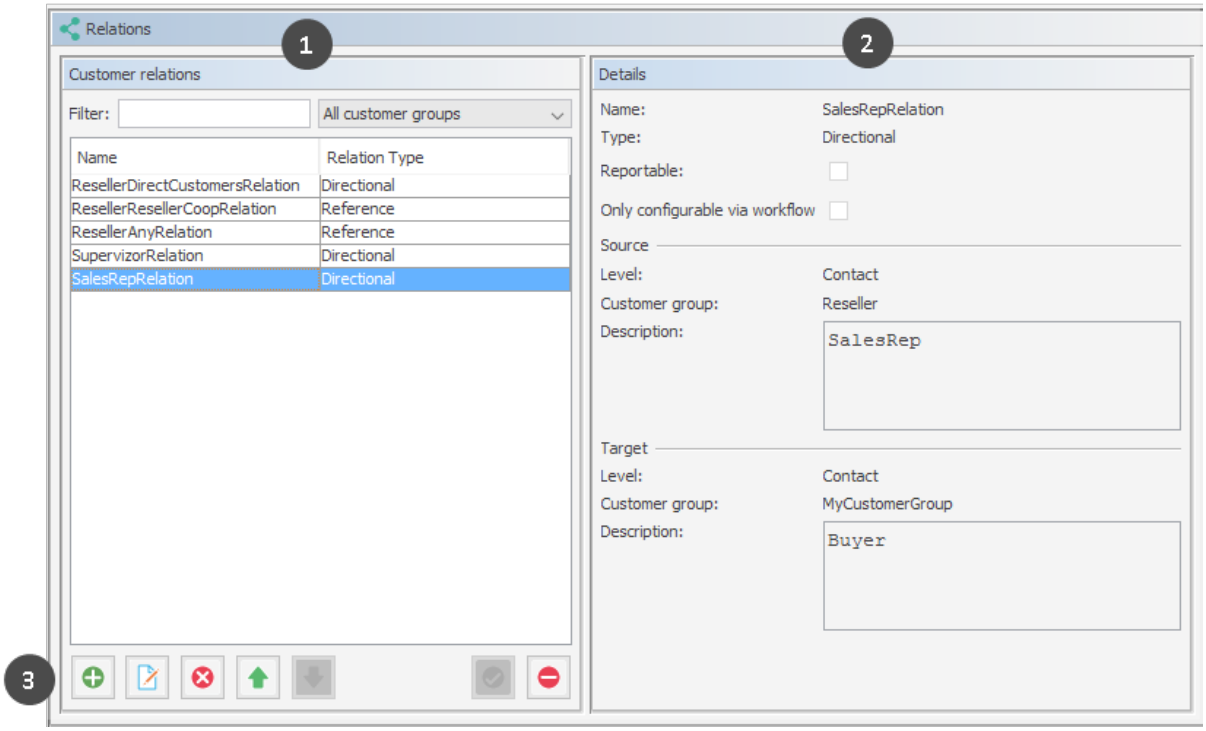


Figure 134: *ConSol CM Admin Tool - Customers, Relations: Managing customer relations*

The following elements are available:

- **List of relations (1)**
- **Filter**
  - Filter for an expression which has to be entered into the field *Filter*. Use the asterisk as a placeholder for any character.
  - Filter for customer groups using the drop-down menu.
- **Relation details (2)**
- **Buttons (3)**
  - **Add button**
    Add a new relation. The pop-up window *Create customer relation* (see next section) is opened.
  - **Edit button**
    Modify the parameters of a relation. The pop-up window *Edit customer relation* (see next section) is opened.
  - **Delete button**
    Delete an existing relation. This is only possible when no relations of this type have been set (using the Web Client).

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Change order (arrows up and down)**
  Place a relation at a specific position in the list. This defines the order of the manual relations as they are displayed in the Web Client.
- **Activate / deactivate relations**
  A deactivated relation is not available in the Web Client, i.e., a relation of this type can no longer be created. Existing relations of this type are not modified and are displayed in the GUI.

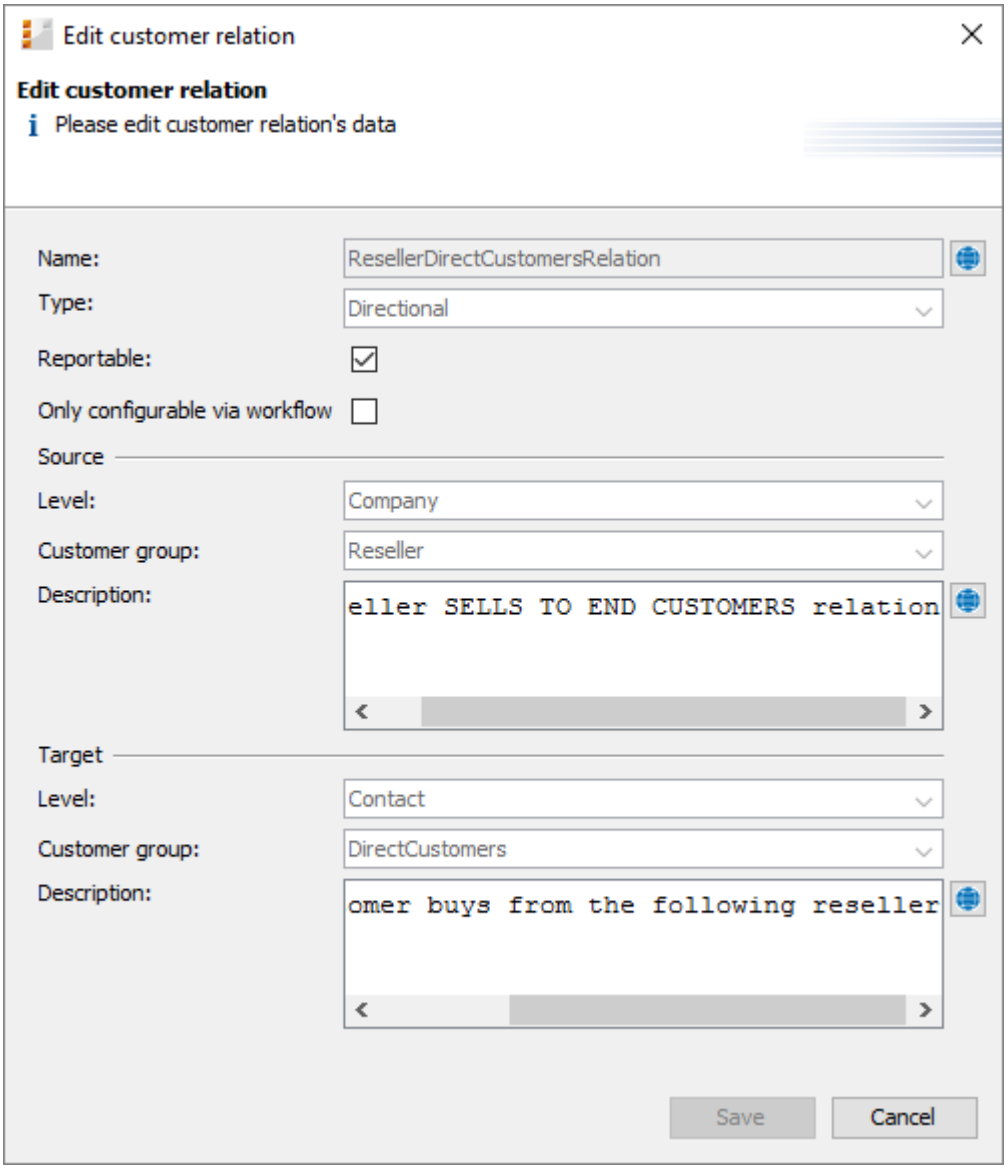


Figure 135: *ConSol CM Admin Tool - Customers, Relations: Details of a customer relation*

To create a new relation, use the *Add* button, to edit a relation use the *Edit* button. In both cases, the detail information pop-up window for a relation is opened where you can edit the following fields:

- **Name**
  Name of the relation. The technical name is used for internal use cases (scripts) whereas the localized name will be displayed in the Web Client (as for most fields in ConSol CM). For details about localization, please refer to section Localization of Objects in General, Type 1.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Type**
  Select one of the two types:
  - **Directional**
    A directional relation has a defined source and a defined target. A customer can be source and target for different relation types at the same time. An example for a directional relation is a reseller (company) to end customer (contact) relation: *sells products to*. A company (reseller) sells products to a contact (end customer). A relation between two contacts of a company: *is boss of*. The other direction, *works for*, can also be used, but designing a consistent structure for the entire system, to avoid misunderstandings, is highly recommended.
  - **Reference**
    A reference is an undirected relation with no hierarchical implications, e.g., *has a cooperation with*.
- **Reportable**
  Defines if the relations of this type should be transferred to the data warehouse.
- **Only configurable via workflow**
  If this checkbox is marked then the relation is not available in the Web Client but can only be created via workflow scripts. Such relations cannot be manipulated manually.
- For a directional relation select:
  - **Source**
    - **Level**
      Level of the relation source, i.e., *company* or *contact* or *any* (choose the latter if the source can be either a company or a contact).
    - **Customer group**
      The customer group of the source customer.
    - **Description**
      Will be displayed in the Web Client as the description of the relation on the source side.
  - **Target**
    - **Level**
      Level of the relation target, i.e., *company* or *contact* or *any* (choose the latter if the target can be either a company or a contact).
    - **Customer group**
      The customer group of the target customer.
    - **Description**
      Will be displayed in the Web Client as the description of the relation on the target side.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.8.3 Creating Customer Relations Using the Web Client

In spite of this being an administrator's manual, we will show you how relations are set using the Web Client, because, as an administrator, you should always know what the consequences of administration modifications are.

As an engineer who has the access permissions to the source and to the target customer group, you can add a relation of one customer to another in the *Relations* section of the source customer. You have to have at least one role with the access permission *Write* for the source customer group and the target customer group to perform this operation. You can set the relations on the customer-specific page, i.e., open the company page or the contact page of the potential source object.

For example, you can create a relation *Sells to end customers* from a company in the *Reseller* customer group to a contact in the *End customers* customer group. Of course, this relation has to be defined in the Admin Tool first. Use the *Add* link in the *Relations* section and then select the relation from the drop-down menu. Enter the target name (e.g., contact name) in the auto-complete text field. You can also add a note. Then click *OK*. The relation can be edited or deleted afterwards using the respective links (*Edit*, *Remove*).

Please refer to the *ConSol CM User Manual* for a detailed explanation of working with customer relations.

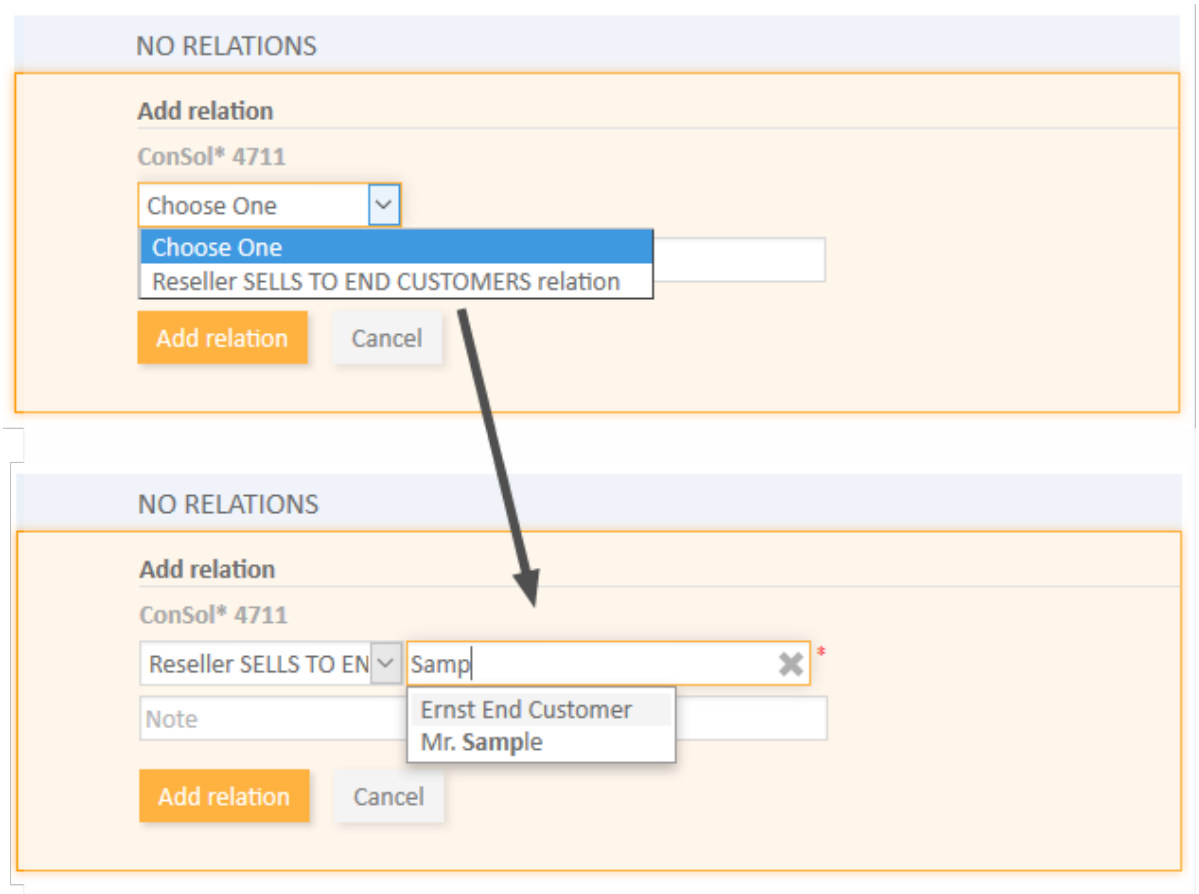


Figure 136: *ConSol CM Web Client - Setting a relation*

C.8.4 Scripting Using Relations

A new class, the `UnitRelationService`, is available. For details please refer to the *ConSol CM API Java Doc*.

> ⚠ In this book we will use the terms *customer* and *customer definition*. In previous versions of ConSol CM, the term *data object* was used to refer to customers. However, the names of the corresponding Java classes are *Unit* and *UnitDefinition*. All other Java classes which deal with customer objects are also still named *Unit*... Please keep that in mind if you work as both a ConSol CM administrator and programmer. Please refer to the *ConSol CM Java API Doc* for details.

```
// 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)
```

Please refer to the *ConSol CM Process Designer Manual* for a detailed explanation on how to write scripts which use customer relations.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.8.5 Customer Relations to Resources

The relations of resources to contacts or to companies are always defined as resource relations, i.e., defined for resources, not for customers. The company or contact objects are assigned as potential target objects when the resource relation is defined. resource relations are explained in section CM/Resource Pool - Resource Relations.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.9 Action Framework - Customer Actions

This chapter discusses the following:

C.9.1 Introduction 234
C.9.2 Managing Customer Actions Using the Admin Tool 236
C.9.3 Using Customer Actions as an Engineer (User) 242
C.9.4 Examples for Customer Action Scripts 242
C.9.5 Scripts for the Action Framework: Programming Customer Actions 245

C.9.1 Introduction

Customer actions (also called *Unit actions*, previously called *Data Object Actions)* are system actions which are either triggered manually or by a certain incident concerning a customer, i.e. a customer is deleted or a customer is modified. Customer actions are components of the ConSol CM Action Framework. They can be performed for a customer, i.e., a contact or a company. The actions can be performed automatically by the system or manually, triggered by an engineer (via Web Client) who has the required permissions. You might want to apply customer actions for use cases like the following:

- Load additional data into a company's data set.
- Build an automatic report about company-specific KPIs.
- Transfer ConSol CM data to another system (e.g., an ERP system).
- Create/update a Google Maps (see Trademarks) link within the address data.

You can use the following types of customer actions:

- **Automatic** actions which are performed by the system after one of the following customer operations:
  - CREATE
  - UPDATE
  - DELETE
  - RELATION
  - SEARCH
- **Manual** actions which are performed by an engineer using *Activities* links in the customer page (company or contact page) of the Web Client (similar to *Workflow activities* for tickets). Which actions are available for manual execution depends on the customer which is currently being displayed, i.e., when the company page is open, company actions will be offered, when the contact page is open, contact actions will be offered.

⚠ Please keep in mind that only engineers who have at least one role with the following access permissions for the respective customer group are allowed to use the customer actions, i.e., only then will the Activities be displayed in the Web Client:

- Act

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

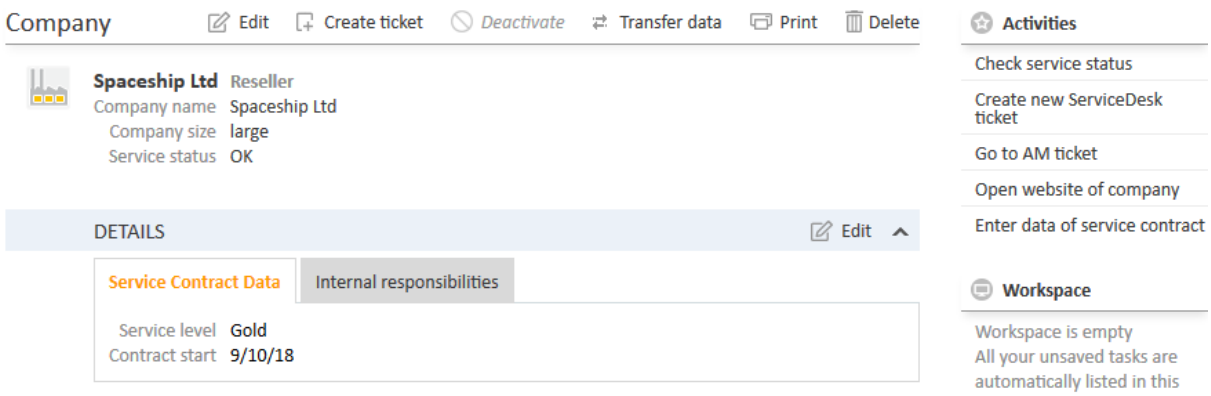


Figure 137: *ConSol CM Web Client - Example for manual customer activities*

Customer actions are based on Groovy scripts which are stored in the *Scripts and Templates* section of the Admin Tool.

The execution of customer actions can be controlled using condition scripts, i.e., you can implement a customer condition script which is run before the action script of the customer action. The action script is only run if this condition script returns “true”.

So there are two types of scripts you have to deal with when you use the ConSol CM Action Framework:

- **Customer action scripts**
  Defines the action which should be performed.
- **Customer condition scripts**
  Defines one or more conditions for the display of the action in the Web Client. Has to return “true” or “false”. If “false” is returned the action is not displayed on the GUI and therefore cannot be performed.

When you want to implement a customer action you have to proceed in three steps:

1. Create a script for the customer action (either an action script only or an action script and a condition script).
2. Create the customer action(s) which use(s) the script(s).
3. Assign the customer action(s) to the customer group(s) where they should be available. You can assign the actions to contacts and/or companies of a customer group.

In the following sections, all three steps are explained in detail.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.9.2 Managing Customer Actions Using the Admin Tool



In this book we will use the terms *customer* and *customer definition*. In previous versions of ConSol CM, the term *data object* was used to refer to customers. However, the names of the corresponding Java classes are *Unit* and *UnitDefinition*. All other Java classes which deal with customer objects are also still named *Unit*... Please keep that in mind if you work as both a ConSol CM administrator and programmer. Please refer to the *ConSol CM Java API Doc* for details.

C.9.2.1 Step 1: Write the Customer Action Script

Create a new Admin Tool script of type *Customer action*. If required, create another script of type *Customer condition*.

For a detailed explanation of Admin Tool scripts, please refer to section Admin Tool Scripts. For an introduction to Admin Tool scripts used for customer actions, please read section Scripts for the Action Framework: Programming Customer Actions in this chapter.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

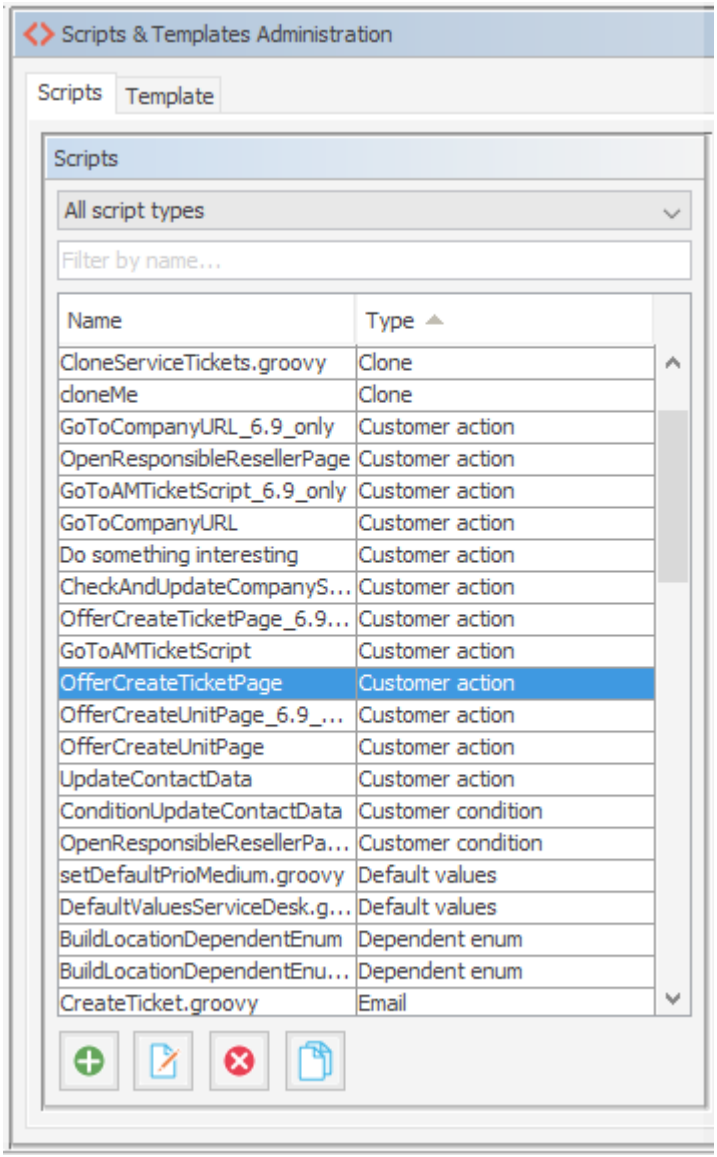


Figure 138: *ConSol CM Admin Tool - System, Scripts and Templates: Scripts for customer actions*

```
// display Create Ticket page for a new Service Desk ticket

def newticket = new Ticket()
def qu = queueService.getByName("ServiceDesk")
newticket.setQueue(qu)
client.goToCreateTicket(newticket).withCustomer(unit)
```

Code example 9: *Script which opens the create ticket page with the customer as the main customer, CM version 6.11 and up*

C.9.2.2 Step 2: Create Customer Action(s) Which Use the Script

Open the navigation item *Actions* in the navigation group *Customers* in the Admin Tool and add (= create) a new action object using the *Add* button . The same pop-up window appears for both adding and editing a customer action.

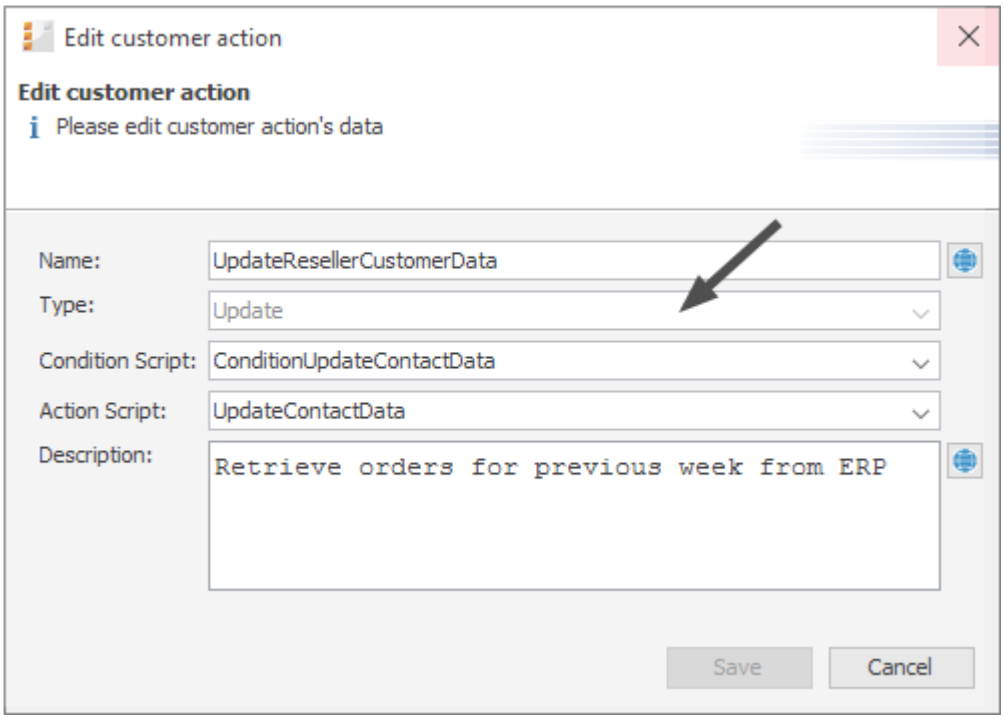


Figure 139: *ConSol CM Admin Tool - Customers, Actions: Editing a customer action*

In the pop-up window, the parameters for the new action have to be defined:

- **Name**
  The unique technical name of the action. Can be localized using the *Localize* button. Localization is required for manual actions because the localized name is displayed under *Activities* in the Web Client. For details about localization, please refer to section Localization of Objects in General, Type 1.
- **Type**
  The action type which defines when it should be executed. Select one of the following types:
  - **Create**
    This script will be executed automatically when the contact/company is created.
  - **Update**
    This script will be executed automatically when the contact/company is updated, i.e., when the data has been modified (either manually or automatically) and is saved again. Starting with CM version 6.10.5.4, the changes which have been performed during the *Update* action can be monitored using the changes object. This is explained in detail in section Working with the Changes Object in Customer Update Actions.
  - **Delete**
    This script will be executed automatically when the contact/company is deleted.
  - **Relation**
    This script will be executed automatically when a relation to or from a customer of this customer group is
    - created
    - deleted

    (The script will not be executed when the comment of a relation is changed.)

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

  - **Manual**
    This script will be available on the contact/company page as a manual activity, provided the engineer has the required access permissions (*act* permission for the respective customer group).

- **Condition Script**
  If a condition script should be run before the action script, the name of the condition script has to be entered here. The action script is only run if the condition script has returned “true”. If there is no condition, just leave this field empty.
- **Action Script**
  The name of the action script which should be run. This has to be the exact name under which the script is stored in the *Scripts and Templates* section of the Admin Tool.
- **Description**
  Enter the description which should be displayed as mouse-over in the Web Client (for manual actions only).

Save the action. You can then assign it to customer groups, as described in the following step.

C.9.2.3 Step 3: Assign Customer Actions to Customer Groups

For the customer action to become effective, you have to assign it to a customer group, at which point it will be available for all customers of this customer group. Depending on the initial definition (contact or company action), the action will be available for contacts or for companies in the customer group. To assign a customer action to a customer group, open the navigation item *Customer Groups* in the navigation group *Customers* of the Admin Tool. Select the customer group you would like to edit and click the *Edit* button to open the pop-up window to assign the customer actions. All customer actions which have been stored under *Actions* (see step 2) will be available here, each for the corresponding action type. For example, an action which has been defined for the type *Update* during definition (see step 2) will only be available as an *Update* action.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

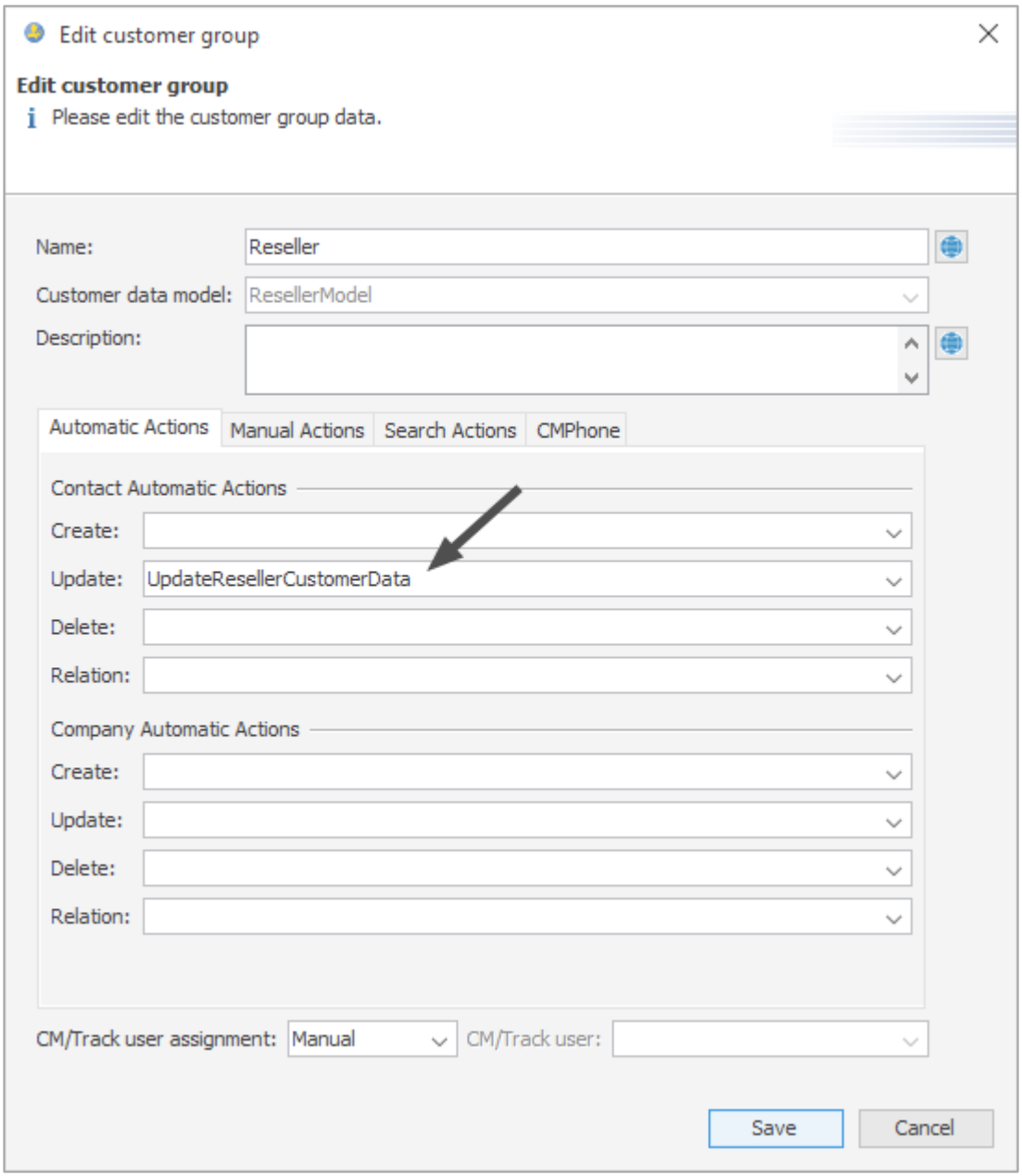


Figure 140: *ConSol CM Admin Tool - Customers, Customer Groups: Assigning customer actions to a customer group*

You can assign the following action types to a customer group:

- **Automatic Actions**
  - **Contact Automatic Actions**
    The script will be executed for the contact.
  - **Company Automatic Actions**
    The script will be executed for the company.

  For each type you can determine the system behavior for the following actions:

- **Create**
  This script will be executed automatically when the contact/company is created.
- **Update**
  This script will be executed automatically when the contact/company is updated, i.e., when the data has been modified (either manually or automatically) and is saved again. Starting with CM version 6.10.5.4, the changes which have been performed during the *Update* action can be monitored using the `changes` object. This is explained in detail in section Working with the Changes Object in Customer Update Actions.
- **Delete**
  This script will be executed automatically when the contact/company is deleted.
- **Relation**
  This script will be executed automatically when a relation to or from a customer of this customer group is
  - created
  - deleted

  (The script will not be executed when the comment of a relation is changed.)

- **Manual Actions**
  This script will be available on the contact/company page as a manual activity, provided the engineer has the required access permissions (*act* permission for the respective customer group).
- **Search Actions**
  These are explained in section Action Framework - Search Actions.

C.9.3 Using Customer Actions as an Engineer (User)

As an engineer (user), only the customer action type *manual* is relevant for you. The *CREATE, UPDATE* and *DELETE* scripts run in the background.

Manual actions are offered in the Web Client, similar to workflow activities for a ticket. Please see *Example 1* in the next section.

C.9.4 Examples for Customer Action Scripts

C.9.4.1 Example 1: Simple Manual Action

A manual action is coded and stored as an Admin Tool script, then a company action is defined using the script, and the action is assigned to a customer group.

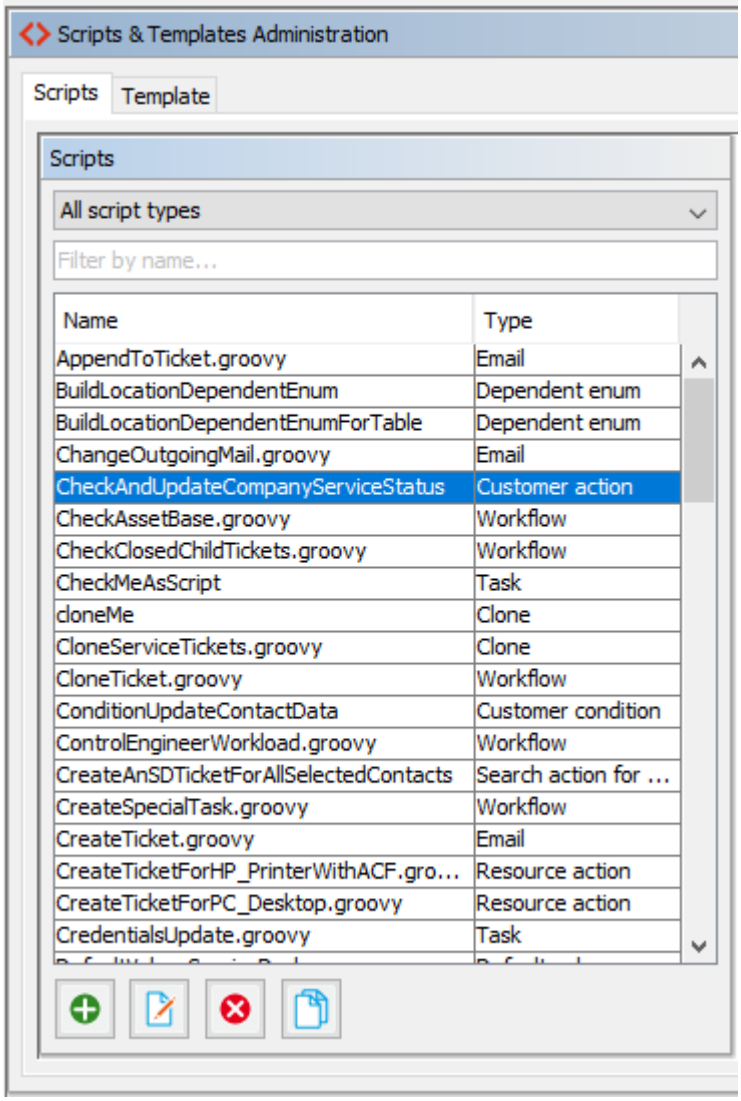


Figure 141: *ConSol CM Admin Tool - System, Scripts and Templates: Customer action script for a company action*

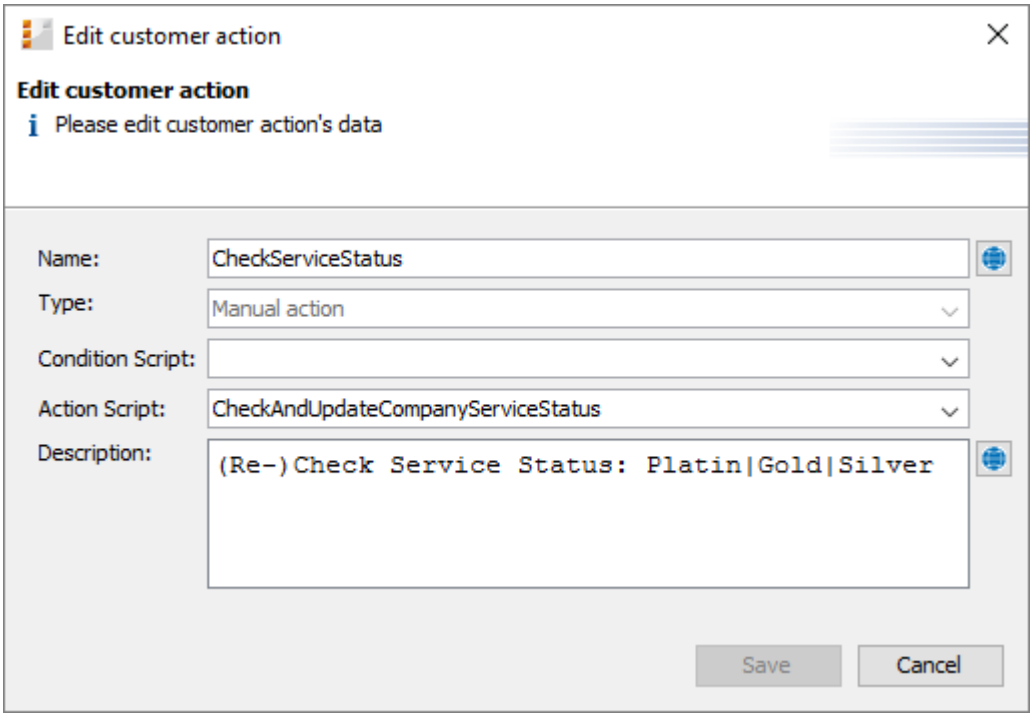


Figure 142: *ConSol CM Admin Tool - Customers, Actions: Defining the company action*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

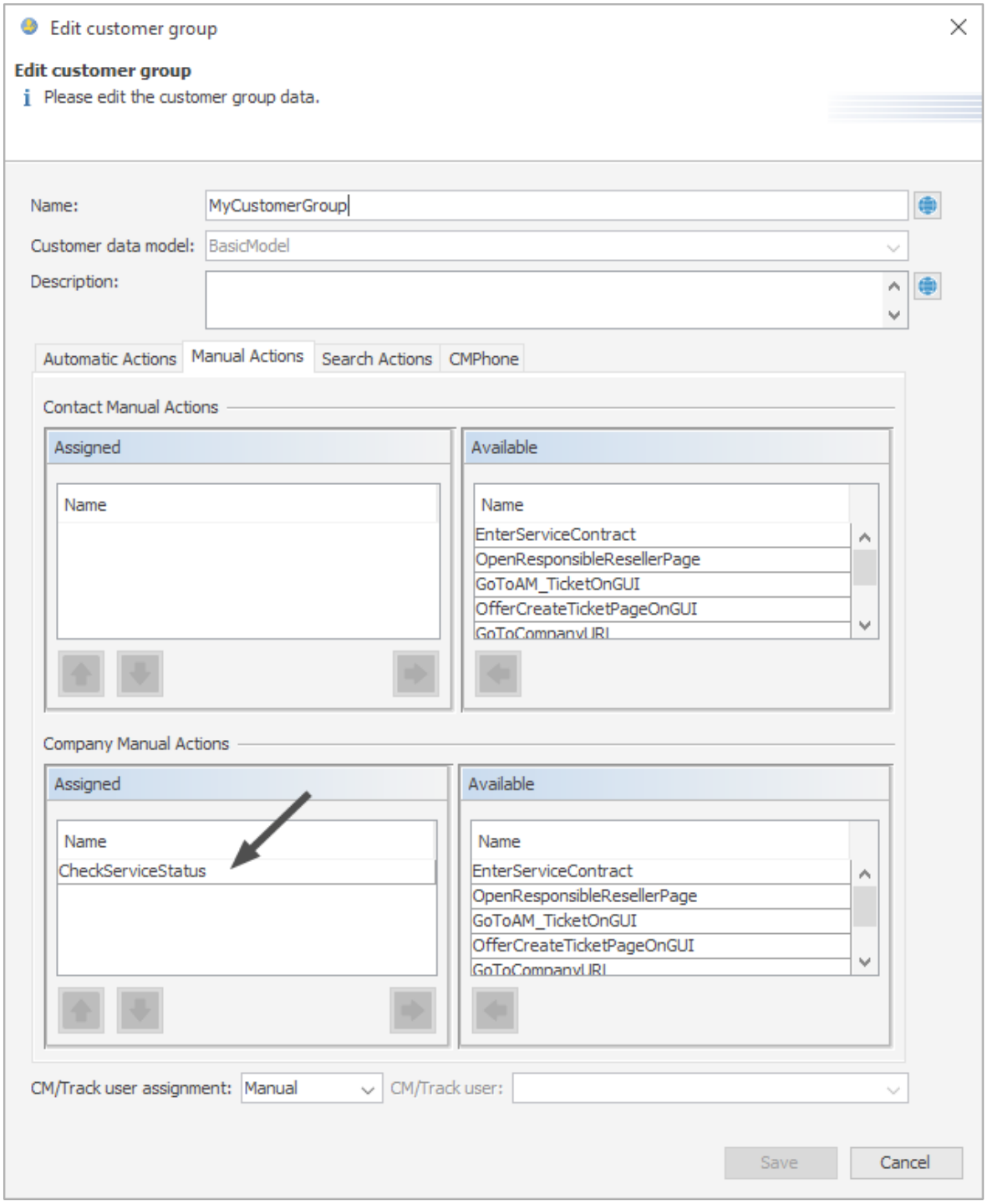


Figure 143: *ConSol CM Admin Tool - Customers, Customer Groups: Assigning a company action to a customer group*

The engineer can use the action manually in the Web Client.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

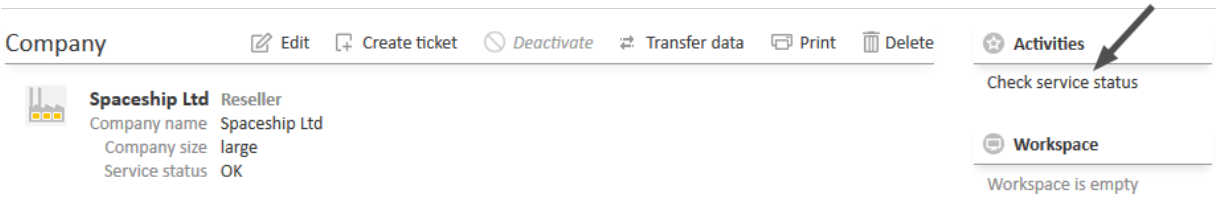


Figure 144: *ConSol CM Web Client - Using a manual company action*

A script similar to the following could be used:

```
// ... 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
if (ser_stat.name == "ok") {
   unit.set("ResellerCompanyData:service_status","ok")
   client.showInfoMessage("info.dataobject.action.success").withRefreshContent();
}
```

Code example 10: *Customer action script, CM version 6.11 and up*

C.9.4.2 Example 2: New Ticket for a Customer

The following script opens the *Create ticket* page for the contact or company the action was performed for. The target queue is *ServiceDesk*, enabling a new Service Desk ticket to be created in no time for the open contact or company. For an introduction to Admin Tool scripts for the Action Framework, please read the following section.

```
// display Create Ticket page for a new Service Desk ticket

def newticket = new Ticket()
def qu = queueService.getByName("ServiceDesk")
newticket.setQueue(qu)
client.goToCreateTicket(newticket).withCustomer(unit)
```

Code example 11: *Script which opens the create ticket page with the customer as the main customer, CM version 6.11 and up*

> Please note that in case a ticket should be created with a company as main customer, the checkbox *Company as customer* must be checked for the customer data model of the respective customer group.

C.9.5 Scripts for the Action Framework: Programming Customer Actions

Customer actions are defined by Admin Tool scripts, i.e., by Groovy scripts which are stored in the *Scripts and Templates* section of the Admin Tool. The predefined object `unit` (i.e., an object of class `Unit`) is available for those scripts. Objects of the class `Unit` can represent a company or a contact,

depending on the context.

There are two types of scripts for the Action Framework:

- Customer action scripts
- Customer condition scripts

C.9.5.1 Customer Action Scripts

The actions in this script are either triggered automatically by the system operations *CREATE*, *UPDATE*, or *DELETE* or by a manual action of the engineer (using *Activities* in the Web Client).

Automatic Customer Action Scripts

```
unit.set("personalData.name", "Skywalker")
unitService.update(unit)
```

Code example 12: *Set a value in customer data and update the unit*

> ⚠ When you use `unitService.update(unit)`, as in the example above, you should use a customer condition script to avoid infinite loops. See the notes in section Customer Condition Scripts.

Manual Customer Action Scripts

For manual customer action scripts you can use the methods of the object `client` (class `OperationResponseBuilder`) to determine the behavior of the Web Client after the action script is executed. The following list shows some examples, you can find the complete list of available methods in Overview of the Methods.

- **goToTicket() / goToResource() / goToUnit()**
  Open a ticket, resource page, or customer page
- **goToCreateResource() / goToCreateTicket() / goToCreateUnit()**
  Open the page to create a resource, ticket, or customer
- **openActivityForm()**
  Open an ACF
- **goToPage()**
  Open a web page (URL)
- **showMessage()**
  Display a message
- **rollback()**
  Rollback the action

Create a Customer

Use the method `client.goToCreateUnit(unit)` to open the page to create a new customer. You need to create a new `unit` object and provide the customer group and customer definition. Optionally, you can pre-fill some data.

```
// used for companies in end customers group to create new contacts easily

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")

client.goToCreateUnit(myunit)
```

Code example 13: *Company script which fills some customer data, CM version 6.11 and up*

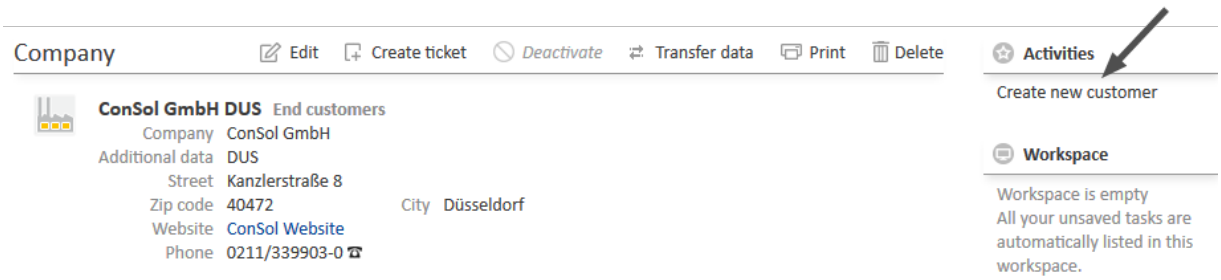


Figure 145: *ConSol CM Web Client - Manual company action*

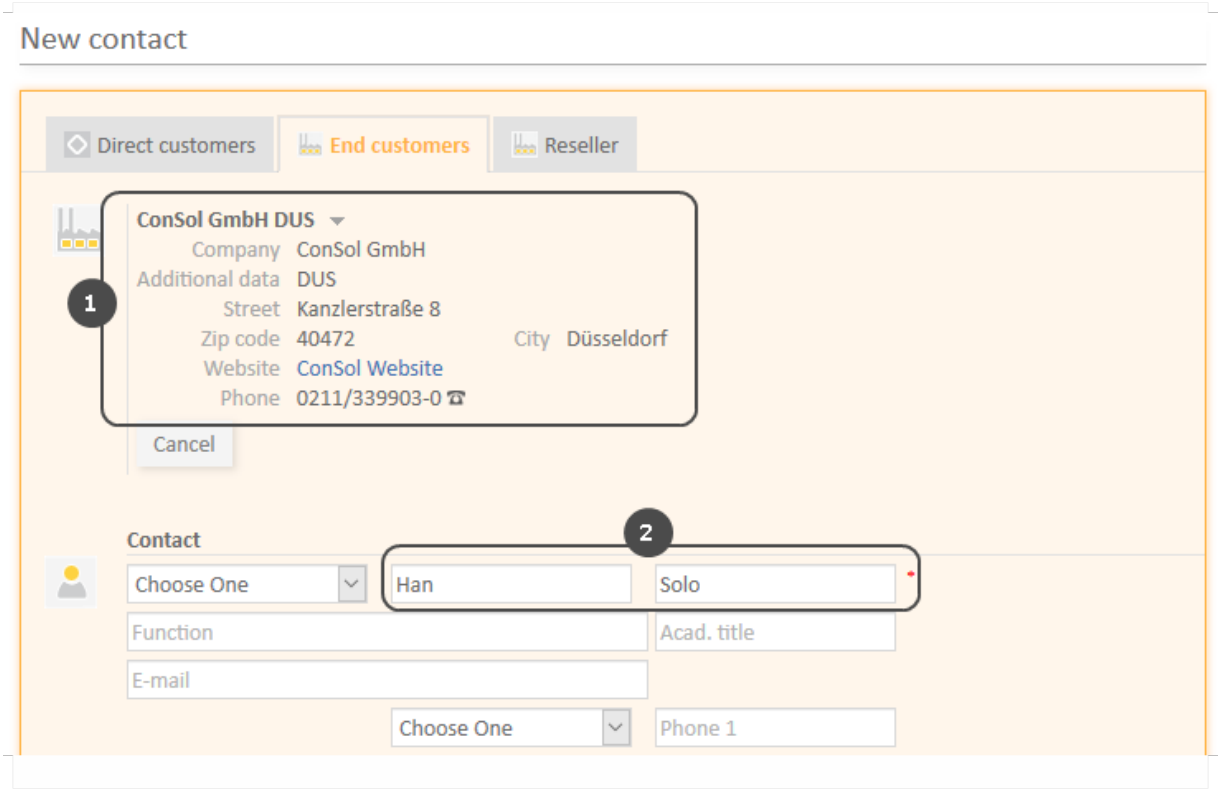


Figure 146: *ConSol CM Web Client - Create customer page opened and pre-filled by company action*

- (1) The company for which the *Create new customer* activity has been performed is preselected.
- (2) Some data fields of the contact data are pre-filled

Of course, the names of the customer fields which are used in the script have to be the ones from the customer data model which has been assigned to the customer group for which a new contact should be created.

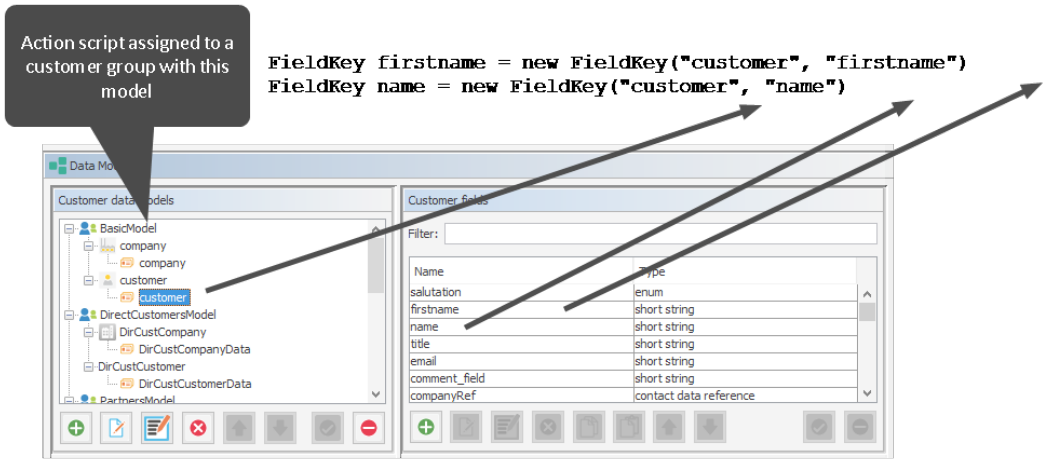


Figure 147: *ConSol CM Admin Tool - Customers, Data Models: Script for creating contact as company action*

Create a Ticket

Use the method `client.goToCreateTicket(ticket)` to open the page to create a new ticket. You need to create a new `ticket` object and provide the queue. Optionally, you can pre-fill some data. If you want to set a customer for the ticket, you can append the method `withCustomer(unit)`. In the example below, `unit` is the customer on whose page the action is executed.

```
// display Create Ticket page for a new Service Desk ticket

def newticket = new Ticket()
def qu = queueService.getByName("ServiceDesk")
newticket.setQueue(qu)
client.goToCreateTicket(newticket).withCustomer(unit)
```

Code example 14: *Script which opens the create ticket page with the customer as the main customer, CM version 6.11 and up*

⚠ Please remember that the customer group for which the script should be applied has to be assigned to the queue where the ticket is to be created (*ServiceDesk* in the example).

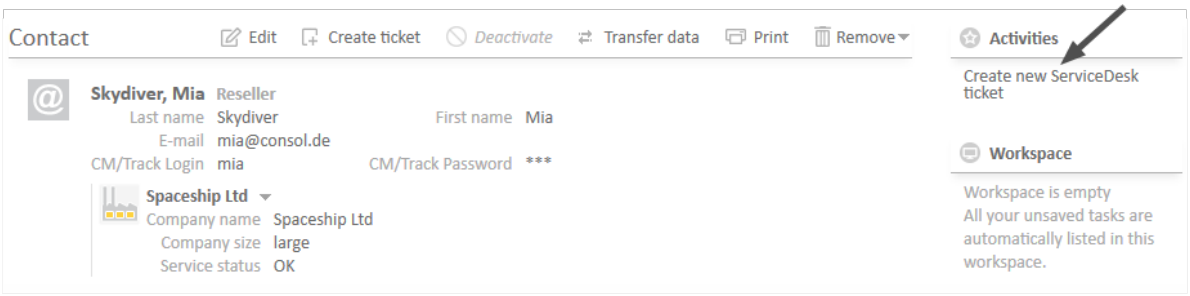


Figure 148: *ConSol CM Web Client - Manual contact action*

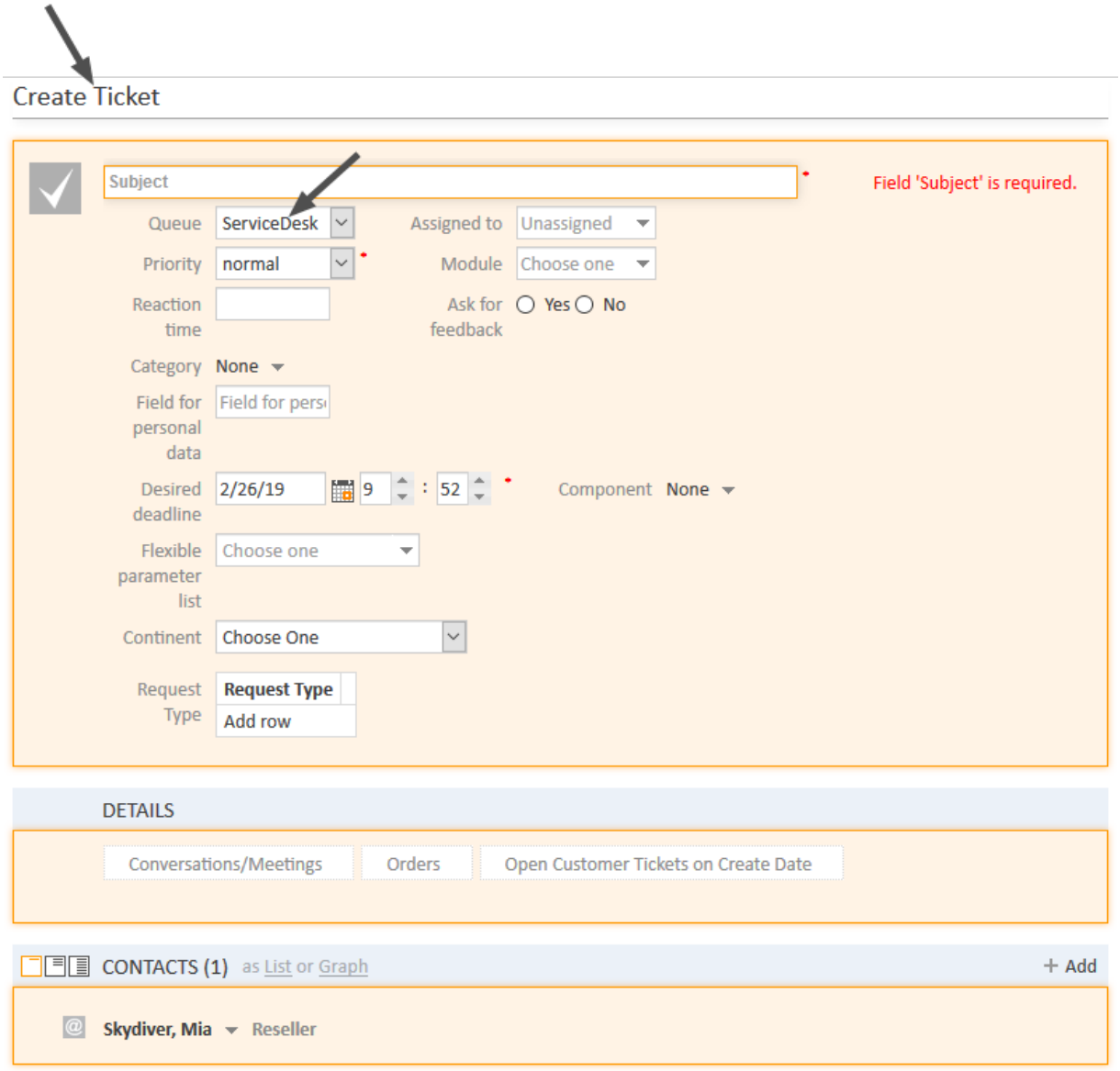


Figure 149: *ConSol CM Web Client - Create ticket page opened and pre-filled by contact action*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Open a Customer Page

Use the method `client.goToUnit(unit)` to open a customer page. In the example below, `unit` is the Reseller company which is responsible for the end customer. A company-contact relation has to be set before. If more than one reseller relation has been defined, the first relation in the list is used.

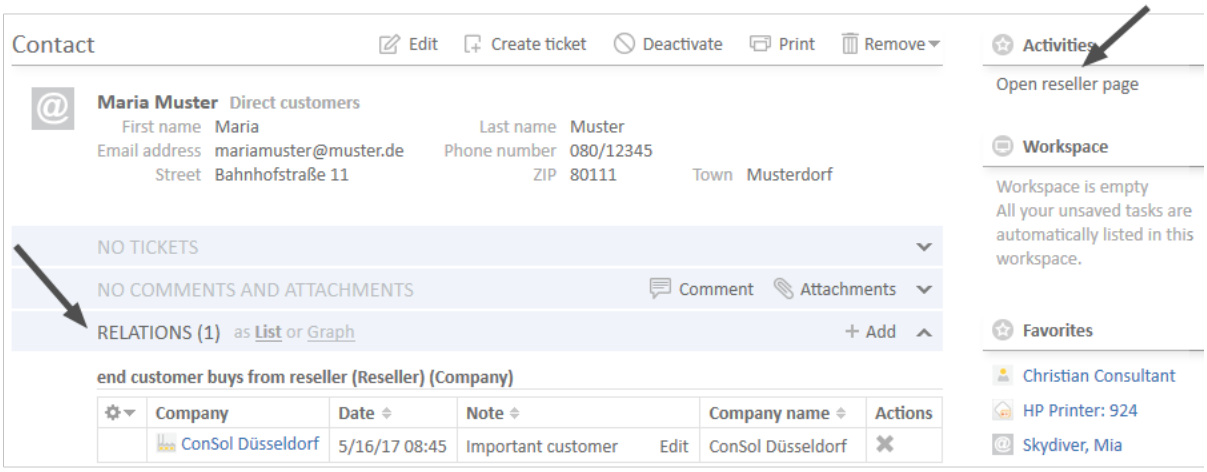


Figure 150: *ConSol CM Web Client - Customer action on contact detail page (activity opens company detail page of responsible reseller, CM version 6.11)*

```
// Open company detail page of responsible reseller company, is only executed if
 this relation exists (checked in a condition script)

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()
   client.goToUnit(source_unit)
}
```

Code example 15: *Script which opens the company page, CM version 6.11 and up*

If the action *OpenResponsibleResellerPage* should only be displayed in the Web Client when a relation to the responsible reseller has been set, you can work with a customer condition script. It has to be created as an Admin Tool script of type *Customer condition* and has to be assigned to the customer action in the *Actions* section.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

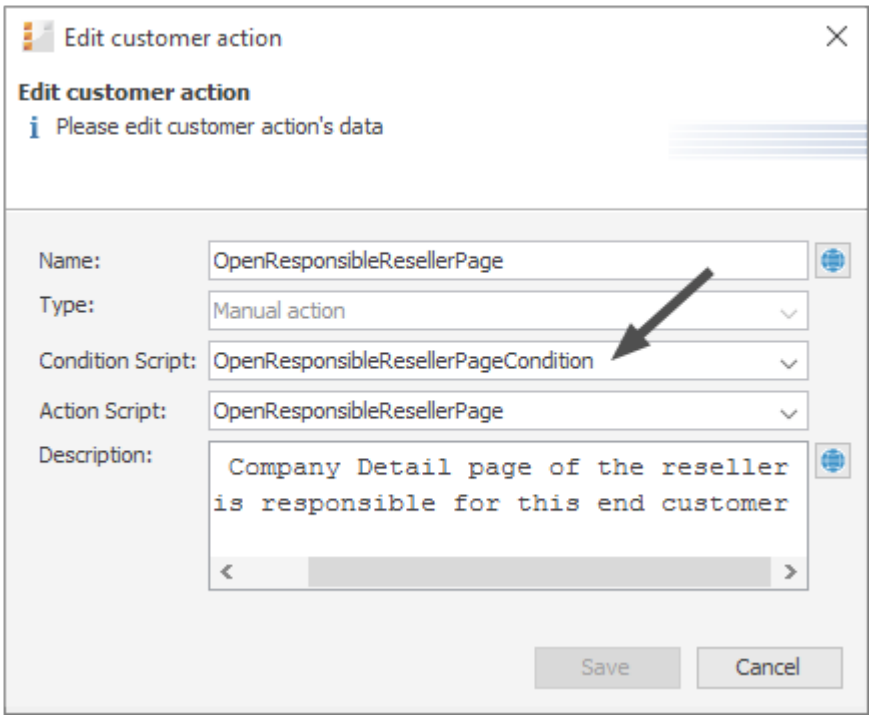


Figure 151: *ConSol CM Admin Tool - Customers, Actions: Assignment of customer condition script*

```
// Checks if reseller relation is set

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 example 16: *Customer condition script which checks if reseller relation is present*

Open a Ticket Page

Use the method `client.goToTicket(ticket)` to open a ticket page. In the example below, `ticket` is the first ticket in the list of account management tickets which have the current company as a main customer.

```
//get tickets in Account Management queue with the current company as main customer

def myqueue = workflowApi.getQueueByName("AccountManagement")
def tickets1 = ticketService.getByQueue(myqueue)
def tickets2 = ticketService.getByMainContact(unit)

def foundTickets = tickets1.intersect(tickets2)

if ( foundTickets ) {
   def AM_tic = foundTickets.first()
   client.goToTicket(AM_tic)
}
```

Code example 17: *Open a ticket page in view mode, CM version 6.11 and up*

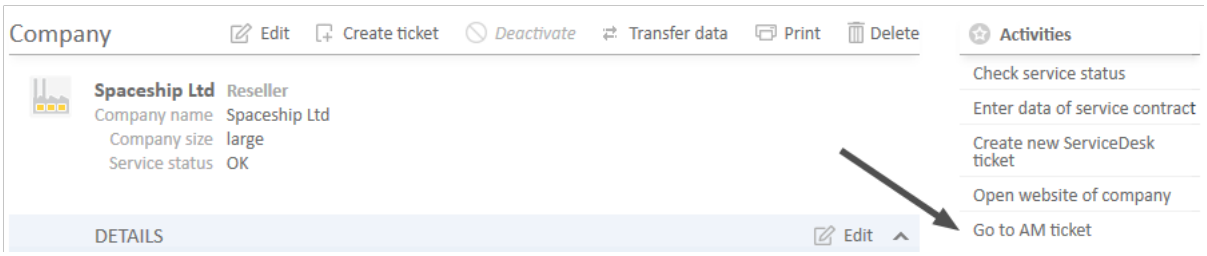


Figure 152: *ConSol CM Web Client - Company action available on company page (1)*

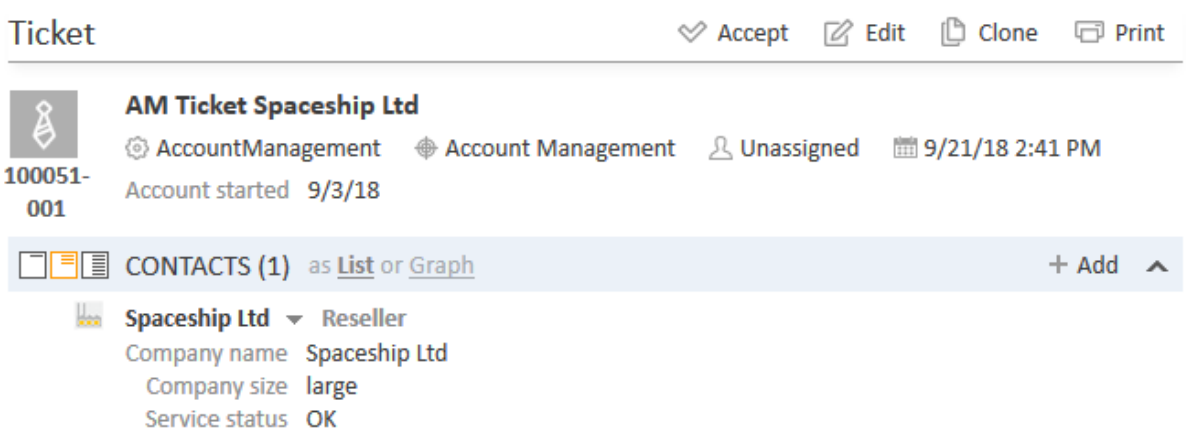


Figure 153: *ConSol CM Web Client - AM page opened after company action Go to AM ticket*

Open a Web Page

Use the method `client.goToPage(string)` to open a URL. In the example below, `string` is the URL saved in a company data field. You can also use this method to open a dynamically built URL.

```
// opens company's web site
def url = unit.get("ResellerCompanyData.url")

if (!url) {
   client.showErrorMessage("error.script.no.url");
} else {
   client.goToPage(url);
}
```

Code example 18: *Script which opens a certain web site (URL), CM version 6.11 and up*

The string *error.script.no.url* is a label which has been defined in the navigation group *Labels*, as described in section Labels.

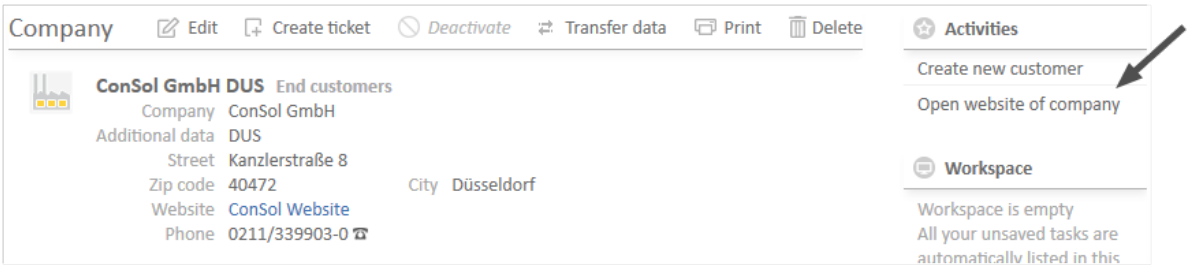


Figure 154: *ConSol CM Web Client - Company action available on company page (2)*

C.9.5.2 Customer Condition Scripts

A customer condition script defines whether an action should be shown in the Web Client or not. It is executed before the customer action script. If it returns “false”, the customer action script will not be executed.

```
if(unit.getFieldValue("customer.personalData") == null) {
   return true
} else {
   return false
}
```

Code example 19: *Customer condition script*

To put a customer condition script into operation, you have to perform the following steps:

1. Write an Admin Tool script of type *Customer condition*. The script has to define the conditions for the return values “true” and “false”, as demonstrated in the example above. At this point you must already know for which customer groups and data models the script will be used, to make sure you reference the correct data fields within the script.
2. Assign the customer condition script to one or more customer action(s). This is done in the *Actions* section, navigation group *Customers*. Once in place, this specific customer condition script will always be executed before the customer action script of the same customer action.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

That's it. You do not have to assign the condition script to a customer group. This happens implicitly when the respective customer action is assigned to contacts or companies of certain customer groups.

C.9.5.3 Important Groovy Objects

Object `client`

In all action scripts, the object `client` (class `OperationResponseBuilder`) is available to determine the behavior of the Web Client after the script is executed. It can be used, for example, to open a ticket page or to display a message.

Programming with the class `OperationResponseBuilder` is explained in detail in section Scripts for the Action Framework.

> ⚠ The customer actions *Create*, *Update*, and *Delete* are executed in the core methods `create`, `update`, and `delete` of the object `unitService`.
>
> Thus, if the update action script updates the customer using the `unitService.update(Unit)` method then a `java.lang.StackOverflowError` error may be thrown because the update action will infinitely recurse. In that case a customer condition script should be used to avoid such infinite loops.

C.9.5.4 Working with the Changes Object in Customer Update Actions

Starting with CM version 6.10.5.4, it is possible to monitor the changes which have been performed during a customer update action. (The same applies to resource *Update* actions, explained in section Working with the Changes Object in Resource Update Actions).

To find out which changes have been performed use the object of class `UnitChanges` in Unit actions.

Please remember, the *Update* script will be executed:

- in an explicit *Update* action
- when comments or attachments are added
- when comments or attachments are removed

There are two methods of the `unitChanges` object which provide information about the changed data:

- `getCustomFieldChangeInfo()`
  provides information about changes of unit data (in customer fields)
- `getContentChangeInfo()`
  provides information about changes in the unit history (comments, attachments)

Since the method return parameters contain rather complex components, we recommend to read the API doc of the `UnitChanges` class. The following code provides an example for a script where a `unitChanges` object is used.

```
// 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 example 20: *Unit Update script where changes are monitored and printed out to server.log*

When for a contact in the customer group *Resellers* changes have been made to the two customer fields *phone* and *vip_person*, the following text is displayed in the `server.log` file.

```
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 example 21: *Log output from the script above*

C.10 Address Autocomplete

This chapter discusses the following:

C.10.1 Introduction 258
C.10.2 Switch on the Address Autocomplete Feature Using the Admin Tool 261
C.10.3 Import Zip/City/Address Data into the ConSol CM Database 263
C.10.4 Define the Address Autocomplete Configuration Using the Admin Tool 264
C.10.5 Edit an Address Autocomplete Configuration 267
C.10.6 Delete an Address Autocomplete Configuration or Address Autocomplete Fields 267

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.10.1 Introduction

In some ConSol CM systems it is required that engineers enter or edit a great number of customer data manually. In such cases it can be helpful to have system support to

- provide suggestions for the input into some data fields (like e.g., zip code or address) to ensure that the entered address really exists
- avoid entering duplicates

To support such cases, ConSol CM offers the feature *Address Autocomplete*. This feature can be switched on/off using a system property. In standard ConSol CM installations, it is switched off. When it has been switched on, the engineer can receive suggestions for the input in one or more of the following customer fields:

- one or more fields which contain the zip code
- one or more fields which contain the city
- one or more fields which contain the address (street and number)

This applies to the following operations in the CM Web Client:

- Create a customer on the *Create customer* page
- Create a customer in the *Customers* section of a ticket
- Edit a customer on the customer page
- Edit a customer in the *Customers* section of a ticket
- Entering customer data on the *Detailed Search* page
- Entering customer data in an ACF (Activity Control Form)

Usually, a data set which is publicly available, e.g., a data collection on CD ROM, serves as source for the import of the address data. In this way, the engineers can implicitly use a vast collection of zip code/city/address mappings.



Please note that in order to use this feature, your company will have to purchase an address collection, i.e., the feature Address Autocomplete is based on the import of external data which is not part of a ConSol CM distribution!

In the following example, a German address collection has been imported into a ConSol CM demo system. Please see the three example figures from the respective ConSol CM Web Client to learn how ConSol CM can help engineers improve their data input quality.

Example 1: The engineer starts typing into the *ZIP* field. Only the existing ZIP codes will be offered. The number of suggestions which are displayed is part of an Address Autocomplete Configuration, please see section Define the Address Autocomplete Configuration Using the Admin Tool for details.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH



Figure 155: *ConSol CM Web Client - Suggestions of the Address Autocomplete feature, example 1*

Example 2: The *ZIP* field has already been filled. For the *City* field, only the correct possible values are suggested.

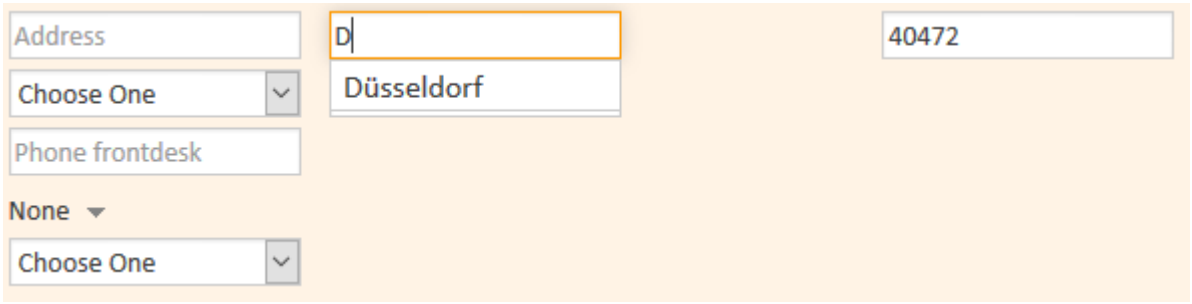


Figure 156: *ConSol CM Web Client - Suggestions of the Address Autocomplete feature, example 2*

Example 3: The *ZIP* field and the *City* field have already been filled. For the *Address* field, only correct possible values are suggested.

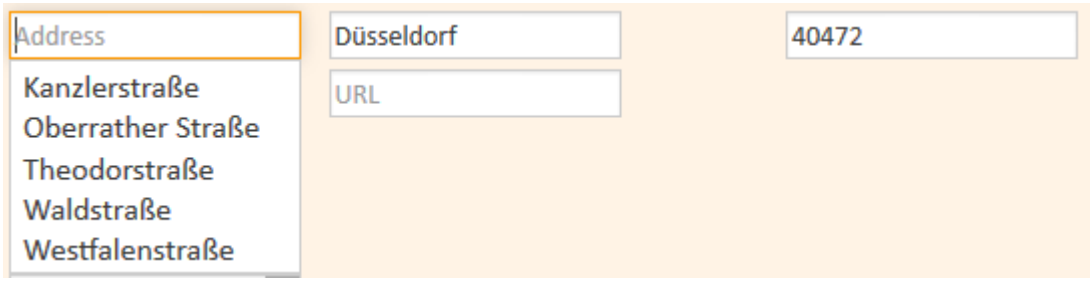


Figure 157: *ConSol CM Web Client - Suggestions of the Address Autocomplete feature, example 3*

Please note that the values which are displayed are only suggestions. The engineer can always overwrite the suggested values by typing into the field manually.

Of course, parallel to this feature, the ConSol CM standard feature *Autocomplete Search* works and will display suggestions for the customers which are already part of the ConSol CM database.

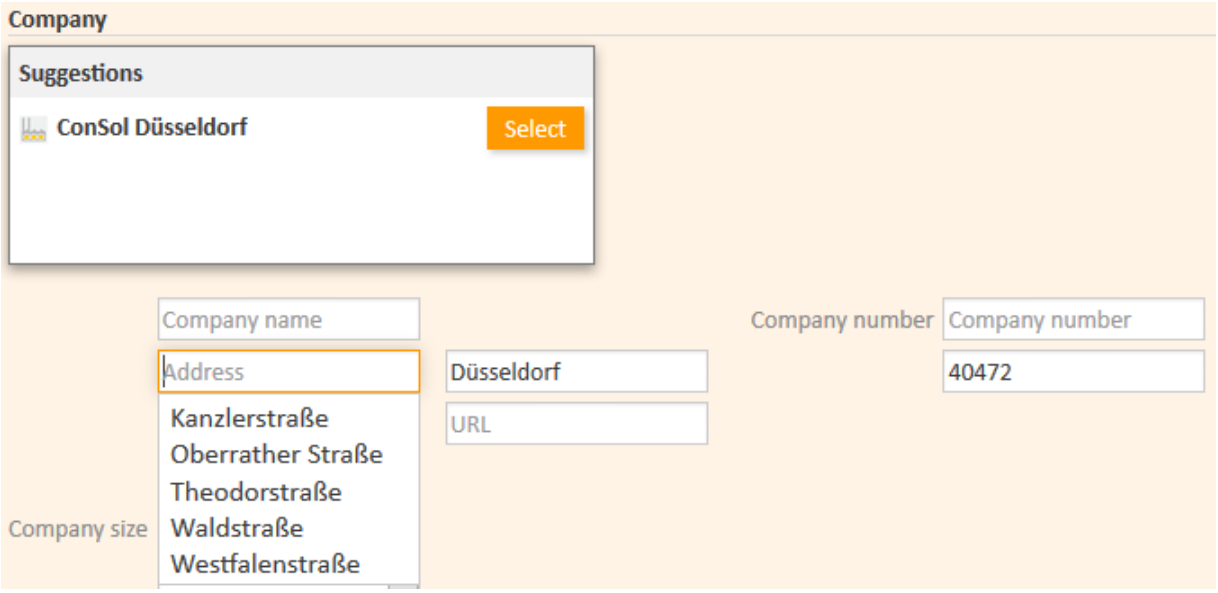


Figure 158: *ConSol CM Web Client - Suggestions of the standard Autocomplete Search feature in combination with Address Autocomplete*

To configure your ConSol CM system for this feature, the following steps have to be performed:

1. Switch on the *Address Autocomplete* feature using the Admin Tool, see section Switch on the Address Autocomplete Feature Using the Admin Tool.
2. Import zip code/city/address data into the ConSol CM database. See section Import Zip/City/Address Data into the ConSol CM Database.
3. Define the autocomplete strategy by creating one or several address autocomplete configurations using the ConSol CM Admin Tool, see section Define the Address Autocomplete Configuration Using the Admin Tool.
4. Refresh the index, as described in section Refresh the Index.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.10.2 Switch on the *Address Autocomplete* Feature Using the Admin Tool

Enter the system property cmas-app-admin-tool, autocomplete.enabledand set its value to “true”. The system property is not present in a standard ConSol CM installation and has to be added manually.

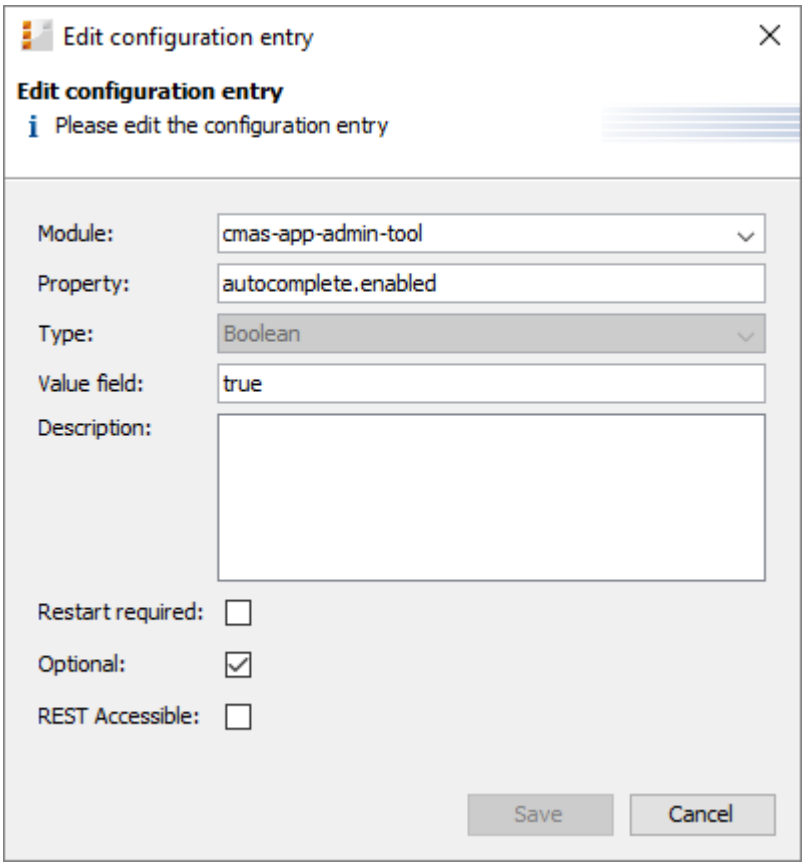


Figure 159: *ConSol CM Admin Tool - System, System Properties: System property for the Address Autocomplete feature*

You will then see the new navigation item *Address Autocomplete* in the navigation group *Customers*. You will learn how to configure the autocomplete strategy in section Define the Address Autocomplete Configuration Using the Admin Tool. But before it makes sense to define the strategy, the source data have to be imported. Please proceed to the next section to learn how to do this.

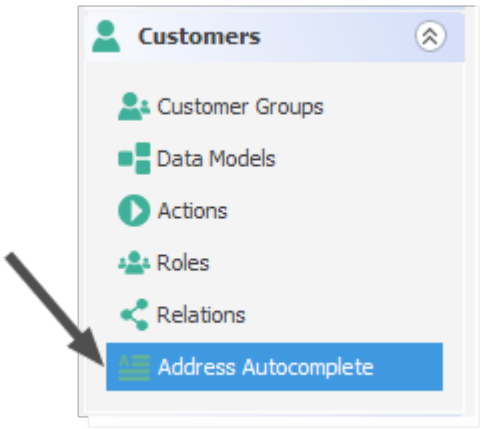


Figure 160: *ConSol CM Admin Tool - Navigation item Address Autocomplete in navigation group Customers*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.10.3 Import Zip/City/Address Data into the ConSol CM Database

Import the data into the table `cmas_autocomplete_address` in your ConSol CM database.

This import has to be implemented by a person who knows how to insert data correctly into a relational database! An import or import script are neither part of the ConSol CM distribution nor part of standard maintenance. You can implement the import script using a tool of your choice. In case you need any support, please ask your ConSol CM Consultant for help and advice.

The import has to comprise three fields for each data set (see the figure below):

- 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    |                |
+--------+--------------+------+-----+---------+----------------+
```

Figure 161: *ConSol CM Database - Table cmas_autocomplete_address*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

C.10.4 Define the Address Autocomplete Configuration Using the Admin Tool

To define the autocomplete strategy, you have to perform the following steps:

1. Create one or more Address Autocomplete Configuration
2. Configure the behavior for each Address Autocomplete Configuration

C.10.4.1 Create One or More Address Autocomplete Configurations

An Address Autocomplete Configuration represents a mapping of a customer field to one of the key fields in the `cmas_autocomplete_address` table, e.g., you want to define "When the engineer starts inputting data in the field *ResellerCompanyData*:*zip*, the system should search in *zip*, and when the engineer starts typing in the customer field *ResellerCompanyData*:*city*, the system should search in the *city* field".

You have to define each of the Address Autocomplete Configurations in the Admin Tool in the navigation item *Address Autocomplete* in the navigation group *Customers*. Add a new Address Autocomplete Configuration by clicking the *Add* button, entering the name of the new configuration, and clicking *Save*.

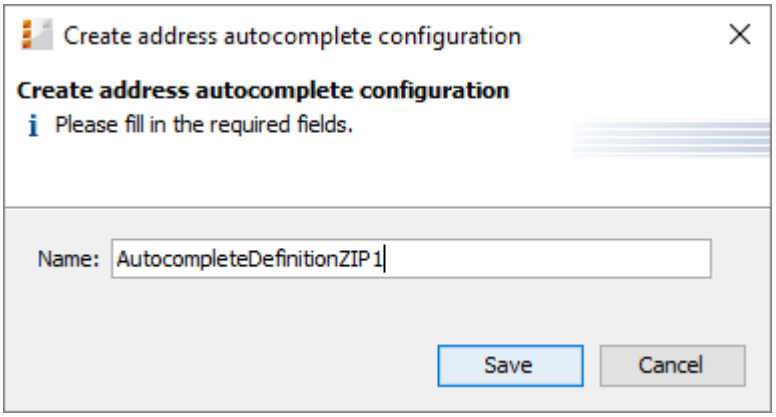


Figure 162: *ConSol CM Admin Tool - Customers, Address Autocomplete: Adding a new Address Autocomplete configuration*

C.10.4.2 Configure the Behavior for Each Address Autocomplete Configuration

To configure an Address Autocomplete Configuration, mark the definition in the list and add one or more Address Autocomplete Fields. These are the mapping rules from the customer fields to the key fields in the `cmas_autocomplete_address` table.

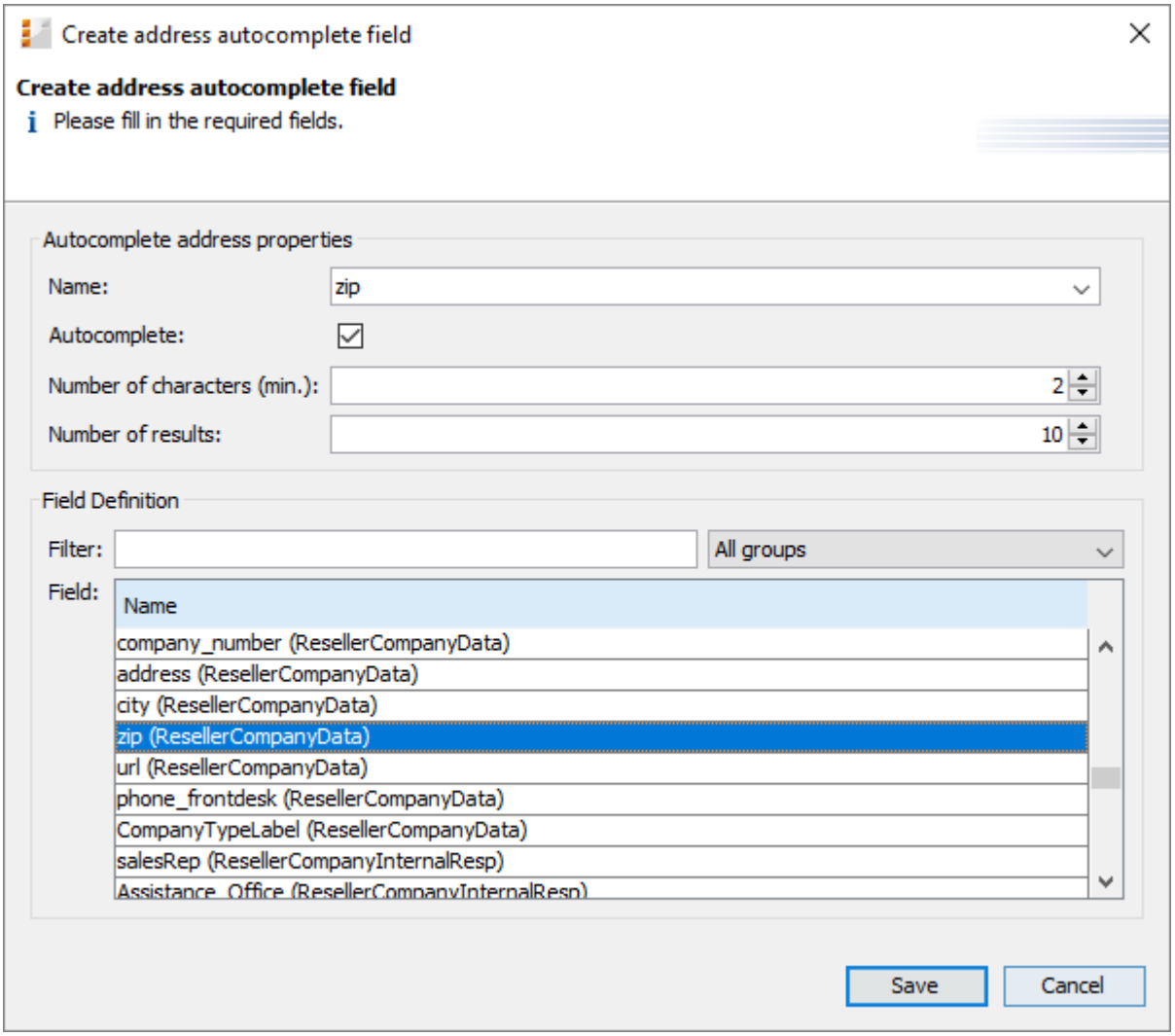


Figure 163: *ConSol CM Admin Tool - Customers, Address Autocomplete: Definition of an Address Autocomplete field*

The following fields have to be defined for each Address Autocomplete Field:

- **Name:**
  Select *city*, *street*, or *zip*. Here you define which key field from the `cmas_autocomplete_address` table is used.
- **Autocomplete:**
  - **If switched on:**
    The field itself will be an autocomplete field without any dependencies on other fields.
  - **If switched off:**
    The field will not be an autocomplete field itself but will be filled depending on other fields. E.g., a *city* field with *Autocomplete* switched off will be automatically filled when the *zip* field or the *address* field is filled but will not directly react to input from the engineer.
- **Number of characters (min.):**
  Here you define the number of characters the engineer has to type into the customer field

before the value recognition and autocompletion start. If you want the system to display the list as soon as the cursor has been placed in the customer field in the Web Client, leave the configuration field here empty.

- **Number of results:**
  Here you define the maximum number of suggestions in the (drop-down) list.
- **Field Definition:**
  Here you select one customer field which should react to the autocomplete input. Only a single selection is possible.

In the following example, two Address Autocomplete Fields have been defined for the Address Autocomplete configuration *AutocompleteDefinitionZIP1*, shown in the following figure.

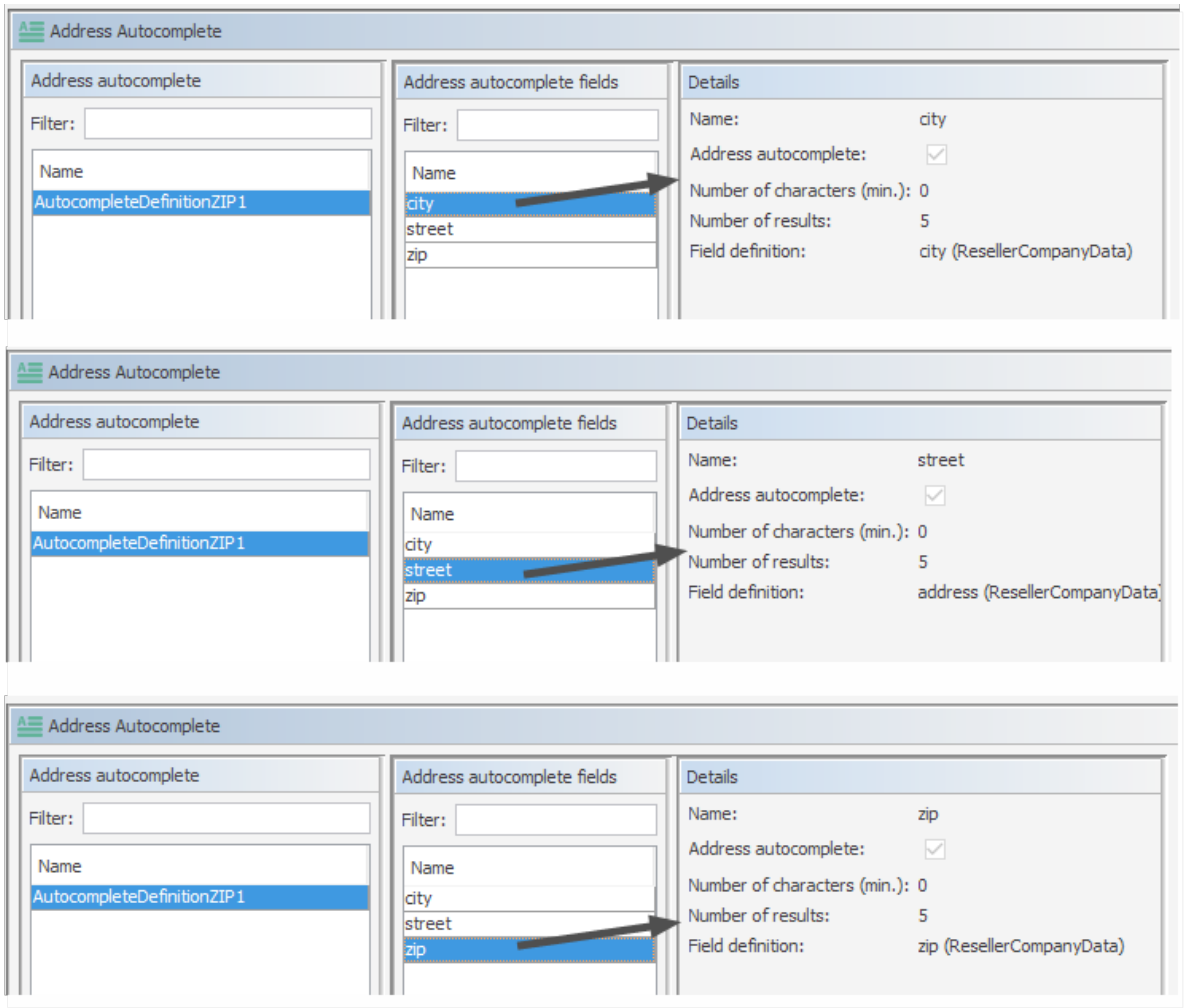


Figure 164: *ConSol CM Admin Tool - Customers, Address Autocomplete: Complete Address Autocomplete configuration*

C.10.4.3 Refresh the Index

When you have entered and configured all required Address Autocomplete Configurations, you have to refresh the index. For details about the index, please refer to section Index.

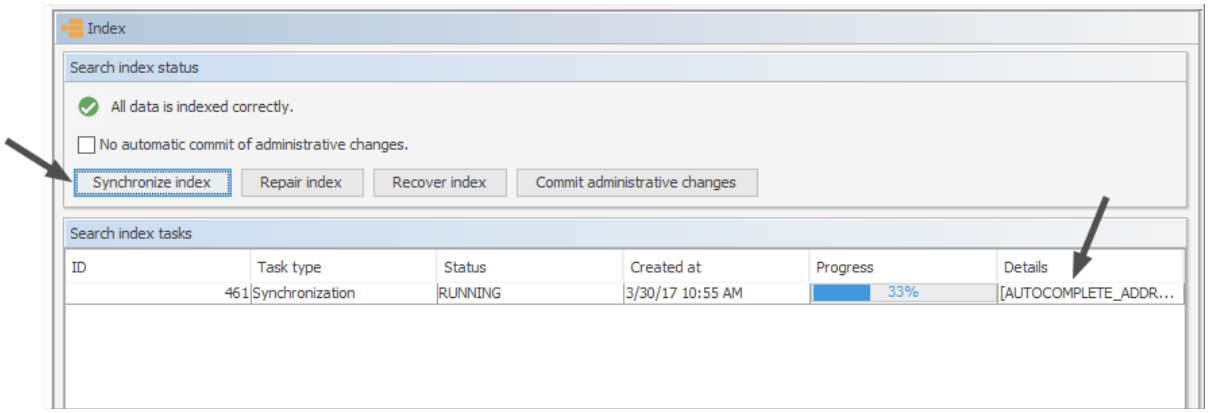


Figure 165: *ConSol CM Admin Tool - Services, Index: Index refresh*

C.10.5 Edit an Address Autocomplete Configuration

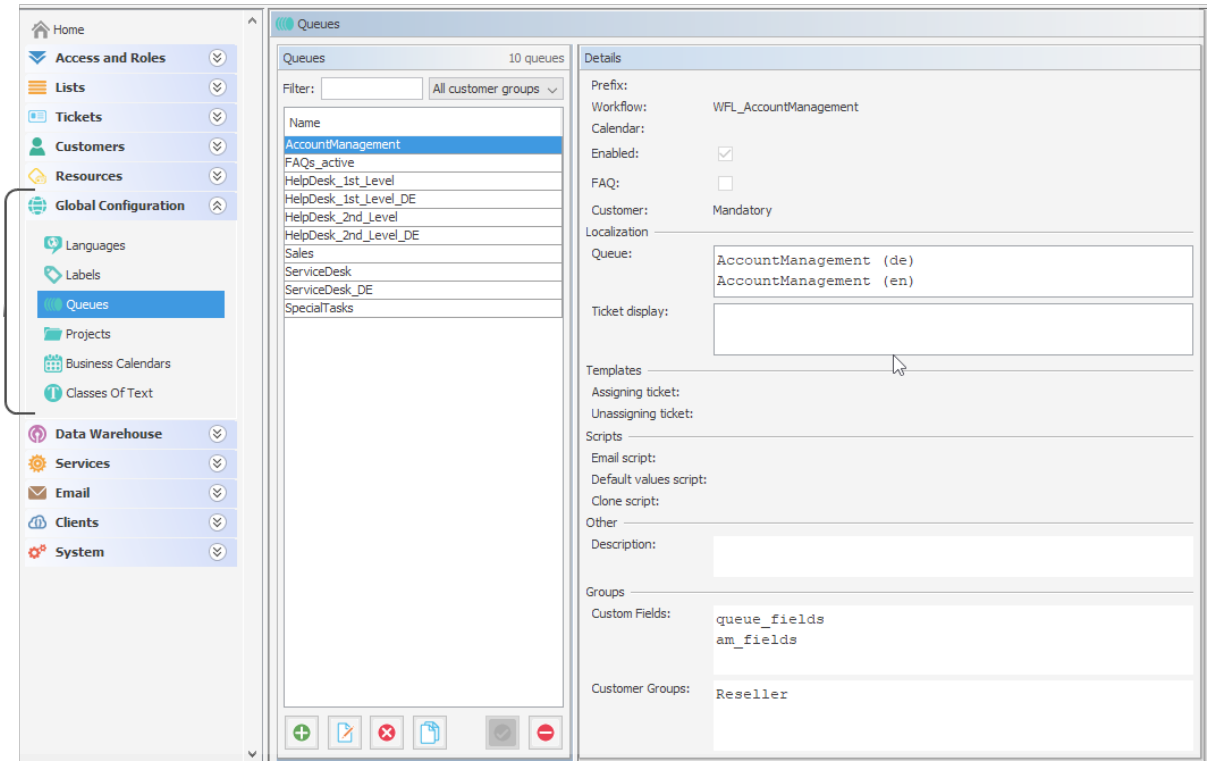
In order to edit an existing Address Autocomplete Configuration, use the *Edit* button . You can edit the Address Autocomplete Configuration and/or each of the Address Autocomplete Fields.

C.10.6 Delete an Address Autocomplete Configuration or Address Autocomplete Fields

In order to delete an Address Autocomplete Configuration or Address Autocomplete Field, mark the configuration or the field in the respective list and press the *Delete* button.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

D - Global Configuration



In this section you will learn how to configure some global settings in the ConSol CM system:

- Languages
- Queues
- Projects
- Working with Calendars
- Business Calendars
- Microsoft Exchange Calendar Integration
- Classes of Text

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

D.1 Languages

This chapter discusses the following:

D.1.1 Languages 269

D.1.2 The Use of Locales 270

D.1.1 Languages

ConSol CM can be configured to offer localization functionalities for the Web Client in one or more of any number of languages. In the Admin Tool, this is configured on the navigation item *Languages* in the navigation group *Global Configuration*. This will also influence the languages which are offered in the Process Designer.

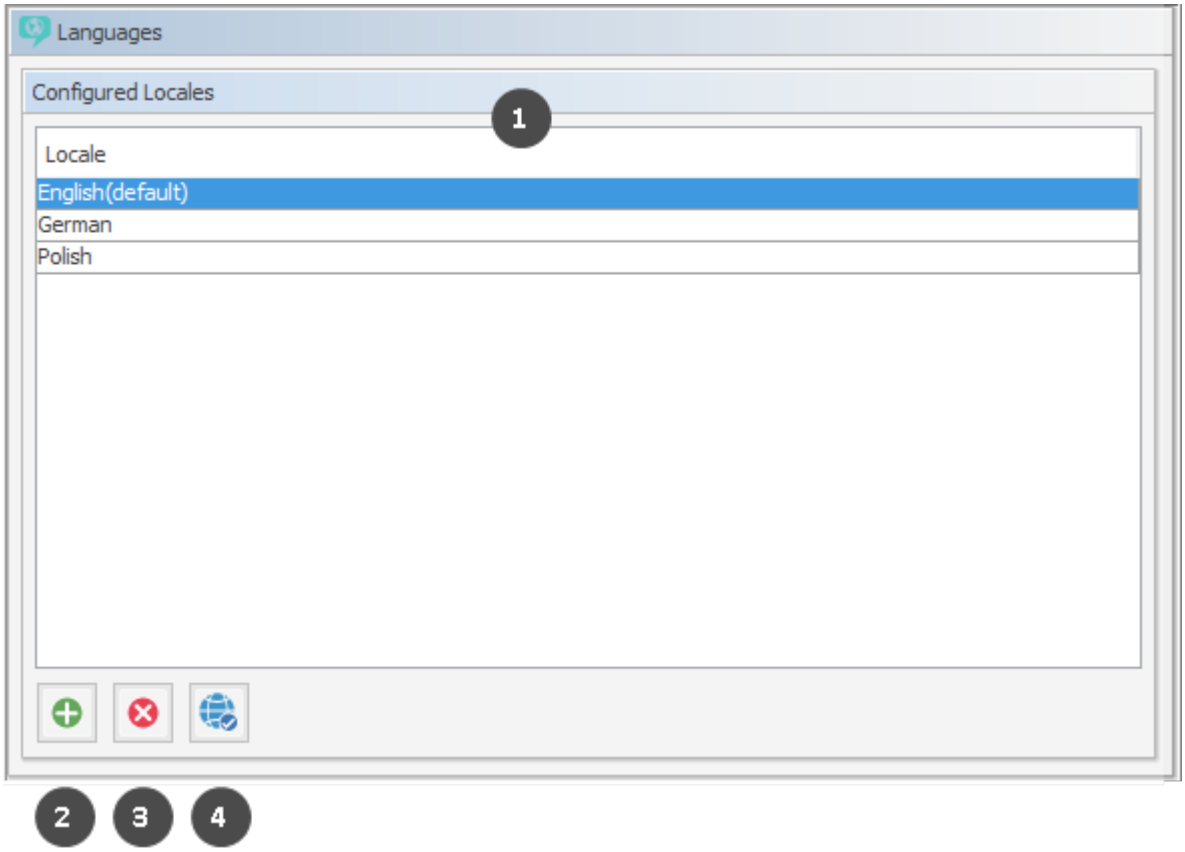


Figure 166: *ConSol CM Admin Tool - Global Configuration, Languages*

Configured Locales:
In this list, the locales (1) which will be available in the entire system are configured. This influences the lists for localized values in the Process Designer (e.g., for activities) and in the Admin Tool (e.g., for ticket field values). The displayed values for those activities or fields then depend on the locale of the web browser which is running the ConSol CM Web Client.

- Click the *Add* button to add more locales (2).
- Click the *Delete* button to remove the selected locale from the list (3).

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- Click the *Locale* button to set the selected locale as the default locale. The default locale is used when the browser's preferred locale is not present in ConSol CM. E.g., if the engineer has set the browser locale to FR and in the ConSol CM administration only English (default), German, and Polish are available, English will be used as the default (4).



Make sure that the configured languages are installed on each machine where ConSol CM is running or is used. This will not be checked automatically.

D.1.2 The Use of Locales

For an engineer who works with the ConSol CM Web Client, the interface is displayed in the language that is configured in the web browser if it is a locale that is configured in ConSol CM. If no matching CM locale can be found, the default locale which has been set in the Admin Tool is used.

Depending on the location in the Admin Tool, the mechanism to enter the localized terms is slightly different. Please read the following section for a details explanation: The Different Modes of Localizing Terms and Labels Using the Admin Tool

In the Process Designer, the locales which have been configured in the Admin Tool are available. However, you can also delete locales in the Process Designer. Please refer to the *ConSol CM Process Designer Manual* for details.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

D.2 Queues

This chapter discusses the following:

D.2.1 Introduction 271

D.2.2 Queue Administration Using the Admin Tool 272

D.2.1 Introduction

The queue is the core component of ConSol CM administration. It contains thematically related tickets which should be handled in the same way and follow the same business process. Every queue has exactly one workflow, which implements the desired course of action. A workflow can be used for several queues. The required ticket data fields are assigned to the queue. In addition, the access permissions, which are granted to the users via roles, are based on queues.

Therefore, the queue determines:

- if the tickets have customers and who are the customers (customer groups)
- how the tickets should look like (ticket data fields)
- how its tickets are processed (workflow)
- who can work on the tickets (permissions)

In addition, some optional settings, as templates, scripts, business calendars, text classes and projects, can be made for the queue.

Queues often reflect the organizational structure of the company. For example, there can be one queue for each department, as each department has its own way of working. A ticket can be passed from one queue to another. In this case it adapts to the new queue, i.e., it receives the data fields of the new queue and only users with permissions for the new queue can work on it.



The objects which are assigned to the queue are defined in other sections. Except for the workflow and the customer assignment, all parameters can be modified after creating a queue, so you can decide in which order you want to perform the configuration.

D.2.2 Queue Administration Using the Admin Tool

In the Admin Tool, queues are managed on the navigation item *Queues* in the navigation group *Global Configuration*.

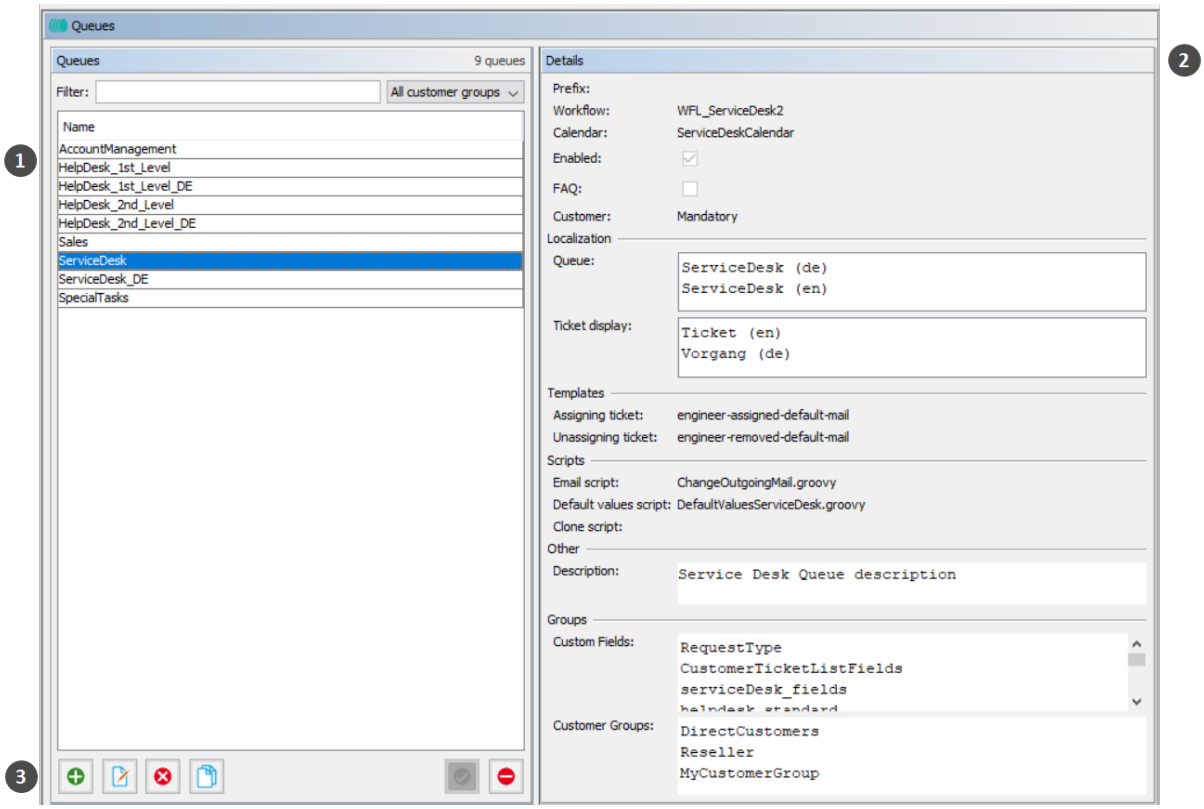


Figure 167: *ConSol CM Admin Tool - Global Configuration, Queues: Queue administration*

The navigation item *Queues* contains a list of all queues (1) which can be filtered using a string filter and/or a pull-down menu where the customer group can be selected. In the *Details* section (2) on the left hand side, all queue details are displayed (in read-only mode).

To add, edit, delete, copy or (de-) activate a queue, use the respective button in the button bar (3) below the queue list.

D.2.2.1 Filter the Queue List

Queues you want to edit or copy can be found most quickly if you enter filter information in the fields above the queue list.

You can filter for queues which

- contain a certain text string (blanks are interpreted, too) and/or
- are assigned to customer groups.

D.2.2.2 Create a Queue

You create a new queue by clicking the *Add* button below the queue list. The pop-up window which appears is the same for adding and for editing a queue. In both cases, you have to edit the queue details.

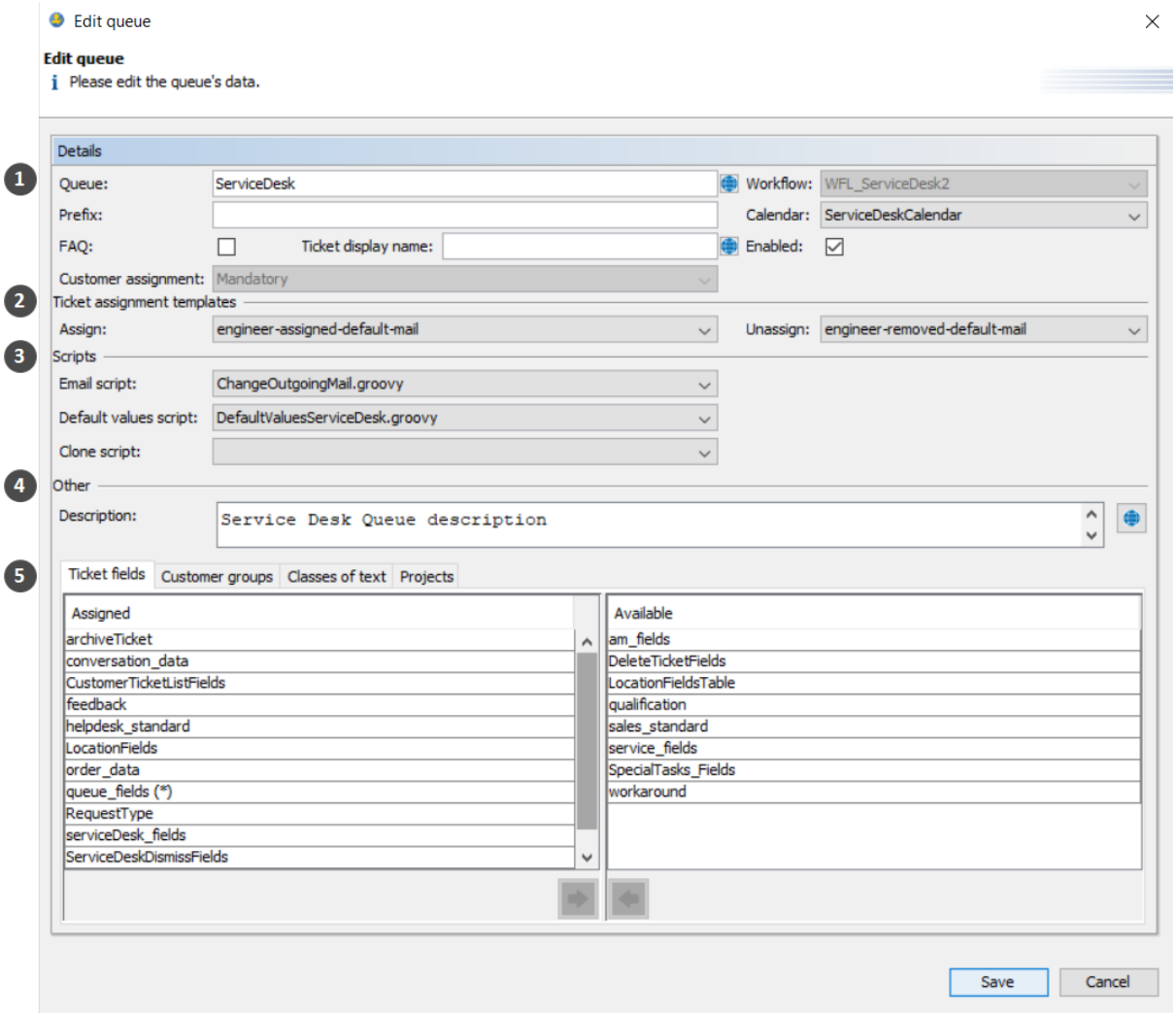


Figure 168: *ConSol CM Admin Tool - Global Configuration, Queues: Create a queue*

The following fields are available:

- **General configuration (1):**
  - **Queue:**
    Enter the technical queue name in this field. Click the *Localize* button to enter the localized queue name for all languages that are available in the system. The localized queue name (depending on the web browser locale) will be displayed in the Web Client in the ticket header. If no localized values are provided, the technical name will be displayed.
  - **Workflow:**
    Choose the workflow for the queue from this list.

    When you have developed and deployed a new workflow, it will only be available in the Admin Tool after a reload of Admin Tool data!

    Once you have assigned a workflow to a queue it cannot be changed anymore!

- **Prefix:**
  You can enter a prefix for the ticket IDs of a queue, e.g., when the ticket ID should indicate to which queue or organizational structure it belongs.

  ⚠ The prefix remains with the ticket name if the ticket is moved to another queue.

  ⚠ Please do not use underscores within the ticket prefix as they might cause layout problems. Up to five digits are supported for prefixes and suffixes in total. This also applies when modifying the ticket name in the workflow.

- **Calendar:**
  Choose the business calendar for the queue from the list. Business Calendars in CM define working hours, holidays and the valid time zone (see section Business Calendars). They are used, e.g., for time triggers in the workflow and have to be activated explicitly for each trigger, i.e., in order to work with time calculations based on a business calendar, it has to be configured in three places:
  - In the navigation group *Global Configuration*, navigation item Business Calendars the calendar is created and the active and vacation times are configured.
  - In the queue administration a calendar is assigned to the queue.
  - For each time trigger in the workflow the use of the queue-specific calendar can be activated or not. Refer to the *ConSol CM Process Designer Manual* for a detailed explanation of working with time triggers.
- **FAQ:**
  Ticking this checkbox marks the queue as a knowledge base for CM/Track users. They can search for tickets in this queue in CM/Track, the ConSol CM Web Portal. Please see also section CM/Track: FAQs in CM/Track for more information about this topic.
- **Ticket display name:**
  Optional. The string entered here will be displayed as ticket header. If a localized value is provided, this will be used, otherwise, the technical name will be used. In this way, you can adapt the CM system to display terms like *Service case*, *Request* or *Case* and you are not limited to the term *Ticket*. If nothing has been defined manually, the default value *Ticket* is used.
  Please note that two manually defined parameters can potentially define the value for the ticket display name.
  - The ticket display name set here, in the queue details definition. This is queue-specific, i.e. for each queue, a different display name can be defined. For example, in the Service Desk queue, this might be *Service case*, whereas in the Sales queue, this might be *Opportunity*.
  - The value which is defined using the label `ticket.base.initcap`. This is used only if the queue-specific ticket display value is not set. For details about labels, please read the section about Labels.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

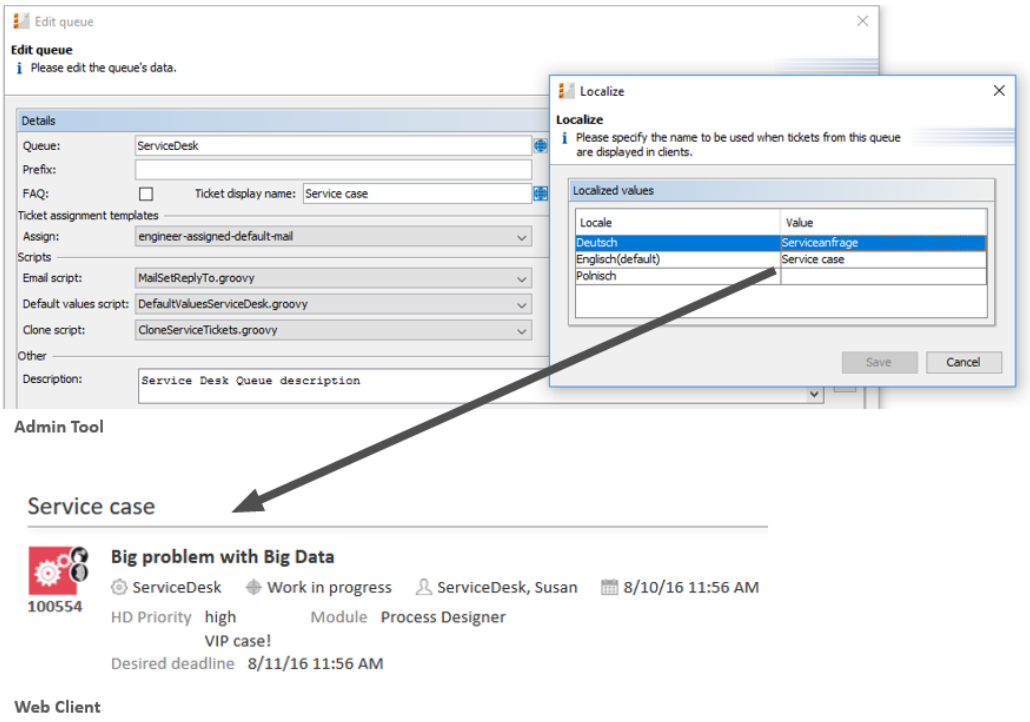


Figure 169: *ConSol CM Admin Tool and Web Client: Setting and viewing a queue-specific ticket header*

- **Enabled:**
  If this checkbox is checked, the queue is immediately available in the system after saving, otherwise the queue is disabled. In enabled queues, you can create tickets. In disabled queues, this is not possible.
- **Customer assignment:**
  Determines if the tickets in the queue have customers or not. There are two settings:
  - **Mandatory**: Tickets in this queue always have a customer. This option is selected by default and represents the standard behavior of tickets in ConSol CM.
  - **None**: Tickets in this queue never have a customer. This is useful for scenarios where customers are not needed, for example, FAQ or news queues, or use cases which do not require customers from a business perspective. Therefore, you cannot assign customer groups to the queue. In the Web Client, tickets in such a queue do not have a *Customers* section and the customer is not shown in the ticket list.

    > In CM/Track, you cannot create tickets in queues without customers. Tickets without customers are only supported for the FAQ and news functionalities in CM/Track.

> ⚠ You can use text templates for tickets without customers. In this case, the content of customer field variables is empty. If the text templates contain scripts which access customer fields, you need to adapt the scripts with conditions and/or null safe operations to avoid exceptions in the log files.

> ⚠ The customer setting cannot be modified once the queue has been created.

> ⓘ **Transferring tickets between queues with a different customer setting**
>
> It is not possible to transfer tickets between queues with different customer settings in the Web Client. For such transfers in scripts, the following applies:
>
> - If you transfer a ticket from a queue with customers to a queue without customers, the customer is removed automatically.
> - If you transfer a ticket from a queue without customers to a queue with customers, you need to provide the customer using the method `workflowApi.addNewPrimaryContact(unit.id, "supporter")`.

- **Ticket assignment templates (2):**
  Here you can choose email templates which are used for automatic emails which are sent to the (new) engineer when a ticket is assigned to an engineer (*Assign*), or to the (old) engineer when a ticket is taken away (*Unassign*). When you have defined the templates in the *Scripts & Templates Administration* of the Admin Tool (see section Admin Tool Templates) they will be available in the drop-down menu. If you do not want the ConSol CM system to send an automatic email in case of the engineer operation, just leave the field empty. Please keep in mind that the system properties cmas-core-server, mail.notification.engineerChange (= “true”) and cmas-core-server, mail.notification.sender have to be set, see sections System Properties and Administrator and Notification Email Addresses for details.
- **Scripts (3):**
  Scripts are used to automate recurring tasks and activities. They are managed and stored in the *Scripts & Templates Administration* (see section Admin Tool Scripts). You can assign:
  - **Email script**
    Choose a script from the list if outgoing emails for this queue should be modified by the script, e.g., to contain default values like the sender or address fields. The email script indicated here is the last script that processes an outgoing email, so all former settings will be overwritten if a variable has been set before. All scripts of type *Email* that are stored in the script section are available for selection, so please make sure to pick the correct one.

> ⚠ **Reply-to addresses in ConSol CM**
>
> There are four potential Reply-to addresses which you deal with:



- **System property `mail.reply.to`**
  If this system property is set, its value is displayed in the email editor in the Web Client. The email address can be overwritten by the outgoing email script. If the page customization attribute *showReplyTo* for the type *mailTemplate* is set to “false”, the Reply-To address is not displayed in the email editor, but used anyway.
- **Queue-specific outgoing email script**
  If a Reply-to address is set in the outgoing email script, this address is used. It overwrites the system property `mail.reply.to`, although the address set in this property is displayed in the email editor.
- **System property `mail.from`**
  Only relevant if the Reply-to address is neither set in the system property `mail.reply.to` nor in the outgoing email script. In this case, most email clients use the From address as Reply-to address.
- **Email address of the current user**
  Only relevant if the Reply-to address is neither set in the system properties `mail.reply.to` or `mail.from` nor in the outgoing email script. In this case, the personal email address of the currently logged in user is used as Reply-to address for emails from the Web Client.

The Reply-to address which was really used is displayed in the ticket history of the Web Client.

Best practice:

Always set a Reply-to address to prevent personal email addresses from being used, as this would lead to customer emails being sent to the users’ personal email accounts instead of ConSol CM. There are two places to set the Reply-to address:

- **System property `mail.reply.to`**
  This is the simplest way to set the Reply-to address. This address is also displayed in the email editor.
- **Queue-specific outgoing email script**
  If different Reply-to addresses are required depending on the queue, we recommend to write one outgoing mail script where queue names are mapped to specific Reply-to addresses. This can then be extended for Bcc, Cc or other addresses.

You can combine the `mail.reply.to` system property and queue-specific Reply-to addresses. In this way the email address set in the outgoing email script is used for queues with such a script, and the email address from the system property is used for the rest of the queues.

Usage in workflow scripts

When sending emails in workflow scripts, you need to set the Reply-to address within the script.



- Retrieve the value of the system property `mail.reply.to` and set it as a Reply-to address in the outgoing email:

```
configurationService.getValue("cmweb-server-
adapter","mail.reply.to")
```

- Use the `Mail` object if the outgoing email script should be used. This overwrites the system property `mail.reply.to`.

```
mail.useDefaultScript()
```

See the *ConSol CM Process Designer Manual* for a detailed explanation.

- **Default values script**
  Here you can select a script to preset values of list boxes when creating a ticket for this queue in the Web Client and/or CM/Track. The script has to be present in the *Scripts & Templates Administration* of the Admin Tool and has to be of type *Default values*. Please refer to section Scripts of Type Default Values for details about this topic.
- **Clone script**
  Here you can select a script which is executed when a ticket in this queue is cloned (duplicated) using the Web Client (*Clone* option in the ticket menu). The script has to be present in the *Scripts & Templates Administration* of the Admin Tool and has to be of type *Clone*. The clone script sets default values for a ticket which is created using the *Clone* operation. Please refer to section Scripts of the Type Clone for details.

- **Other (4)**
  - **Description:**
    You can enter a free-form description in this field, e.g., to document the purpose of the queue. This information is shown in the Admin Tool only.
- **Tabs for assignment of fields, customer groups, classes of text, and projects (5):**
  - **Tab *Ticket fields*:**
    In order to show data fields (ticket fields) in tickets of the queue, you have to assign the respective ticket field groups here. See section Ticket Fields (Setting Up the Ticket Data Model) for details about the definition of ticket fields.

    Tab *Customer groups*:
    Tickets in the queue can only be created for customers from the selected customer groups. Please make sure that the engineers who are supposed to work with tickets of the queue also have the respective access permissions to the customer (group) data.

    Tab *Classes of text*:
    Here you can assign the classes of text which should be available in tickets of this queue. Please see section Classes of Text for an explanation of the text class definition.
  - **Tab *Projects*:**
    Here you can assign projects to the queue. Engineers who work on a ticket in the queue can book times on the projects that have been assigned to the queue. Projects are defined on the Projects page.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

On each tab you can assign a selected entry by clicking the *Assign* button and remove it by clicking the *Unassign* button.

Click *Save* afterwards to create the queue. The details of the new queue are displayed on the right-hand side of the page.

D.2.2.3 Edit a Queue

If you want to edit a queue, select it in the list and click the *Edit* button or double-click the name of the queue. Modify the queue details and click *Save* to store your modifications.

> ⚠ You cannot change the workflow of a queue once the queue has been saved for the first time!

D.2.2.4 Delete a Queue

Select the queue you want to delete in the list and click the *Delete* button. If you confirm the following dialog with *Yes*, the queue will be deleted and is no longer available in the system.

> ⚠ If there are still tickets for a queue it cannot be deleted. You have to move the tickets to another queue before you can delete it.

D.2.2.5 Copy a Queue

The *Copy* button allows you to save time when creating a queue. The selected queue will be copied. The new queue has the same name as the copied queue. Double-click the name or click the *Edit* button to open the edit window where you can modify the name and details of the queue. Click *Save* to store your modifications.

> ⚠ You cannot change the workflow of a queue once the queue has been saved for the first time!

D.2.2.6 Enable or Disable a Queue

You can disable a queue to prevent that new tickets can be opened in this queue. This allows a queue to be made temporarily unavailable without having to delete it. To disable a queue, select the queue in the queue list and click the *Deactivate* button. The entry in the list is now shown in italics. Just click the *Activate* button at the bottom of the page to enable the queue again.

What engineers in the Web Client can do in a disabled queue, provided they have the required permissions:

- execute workflow activities thereby moving the ticket through the business process
- edit the ticket
  - edit ticket fields
  - add comments, emails, attachments

What engineers in the Web Client can NOT do in a disabled queue:

- open / create new tickets
- move tickets from another queue to this queue using the drop-down menu in the ticket

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

D.3 Projects

This chapter discusses the following:

D.3.1 Introduction 280

D.3.2 Managing Projects Using the Admin Tool 280

D.3.1 Introduction

With ConSol CM you can book working time to projects. Please see the Time Booking Using ConSol CM section for a detailed explanation.

D.3.2 Managing Projects Using the Admin Tool

Projects are managed via the navigation item *Projects* in the navigation group *Global Configuration*. For a project to be active and available in the Web Client, it has to be assigned to a queue (see section Queues). In the Web Client you can then book working time to tickets that are in one of the queues where the project has been assigned. Engineers can see their time bookings on the engineer profile page.

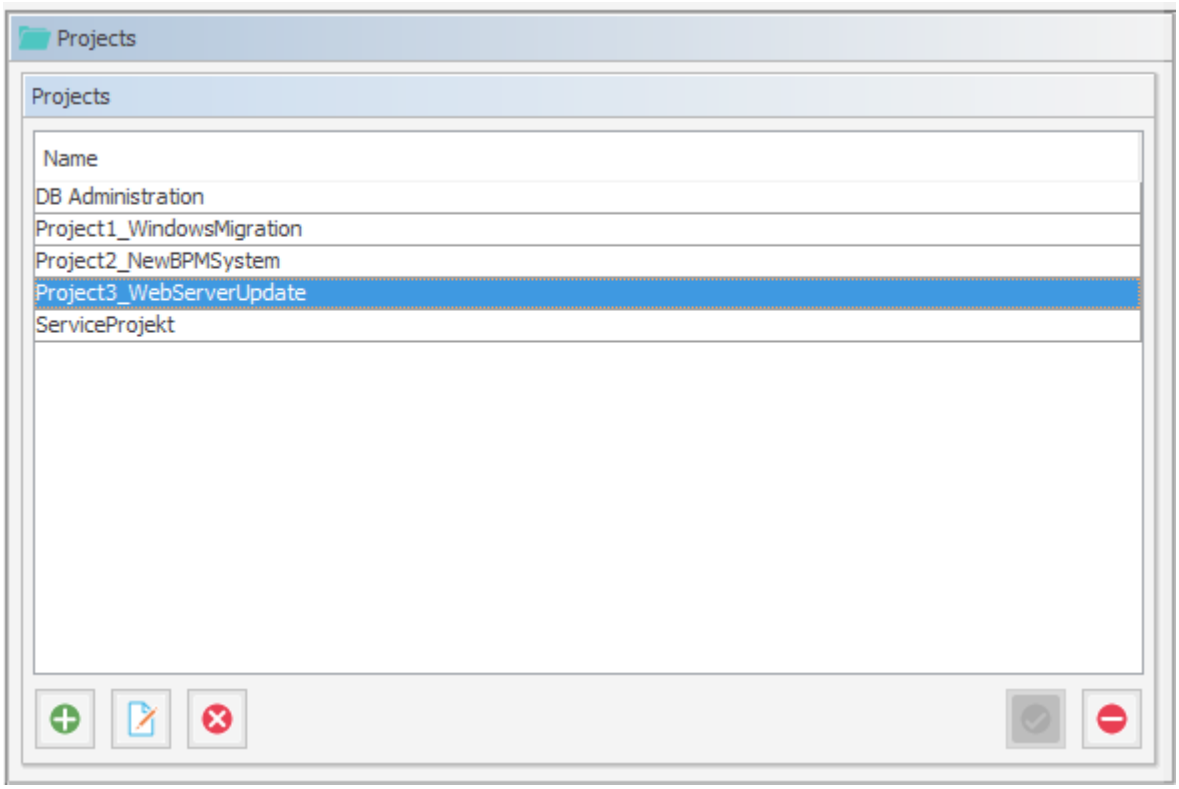


Figure 170: *ConSol CM Admin Tool - Global Configuration, Projects*

D.3.2.1 Create or Edit a Project

A project is defined by its name. Click the *Add* button to open a pop-up window where you can enter the name. Using the *Localize* button next to the name field you can localize the name (see Localize a Project). The checkbox *Enabled* is pre-selected to activate the project in the system (see also Disable or Enable a Project). You will get the same window when you click the *Edit* button to edit a project.

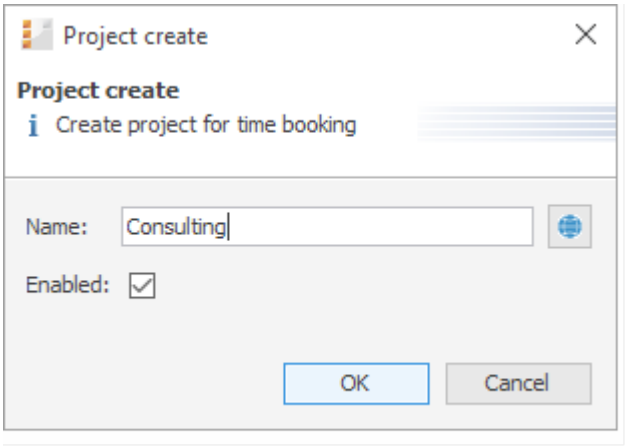


Figure 171: *ConSol CM Admin Tool - Global Configuration, Projects: Create or edit a project*

D.3.2.2 Delete a Project

A project can only be deleted if it is not assigned to any queues and has not been used for any time bookings. Otherwise you get a warning and can only disable this project (see below).

In order to delete a project, select it in the list and click the *Delete* button. After choosing *Yes* in the confirmation dialog the project will be removed from the list and the system.

D.3.2.3 Disable or Enable a Project

If a project is still assigned to a queue or has been used for a time booking in a ticket, but is not needed anymore, you can disable it. To do this select the project and click the *Deactivate* button. The entry in the list is shown in italics afterwards. The project is not available for new time bookings anymore. Just click the *Activate* button at the bottom of the page if you want to enable the project again.

You can also enable or disable a project in the window used for editing projects by selecting or deselecting the checkbox *Enabled*. When you create a project this checkbox is automatically selected.

D.3.2.4 Localize a Project

Click the Localize button in the create or edit window to enter a localized name for a project. See section Localization of Objects in General, Type 1 for a details explanation of the localization mechanism.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

D.4 Working with Calendars

In ConSol CM, there are two configuration options for calendars:

1. You can define business calendars which can be used to manage business hours for one or more teams which work with ConSol CM. This is covered in section Business Calendars.
2. You can configure ConSol CM to access Microsoft Exchange Server calendars. This is explained in section Microsoft Exchange Calendar Integration.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

D.4.1 Business Calendars

D.4.1.1 Introduction

A business calendar defines working hours. This can be used, for example, to represent the business hours of the Service Desk team to avoid system escalations being fired during non-business hours.

In ConSol CM, you can define as many business calendars as your company's environment requires. In this way you can configure specific working hours for each team.

> Tickets which have not been assigned to an engineer within an hour of opening the ticket are automatically moved to an escalation level. If a calendar defines working hours from 8 a.m. to 5 p.m. and a ticket arrives at 4:45 p.m., the ticket will not escalate at 5:45 p.m. but at 8:45 a.m. the next day. This time is calculated as follows: 15 minutes between ticket arrival and end of the working hours plus 45 minutes from next beginning of the working hours until the full hour given by the escalation limit is reached.

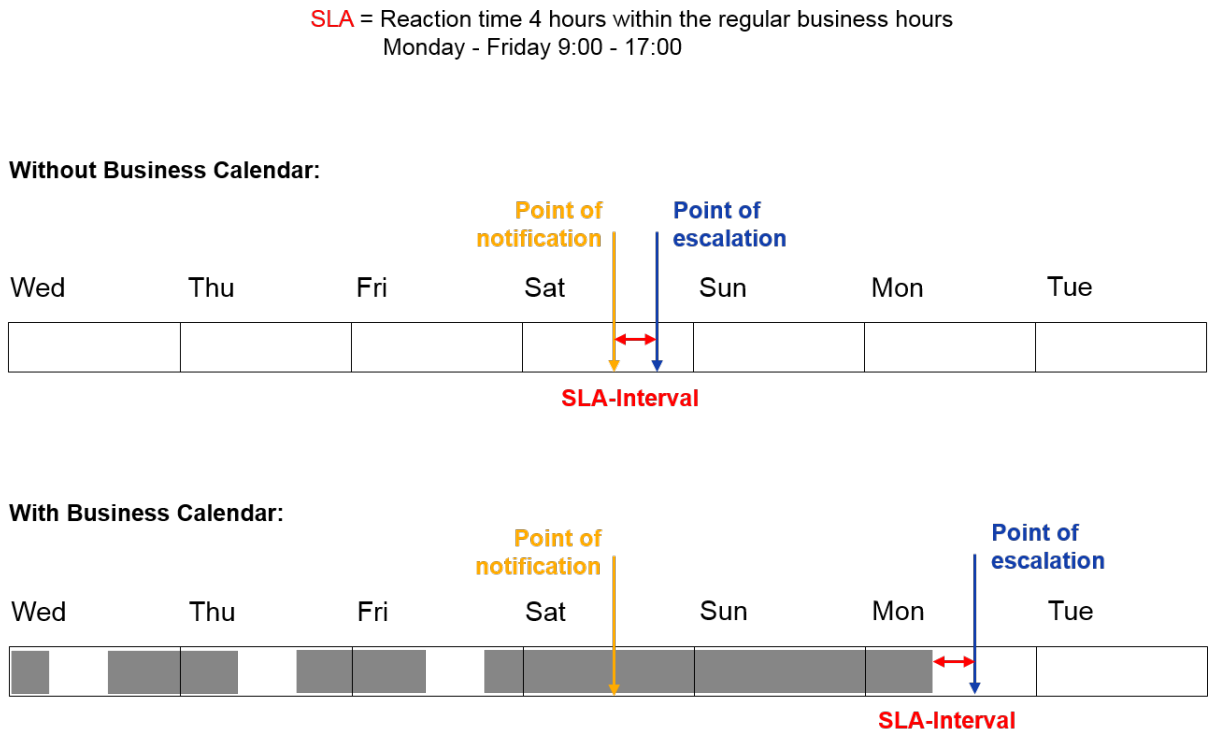


Figure 172: *ConSol CM Principle - Business calendar*

Aside from working hours, you can define holidays, too. On these days the automatic escalation pauses entirely. Holidays have to be defined per calendar. It is not possible to define a holiday that is valid for all existing calendars simultaneously.

If you want to work with times which are defined in a business calendar (e.g., use active time for a timer trigger in a workflow for an escalation), you have to perform three steps:

- create the business calendar with its active/inactive times (Admin Tool, see explanation below)
- assign the business calendar to all queues where it should be in operation, see section Queues (Admin Tool)

- assign the use of a calendar to every single workflow element where the calendar should be used as the basis for time calculations (Process Designer), as explained in the *ConSol CM Process Designer Manual*.

D.4.1.2 Configuration of Business Calendars Using the Admin Tool

In the Admin Tool, business calendars are defined via the navigation item *Business Calendars* in the navigation group *Global Configuration*.

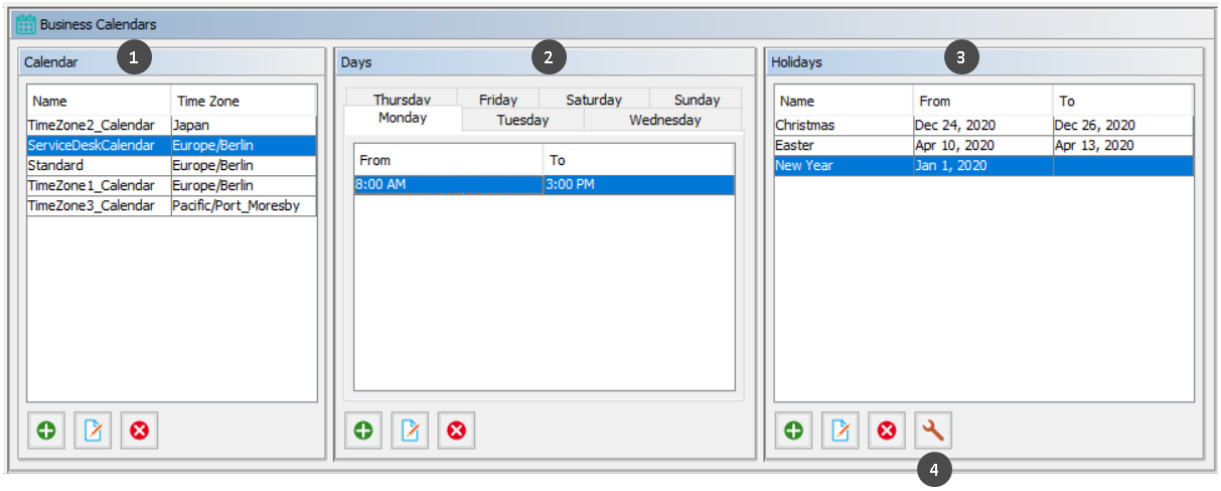


Figure 173: *ConSol CM Admin Tool - Global Configuration, Business Calendars*

- List of available business calendars (1)
- Configured working hours for the selected calendar (2)
- Holidays in the selected calendar (3)
- Button to import holiday definitions from an external csv file (4)

Creating a New Calendar

Click the *Add* button in the left part of the page to create a new calendar. A window with the following fields is displayed:

- **Name:**
  Enter a unique name for the calendar.
- **Timezone:**
  Choose the time zone to be used for the calendar.

> ⚠ This field only describes to which time zone the defined hours refer. The calendar itself is valid worldwide for the respective workflow!
>
> **Example:**
>
> The ConSol CM server is located in Detroit, MI, USA. In the business calendar, Europe/Berlin is set as time zone. A time trigger which uses the business trigger would fire based on Berlin time, not Detroit time.

Click *Save* afterwards to create the calendar.

By clicking the *Edit* button, you can modify a selected calendar in the same way. Click the *Delete* button to delete the selected calendar.

Defining the Working Hours for a Calendar

Select a calendar on the left and click the *Add* button in the middle part of the page to create the days and hours for this calendar. A window with the following fields is displayed:

- **Time range**
  Enter the time range for which the automatic workflow escalations shall be active.
- **Days**
  Select the checkboxes of the days for which the time range shall be valid. It is possible to choose individual days or all days at once (checkbox *All*).

> If the system detects an inconsistency between the time defined here and an already existing time, you will get a corresponding message.

Click *Save* afterwards to create this time range for the marked days.

If you want to edit the time range later, you have to do it separately for each day. Select the respective day, click the *Edit* button and change the time range in the window that appears. Or click the *Delete* button to delete the time range for a selected day. It is not possible to edit or delete the time range for multiple days at once.

Defining the Holidays for a Calendar

You can define the dates and time periods for holidays using one of two approaches:

- Defining the holidays manually.
- Importing the holidays from an Excel file.

Defining the Holidays for a Calendar Manually

Select a calendar and click the *Add* button in the right side of the page to create a new holiday entry. A window with the following fields is displayed:

- **Name:**
  The name of the holiday.
- **From:**
  The date of the holiday.
- **To:**
  If it is a multi-day holiday (e.g., Christmas), you can enter the last date of the holiday here.

> It is not possible to define holidays that last only half a day.

Click *Save* afterwards to create the holiday.

If you want to edit a selected holiday entry just click the *Edit* button. Clicking the *Delete* button deletes one or more selected entries.

Importing Holidays for a Calendar from a .csv File

Holiday data can be imported from a .csv file conforming to the following format:

- First column: title/name of the holiday
- Second column: start date
- Third column: end date (use the same date as start date if it's a one-day holiday)
- Separator: comma (no comma at the end of the line)
- Slashes for the dates

```
Christmas,24/12/2020,26/12/2020
New Year,01/01/2020,01/01/2020
Easter,10/04/2020,13/04/2020
```

Please note that all dates for the holiday import must be written in the following format (as shown in the example): DD/MM/YYYY!

In the Admin Tool navigation group *Global Configuration*, navigation item *Business Calendars*, select a calendar and click the *Import holidays* button and enter the path to the .csv import file.

The new holidays will be imported in the holiday list of the selected business calendar.

D.4.2 Microsoft Exchange Calendar Integration

D.4.2.1 Introduction

Starting with CM version 6.10, it is possible to include a Microsoft Exchange calendar view in the Web Client.

Please refer to the ConSol CM System Requirements of your CM version for a list of supported Exchange Server versions.

The calendar view can be displayed...

- on the ticket page
- on the customer page, i.e.,
  - on the contact page
  - on the company page
- on the resource page

The calendar will be displayed in a distinct section of the ticket, customer, or resource page.

An engineer who works with the calendar view can ...

- display the monthly or weekly view
- move existing appointments using drag-and-drop
- create new appointments (if full access has been configured)

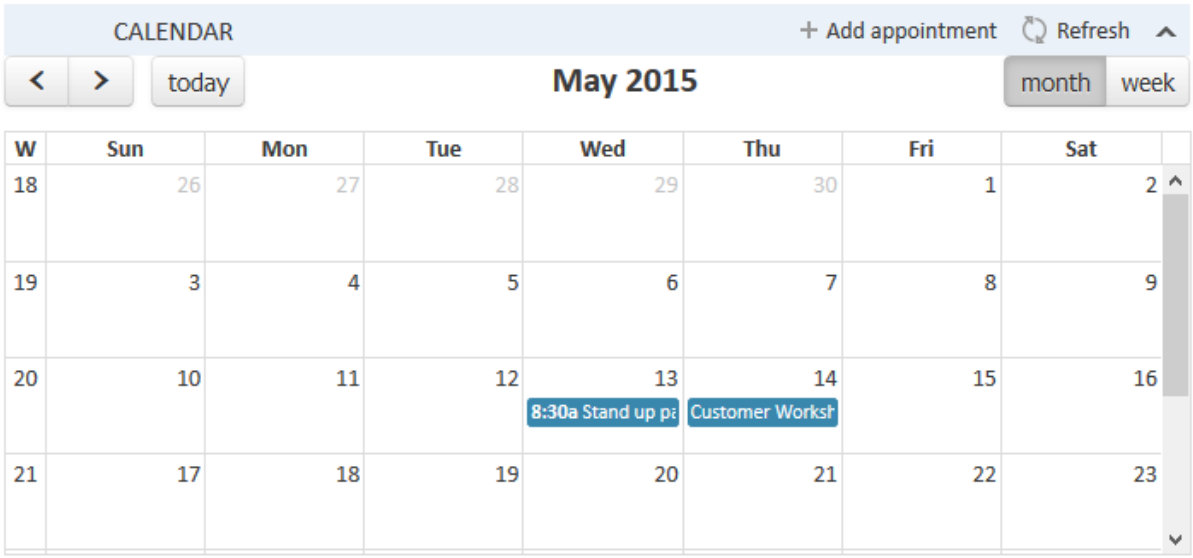


Figure 174: *ConSol CM Web Client - Ticket with Calendar section (monthly view)*

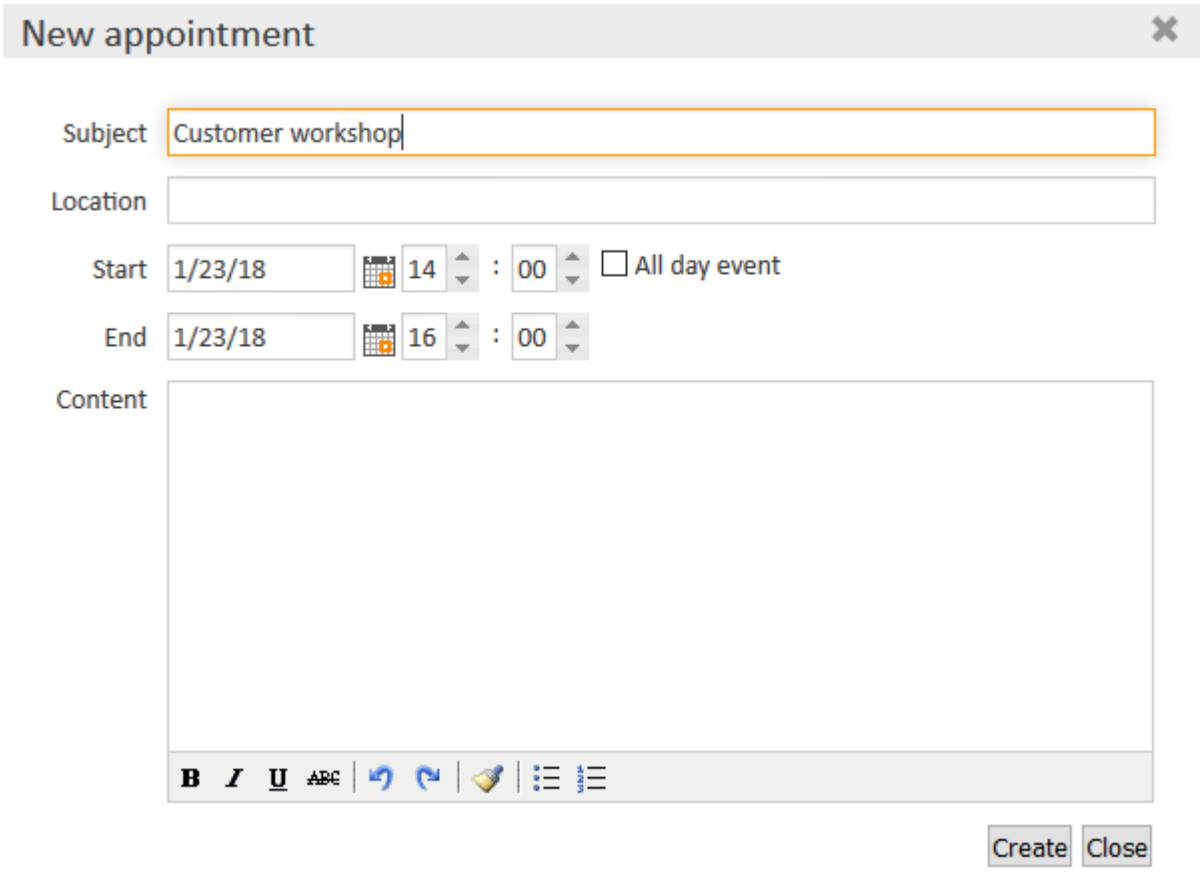


Figure 175: *ConSol CM Web Client - Adding an appointment in the calendar*

D.4.2.2 Configuring the Microsoft Exchange Calendar Integration

Basic Configuration

The integration of a Microsoft Exchange server to provide calendar data is done based on *page customization*. For a detailed introduction to this topic, please read the section about Page Customization. Here, only the calendar-specific configuration is explained.

Perform the following steps:

1. **Make the calendar section visible (example for the Ticket Edit page):**

   Log in as an administrator and open a ticket. Select *Enable page customization* in the main menu. Since the calendar section is not yet displayed, you cannot mark the element you want to configure, but instead have to select it in the page customization tree. Select *calendar/ticketEditPage/calendarSection* and set the attribute *state* from “hidden” to “expanded”. Alternatively, you can set “collapsed”. This will initially display a collapsed *Calendar* section and the engineer can expand it manually. In both cases, the calendar section of the ticket will be visible. As header *No Calendar* will be displayed. The configuration of the calendar follows in step 2.

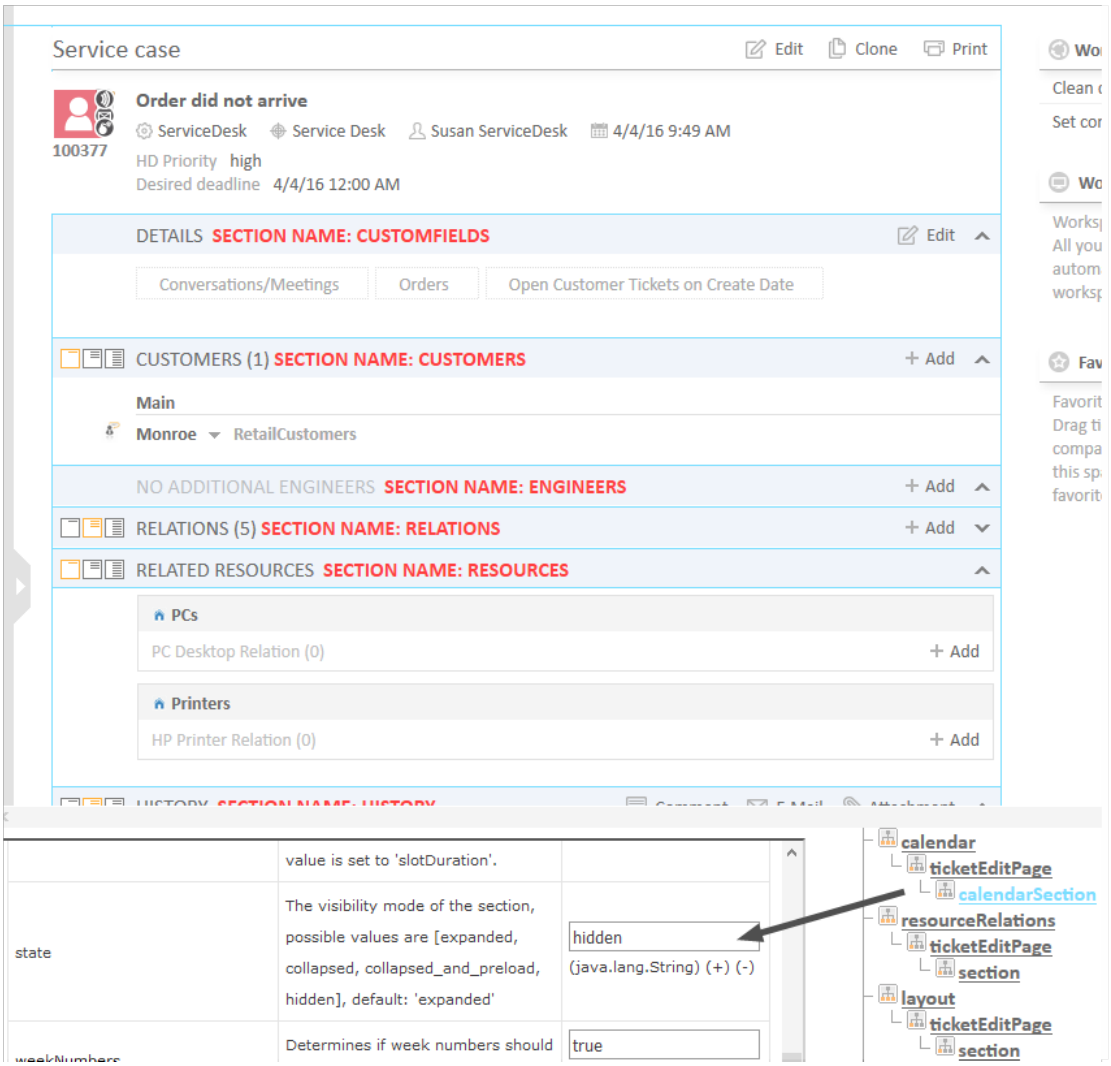


Figure 176: *ConSol CM Web Client - Using page customization to make the Calendar section of a ticket visible*

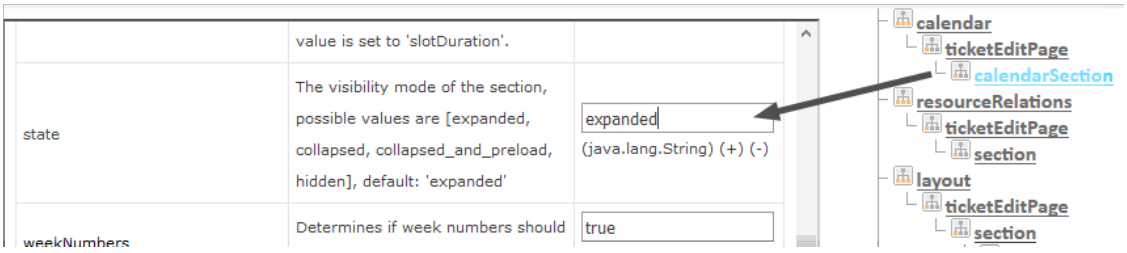


Figure 177: *ConSol CM Web Client - Page Customization Definition Section (PCDS) for the Calendar section of tickets*

2. **Configure the calendar script:**

   The connection to a Microsoft Exchange server calendar is implemented using an Admin Tool script. The script has to be stored in the Admin Tool script section, i.e., navigation group *System*, navigation element *Scripts and Templates*, and must be of script type *Calendar*

integration. The name of this script has to be entered as the value for the attribute *calendarInitializationScript* in the page customization (see step 1). This Admin Tool script is used to initialize the calendar source and has to return a map which contains all parameters describing the source. If the returned map is empty or is null, no calendar will shown (the section will be empty with the label *No calendar*).

The script has to be coded as shown in the following example:

```
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 example 22: *Example calendar integration script*

The following parameters can be used in the script:

- **name**
  String. The name of the source. For now it has no functional use, but in future CM versions, when multiple sources can be used in the calendar section, it will be used to identify them in the user interface.
- **color**
  String. Background color for appointment entries. Format: HTML colour format (e.g., red or #D80000)
- **editable**
  Boolean. Indicates whether appointment creation/editing/removal should be enabled. If this value has not been set explicitly, the value from the section customization will be used, which by default is set to “false”. NOTE: appointments are editable only if the current user has write permissions to the ticket in which the calendar section is shown.
- **access.***
  Properties describing access to the calendar: credentials, connection details, URLs ... etc.
  - **access.type**
    String parameter. Possible values EXCHANGE (MS Exchange Server), RANDOM (randomly generated appointments for testing purposes).

    For access.type EXCHANGE:
    - **access.url**
      URL. The url of the .asmx file (Active Server Method File) on the Exchange server. The file is usually located in the EWS (Exchange Web Services) directory and provides the exchange access web service.
    - **access.username**
      String. The name of a technical user under which the access/login at the Exchange server is performed.

- **access.password**
  String. The password of the technical user under which the access/login at the Exchange server is performed.
- **access.domain**
  String. The Windows domain of the Exchange server. The technical user (used for access.username and access.password, see above) has to be a member of this domain.
- **access.impersonation**
  String. Email account of an Exchange calendar user. The calendar on the ticket page or on the customer page will be based on the view of this Exchange calendar user. For example, *somebody* could be the Exchange login of the ConSol CM engineer who is currently logged in. If the name of an engineer is the same in Exchange and ConSol CM, you can simply pass the current engineer's name to Exchange using the method `workflowApi.getCurrentEngineer().getName()` or `engineerService.current.name`.

For access.type RANDOM:

- **access.calendar**
  Name of the calendar file. Random calendar provider stores generated appointments on disk (files will be automatically removed when CM server is stopped). Using same name in configuration ensures that user will have the same set of appointments.

Additional variables available in the script:

- Ticket context: `ticket`
- Contact/customer context: `unit`

3. **Enable edit mode for the calendar:**
   To allow full access (i.e., to create and/or edit appointments) for engineers, set the page customization attribute *editable* to “true”.

Basic principle of Exchange server calendar access

The configured Exchange server is contacted, namely the Web Service indicated in the url. The login at this server is performed using the technical user (access.username, access.password). Then the user is changed to the person/user indicated under *access.impersonation*. The latter is performed using the Exchange server function *Impersonation*. The calendar of this impersonated user is the displayed on the ticket or customer page.



The impersonation function can only be used by an account which has been granted the *ApplicationImpersonation* role by the Exchange administrator. Please make sure all security aspects are taken into consideration when you set up this role!

Advanced Configuration: Page Customization Parameters for the Microsoft Exchange Calendar Integration

Short Background Information about the Microsoft Exchange Calendar Integration

The integration (more precisely, the *display*) of Microsoft Exchange calendars in ConSol CM is based on the jQuery `fullcalendar` plugin. For complete details on that API, please refer to the fullcalendar web site.

Pages for the Configuration of the calendarSection

The *Calendar* section can be configured on these pages:

- **Ticket page**
  Use *calendar/ticketEditPage/calendarSection*.
- **Customer pages**
  - **Contact page**
    Use *calendar/contactEditPage/calendarSection*.
  - **Company page**
    Use *calendar/companyEditPage/calendarSection*.
- **Resource pages**
  Use *calendar/resource/calendarSection*

In addition, the calendar can be displayed as a widget on the Web Client dashboard. See Calendar widgets for details.

Page Customization Attributes

The following attributes can be used to configure the appearance and behavior of the integrated Exchange calendar.

Attributes:

- **allDaySlot**
  Boolean. Determines if the all-day slot is displayed at the top of the calendar. Default “true”.
- **appointmentBackgroundColor**
  String. Sets the background color for all appointments in the calendar. You can use any CSS color format, such as “#f00”, “#ff0000”, “rgb(255,0,0)”, or “red”.
- **appointmentBorderColor**
  String. Sets the border color for all appointments in the calendar. You can use any CSS color format, such as “#f00”, “#ff0000”, “rgb(255,0,0)”, or “red”.
- **appointmentColor**
  String. Sets the background and border colors for all appointments in the calendar. You can use any CSS color format, such as “#f00”, “#ff0000”, “rgb(255,0,0)”, or “red”.
- **appointmentConstraint**
  String. Limits appointment dragging and resizing to certain windows of time.

  Possible values:

  - **<appointment_ID>**
    Appointments that are being dragged or resized must be fully contained by at least one of the appointments linked to by the given appointment ID.
  - **businessHours**
    Appointments being dragged or resized must be fully contained within the week's business hours (default: Monday-Friday 9am-5pm), see *businessHours* attribute.
  - **<start-time>-<end_time>;<days_of_week>**
    A custom time window in the same format as the *businessHours* attribute. Days of week are optional.
    Examples: “10:00-18:00; 1,2,3,4” or “10:00-18:00”
- **appointmentDurationEditable**
  Boolean. Allows appointments’ durations to be editable through resizing. Default “true”.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **appointmentOverlap**
  Boolean. Determines if appointments in the calendar, when dragged and resized, are allowed to overlap each other. Default “true”.
- **appointmentStartEditable**
  Boolean. Allows appointments’ start times to be editable through dragging. Default “true”.
- **appointmentTextColor**
  String. Sets the text color for all appointments in the calendar. You can use any CSS color format, such as “#f00”, “#ff0000”, “rgb(255,0,0)”, or “red”.
- **aspectRatio**
  String. Determines the width-to-height aspect ratio of the calendar. Default value “2.8”. If empty, the calendar component will use the internal default value 1.35.
- **businessHours**
  String. Emphasizes certain time slots in the calendar.
  Format: <start-time>-<end_time>;<days_of_week>
  Example: 10:00-18:00; 1,2,3,4 (from 10am to 6pm, Monday-Thursday)
- **calendarEventHandlerScript**
  String. Name of the script which handles calendar events. Besides standard context variables like `ticket`, there are additional ones:
  - **eventType**
    enum (values: CREATE, UPDATE, DELETE)
  - **appointment**
    with appointment data (uid, subject, location, etc.).
- **calendarInitializationScript**
  String. Name of the script which produces the calendar configuration. If the value is empty or the script returns “null” the calendar is not shown.
- **contentHeight**
  String. Makes the calendar’s content area this many pixels tall. By default, this option is not set and the calendar’s height is calculated based on the *aspectRatio* attribute.
- **defaultAllDayAppointmentDuration**
  String. A fallback duration for all-day appointments without a specified *end time* value. Default value “1” (one day).
- **defaultDate**
  String. The initial date displayed when the calendar first loads. Accepts an ISO8601 date string like “2020-02-01”.
- **defaultTimedAppointmentDuration**
  String. A fallback duration for timed appointments without a specified *end time* value. If not set, the default value is “02:00:00” (2 hours). This attribute also affects the default duration of appointments during creation.
- **DefaultView**
  String. Default calendar view.
  Possible values: month, basicWeek, basicDay, agendaWeek, agendaDay. Default “agendaWeek”.
  See Available Views for view examples.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **editable**
  Boolean. Whether the appointments can be created, dragged and resized. This value overwrites the source configuration. Default “false”.
- **firstDay**
  String. The day that each week begins with. (Sunday=0, Monday=1, Tuesday=2, etc.). If empty, the value is based on browser’s locale.
- **forceAppointmentDuration**
  Boolean. A flag to force calculation of an appointment’s end if it is unspecified. Default “false”.
- **handleWindowResize**
  Boolean. Whether to resize the calendar automatically when the browser window resizes. Default “true”.
- **headerCenter**
  String. Defines the buttons and title at the top/center of the calendar. See *headerLeft* description for details. Default “title”.
- **headerLeft**
  String. Defines the buttons and title at the top/left of the calendar. Comma- or space-separated list values (values separated by a comma are displayed adjacently). Default “prev,next today”.

  Possible values:

  - **title**
    Text for the current month/week/day.
  - **prev**
    Button for moving the calendar back one month/week/day.
  - **next**
    Button for moving the calendar forward one month/week/day.
  - **prevYear**
    Button for moving the calendar back one year.
  - **nextYear**
    Button for moving the calendar forward one year.
  - **today**
    Button for moving the calendar to the current month/week/day.
  - **<view name>**
    Button that will switch the calendar to any of the available views (see *defaultView* description for available views).

  The header is not shown if all three header attributes (*headerCenter*, *headerLeft*, *headerRight*) are empty.
- **headerRight**
  String. Defines the buttons and title at the top/right of the calendar. See *headerLeft* attribute description for details. Default “month,agendaWeek”.
- **height**
  String. Sets the height, in pixels, of the entire calendar (including header). By default, this option is not set and the calendar’s height is calculated based on the *aspectRatio* attribute.
- **hiddenDays**
  String. Excludes certain days of the week from being displayed. Comma separated list of day-of-week indices (Example: “1,3,5”). Each index is zero-based (Sunday=0) and ranges from 0-6.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **lazyFetching**
  Boolean. Determines when appointment fetching should occur. Setting this attribute to “false” makes sense when there are a lot of external changes to the user’s calendar. Default “true”.
- **maxTime**
  String. Determines the end time (exclusively) which will be displayed, even if the scrollbars have been scrolled all the way down. Default value is “24:00:00”.
- **minTime**
  String. Determines the starting time that will be displayed, even if the scrollbars have been scrolled all the way up. Default value is “00:00:00”.
- **nextDayThreshold**
  String. When an appointment’s end time spans into another day, the minimum time it must be in order for it to render as if it were on that day. Default: “09:00:00” (9am). Only affects timed appointments that appear on whole days. Whole day cells appear in *month view*, *basicDay*, *basicWeek* and the *all-day slots* in the agenda views.
- **readable**
  Boolean. Determines if the user can see the calendar entries. Set “true” to make the calendar entries visible and “false” if the user should not see the calendar entries. In the latter case, the calendar widget is still displayed. Only available if the calendar is displayed as a calendar widget on the dashboard.
- **rightToLeftMode**
  Boolean. If enabled, displays the calendar in right-to-left mode. Default “false”.
- **scrollTime**
  String. Determines how far down the scroll pane is initially scrolled. Default is “06:00:00” (6am).
- **slotAppointmentOverlap**
  Boolean. Determines whether timed appointments in agenda view should visually overlap. Default “true”.
- **slotDuration**
  String. The frequency for displaying time slots. Default is “00:30:00” (30 minutes).
- **snapDuration**
  String. The time interval at which a dragged appointment will snap to the agenda view time grid. Also affects the granularity at which selections can be made. Default value is set to *slotDuration*.
- **state**
  String. The visibility mode of the section, possible values are [*expanded, collapsed, collapsed_and_preload, hidden*]. Default: “expanded”. Only available if the calendar is displayed on the case, contact, or resource page.
- **weekNumbers**
  Boolean. Determines if week numbers should be displayed in the calendar. Default “true”.
- **weekends**
  Boolean. Whether to include Saturday/Sunday (i.e., weekend) columns in any of the calendar views. Default “true”.

See http://fullcalendar.io/docs for further details.

Defining Event Handlers for Appointment Events

Using the Page Customization attribute *calendarEventHandlerScript*, you can define actions which should be triggered in case a certain event has occurred. The events which can be used are:

- a new appointment has been created (event type CREATE)
- an existing appointment has been edited (event type UPDATE)
- an existing appointment has been removed (event type DELETE)

For example, you can define that when a new appointment is made, an email is automatically sent to all contacts of the ticket.

The following variables are available in a calendarEventHandlerScript:

- **ticket** (only in ticket context)
- **unit** (only in contact/company context)
- **eventType** - type of event, possible values are: CREATE, UPDATE or DELETE. It is an enum of type com.consol.cmweb.server.common.model.calendar.AppointmentEventType.
- **appointment** - appointment object (class `com.consol.cmweb.server.common.model.calendar.AppointmentVo`).
  Properties (Some properties may not be available. It depends on the type of calendar server and the respective version):

  Basic:

  - **subject**
    String. Subject/title of the appointment.
  - **startDate**
    Date. Start date/time of the appointment.
  - **endDate**
    Date. End date/time of the appointment.
  - **allDayEvent**
    Boolean. Defines whether an appointment is an all day event: which means that is lasts all day (or many days).
  - **location**
    String. Location/place of the appointment.
  - **meeting**
    Boolean. Is “true” when an appointment is a meeting. (MS Exchange Server specific property: means that attendees were invited appointment became a meeting)
  - **cancelled**
    Boolean. Indicates if an appointment is marked as *cancelled*.
  - **recurring**
    Boolean. Whether appointment is a part of recurring set.
  - **busyStatus**
    AppointmentVo.BusyStatus. Possible values: FREE, TENTATIVE, BUSY, OUT_OF_OFFICE, WORKING_ELSEWHERE or NONE.
  - **body**
    String. Body, content of the appointment. It can be text or HTML (depends on *bodyType* property)
  - **bodyType**
    AppointmentVo.BodyType. Possible values TEXT, HTML or NONE

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Advanced:

- **uid**
  String. Unique identifier of an appointment within calendar server.
- **start**
  org.joda.time.DateTime. Start date/time (joda)
- **end**
  org.joda.time.DateTime. End date/time (joda)
- **timeZone**
  org.joda.time.DateTimeZone. Timezone of the appointment (used to correctly show all-day appointments because days starting at different time in each timezone.

The following script shows an example of a `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 example 23: *Admin Tool script, example of a calendarEventHandlerScript*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

D.5 Classes of Text

This chapter discusses the following:

D.5.1 Introduction 299

D.5.2 Managing Classes of Text Using the Admin Tool 301

D.5.1 Introduction

A **class of text** is a classification that you assign to a ticket entry. This entry can be:

- a comment
- an email that was sent from the ticket
- an email that was received in the ticket
- an attachment

Assigning a class of text can serve one or more of the following purposes:

- Highlighting the text in the ticket with a special color to make it easier to find (e.g., an important note, as shown in the following figure). An icon can also be used for each class of text.
- Marking a ticket entry to make it visible in CM/Track, i.e., to make it available for customers who log in to the ConSol CM customer portal.
- Marking the entry to control the process flow, e.g., a ticket can only be finished when exactly one entry has been marked as *solution*.
- Marking the entry for hand-off to another process, e.g., the entries marked *question* and *answer* are automatically used for an FAQ ticket.

Thus, with classes of text you can organize ticket information within the ticket and can also control the process flow and the availability of information.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

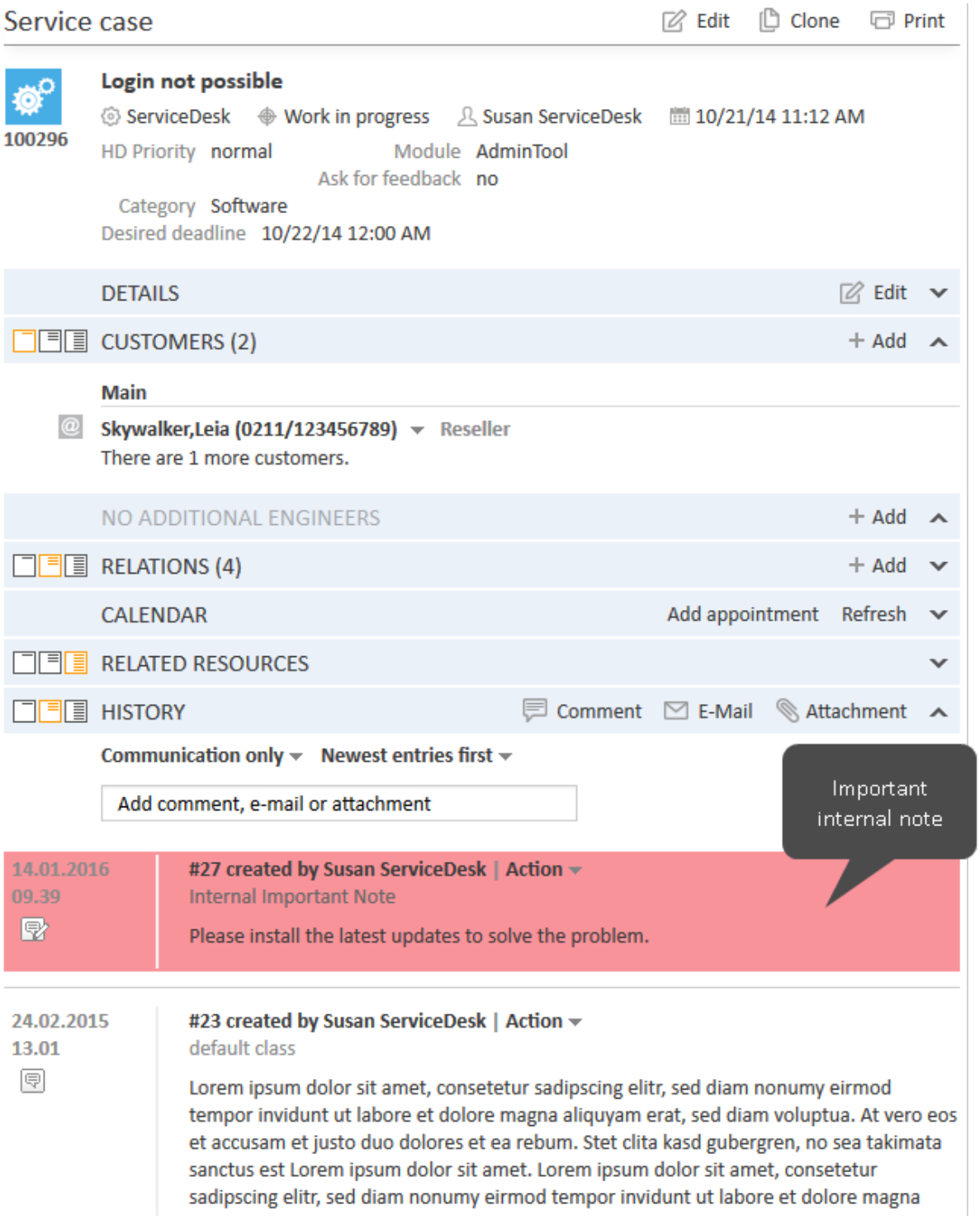


Figure 178: *ConSol CM Web Client - Using a class of text for an internal important note*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

D.5.2 Managing Classes of Text Using the Admin Tool

D.5.2.1 Installing a New Class of Text

Two steps are required to install a new class of text for tickets in a certain queue:

1. Defining the class of text in the navigation item *Classes of text*.
2. Assigning the class of text to the queue where it should be available for tickets (see section Queues for more information).

Defining a Class of Text

Classes of text are defined and managed in the corresponding navigation item in the navigation group *Global Configuration* in the Admin Tool (see the following figure).

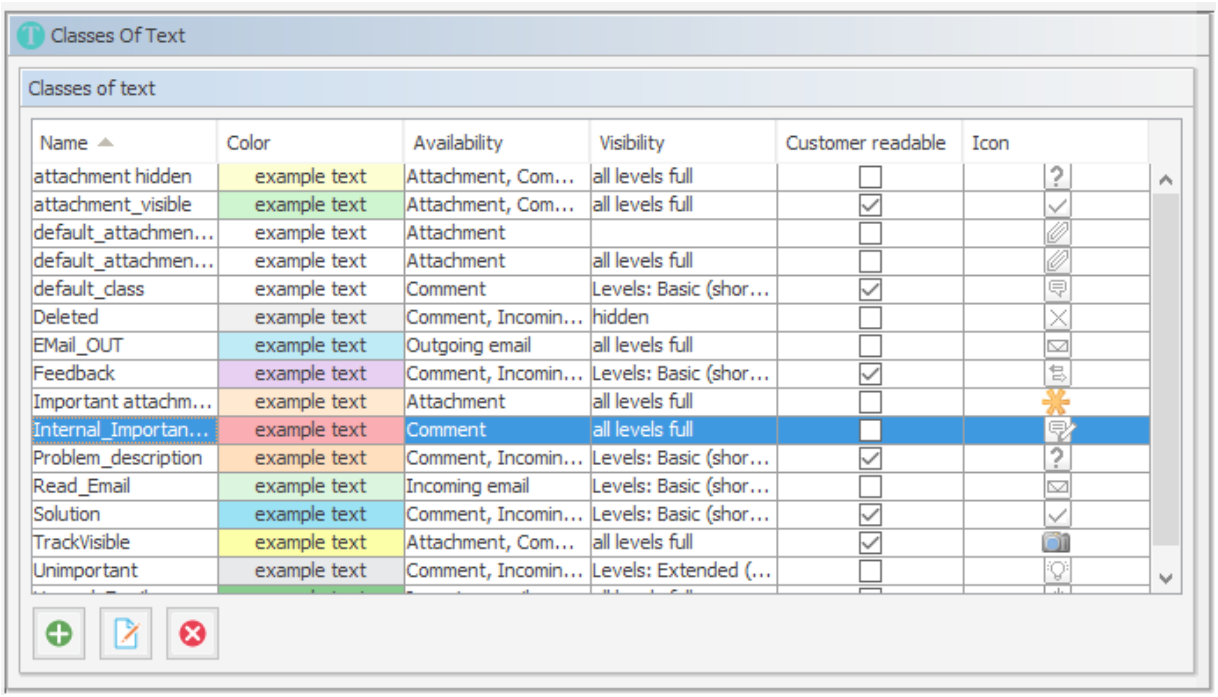


Figure 179: *ConSol CM Admin Tool - Global Configuration, Classes of Text*

You define a new class of text by clicking the *Add* button below the list. The following pop-up window appears:

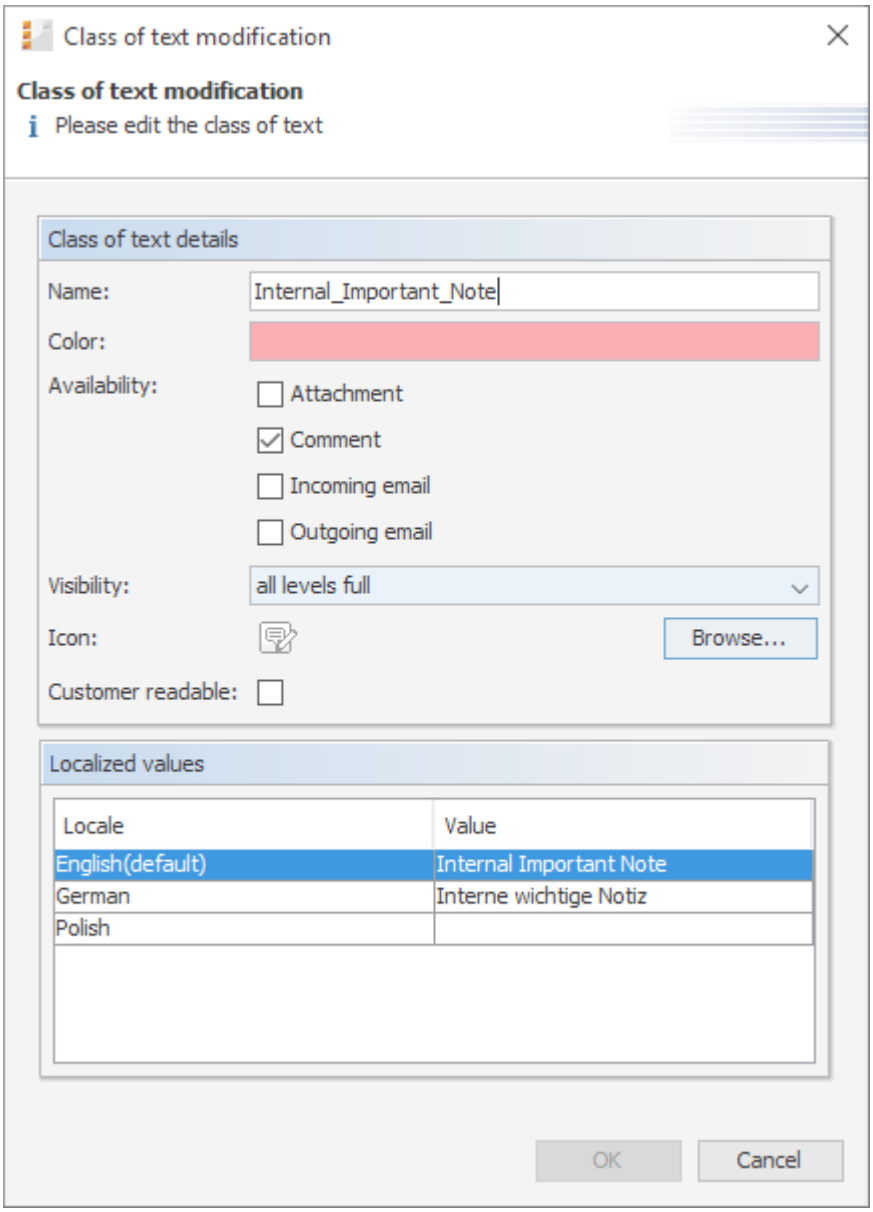


Figure 180: *ConSol CM Admin Tool - Global Configuration, Classes of Text: New Class of Text*

Here, you have to define the details of the class of text:

- **Name**
  Enter a name for the new class of text. The name must be unique.
- **Color**
  When you click into the *Color* field a pop-up window appears. It contains a range of colors from which you can choose the desired color for the class of text. You can see the selected color in the *Preview* area. Click *OK* to save your choice. Click *Reset* to return to the last saved color.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

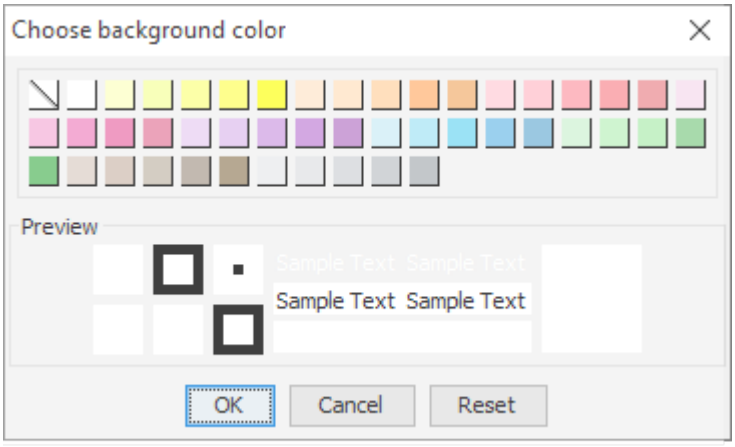


Figure 181: *ConSol CM Admin Tool - Global Configuration, Classes of Text: Choose a color*

- **Availability**
  You can choose here for which ticket information the class of text shall be available. Mark one or several of the following options:
  - Attachment
  - Comment
  - Incoming mail
  - Outgoing mail
- **Visibility**
  Sets the visibility level for comments and emails, is not applied to attachments. See also General Information about the Visibility of Ticket History Entries in the Web Client

  There are three ticket history levels in the Web Client:

  - Basic (1st level)
  - Extended (2nd level)
  - Detail (3rd level)

  Thus, the configuration of a class of text concerning one or more of those levels determines whether a text entry marked with this class of text is displayed at a given level.

The addition *short* and *full* indicates whether entries are shown **within** the chosen visibility level in full length (*full*) or are truncated after a certain number of characters (*short*):

- **short** - shortened
- **full** - full length

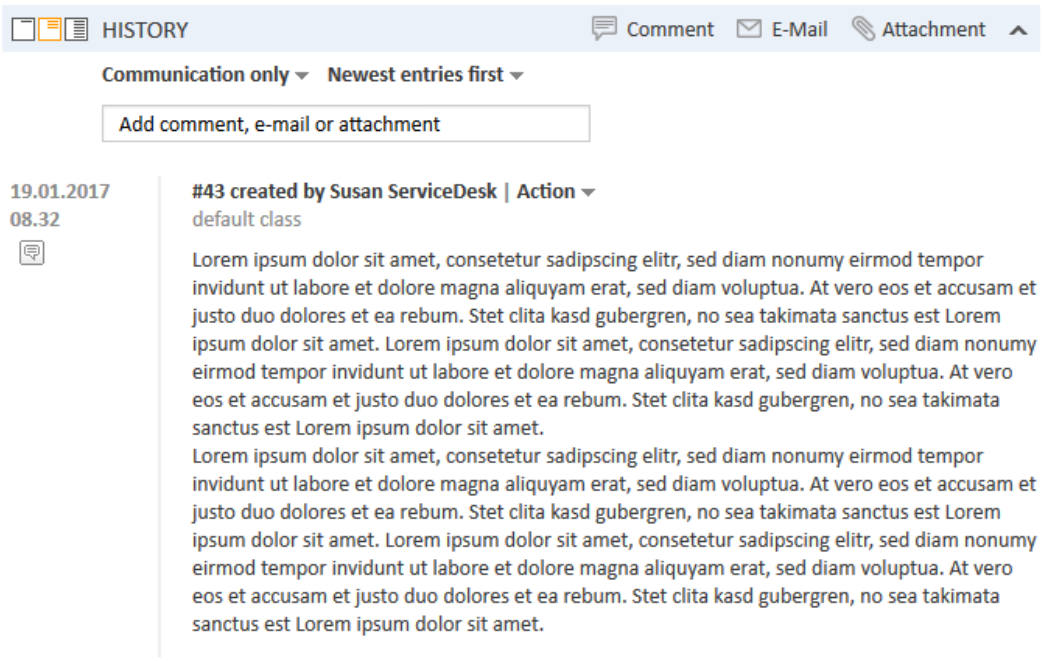


Figure 182: *ConSol CM Web Client - Example: Visibility level "Extended" (2nd level) shows entries in full length ("full")*

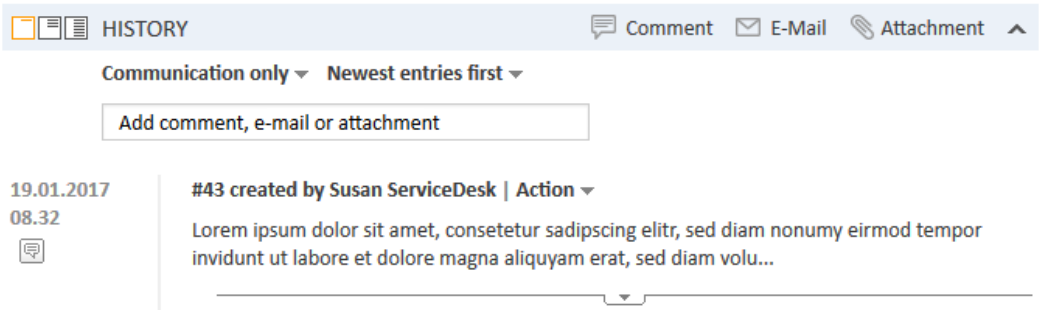


Figure 183: *ConSol CM Web Client - Example: Visibility level "Basic" (1st level) shows entries shortened ("short")*

Select, in the drop-down menu, the history levels at which the marked ticket information should be visible (see picture below).

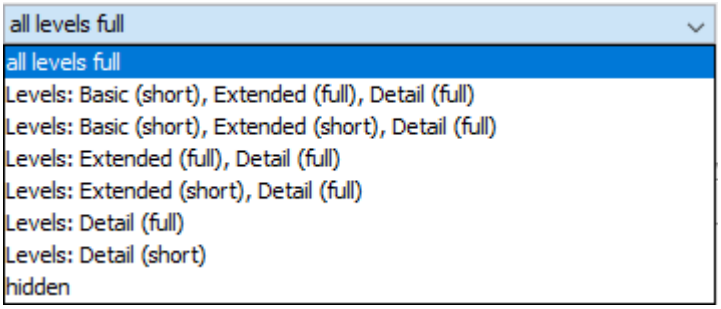


Figure 184: *ConSol CM Admin Tool - Global Configuration, Classes of Text: Choose a visibility level*

If you choose *hidden*, the ticket History entry marked by this class of text will not be visible in the ticket history. For a detailed explanation of visibility level and display mode for ticket History entries, please refer to section General Information about the Visibility of Ticket History Entries in the Web Client.

- **Icon**
  When you click the box next to *Icon* you will get a selection of standard ConSol CM icons. Select one of these icons for the new class of text or load your own individual icon by clicking the *Browse...* button. An icon for a class of text must have the size 16 (height) * 24 (width) px to be correctly placed and aligned in the drop-down menu in the Web Client.



Figure 185: *ConSol CM Admin Tool - Global Configuration, Classes of Text: Choose an icon*

- **Customer readable**
  Select this checkbox if ticket information marked with this class of text shall be visible for customers in CM/Track, the ConSol CM customer portal.
- **Localized values**
  You can localize the name of a class of text. Enter the corresponding class name in the *Value* field for each additional language. A detailed explanation of the localization mechanism is given in section Localization of Objects in General, Type 2.

Click *OK* to save the details of the new class of text and to close the window.

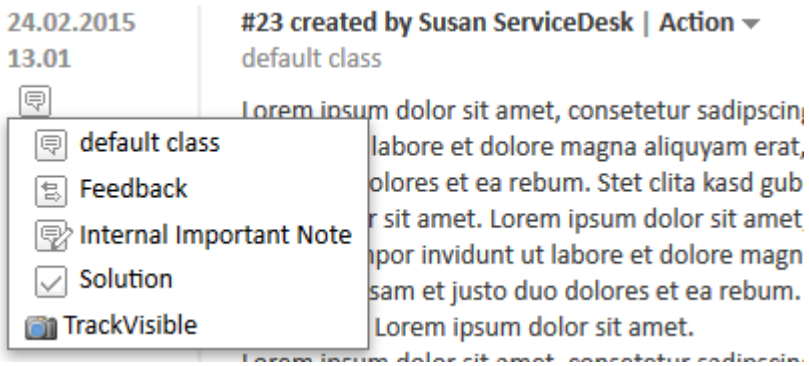


Figure 186: *ConSol CM Web Client - Context menu for classes of text in ticket history section*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Assigning the Class of Text to a Queue

After assigning the class of text to a queue within the Queues it will be available for tickets of this queue in the Web Client.

D.5.2.2 Editing a Class of Text

If you want to edit a class of text, select it in the list and click the *Edit* button. The same window as described above for creating a class of text will appear. You can modify all details and save your changes by clicking *OK*.

D.5.2.3 Deleting a Class of Text

You can only delete a class of text if it is not used within any tickets and if it is not assigned to a queue. In order to delete a class of text select it in the list and click the *Delete* button. If you confirm the following dialog with *Yes*, the class of text will be removed from the list and the system.

D.5.2.4 Setting the Default Class of Text

To define the default class of text, use the system property cmweb-server-adapter, defaultContentEntryClassName (see System Properties). The default class of text will be applied to any ticket entry which is not explicitly marked with another class of text.

Depending on the visibility configuration of this class of text, the regular comments in the ticket (i.e., the comments which have the class of text *default class*) will be displayed, see section visibility.

D.5.2.5 Working With Classes of Text in Scripts

You can work with classes of text in scripts which are used in workflow activities or which are located in the *Script* section of the Admin Tool. For details about programming, please read the *ConSol CM Process Designer Manual*.

In scripts, you can assign a class of text to a TextEntry even if the class of text is not assigned to the queue! This can help automate certain processes.

D.5.2.6 Transferring Classes of Text to the DWH

All classes of text (default classes and special ones) used in history entries (comments, emails) are transferred from ConSol CM to the data warehouse (DWH) automatically in all transfer modes (LIVE and ADMIN). There is no configuration required for this feature, and the installation is upgraded automatically. A new initialization and initial transfer are required, however.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E - GUI Design and Advanced Customization

ConSol CM provides some features which allow an advanced customization of the Web Client. This includes the possibility to determine which components should be displayed on each page and how the Web Client should look like.

The following mechanisms exist for the Web Client:

- GUI texts can be adapted. Please see Labels.
- The relations sections on ticket, customer, and resource pages can be configured to show a graphical representation of the relations. Please see Configuration of the Graphical Representation of Relations.
- The page customization allows to configure different aspects of the Web Client pages. For example, it can be used to determine which sections should be displayed on a page, or to fine-tune the behavior of page components. The page customization settings are managed per page. Please see Page Customization for a detailed description of the available configuration options.
- Dashboards can be displayed in two places of the Web Client: the overview page and the resource dashboard. You can determine which kinds of widgets you want to display on the dashboards. Please see Dashboards for detailed information about using dashboards.
- Skinning can be used to determine the layout of the Web Client. For example, you can adapt the colors to your corporate design and use your company's logo in the Web Client header. Please see Layout Customization Using Skinning for a detailed description.

In addition, it is possible to customize some aspects of CM/Track. Please see Design and Configuration of REST-based ConSol CM Client GUIs for further information.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.1 Labels

This chapter discusses the following:

E.1.1 Introduction 308

E.1.2 Configuring Labels Using the Admin Tool 308

E.1.1 Introduction

It is possible to manipulate labels which are displayed on the GUI, i.e. in the Web Client. In this way, you, as an administrator, can customize the graphical user interface according to your company's requirements concerning terminology and/or Corporate Design. For example, in your company , it might be required to talk about cases instead of tickets, and a queue should be referred to as process. All this can be manipulated using the label settings in the Admin Tool.

E.1.2 Configuring Labels Using the Admin Tool

In the Admin Tool, labels are configured on the navigation item *Labels* in the navigation group *Global Configuration*.

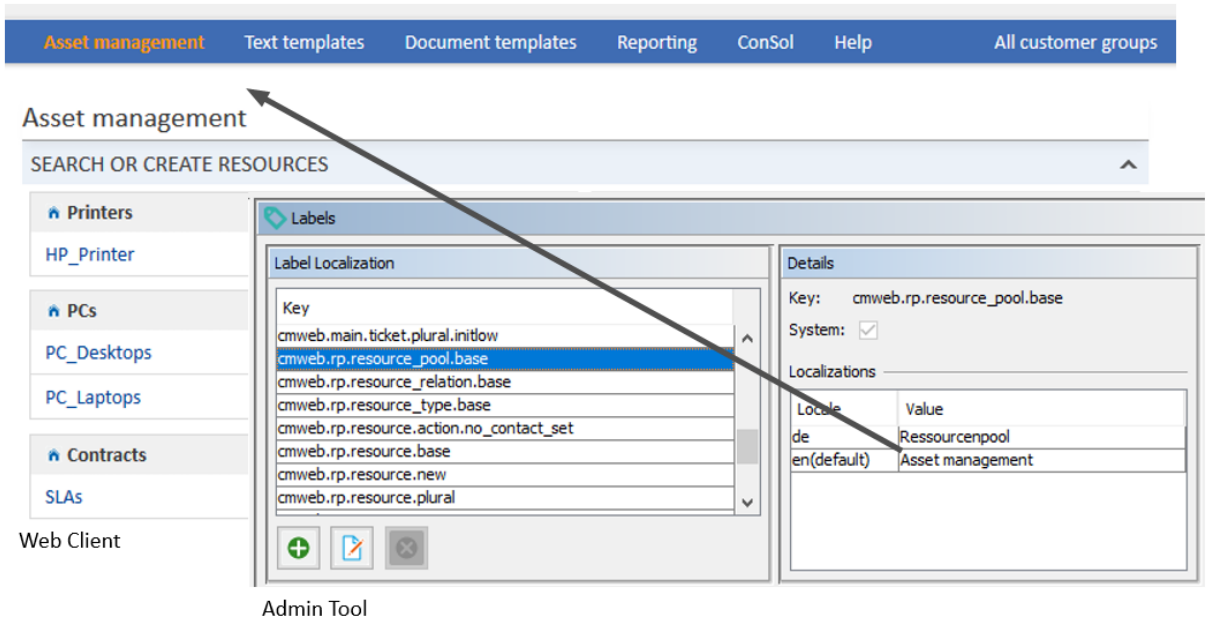


Figure 187: *ConSol CM Admin Tool - Global Configuration: Labels*

The following labels are currently available for tickets, engineers, and resources:

Label	GUI location	Default value
cmweb.main.engineer.base.initcap	Term for one engineer, capitalized. Used, e.g., in ticket header section.	Assignee
cmweb.main.engineer.base.initlow	Term for one engineer.	assignee

Label	GUI location	Default value
cmweb.main.engineer.plural.initcap	Term for more than one engineers, capitalized.	Assignees
cmweb.main.engineer.plural.initlow	Term for more than one engineers.	assignees
cmweb.main.queue.base.initcap	Term for one queue, capitalized. Used, e.g., in ticket list filter.	Process
cmweb.main.queue.base.initlow	Term for one queue.	process
cmweb.main.queue.plural.initcap	Term for more than one queue, capitalized.	Processes
cmweb.main.queue.plural.initlow	Term for more than one queue.	processes
cmweb.main.ticket.article.dative.initcap	Demonstrative pronoun (dative) for one ticket, capitalized	This
cmweb.main.ticket.article.dative.initlow	Demonstrative pronoun (dative) for one ticket	this
cmweb.main.ticket.article.demonstrative.initcap	Demonstrative pronoun (nominative) for one ticket, capitalized	This
cmweb.main.ticket.article.demonstrative.initlow	Demonstrative pronoun (nominative) for one ticket	this
cmweb.main.ticket.article.initcap	Article for one ticket, capitalized	The
cmweb.main.ticket.article.initlow	Article for one ticket	the
cmweb.main.ticket.base.initcap	Term for one ticket, capitalized	Case
cmweb.main.ticket.base.initlow	Term for one ticket	case
cmweb.main.ticket.new	Entry in main menu for "Create new ticket"	Create case
cmweb.main.ticket.plural.initcap	Term for more than one ticket, capitalized	Cases
cmweb.main.ticket.plural.initlow	Term for more than one ticket	cases
cmweb.main.ticket.relations.fixed.child	Term for the ticket on the lower hierarchical level in a relation which can only be set using scripts	Child (fixed)

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Label	GUI location	Default value
cmweb.main.ticket.relations.fixed.parent	Term for the ticket on the upper hierarchical level in a relation which can only be set using scripts	Parent (fixed)
cmweb.main.ticket.relations.lax.child	Term for the ticket on the lower hierarchical level in a relation which can be set in the Web Client	Child
cmweb.main.ticket.relations.lax.parent	Term for the ticket on the upper hierarchical level in a relation which can be set in the Web Client	Parent
cmweb.main.ticket.relations.reference.plural	Term for tickets in a non-hierarchical relation	References
cmweb.main.ticket.relations.reference.singular	Term for a ticket in a non-hierarchical relation	Reference
cmweb.rp.resource.base	Title for the resource page and for the resource list on the resource type page	Resource
cmweb.rp.resource.new	Entry for "Create a new resource"	New resource
cmweb.rp.resource.plural	Term for more than one resources	Resources
cmweb.rp.resource.to	Used for the context menu Jump to ... (resource)	Jump to resource
cmweb.rp.resource_pool.base	Entry in main menu.	Resource pool
cmweb.rp.resource_relation.base	Expression for resource relation.	Resource relation
cmweb.rp.resource_type.base	Expression for resource type, e.g. on Detail Search page for resources.	Resource type

The terms used for the objects in the customer data model (*customer*, *company*, and *contact*) can be configured as well.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Starting with ConSol CM version 6.11.2.0, the default values for the GUI terms have changed for new installations. The following terms are now used in the Web Client:

Definition	Old wording	New wording
General term for the unit entity in the system	Customer	Contact
Unit on the first level of a two-level unit definition or unit of a one-level unit definition	Contact	Person
Unit on the second level of a two-level unit definition. Can include several units from the first level.	Company	Company
Generic description of a ticket-unit relation	Customer role	Contact role
Primary unit related to a ticket	Main customer	Main contact
Additional unit related to a ticket	Additional customer	Additional contact

For updates from a previous version of ConSol CM to ConSol CM version 6.11.2.0, the previous values are still used by default.

The following table contains a list of the available labels with the default values for update installations (*Value on update*) and new installations (*Value on setup*):

Label	Value on update	Value on setup
cmweb.main.unit.additionalcontact.base.initcap	Additional customer	Additional contact
cmweb.main.unit.additionalcontact.plural.initcap	Additional customers	Additional contacts
cmweb.main.unit.company.article.initcap	This	This
cmweb.main.unit.company.base.initcap	Company	Company
cmweb.main.unit.company.base.initlow	company	company
cmweb.main.unit.company.jump.to	Jump to company	Jump to company
cmweb.main.unit.company.plural.initlow	companies	companies
cmweb.main.unit.company.tickets.filter.all.text	of this company and its contacts	of this company and its persons

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Label	Value on update	Value on setup
cmweb.main.unit.company.tickets.filter.all.tooltip	which are relevant for this company. Either the company itself or one of its contacts are main or additional customer	which are relevant for this company. Either the company itself or one of its persons are main or additional contact
cmweb.main.unit.company.tickets.filter.main.text	of this company (only as main contact)	of this company (only as main contact)
cmweb.main.unit.company.tickets.filter.main.tooltip	where this company is assigned as main customer	where this company is assigned as main contact
cmweb.main.unit.company.tickets.filter.ofcompany.text	of this company	of this company
cmweb.main.unit.company.tickets.filter.ofcompany.tooltip	where this company is assigned as main or additional customer	where this company is assigned as main or additional contact
cmweb.main.unit.company.tickets.filter.ofcontacts.text	of contacts	of persons
cmweb.main.unit.company.tickets.filter.ofcontacts.tooltip	which carry one of this company's contacts as main or additional customer	which carry one of this company's persons as main or additional contact
cmweb.main.unit.company.transfer.select.text	select another company	select another company
cmweb.main.unit.contact.base.initcap	Contact	Person
cmweb.main.unit.contact.base.initlow	contact	person
cmweb.main.unit.contact.jump.to	Jump to contact	Jump to person
cmweb.main.unit.contact.plural.initcap	Contacts	Persons
cmweb.main.unit.contact.plural.initlow	contacts	persons
cmweb.main.unit.contact.tickets.filter.all.text	of this contact	of this person

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Label	Value on update	Value on setup
cmweb.main.unit.contact.tickets.filter.all.tooltip	where this contact is assigned as main or additional customer	where this person is assigned as main or additional contact
cmweb.main.unit.contact.tickets.filter.main.text	of this contact (only as main customer)	of this person (only as main contact)
cmweb.main.unit.contact.tickets.filter.main.tooltip	where this contact is assigned as main customer	where this person is assigned as main contact
cmweb.main.unit.contact.tickets.filter.ofcompany.text	of the company	of the company
cmweb.main.unit.contact.tickets.filter.ofcompany.tooltip	which are relevant for this person's company. Either the company itself or one of its contacts are main or additional customer	which are relevant for this person's company. Either the company itself or one of its persons are main or additional contact
cmweb.main.unit.contact.transfer.select.text	select another contact	select another person
cmweb.main.unit.general.article.initcap	This	This
cmweb.main.unit.general.base.initcap	Customer	Contact
cmweb.main.unit.general.base.initlow	customer	contact
cmweb.main.unit.general.create	New customer	New contact
cmweb.main.unit.general.plural.initcap	Customers	Contacts
cmweb.main.unit.general.plural.initlow	customers	contacts
cmweb.main.unit.maincontact.base.initcap	Main customer	Main contact
cmweb.main.unit.maincontact.base.initlow	main customer	main contact
cmweb.main.unit.role.general.initcap	Customer role	Contact role

The terms which are used in the column names of DWH cubes generated using the ConSol CM API are set in the following labels:

Label	Default English	Default German	Usage
cmrf.cube.ticket.main	Case	Vorgang	Columns for basic ticket data and ticket fields
cmrf.cube.queue.main	Process	Prozess	Columns for the technical and localized queue name
cmrf.cube.scope.main	Scope	Bereich	Columns for the technical and localized scope name
cmrf.cube.engineer.main	Assignee	Bearbeiter	Columns for the data of the assigned user
cmrf.cube.maincontact.main	Contact	Kontakt	Columns for the contact ID and contact fields
cmrf.cube.company.main	Company	Firma	Columns for the company ID and company fields
cmrf.cube.customergroup.main	Customer group	Kundengruppe	Columns for the technical and localized customer group name

E.1.2.1 For CM Programmers: Defining Labels for Messages

You can also use labels to define error or information messages which should be displayed in the Web Client in certain situations. In this way, you do not have to use only standard CM properties but you can define very specific messages. The following three figures provide an example.

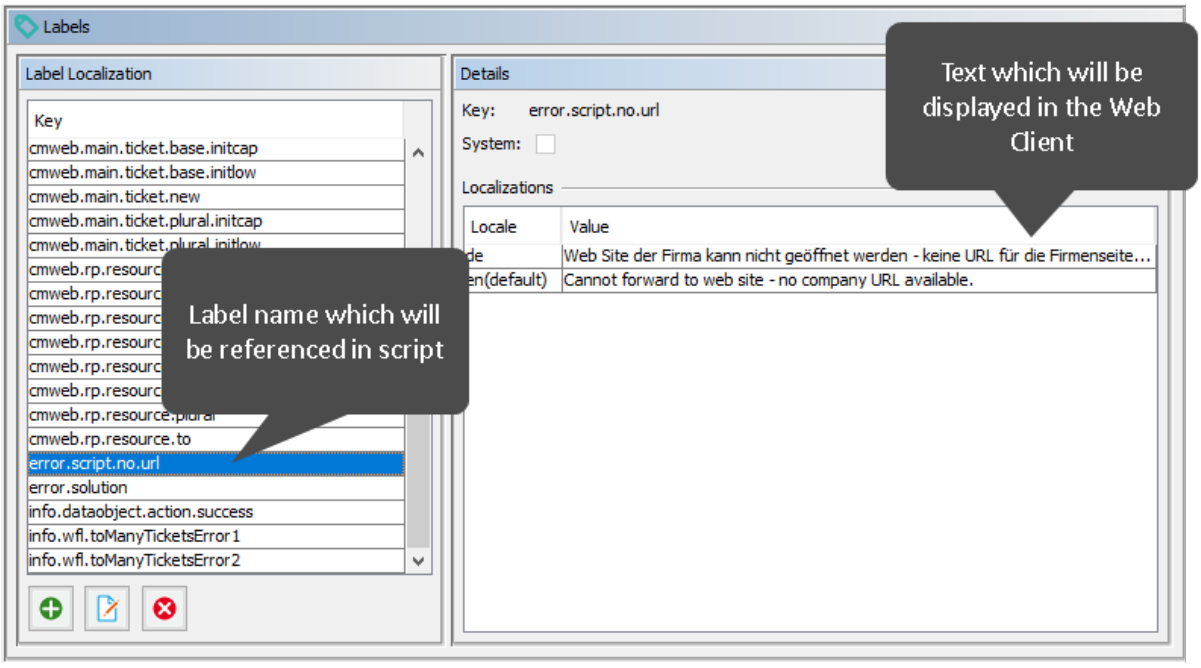


Figure 188: *ConSol CM Admin Tool, Global Configuration, Labels - System-specific label used for an error message (1): Label definition*

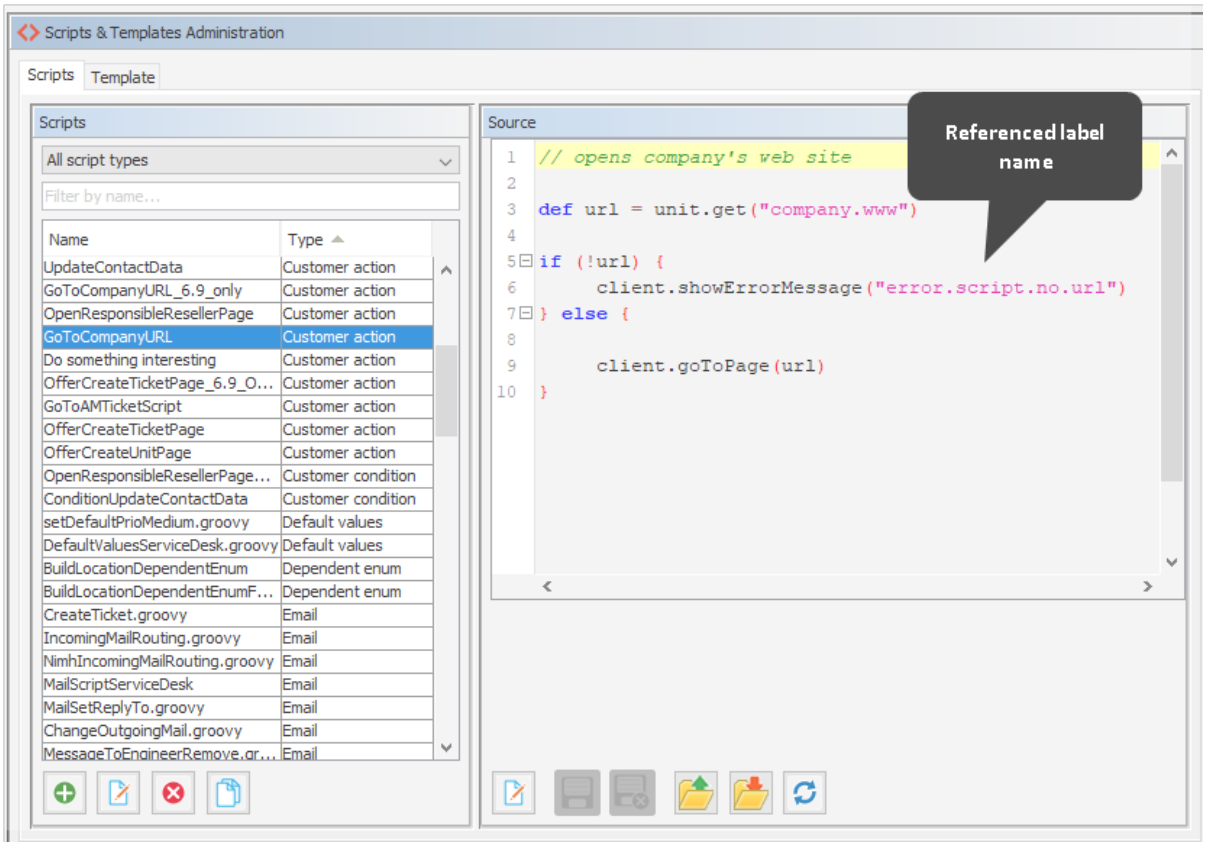


Figure 189: *ConSol CM Admin Tool - System, Scripts and Templates: System-specific label used for an error message (2): Script where the label is referenced*

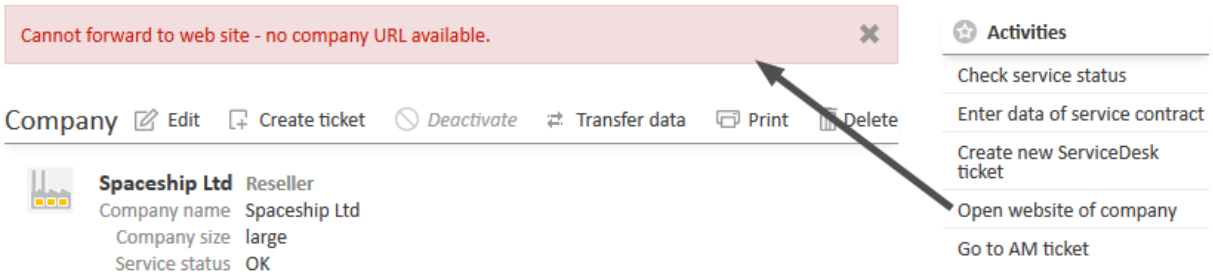


Figure 190: *ConSol CM Web Client - System-specific label used for an error message (3): Display in the Web Client*

You can also work with the `MessageProviderService`. This class provides a very generic way of using labels, not only for messages, but also for other GUI elements like names in charts or other objects. As you know from other services, a singleton of the class `MessageproviderService` is potentially available as `messageProviderService` object in all scripts.

You might need the method `getMessage(String pKey)`. This takes the key of a label as parameter. As a result, the localized value of the label is displayed, according to the default system locale.

Examples, taken from a widget script:

```
 //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")
```

In case you want to display the message in a specific locale, you can work with the other signature of the method `getMessage`:

`getMessage(String pKey, Locale pLocale)`. This can also be used for another locale than the standard one.

Example, taken from a workflow script which checks the engineer workload. Here, the standard locale of the engineer who is logged in is used.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

```
// 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 example 24: *Script of activity "New IT ticket (Accept ticket)", engineer workload is checked*

A message like this might be displayed in this case:

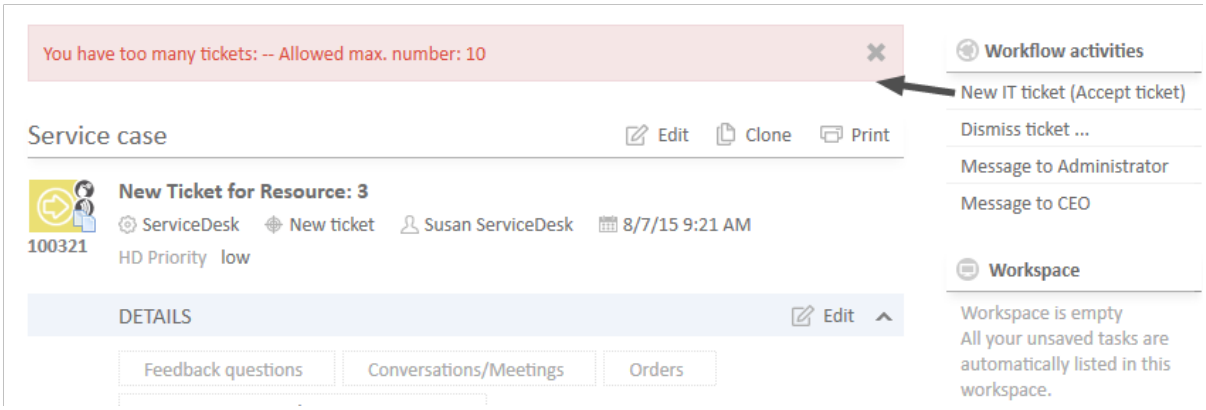


Figure 191: *ConSol CM Web Client - Message for English browser locale if the engineer has too many tickets to accept another one*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.2 Configuration of the Graphical Representation of Relations

E.2.1 Configuration of the Graphical Display in Standard Mode

In several locations of the ConSol CM Web Client, relations between objects can be established and are displayed:

- On the ticket page
  - Relationen of the ticket to one or more customers
  - Relations of the ticket to other tickets (references or parent/child relations)
  - Relations of the ticket to one or more resources (if CM/Resource Pool is active)
- On the company page:
  - Relations of the company to tickets
  - Relation of the company to (its) contacts
  - Relations of the company to other customers (contacts, companies)
  - Relations of the company to one or more resources (if CM/Resource Pool is active)
- On the contact page:
  - Relations of the contact to tickets
  - Relations of the contact to other customers (contacts, companies)
  - Relations of the contact to one or more resources (if CM/Resource Pool is active)

If CM/Resource Pool is active:

- On the resource page:
  - Relations of the resource to one or more other resources
  - Relations of the resource to customers (contacts, companies)
  - Relations of the resource to tickets

The relations can be displayed as list and as graph. The list display is activated per default and cannot be switched off. The graph view can be additionally activated using **page customization**. If it is activated, the engineer can select between list and graph view.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

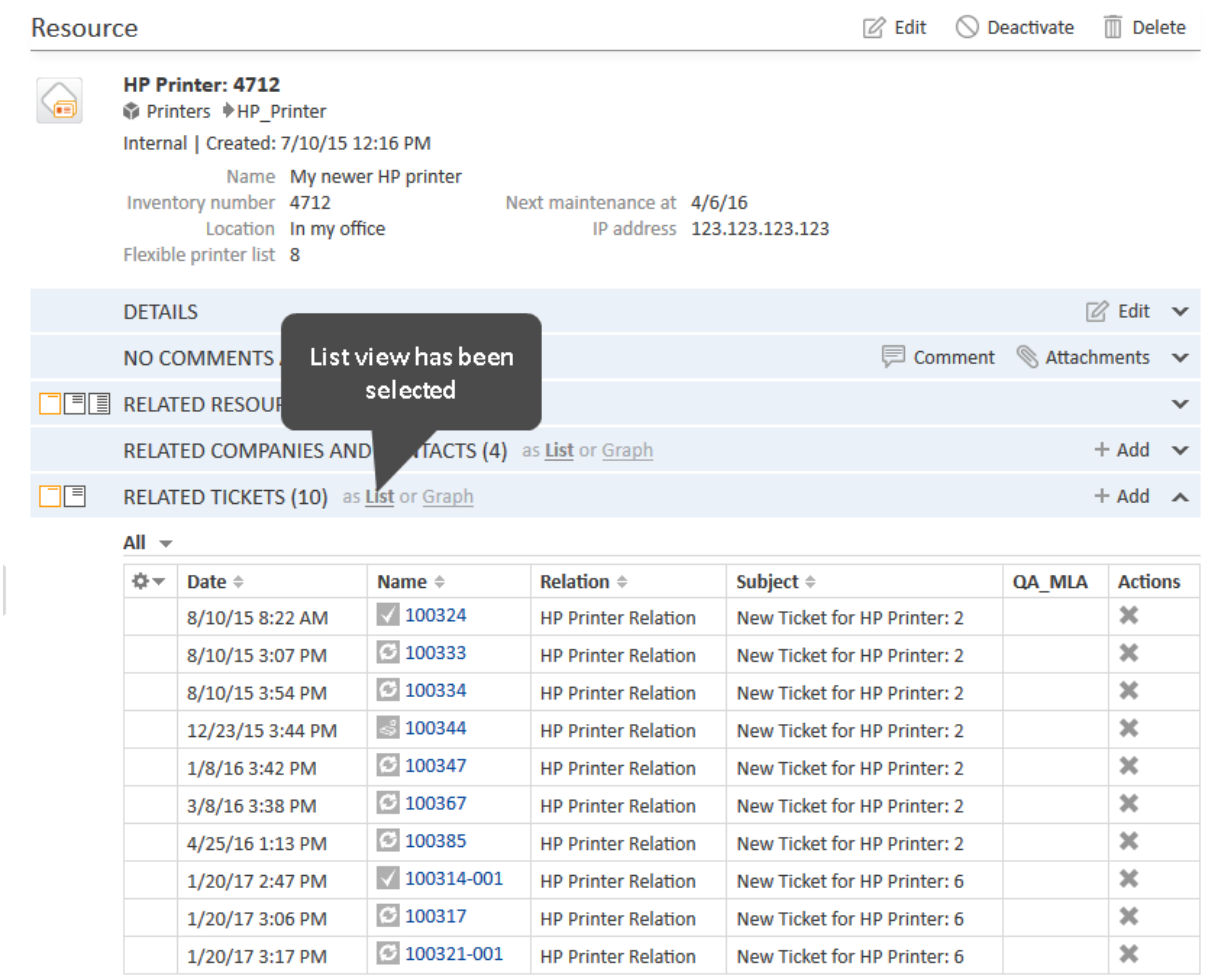


Figure 192: *ConSol CM Web Client - Resource page of a printer: Related tickets section in list mode*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

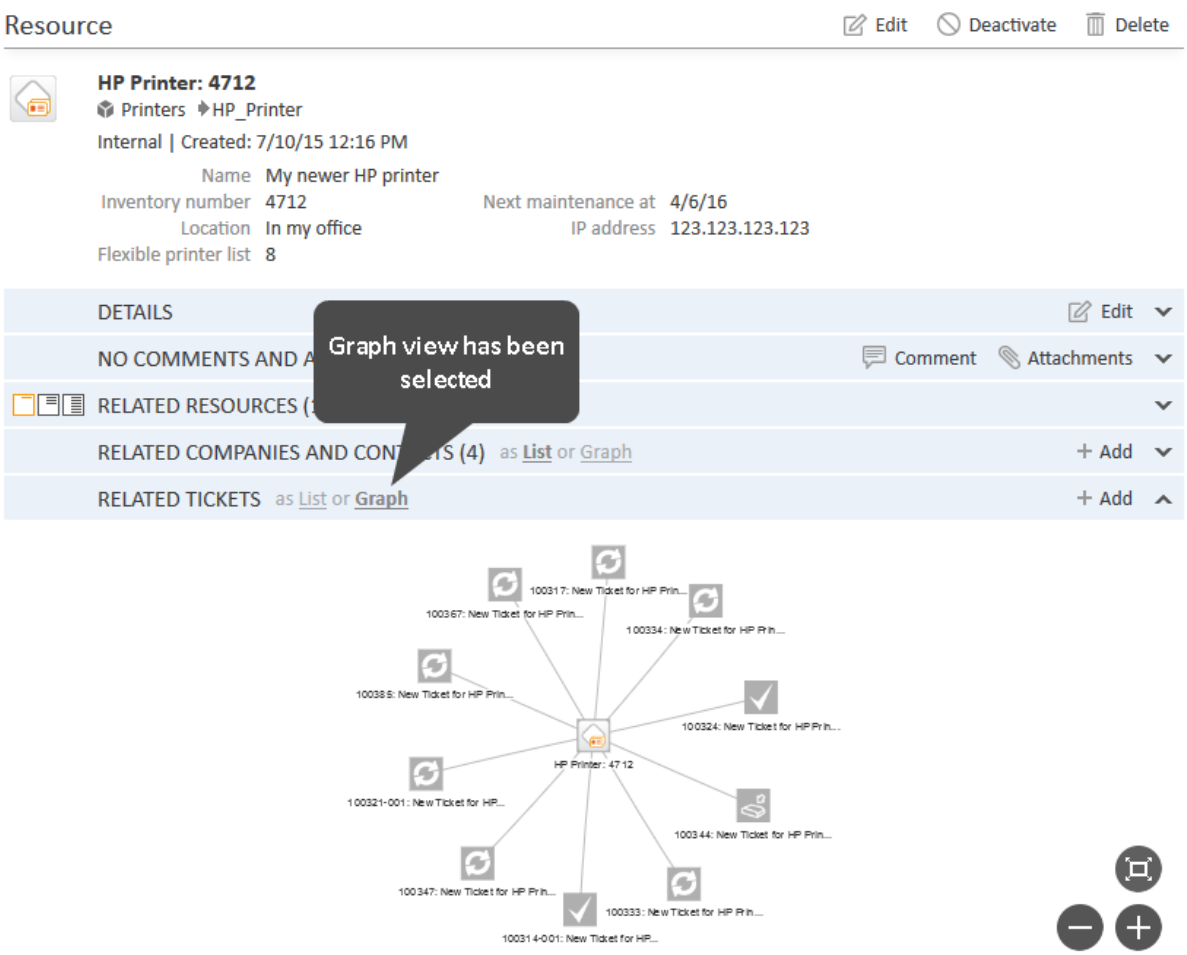


Figure 193: *ConSol CM Web Client - Resource page of a printer: Related tickets section in graph mode*

The graph view has to be activated for each section (see list above) separately. It is configured using page customization attributes. Please see section Page Customization for the Graphical Representation of Relations for a detailed explanation of the configuration.

E.2.2 Configuration of More Relation Displays - Expert Mode

In addition to the standard relations sections in ticket, customer, and resource pages, an additional section can be displayed on each of these pages. In this section, a relations graph can be displayed which is implemented specifically for the system and is based on a script. Please see section Page Customization for the Graphical Representation of Relations for a detailed explanation of the configuration.

E.3 Page Customization

This chapter discusses the following:

E.3.1 General Introduction to Page Customization 323
E.3.2 Page Customization in the Web Client 324
E.3.3 Order and Priorities of Page Customization 333
E.3.4 Page Customization Using Attributes 334
E.3.5 Page Customization Attributes for Types, Scopes and Subscopes, in Alphabetical Order 343
E.3.6 Page Customization for the Graphical Representation of Relations 395

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.3.1 General Introduction to Page Customization

In addition to the design of the Web Client GUI in the process of defining ticket fields (see section Ticket Fields (Setting Up the Ticket Data Model)), customer fields (see section Customer Field Management and GUI Design for Customer Data), and maybe resource fields (see section CM/Resource Pool - Setting Up the Basic Resource Model) an administrator can configure more GUI layout features and functionality using **page customization**. The configuration is performed by assigning values to *attributes*.

When you log in to the Web Client as an administrator, you see the item *Enable page customization* in the main menu. Depending on the context, i.e., on the page that is currently displayed, the page customization offers different, page- and context-specific functionality.

For example, when you have opened a ticket and start the page customization, you can configure attributes for the ticket in general. When the Ticket Email Editor is open, you can also configure email editor-specific attributes.

In the following sections, the general principle of page customization and all available page customization attributes are described and explained in detail. In all other sections of this *ConSol CM Administrator Manual,* references to this sections will be included where required.



The page customization settings are part of the ConSol CM configuration data. You can export and import them using the Admin Tool, as described in section Deployment (Import/Export). Only the page customizations which have been defined specifically for the system (i.e. not the default values) are exported. These are the values which are stored in the database tables `cmas_web_customization` and `cmas_web_customization_values`. Default values are exported only if they were set to another value and have been changed back to the default value manually by clicking the *Update* button.

E.3.2 Page Customization in the Web Client

When you start the page customization mode, the *Page Customization Definition Section* is displayed in the lower half of the GUI. On the right side you see a tree that reflects the structure of the current page. Within the page, every element of the page is marked by a blue border. When you move the mouse over an element, its name is displayed. When you click on this name, the definition section for this element is opened, and the element is marked in the tree. In this way, you can easily identify the component you would like to modify.

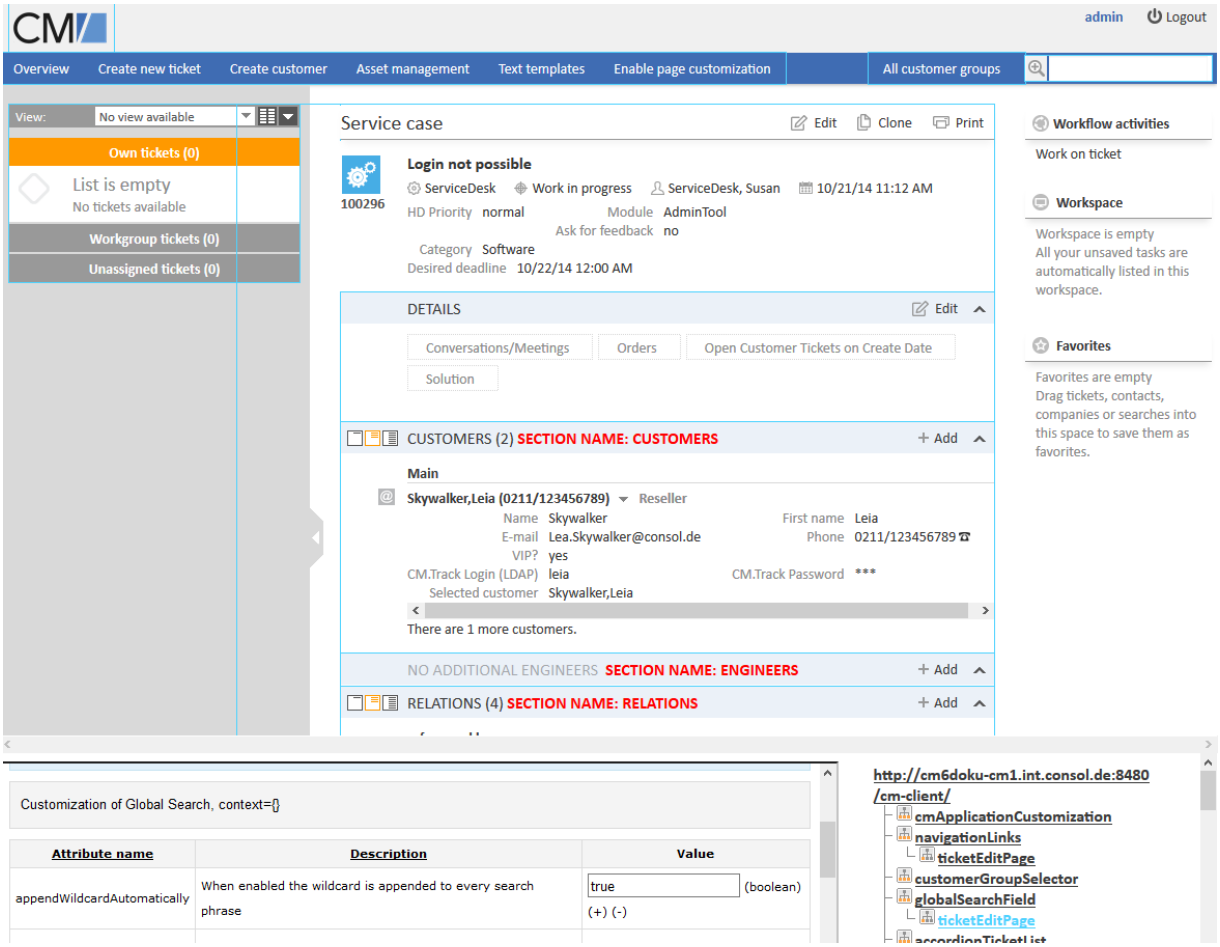


Figure 194: *ConSol CM Web Client - Page Customization Definition Section*

The tree might display the following elements (see next figure). Since the Page Customization Definition Section is rather small you might have to scroll to see all elements. In this example (see figure above), the administrator has opened the *ticketEditPage* (see following paragraphs for details).

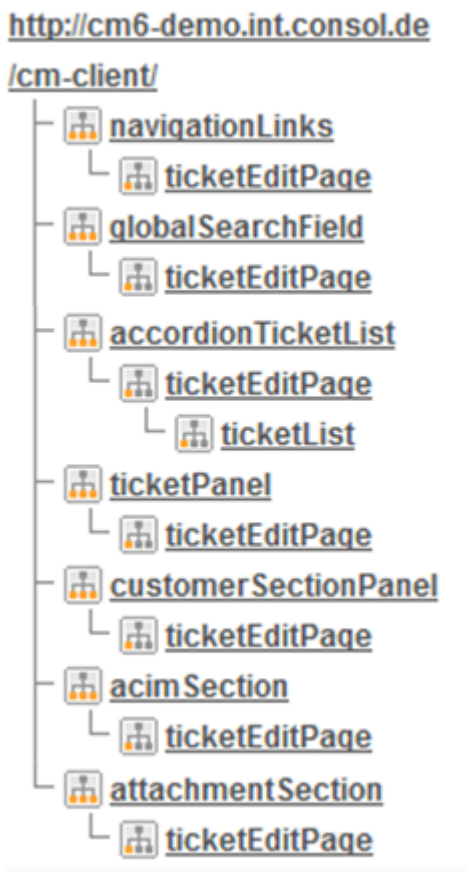


Figure 195: *ConSol CM Web Client - Page Customization tree*

Icon	Description
	Configuration of all elements of this type
	Configuration of this specific element deployed within the identified scope

You can also click on an element name in the tree to open the editor for this element in the left area of the Definition Section, e.g., for the element *navigationLinks* (see following figure).

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

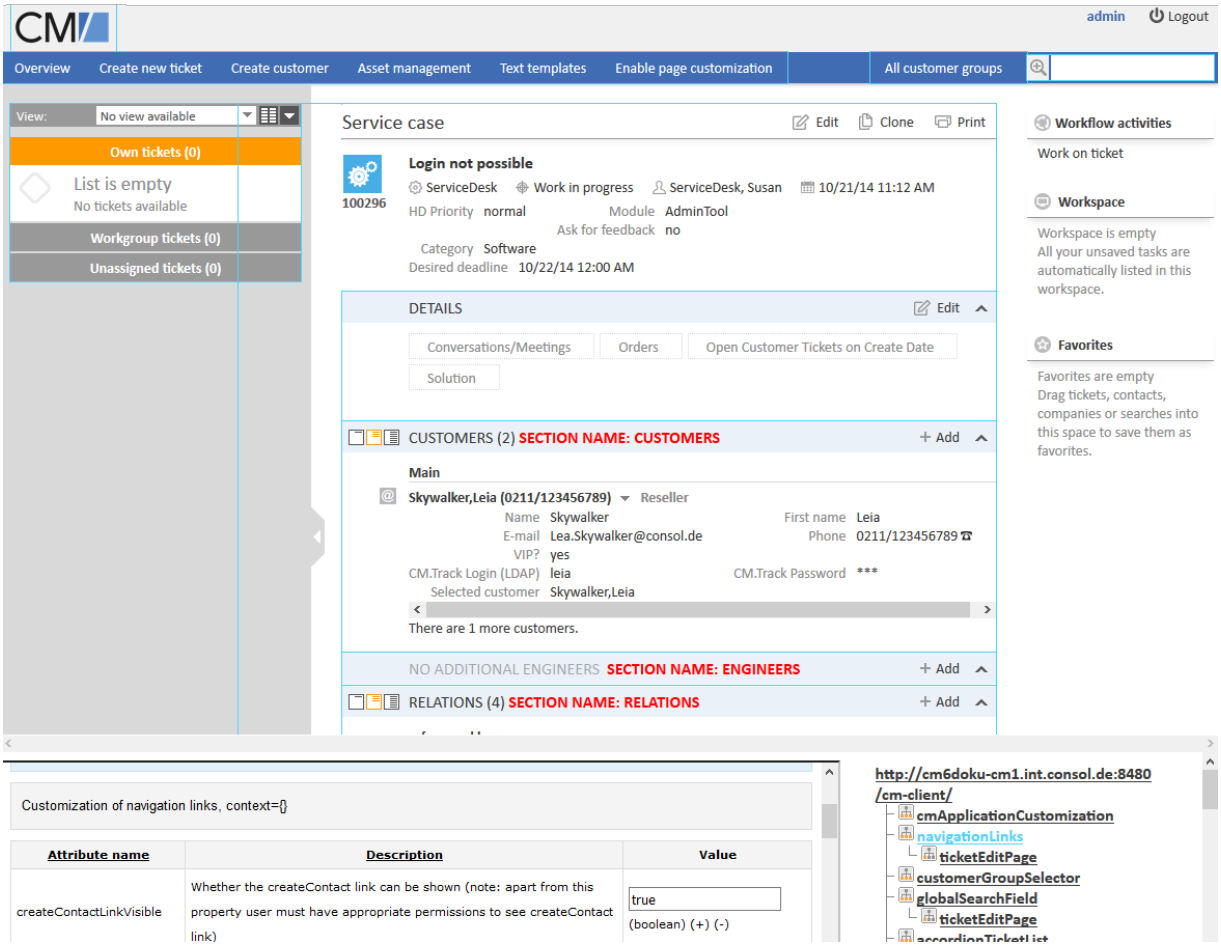


Figure 196: *ConSol CM Web Client - Selected tree element in the definition section*

The entire page is built according to a strictly hierarchical structure and every element is defined by

- a type (mandatory)
- a scope (mandatory)
- a subscope (optional)
- a class (the respective Java class, mandatory)

These are displayed in the blue header section of the editor in the Definition Section when you mark an element either in the tree or in the GUI. Using the page customization, you can decide on which level you want to configure attributes:

- When you work on the *type* level, you define the attributes for all sub-elements of this type, i.e., for all scopes (subscopes).
- When you work on the *scope* level you define the attributes for all (sub-)elements of this scope and all subscopes.
- When you work on *subscope* level you define the attributes for this subscope only.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

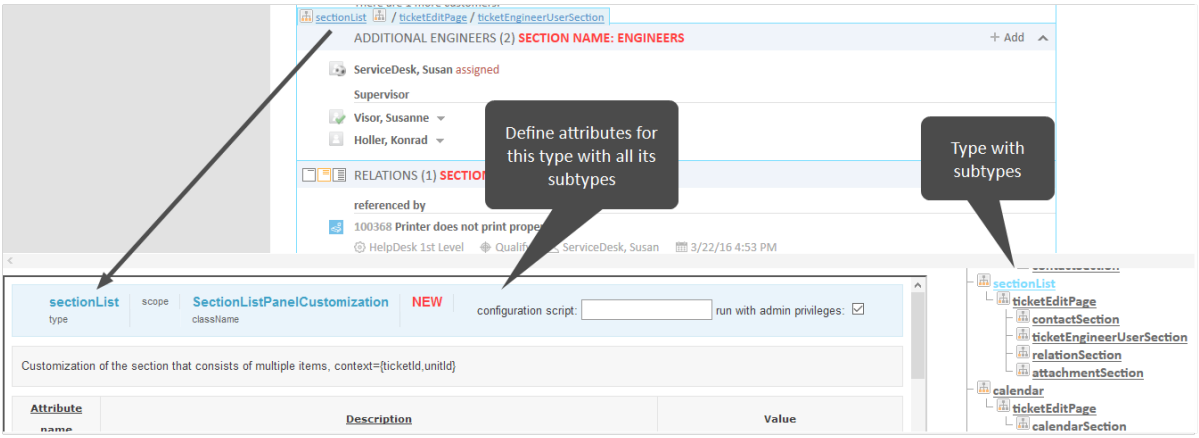


Figure 197: *Selecting the level for the Page Customization, here: type.*

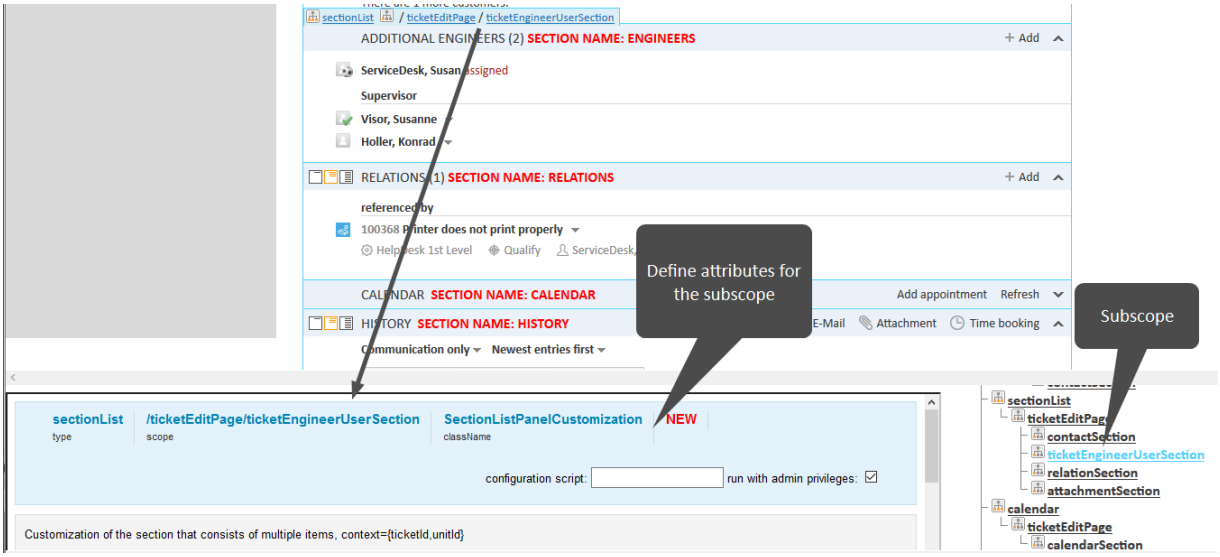


Figure 198: *Selecting the level for the Page Customization, here: subscope.*

Please see the following example for ticket list configuration.

- **Type level:**

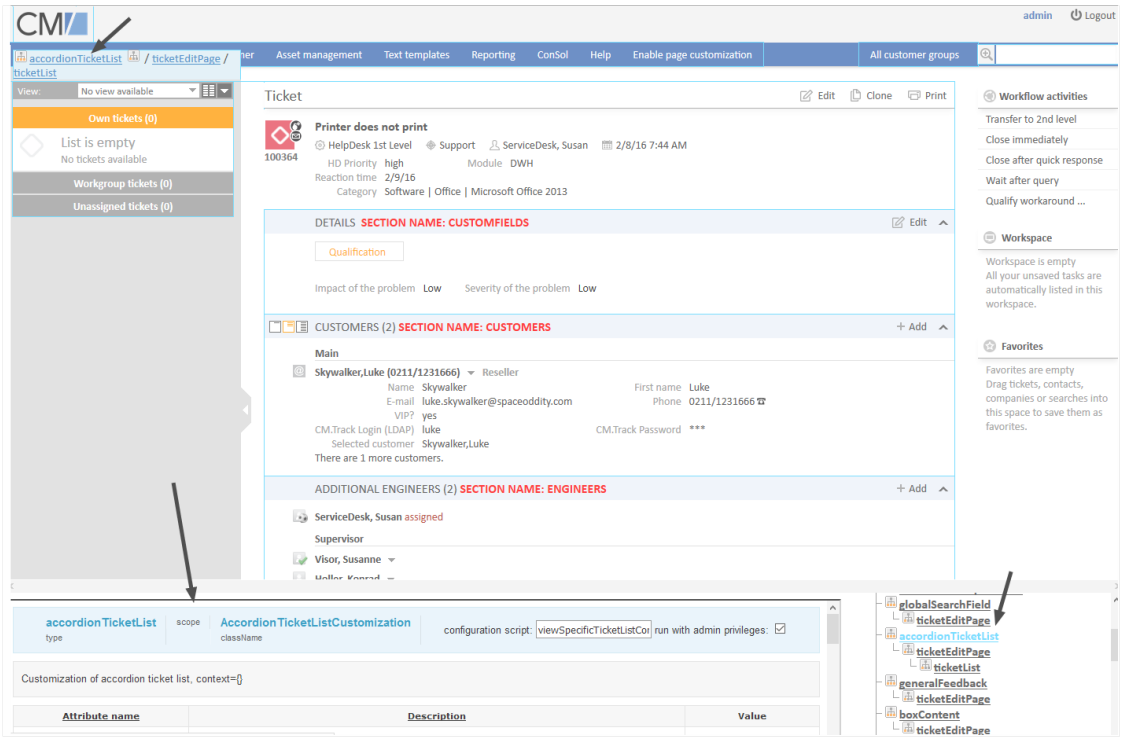


Figure 199: *ConSol CM Web Client - Defining attributes on Type level*

- **Scope level:**

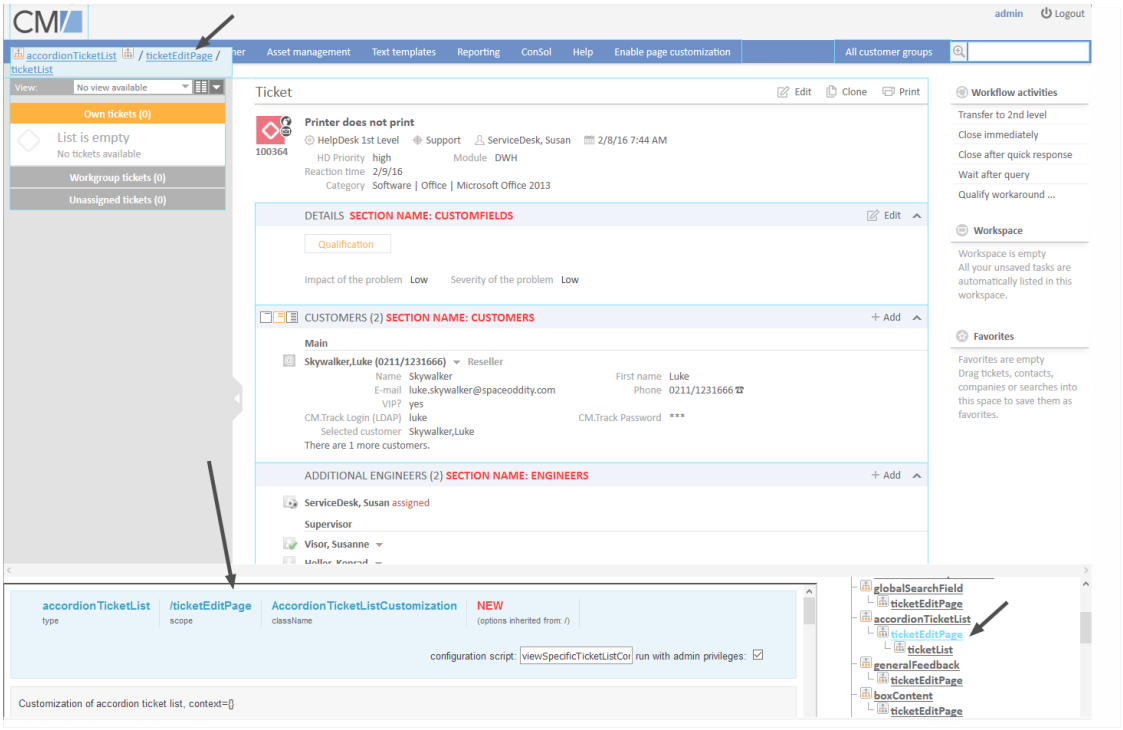


Figure 200: *ConSol CM Web Client - Defining attributes on Scope level*

- **Subscope level:**

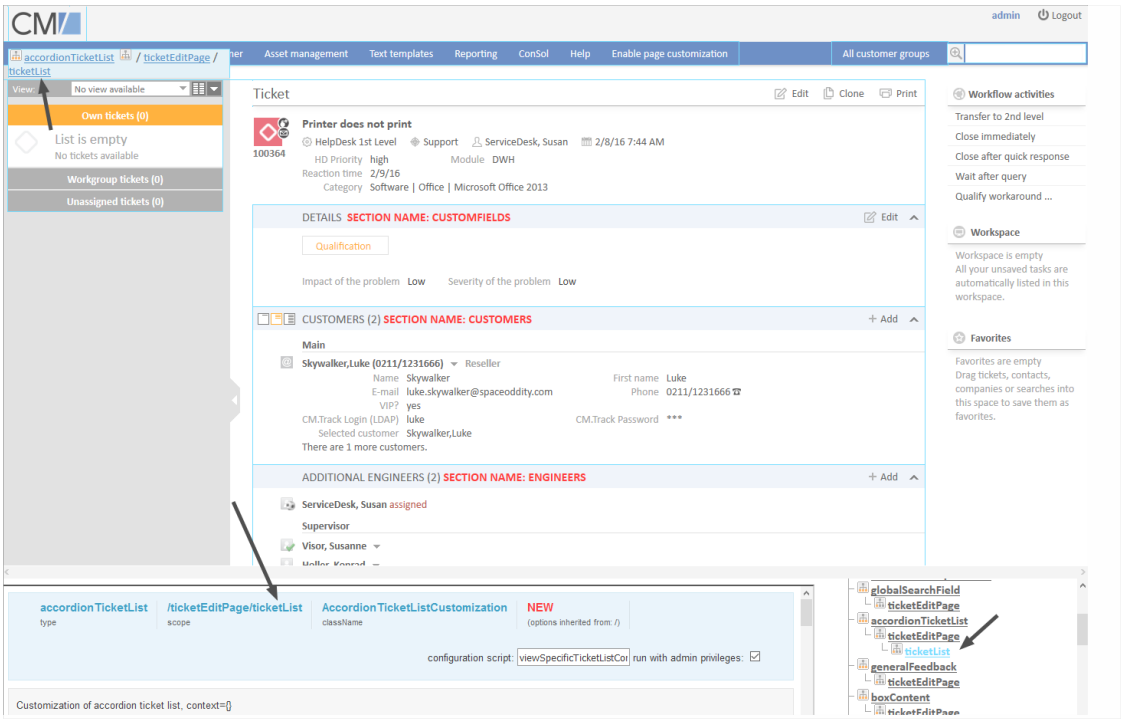


Figure 201: *ConSol CM Web Client - Defining attributes on Subscope level*

For some elements there may be two hierarchical paths, as shown in the following example for the subscope *contactSection*:

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

So, in this example, a *ticketEditPage / contactSection* subscope can be configured for two different types: *customerSectionPanel* and *sectionList*.

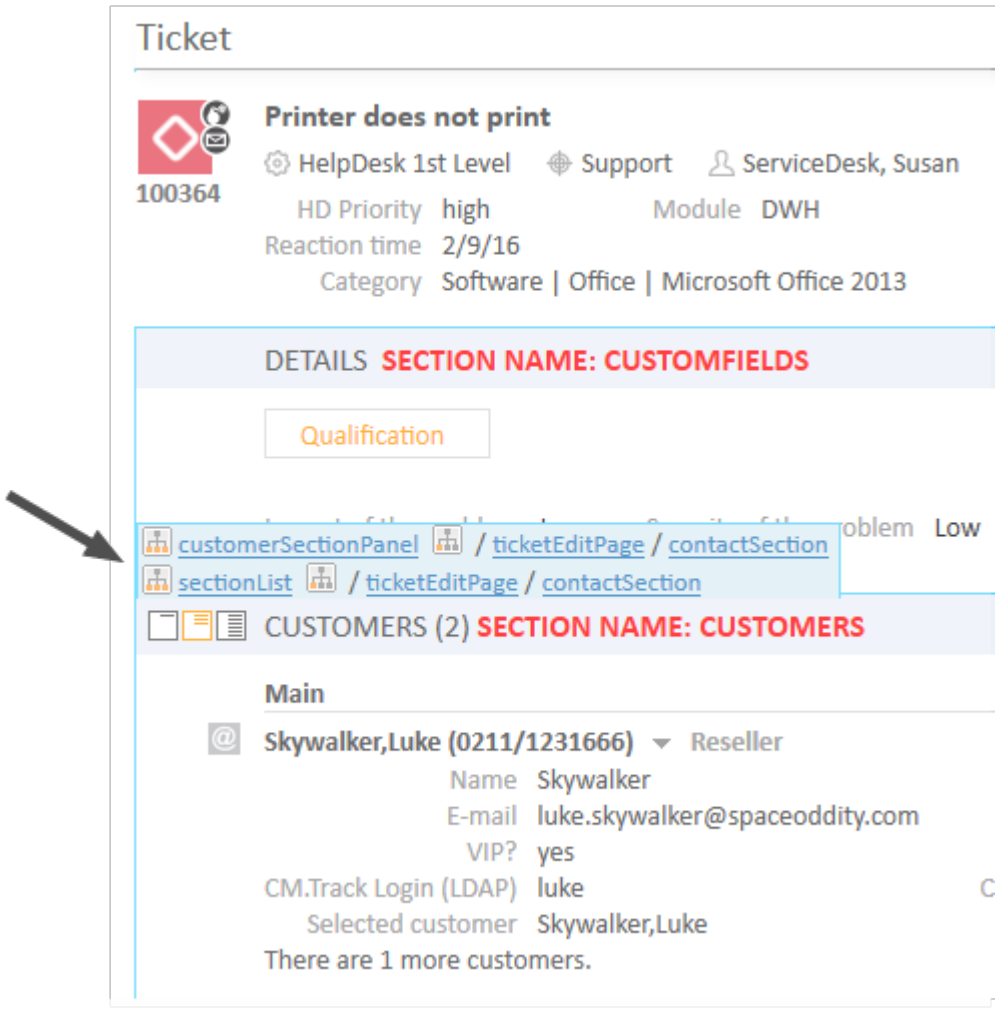


Figure 202: *ConSol CM Web Client - Two paths to the Page Customization element contactSection*

For every element, there is also a configuration script that can dynamically define values. The script is executed in the context of the engineer who is currently logged in. Therefore, the engineer permissions might exert an influence on the script result, e.g., when you use the code to select *all tickets*, the amount of tickets will be different depending on the queue permissions of the current engineer. When you need global access within the script, you can always execute it with admin privileges by ticking the respective checkbox. Within the script, the objects of the context (see following figure: example of *ticketId*) are available.

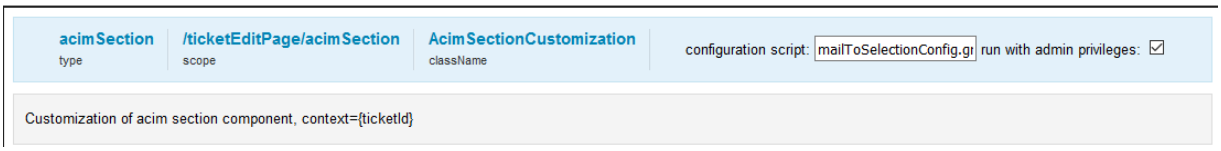


Figure 203: *ConSol CM Web Client - Configuration script for defining values*

For example, if there should be only one email recipient (here: the main customer) for medium and low priority tickets, but emails for a high priority ticket should be sent to all customers of the ticket, the following script can be used:

```
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'];
```

The script has to be stored in the *Scripts and Templates* section of the Admin Tool (see section Admin Tool Scripts for details) and its name has to be entered in the field *configuration script*.

The return values define the values for the attributes which can also be defined using the string input fields. Please note the order of all components involved, see section Order and Priorities of Page Customization.

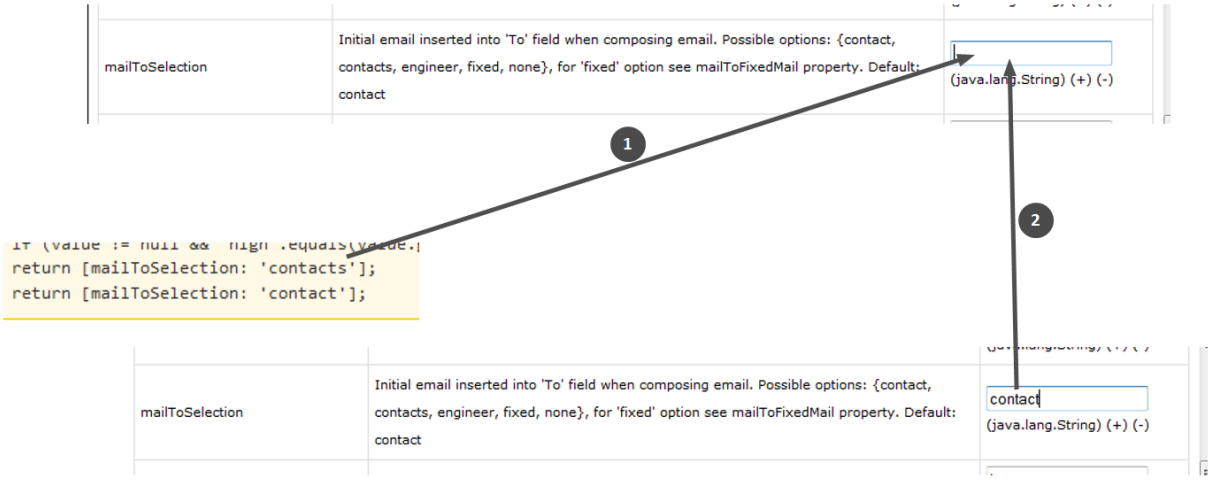


Figure 204: *Two alternatives for setting Page Customization attributes*

- Alternative 1: return attribute via script (1)
- Alternative 2: set attribute in string field (2)

Please note that in scripts Boolean values (“true”/“false”) might have to be returned. In case of ConSol CM Page Customization scripts, a syntax is used where those values have to be returned as strings!

Example: The value of the attribute *emailFeature* is “false”. In the script you will have to write:

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

E.3.2.1 Buttons for Setting Page Customization Attributes

Three buttons are available:

- **Create**
  Creates a new Page Customization (in the database). Use this to (re-)define attributes. All changes which you have made in the form will be saved.
- **Delete**
  Deletes the entries from the database for this type, scope or subscope, i.e. all attributes available in the form will be set to their original values.
- **Reset**
  Does not commit a database operation. Only reverses the entries which you have made while editing the form and which have not yet been saved.



If you enter an incorrect value for a page customization attribute, an error message is displayed upon saving the changes. You need to correct the value by opening the page customization again.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.3.3 Order and Priorities of Page Customization

In case more than one value is set for an attribute, the following hierarchy is applied:

1. **Highest priority:** script
2. **Medium priority:** scope definition
3. **Lowest priority:** type definition

Example for the value of *maxHints* in the *GlobalSearchField* on the *ticketEditPage*:

- **Variant A:**
  - script: no value
  - scope definition (GlobalSearchField/ticketEditPage): maxHints = 10
  - type definition (GlobalSearchField): maxHints = 5
  - => maxHints will be 10
- **Variant B:**
  - script: maxHints = 7
  - scope definition (GlobalSearchField/ticketEditPage): no value
  - type definition (GlobalSearchField): maxHints = 5
  - => maxHints will be 7
- **Variant C:**
  - script: no value
  - scope definition (GlobalSearchField/ticketEditPage): no value
  - type definition (GlobalSearchField): maxHints = 5
  - => maxHints will be 5

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.3.4 Page Customization Using Attributes

In the following sections, all configuration attributes for page customization will be explained. A short description is also given for each attribute in the editor section.

E.3.4.1 Possible Pages (Scopes) for Page Customization

The following main scopes are available. i.e., when you have opened the respective page you can configure page customization attributes which are visible only on this page for the given attribute.

companyEditPage

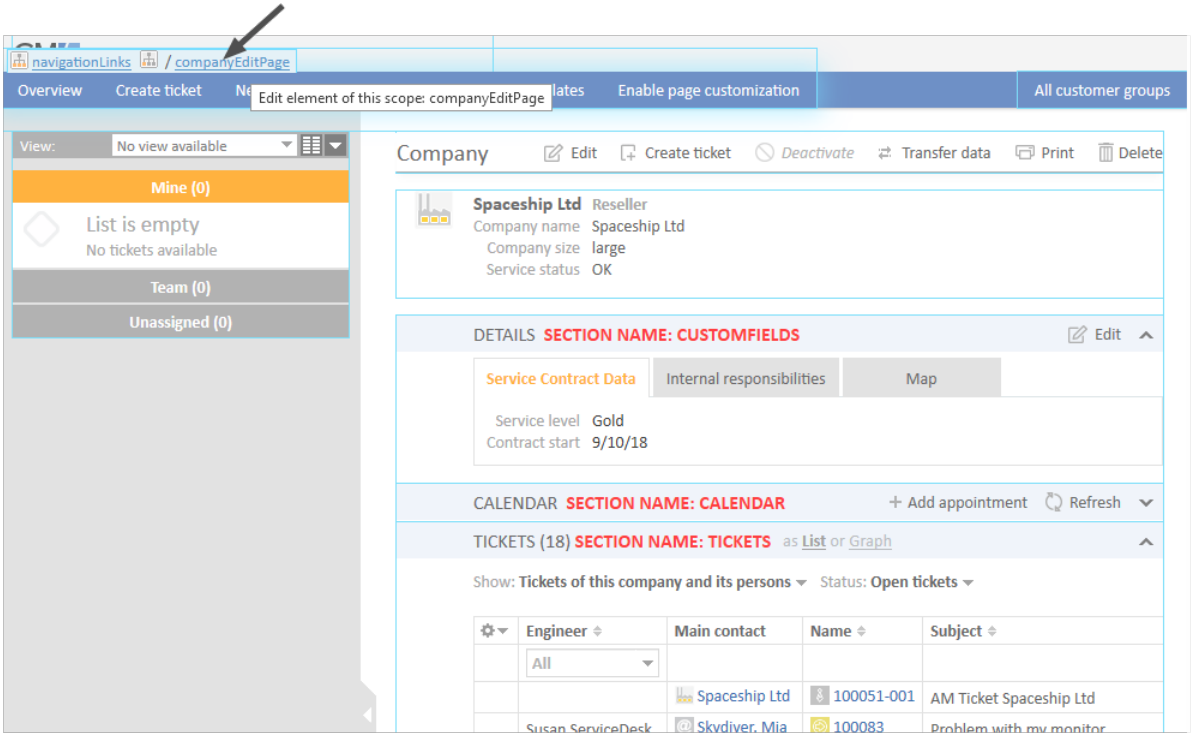


Figure 205: *ConSol CM Web Client - Page Customization for the company page*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

contactCreatePage

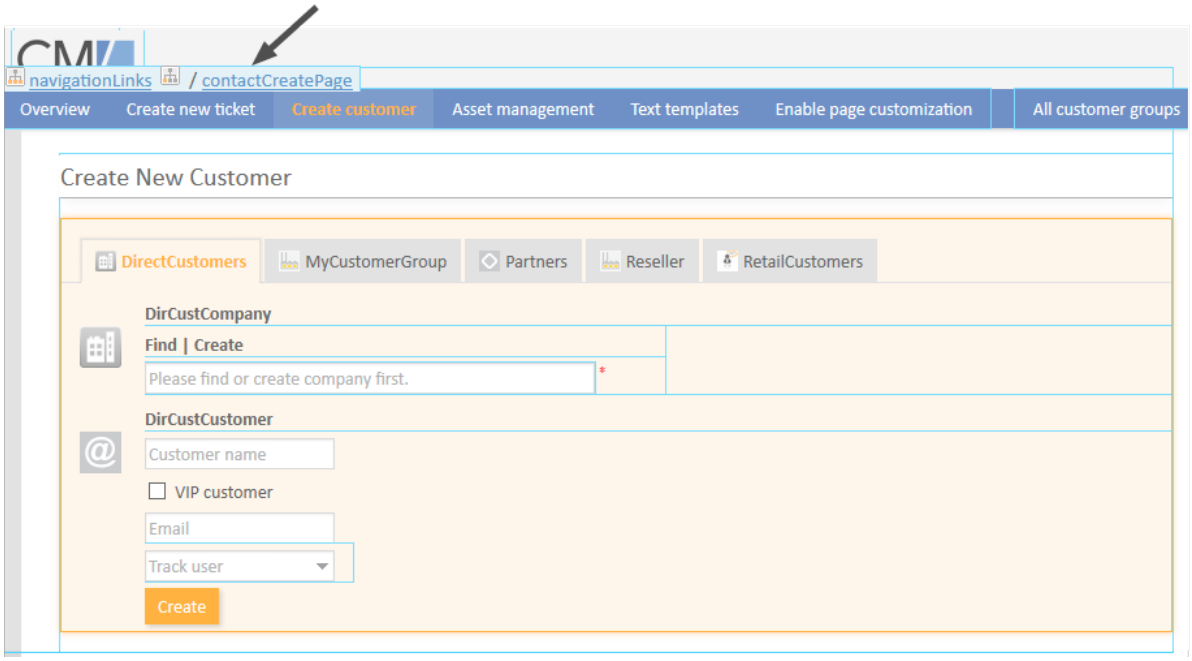


Figure 206: *ConSol CM Web Client - Page Customization for the Create New Customer page*

contactEditPage

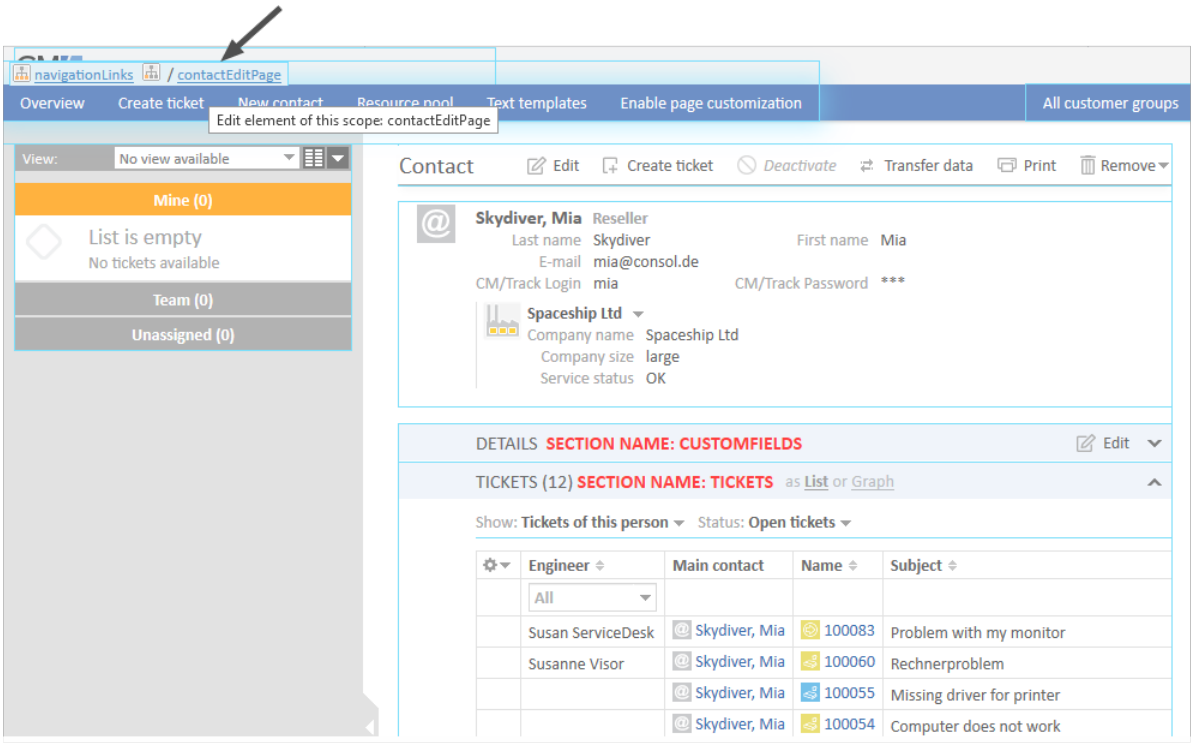


Figure 207: *ConSol CM Web Client - Page Customization for the contact page*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

resourceDashboard

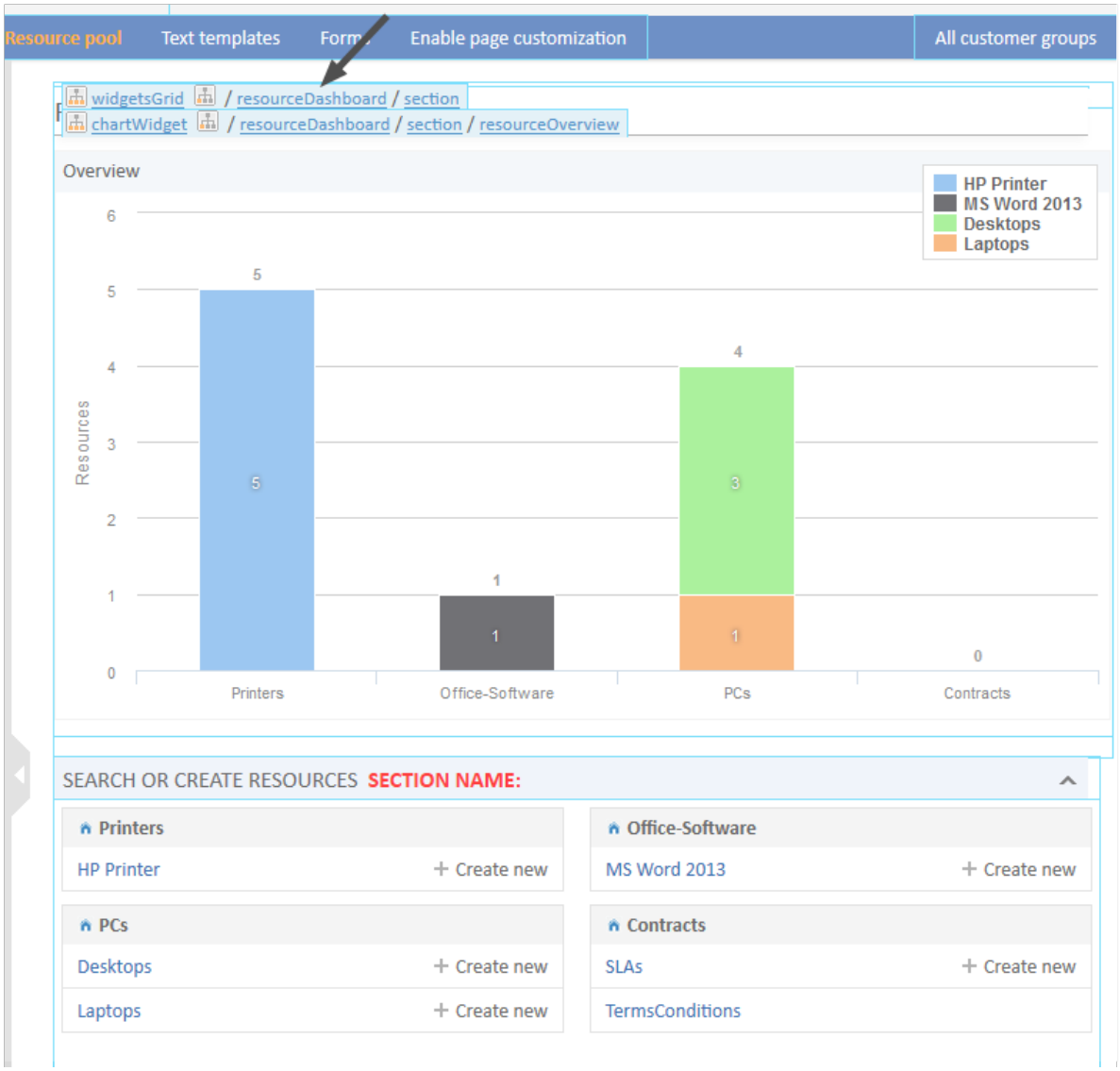


Figure 208: *ConSol CM Web Client - Page Customization for the Resource Pool Dashboard*

resourceType (Resource Type Page)

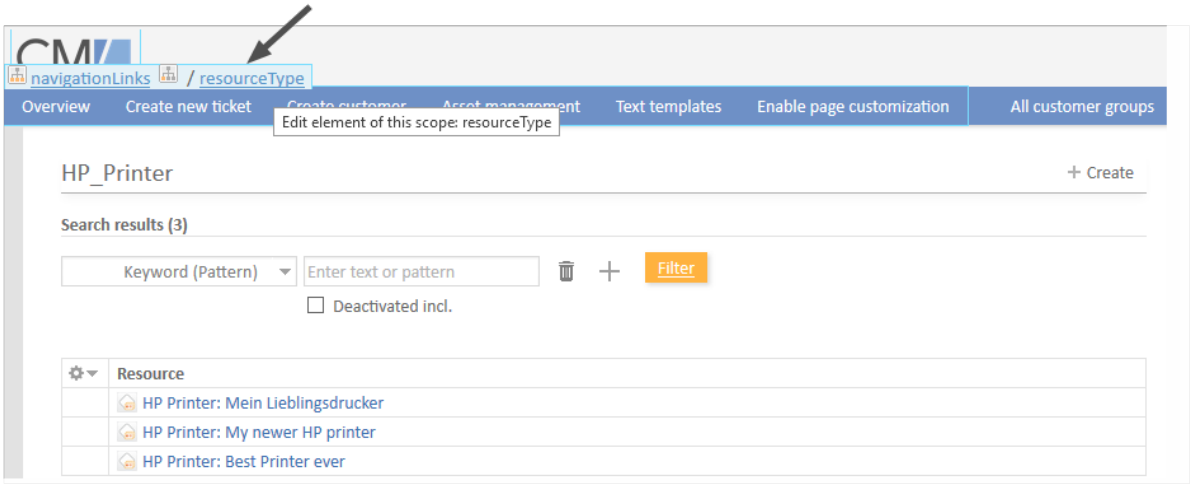


Figure 209: *ConSol CM Web Client - Page Customization for the resource type page*

resource (Resource Page)

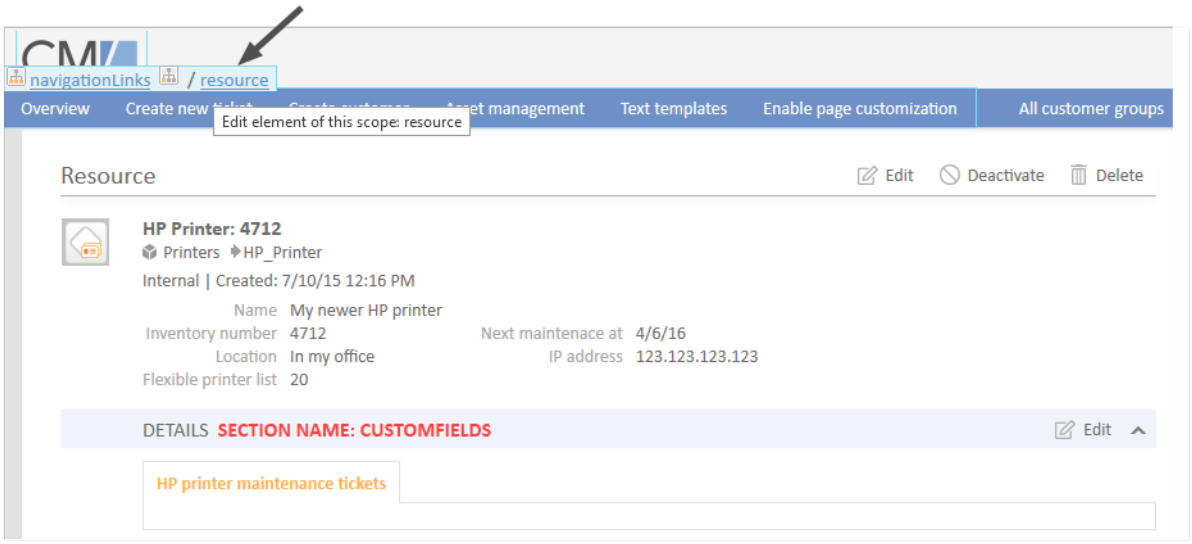


Figure 210: *ConSol CM Web Client - Page Customization for the resource page*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

officeTemplatePage (Only When CM/Doc Is Activated)

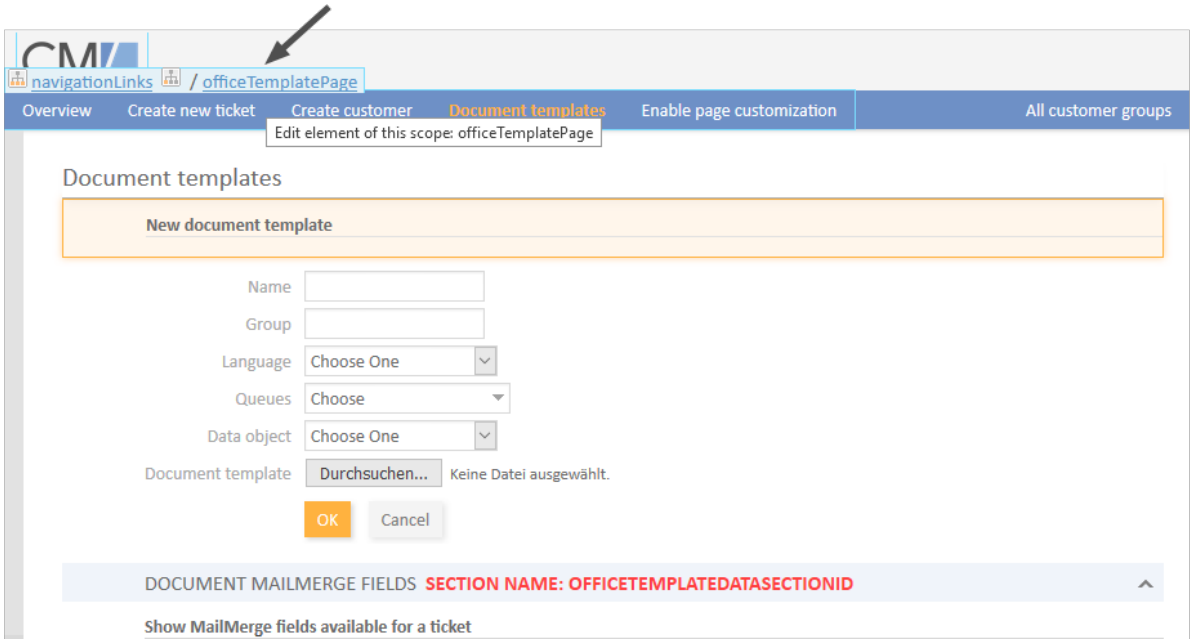


Figure 211: *ConSol CM Web Client - Page Customization for the Document templates page (only when CM/Doc is activated)*

searchDetailPage

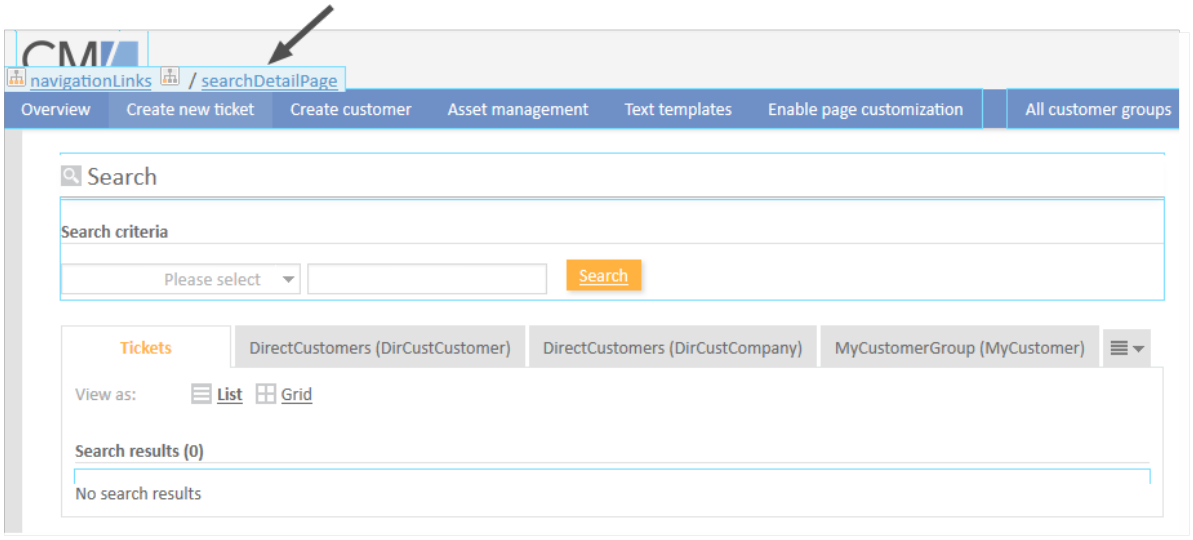


Figure 212: *ConSol CM Web Client - Page Customization for the Detailed Search*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

templateViewPage

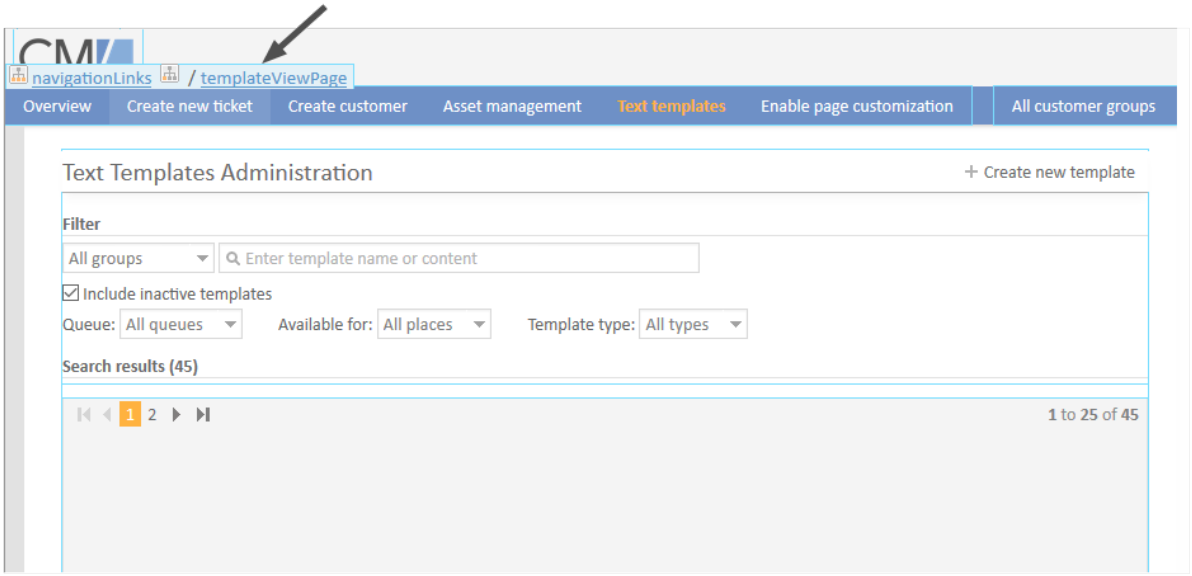


Figure 213: *ConSol CM Web Client - Page Customization for the Text templates page (view mode)*

templateEditPage

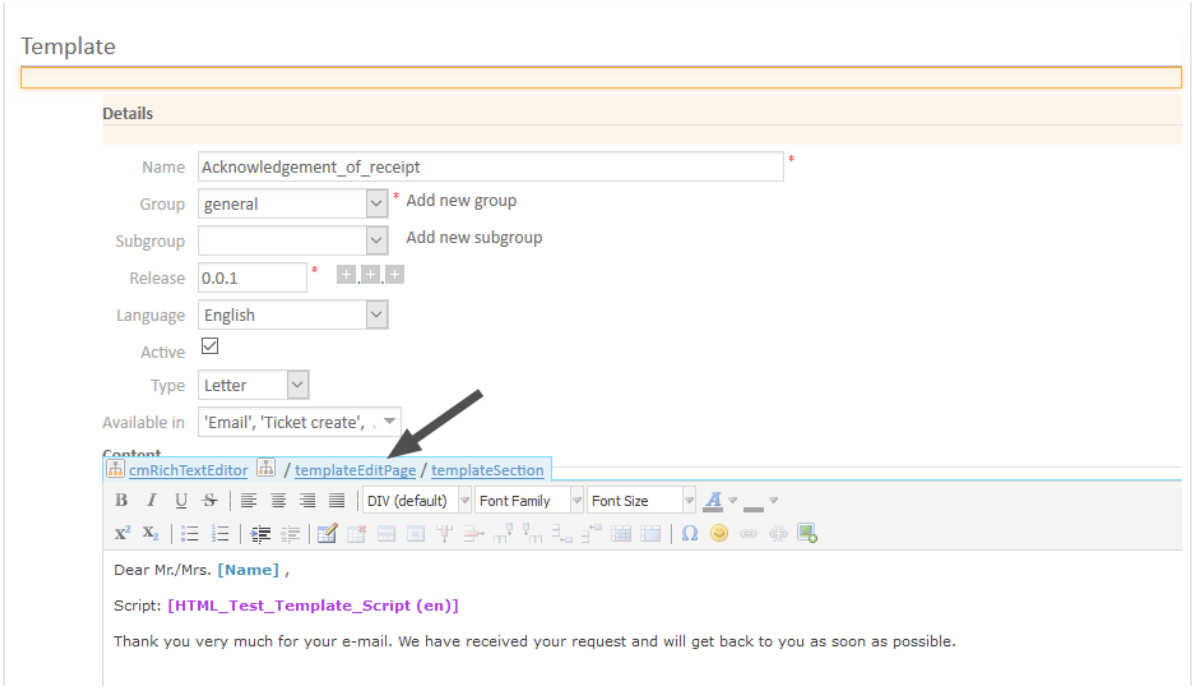


Figure 214: *ConSol CM Web Client - Page Customization for the Text templates page (edit mode)*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

ticketCreatePage

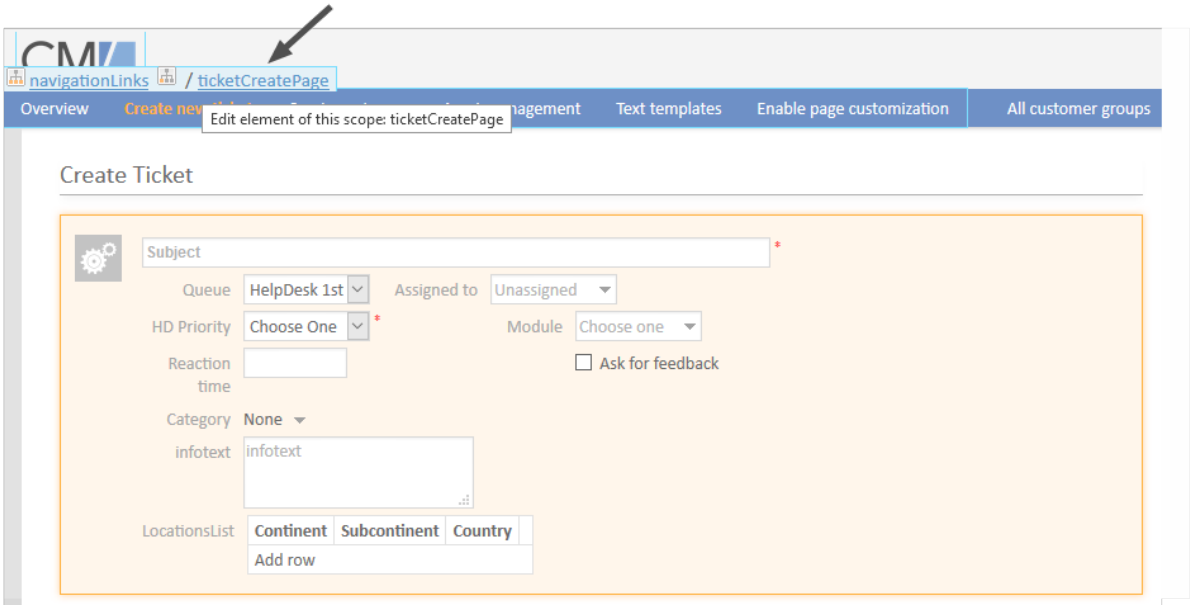


Figure 215: *ConSol CM Web Client - Page Customization for the New ticket page*

ticketEditPage

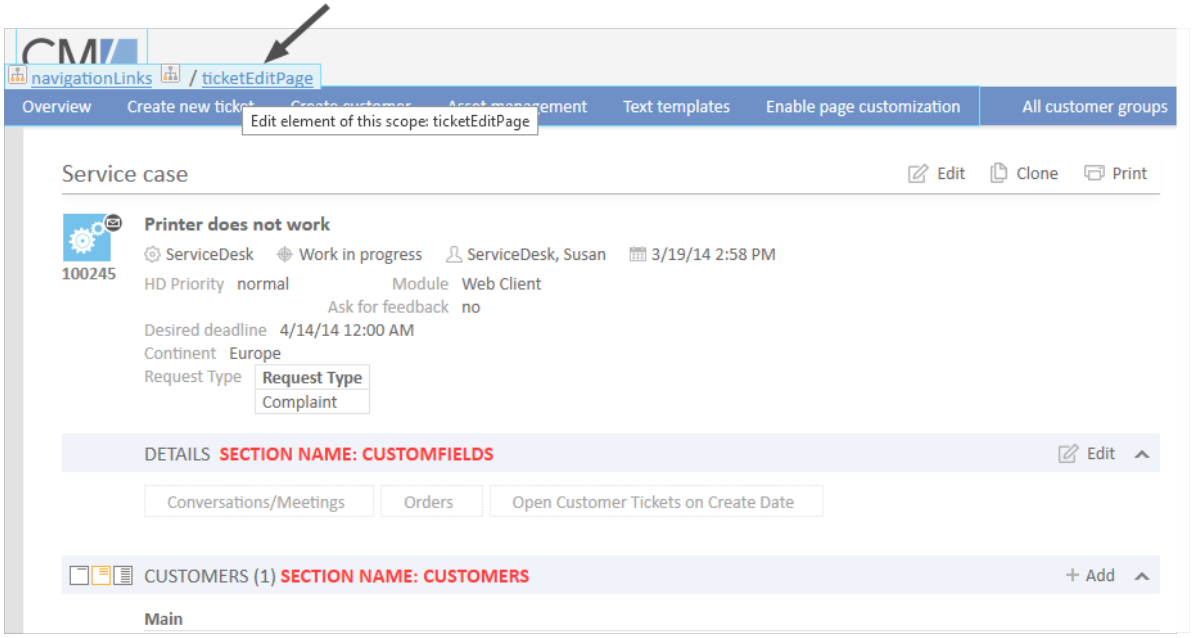


Figure 216: *ConSol CM Web Client - Page Customization for the ticket page*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

userProfilePage

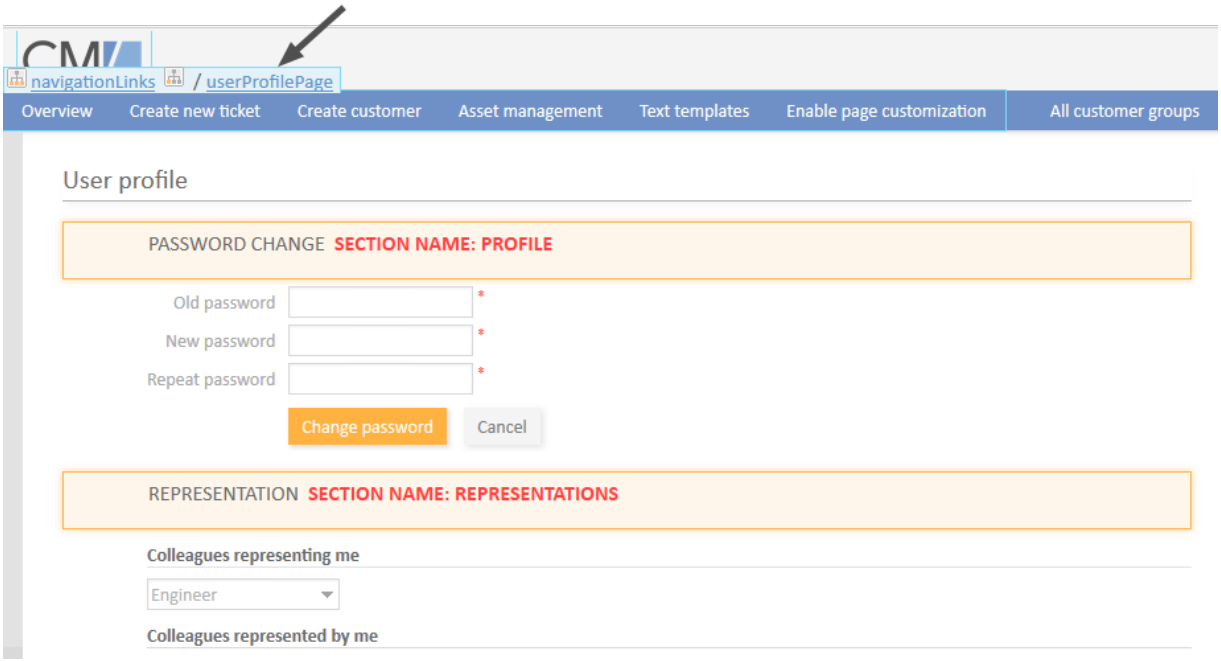


Figure 217: *ConSol CM Web Client - Page Customization for the engineer profile*

welcomePage (Overview Page)

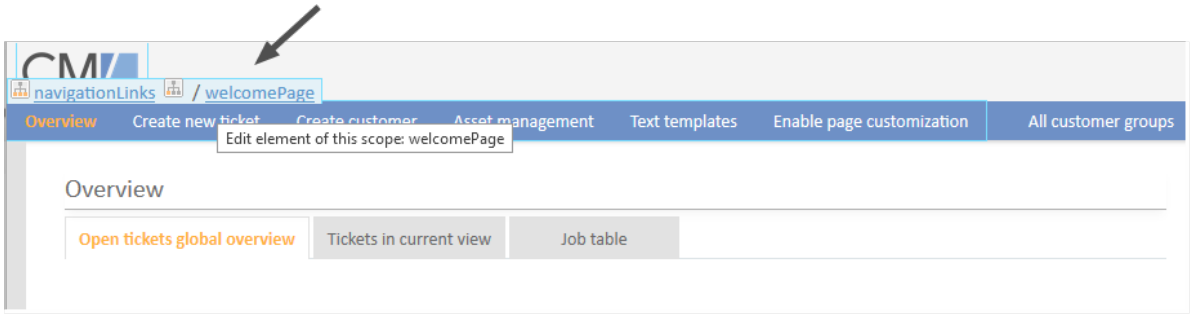


Figure 218: *ConSol CM Web Client - Page Customization for the Welcome page*

For example, for the type *globalSearchField* (see subsequent section) the following page-specific scopes can be configured. That means the behavior of the *globalSearchField* (type) can be configured for each of the following pages (scopes) where it is available:

- globalSearchField/companyEditPage
- globalSearchField/contactCreatePage
- globalSearchField/contactEditPage
- globalSearchField/resource
- globalSearchField/resourceDashboard
- globalSearchField/resourceType
- globalSearchField/searchDetailPage
- globalSearchField/ticketCreatePage

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- globalSearchField/ticketEditPage
- globalSearchField/welcomePage

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.3.5 Page Customization Attributes for Types, Scopes and Subscopes, in Alphabetical Order

The following sections contain a list of Page Customization types, scopes and subscopes. The attributes are described on the respective page.

accordionTicketList (Type) 345
acimSection (Type or Subscope) 346
analyticsPage (Type) 355
attachmentSection (Type or Subscope) 356
autocomplete 356
boxContent 357
calendarSection 358
cmApplicationCustomization (Type) 358
cmRichTextEditor 360
customerGroupSelector 361
customerSectionPanel 362
customerTickets 365
detailSearch (Type) 367
engineerAutocomplete 374
enumAutocomplete 375
froala (Type) 375
generalFeedback 376
globalSearchField (Type) 376
mailTemplate (Type) 378
markersLibrary (Type) 379
navigationLinks (Type) 380
preview (Type) 381
resourceRelations 383
resourceRelationsPanel 383
resourceTypes 384
section (Type) 384
sectionList (Type) 384
table 386
template 386
ticketCreatePanel 387
ticketList (Subscope) 388
ticketPanel 388
TicketRelation 388

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

ticketsAutocomplete 389
ticketsBookingAutocomplete 389
timeBookingSection 389
unitAutocomplete 390
unitFormPanel 390
unitRelationSection (Type UnitSection) 390
UnitResourceRelation 391
unitSearch 392
unitSearchHeader 392
viewDiscriminatorsSection (Type) 392
welcomePage 394

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

accordionTicketList (Type)

Here you can define attributes for the ticket list. Only one subscope is available: *ticketList*

Attributes:

- **hideEmptyGroups**
  Defines whether empty groups are hidden from the ticket list. If set to “false” (default), all groups are displayed. If set to “true”, empty groups are hidden.

  Note that a ticket list group is not hidden if it has the focus. In this case the group is displayed when the ticket list is loaded, and hidden once the user clicks another group.
- **loadingTicketListMode**
  The mode used to render the ticket list. There are four options to select from:
  - LAZY_SYNCH (default)
    The waiting indicator is shown in the place of the ticket list while the rest of a page is rendered, then the ticket list is loaded. The main benefit of this approach is the possibility to show/read the main content faster.
  - LAZY_ASYNCH
    (deprecated since 6.8.2, will be treated as LAZY_SYNCH mode)
    The modification of the LAZY_SYNCH strategy. It does not wait when the page is being rendered but sends the second request and then loads the ticket list. This strategy will load the ticket list faster but may reduce the benefit of having the main page immediately.
  - INCLUDED
    The ticket list is loaded along with the rest of a page.
  - LAZY_ASAP
    The waiting indicator is shown in the place of the ticket list while the rest of a page is rendered. The request for the ticket list is sent as soon as two libraries have been loaded. In this approach a request for the ticket list is sent and processed concurrently with the first request. The ticket list will appear much faster on the page. (java.lang.String)
- **mainCustomerDescriptionVisible**
  The page customization attribute *accordionTicketList.**mainCustomerDescriptionVisible***={true, false} replaces the annotation `show-contact-in-ticket-list` (which is valid until CM version 6.8). When this value is set to “true”, the customer data of the main customer is displayed in the ticket list representation of the ticket. Default is “true”. (boolean)
- **pauseRefreshWhileTabInBackground**
  Configures the refresh behavior of the ticket list. If set to “true” (default value), the ticket list is only refreshed if the tab is active. If it is set to “false”, the ticket list is also refreshed if the tab is in the background, the browser is minimized or the computer is locked / in screensaver mode.
- **quickAssignLinkShowsTicketPageFlag**
  Whether the quick assign link (represented by the arrow rendered for each unassigned ticket) opens the assigned ticket immediately (boolean, default is “false”).
- **ticketDataConfigQueueGroupingScript**
  Defines the ticket information display when grouping by queue is selected. See section Ticket List.
- **ticketDataConfigEngineerGroupingScript**
  Defines the ticket information display when grouping by engineer is selected. See section Ticket List.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **ticketDataConfigCustomGroupingScript**
  Defines the ticket information display when grouping by a ticket field is selected. See section Ticket List.
- **ticketDataConfigNoGroupingScript**
  Defines the ticket information display when no grouping criterion is selected. See section Ticket List.

acimSection (Type or Subscope)

An *ACIM* (activity item) is an entry in the *History* section of a ticket. This can be ...

- a comment
- an email which has been sent from the ticket
- an email which has been received by the ticket
- an attachment
- a time booking entry

An ACIM group is a group of entries which has a distinct date/time stamp. An ACIM item is a single entry within the ACIM group. It has only a time stamp.

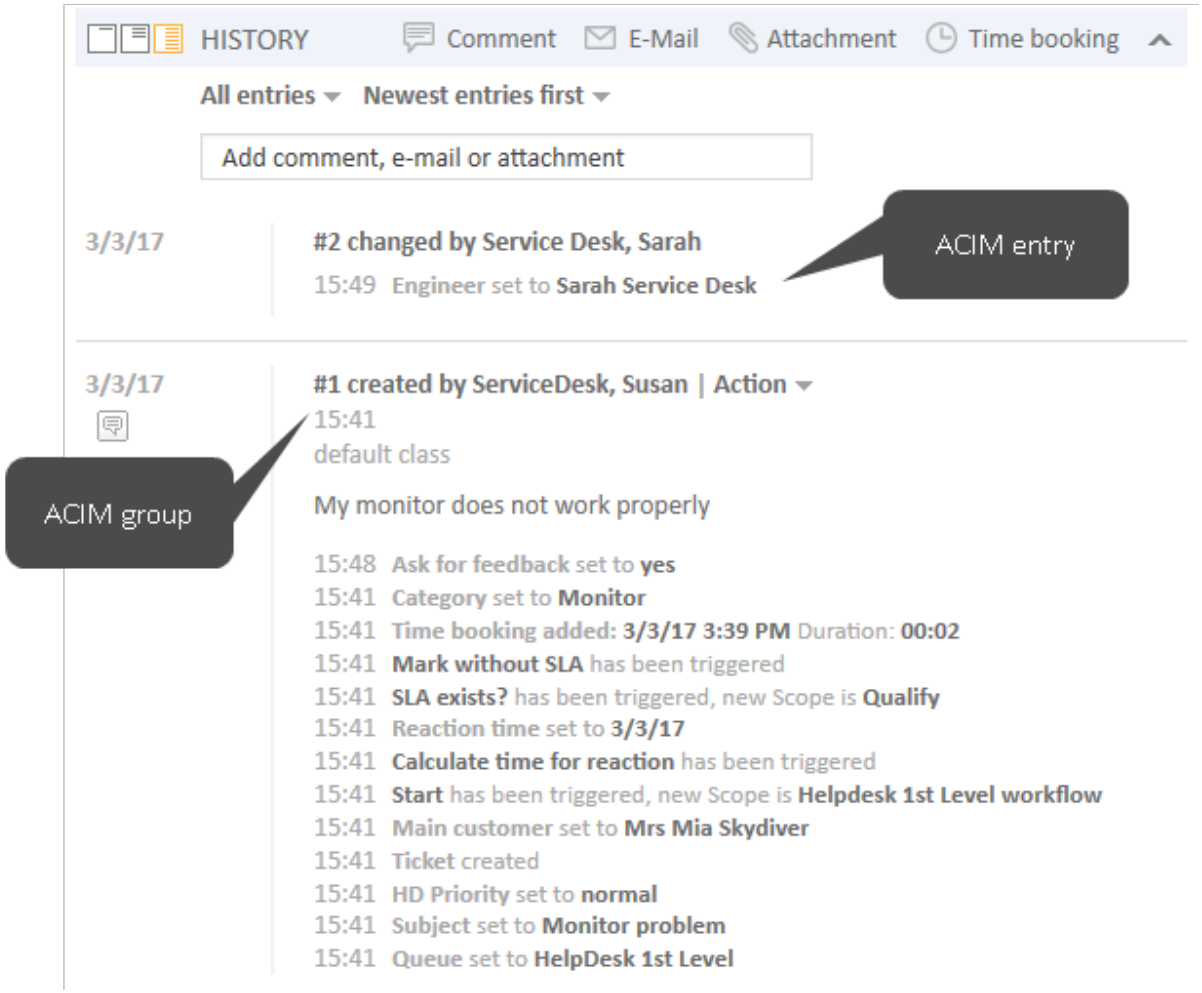


Figure 219: *ConSol CM Web Client - ACIM group and item*

⚠ Please make sure that the date format you have entered for one of the following date attributes is correct! If the date format is not correct, the entire page cannot be displayed! The Web Client will not work! Please see the correct date formats in the following table. By using an empty text (' ') as value it is possible to hide the date/time stamp completely.

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

Figure 220: *ConSol CM Web Client - Valid date formats for ACIM date configuration*

Attributes:

- **acimGroupActionEntryDateFormat**
  Date format for group of ACIM without text or email entry (i.e., for automatic actions). If nothing has been entered as pattern, the default one will be used.
  Syntax: dateFormatFirstLevelOfDetails|secondLevel|thirdLevel
  (java.lang.String, default = dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm)

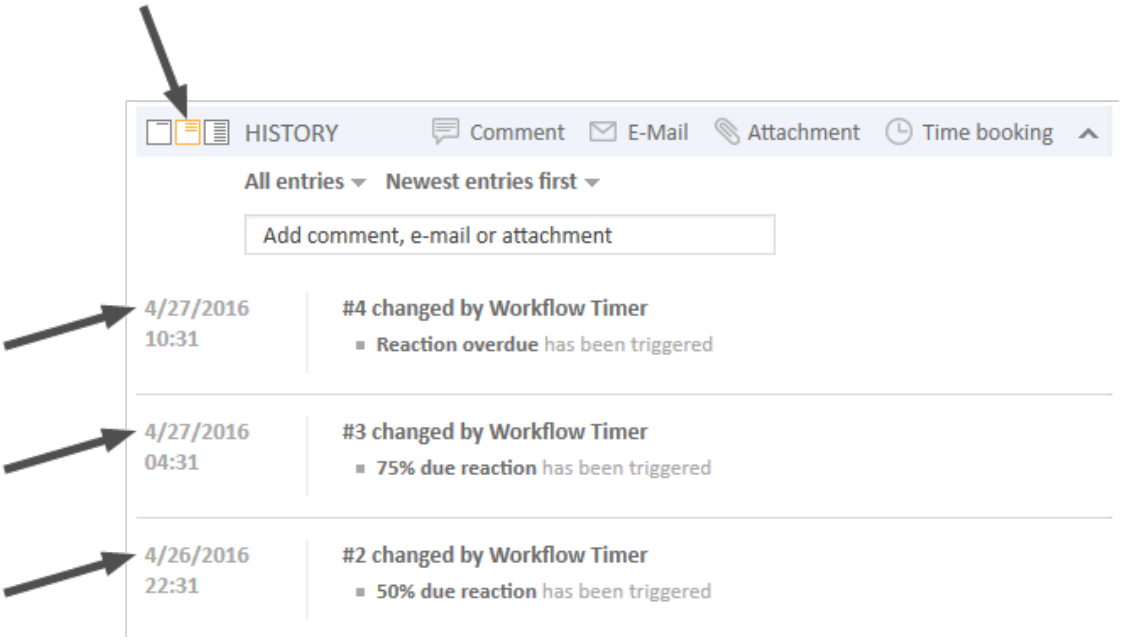


Figure 221: *ConSol CM Web Client - Display for format: dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm*

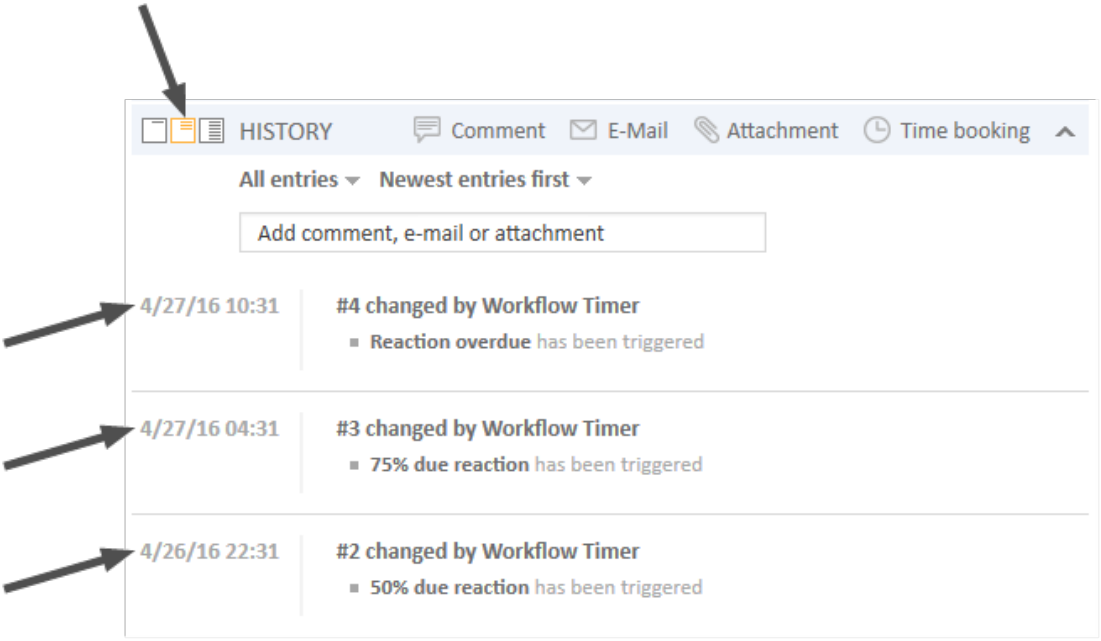


Figure 222: *ConSol CM Web Client - Display for format: dd.MM.yy HH.mm|dd.MM.yy HH.mm|dd.MM.yy HH.mm*

- **acimGroupTextEntryDateFormat**
  Date format for group of ACIM with text, email, or attachment entry. If nothing has been entered as pattern, the default one will be used.
  Syntax: dateFormatFirstLevelOfDetails|secondLevel|thirdLevel
  (java.lang.String, default = dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm)

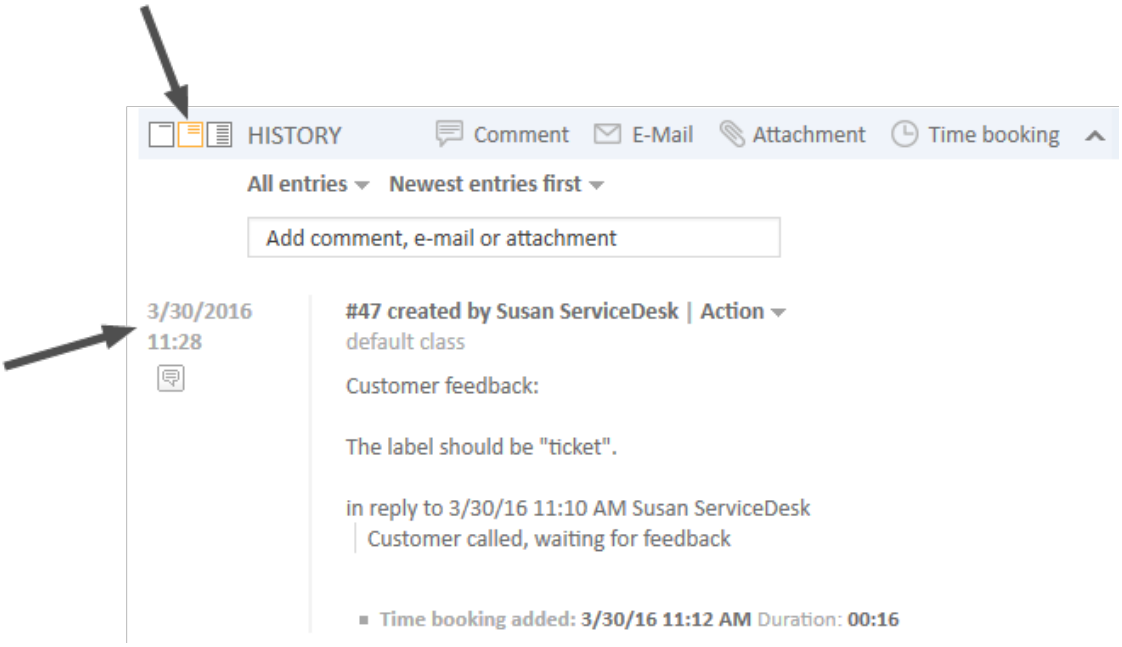


Figure 223: *ConSol CM Web Client - Display for format: dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm*

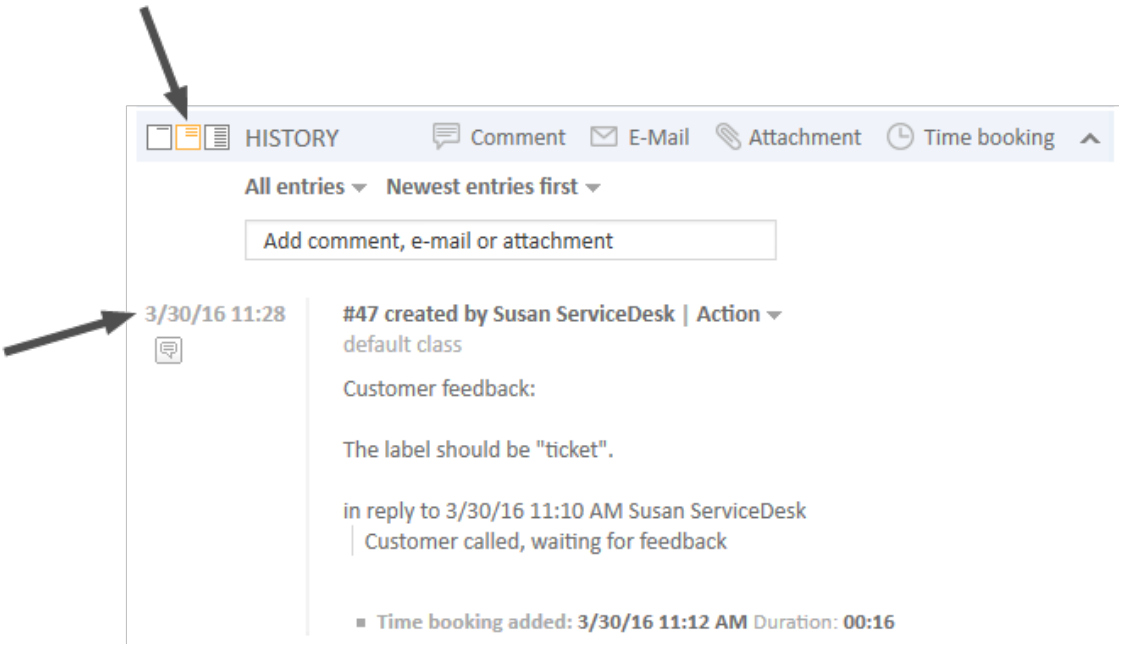


Figure 224: *ConSol CM Web Client - Display for format: dd.MM.yy HH.mm|dd.MM.yy HH.mm|dd.MM.yy HH.mm*

- **acimItemActionEntryDateFormat**
  Date format for item of ACIM entry. If nothing has been entered as pattern, the default one will be used.
  Syntax: dateFormatFirstLevelOfDetails|secondLevel|thirdLevel
  (java.lang.String, default = dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm)

- **acimItemTextEntryDateFormat**
  Date format for text or email entry. If nothing has been entered as pattern, the default one will be used.
  Syntax: dateFormatFirstLevelOfDetails|secondLevel|thirdLevel
  (java.lang.String, default = dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm)
- **allowedTimeToEditTextEntry**
  Period in minutes during which the engineer can edit his own comments. The default value is “120” (2 hours). Set the value to “0” to deactivate the comment editing functionality.
- **showCloneOption**
  Enables clone option for text ACIM entry (comment or email entry). (boolean)

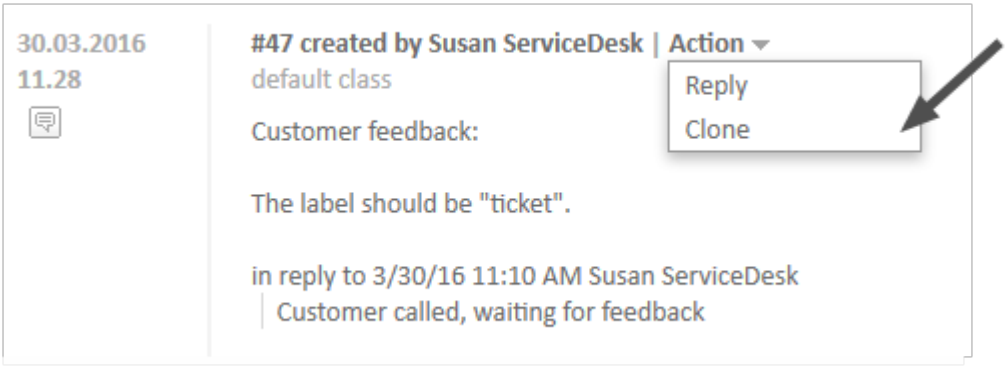


Figure 225: *ConSol CM Web Client - Clone option for text ACIM entry*

- **appendOrReplaceOnClone**
  Works only if clone option is set to “true”. If the editor is opened and already contains some text, you can append or replace its content when the clone of another item is selected simultaneously. Possible values are “append”, “replace”. Default is “append”. (java.lang.String)
- **attachmentDeletionAllowedManuallyUploaded**
  Boolean. Enables the delete functionality (entry in context menu) for ticket attachments which have been uploaded manually as a file. Default “true”.
- **attachmentDeletionAllowedIncomingEmail**
  Boolean. Enables the delete functionality (entry in context menu) for attachments from incoming emails. Default “false”.
- **attachmentDeletionAllowedOutgoingEmail**
  Boolean. Enables the delete functionality (entry in context menu) for attachments from outgoing emails sent from the system. Default “false”

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

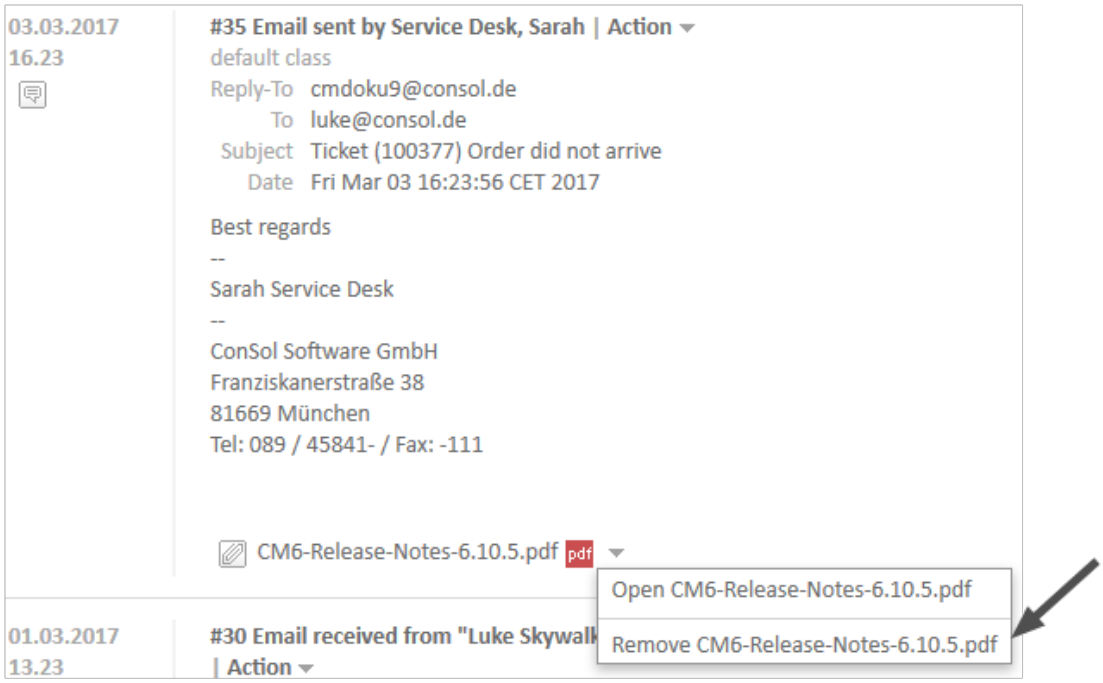


Figure 226: *Excerpt of a ticket page: attachmentDeletionAllowedManuallyUploaded set to true, manually uploaded attachment can be removed/deleted*

- **headHistoryElementsCount**
  Lazy loading - Number of groups in ACIM section that will be loaded from the top of the history. Default number is “0” (= lazy loading switched off). Customization works only when configured by type, not location. If head and tail elements count is “0”, then all history is loaded at once. (int)
- **tailHistoryElementsCount**
  Lazy loading - Number of groups in ACIM section that will be loaded from the bottom of the history. Default number is “0” (= lazy loading switched off). Customization works only when configured by type, not location. If head and tail elements count is “0”, then all history is loaded at once. (int)

> The page customization attributes *headHistoryElementsCount* and *tailHistoryElementsCount* are available on three different scope levels:
>
> - *acimSection* type on top level without scope
> - *acimSection* type with */ticketEditPage* scope
> - *acimSection* type with */ticketEditPage/acimSection* (sub-)scope
>
> A non-zero value activating the lazy loading feature must be set either on the top level without scope alone or on all three levels.
>
> Other scope uses may lead to unwanted behavior.

Example 1: Lazy loading switched off

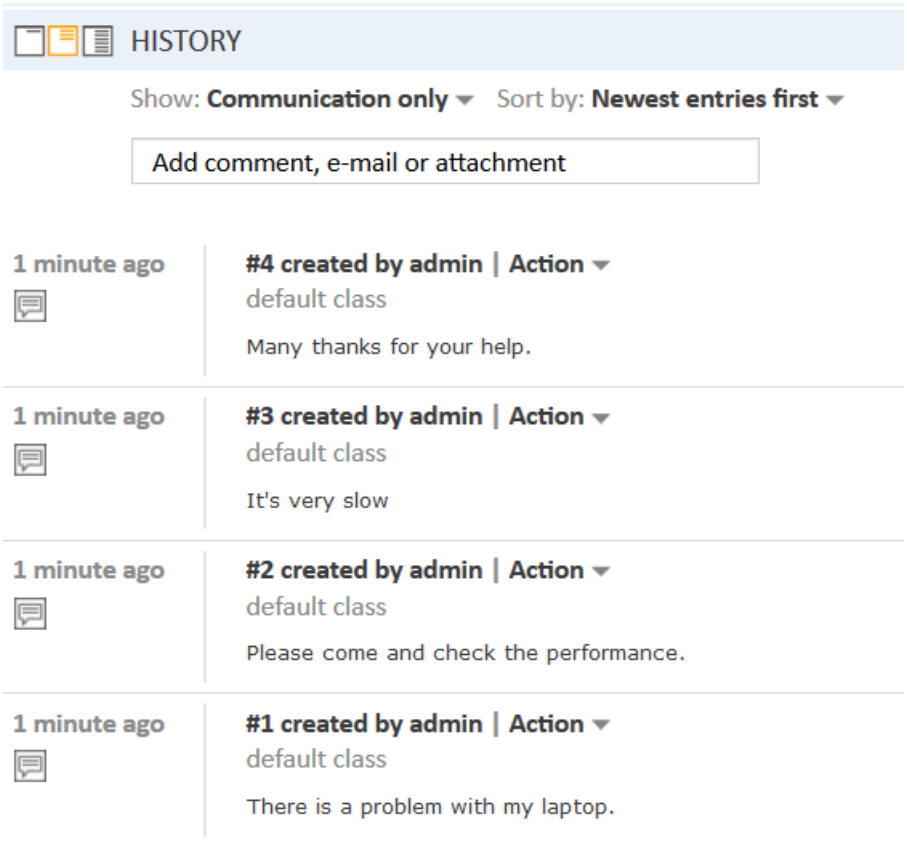


Figure 227: *ConSol CM Web Client - Lazy Loading switched off*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Example 2: headHistoryElementsCount and tailHistoryElementsCount each set to “1”

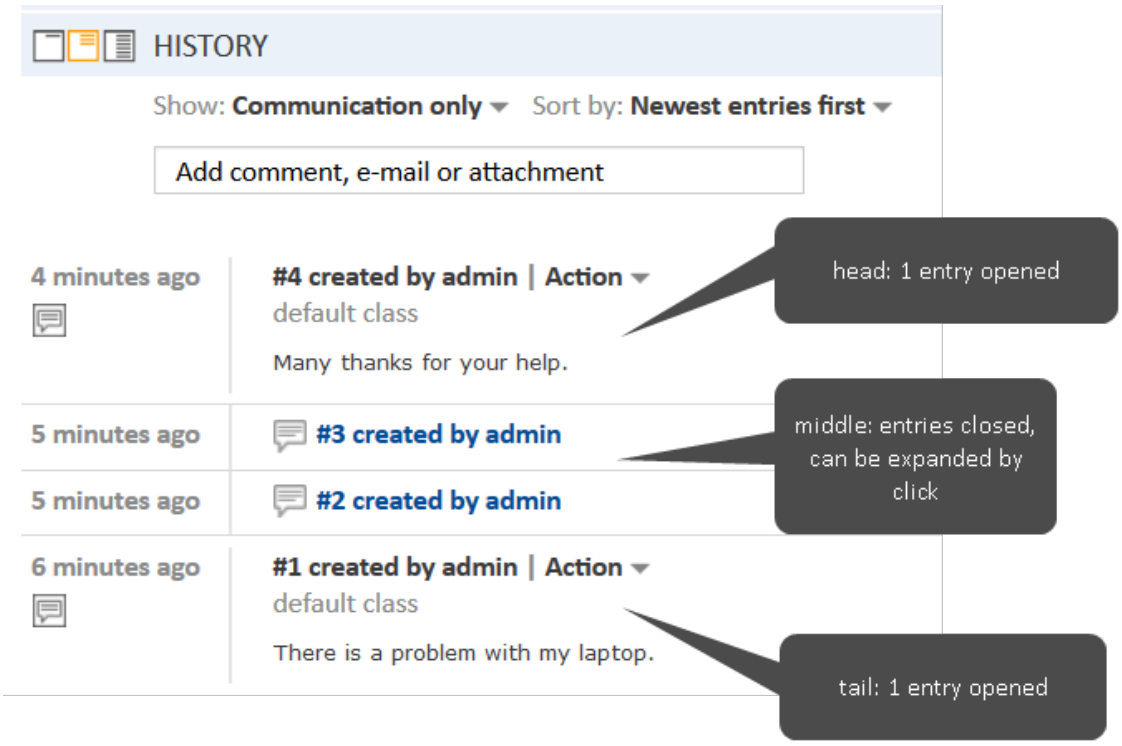


Figure 228: *ConSol CM Web Client - headHistoryElementsCount and tailHistoryElementsCount set to 1*

- **mailToSelection**
  Initial email address inserted into To field when composing email. Default: “contact” (java.lang.String)
  Possible options:
  - **contact**
    The email address of the main contact is copied into the To field when the Ticket Email Editor is opened.
  - **contacts**
    The email addresses of the main contact and all additional contacts are copied into the To field when the Ticket Email Editor is opened.
  - **engineer**
    The email address of the engineer is copied into the To field when the Ticket Email Editor is opened.
  - **fixed**
    for *fixed* option see *mailToFixedMail* attribute.
  - **none**
    The To field is left empty when the Ticket Email Editor is opened.
- **mailToFixedMail**
  Fixed email address used when attribute *mailToSelection* is set to “fixed”. (java.lang.String)
  Example: Email to a fixed email address, *mailToFixedMail* set to “ foo@bar.de”, *mailToSelection* set to “fixed”.

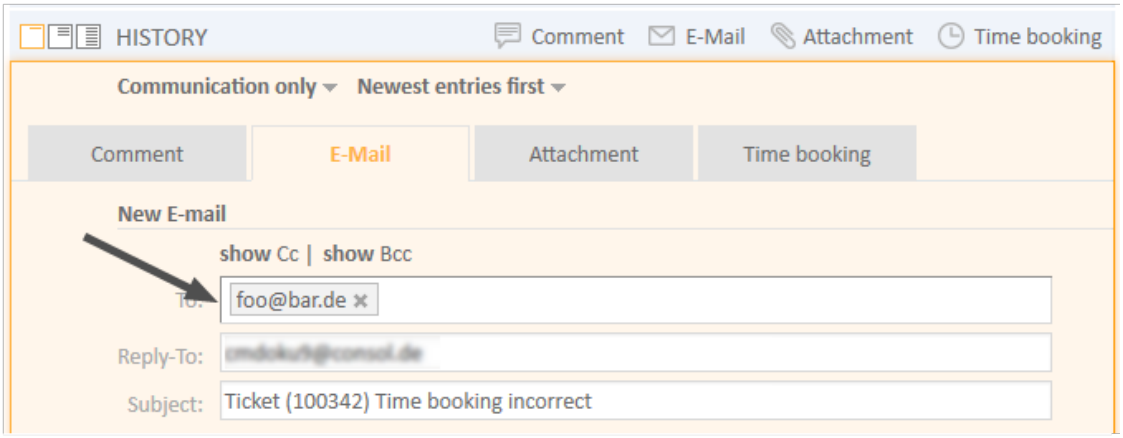


Figure 229: *ConSol CM Web Client - Email to a fixed email address*

- **recordLastUsedAcimTab**
  Boolean. Records last used ACIM tab. i.e., when you open the editor again, the tab (comment/email/attachment/time booking) will be opened that was open and active (i.e. some actions were performed, e.g., writing a comment) when you closed the editor last time.
- **reloadPageIfIE8onAcimSubmit**
  Boolean. Reloads page after a new ACIM submit, only for ***IE8***. This is a workaround for the problem that adding comments/emails may take a long time when using IE8.
- **removeContentOnTabSwitch**
  Clears text area content each time the tab panel in the editor is being switched. (boolean, default = "false", i.e., when you switch, for example, from the Ticket Email Editor to the Comment Editor, the text you have typed will remain in the editor panel and will not be removed)
- **timeBookingFeature**
  Boolean. Activates or deactivates the time booking support in acimSection (i.e., display the link to time booking and the tab in the editor), default = "true".

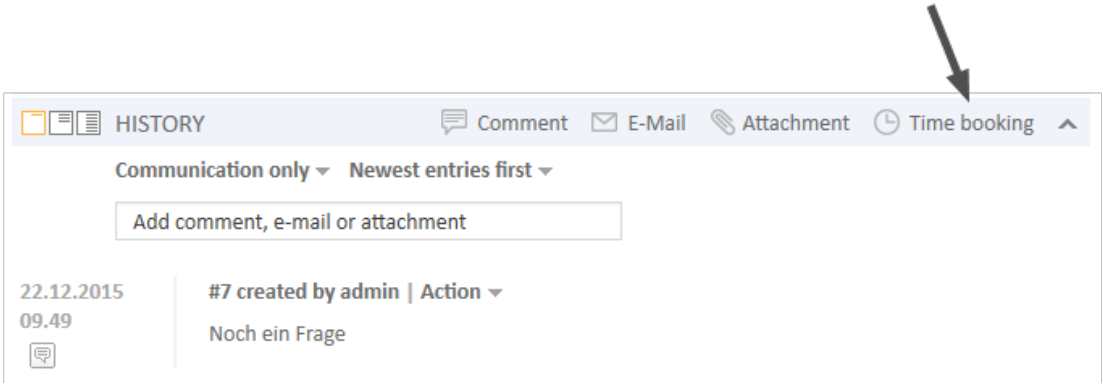


Figure 230: *ConSol CM Web Client - Activate time booking support (timeBookingFeature = true)*

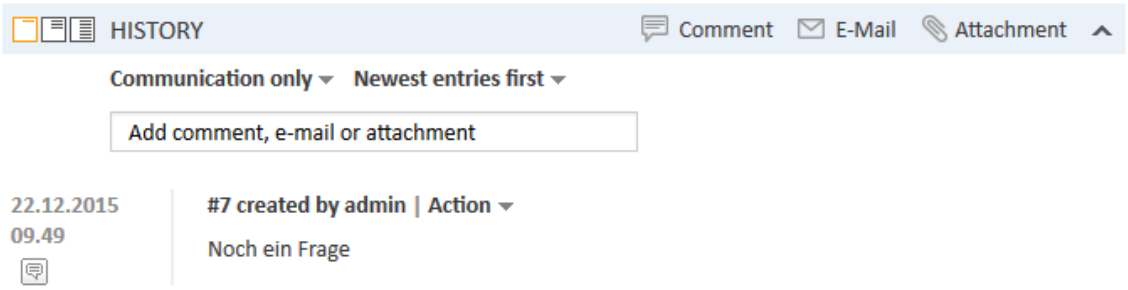


Figure 231: *ConSol CM Web Client - De-activate time booking support (timeBookingFeature = false)*

> Please note that
>
> - the value "false" in the *timeBookingFeature* hides the hyperlink from the time booking editor (see figure above). The engineer cannot change to display mode for it.
> - the visibility of the time booking section on the *userProfilePage* is configured via the *timeBookingSection* attribute *visible*!

- **extendedViewCharactersLimit**
  Integer. Defines the maximum number of characters which will be displayed in the ticket history for comments in extended view if this level is defined as *Extended (short)* in the class of text which is applied to the entry. Default 350. Please see the example in section Modifying the Display Behavior for the Basic and Extended Level to Optimize Ticket History Display.
- **basicViewCharactersLimit** (formerly **standardViewCharactersLimit**)
  Integer. Defines the maximum number of characters which will be displayed in the ticket history for comments in basic view if this level is defined as *Basic (short)* in the class of text which is applied to the entry. The text is cut off after the maximum number of characters defined in a page and formatting is removed. Emails will not show any header information. This provides a very concise informative view using very little space. Default 150. Please see the example in section Modifying the Display Behavior for the Basic and Extended Level to Optimize Ticket History Display.

analyticsPage (Type)

Here you can define attributes for CM/EBIA, opened using the *Analytics* link.

Attributes:

- **adminLinkVisible:**
  Determines whether the *Admin* link is displayed. Default: true.
- **iframeHeight:**
  Determines the height of the iFrame with the CM/EBIA application. Default: 1000
- **navigationLinksVisible:**
  Determines whether the links *Home*, *Create question* and *Open full application* are displayed. Default: true

attachmentSection (Type or Subscope)

Please note that in ConSol CM version 6.11, one page customization concerning the attachment section has been removed. The details are explained in the following figure.

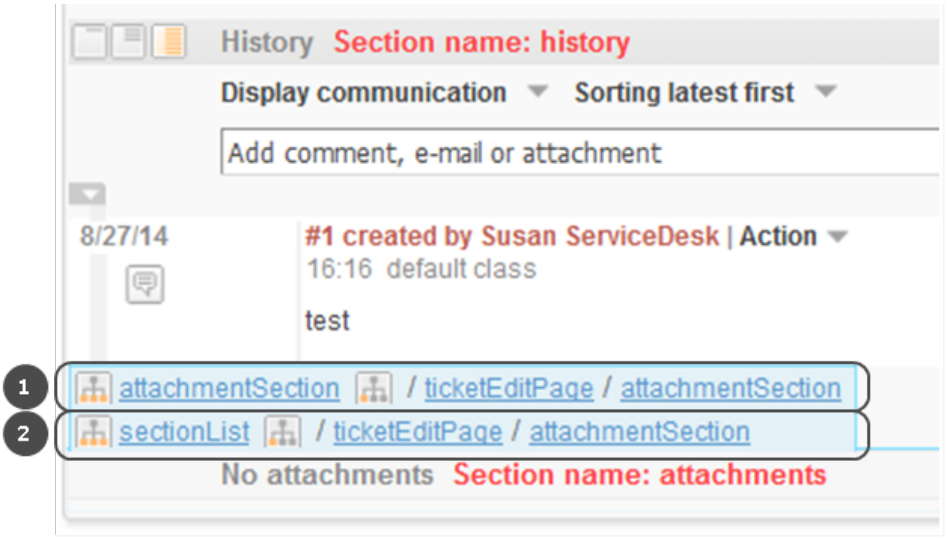


Figure 232: *Web Client in ConSol CM version 6.10.5. The page customization (1) has been removed in version 6.11, the page customization (2) is still available in version 6.11.*

autocomplete

(available e.g., on *userProfilePage*)

Attributes:

- **suffixCharactersToRemove**
  Any occurrences of these characters will be removed from the tail of each search pattern word. (java.lang.String)
  Example: If you search by using patterns, i.e., of the form “<Last name>, <First name>” then the search will not find any results because the word “<Last name>,” is not found in the search index because of the “,” at the end of the word. It is now possible to configure that certain characters should be removed from the tail of the search pattern word. So in this case the “,” will be ignored during the search and (see following figure) the engineer *Holler* is found.

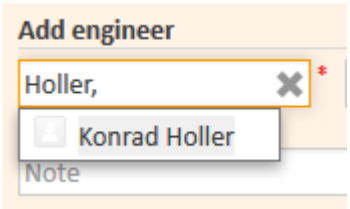


Figure 233: *ConSol CM Web Client - Input in a search field where characters should be ignored in the search*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

boxContent

Available for the following scopes:

- ticketEditPage
- userProfilePage

Attributes:

- **order**
  Specify the order of the sections within a ticket in csv format (e.g.: header, history, relations).
  Default values for standard installation:
  - **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



For other pages or custom projects check the section names shown in the header or in your ContentBuilder implementation.

This attribute has to be created before it can be changed. Use the *Create* button of the page customization.

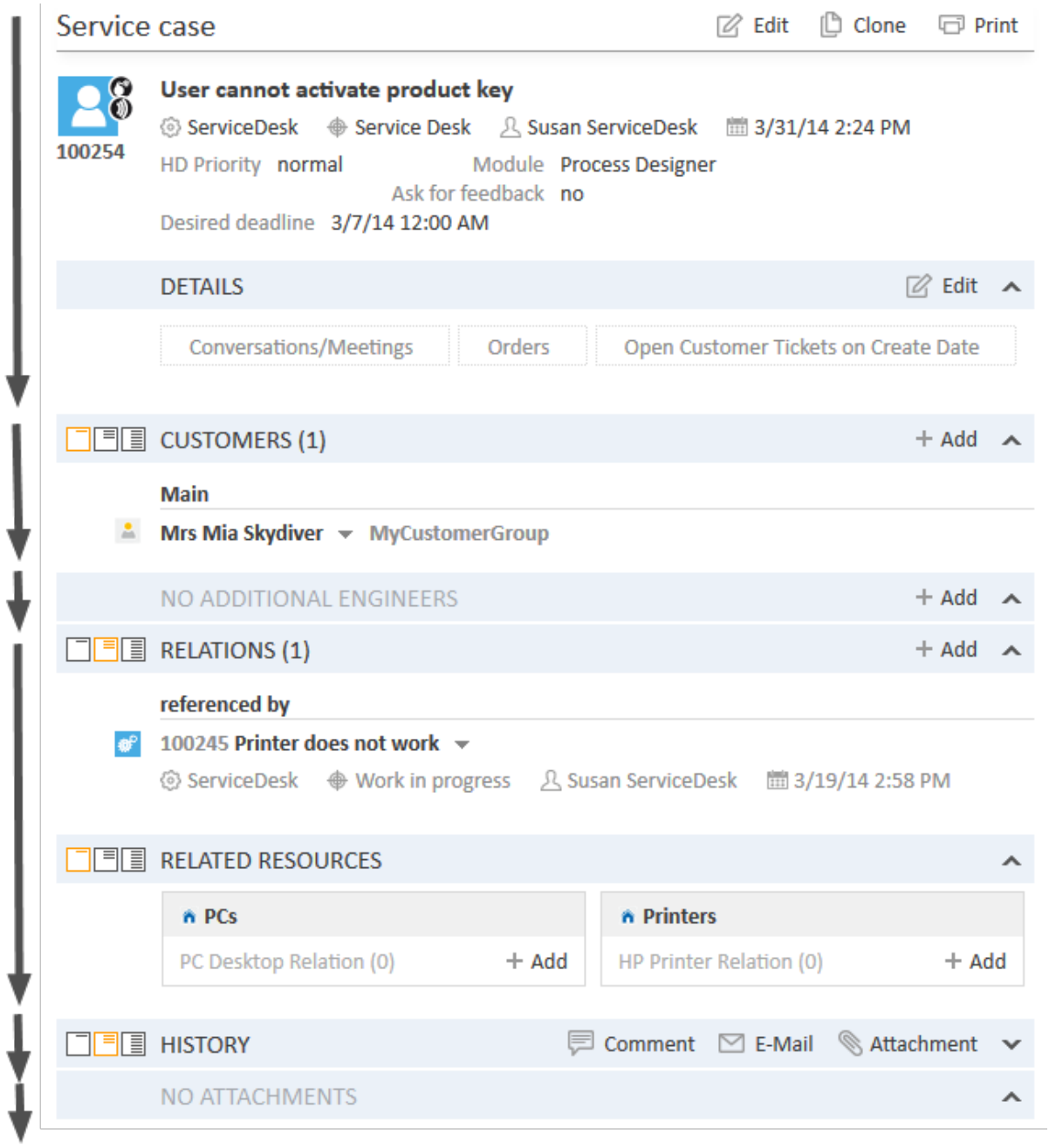


Figure 234: *ConSol CM Web Client - Default value for attribute boxContent / order*

calendarSection

This is used to configure the Calendar section on ticket, customer, and resource pages. Currently the respective attributes are used to configure the integration of Microsoft Exchange calendar views. Please see section Microsoft Exchange Calendar Integration for a detailed explanation of the feature and a list of all available attributes.

cmApplicationCustomization (Type)

Attributes:

- **activityKeyBindingEnabled**
  Boolean. Enable/disable shortcuts for activities/actions. Starting with version 6.10.1, the engineer can use shortcuts to address workflow activities of the ticket. Default “true”.
- **globalSearchFocusKeyBindingEnabled**
  Boolean. Enable/disable shortcuts for the Quick Search. If this attribute is enabled, it is possible to jump to the Quick Search field by pressing the “F” key. Default “true”.
- **systemNotificationTitle**
  String. Set the title of the notification in the Windows system notifications area.
- **submitACFOnEnterEnabled**
  Boolean. Enable/Disable submit ACF form on Enter pressing. Starting with version 6.10.1, the engineer can press ENTER to submit Activity Control Forms. Default “true”.
- **timeToShowWarningMessage**
  Integer. Time (in seconds) when a warning message is displayed before a session timeout occurs. The user can click the *Yes, extend my session* button in the message to avoid being logged out. The default value is “180”. The message can be disabled by setting the value to “0”.

The following two attributes influence the session timeout for engineers in the Web Client. In order to prevent the session from terminating while an engineer edits important information (and loosing the entered data with the end of the session), the following two attributes can be used to configure the automatic session renewal.

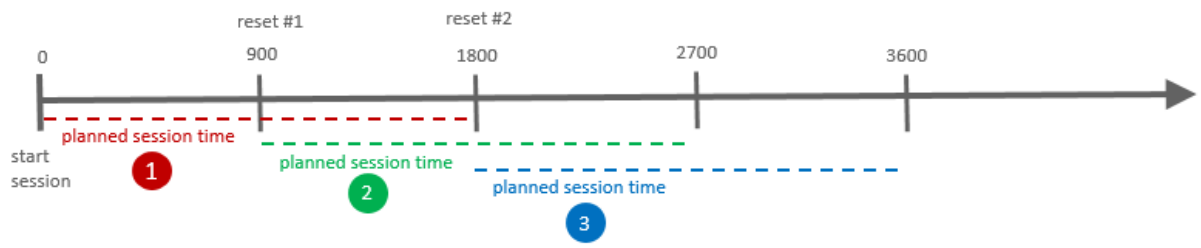


Figure 235: *Principle of renewing engineer session, example with updateTimeServerSessionActivity = 900.*

- **updateTimeServerSessionActivity**
  Integer. Set time in seconds to autoupdate server session activity. Default 900.
- **updateTimeServerSessionActivityEnabled**
  Boolean. Enable/Disable autoupdate server session activity. Default “true”. When set to “false”, the session renewal is not active.

The following actions underlie the control of the `updateTimeServerSession`:

- Creating a ticket, customer, or resource
- Creating a comment or an email in a ticket
- Creating a comment for a customer or for a resource
- Editing data fields of a ticket, customer, or resource
- Editing data field groups
- Activity control forms

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

cmRichTextEditor

Two different editors are available: TinyMCE (previous editor) and Froala (new editor). The used editor is determined in the system property cmweb-server-adapter, tinymce.enabled.

Available on the following pages:

- ticket edit page with opened editor
- ticket create page with opened editor
- template edit page

Attributes:

- **configuration**
  Only relevant for Froala. The configuration of the editor in JSON format, see https://www.froala.com/wysiwyg-editor/v2.0/docs/options for all the options.
- **editorFeatures**
  Only relevant for TinyMCE. Additional editor features. By default, all values are set, i.e., the respective buttons are available.
  The following table shows the available settings for TinyMCE and the Froala counterpart:

Value TinyMCE	Description	Froala configuration
SUB_SUP	Allows subscript and superscript.	subscript, superscript
INDENTS	Allows the ability to indent text.	indent, outdent
LISTS	Allows inserting lists and list formatting.	formatOL, formatUL
TABLES	Allows inserting tables.	insertTable
INSERT	Allows inserting text elements; for finer control you can use INSERT_EMOTICON, INSERT_CHAR and INSERT_IMAGE	insertImage, insertTable, insertLink
INSERT_EMOTION	Allows inserting emoticons.	not supported
INSERT_CHAR	Allows inserting special characters.	specialCharacters
INSERT_IMAGE	Allows inserting images.	insertImage
INSERT_LINK	Allows inserting/editing and removing hyperlinks. Valid protocol prefixes for the URL are “http://”, “https://”, “ftp://”, “ftps://”, “file://”, and “mailto:”. When omitting the protocol, the prefix “http://” is added automatically. For file links a change in the browser configuration might be necessary (see Types of Data Fields, *How can I display a file link?*).	insertImage

- **editorFonts**
  Only relevant for TinyMCE. The list of fonts for the editor in form <font display name>=<font name in system>.
  Fonts are separated by ';'. You can specify a comma-separated list of possible system names for each font.
  Example: Arial=arial,helvetica,sans-serif;Courier New=courier new,-courier;Verdana=verdana,geneva
- **font**
  This is the font used in the editor by default.
- **fontSizeValue**
  This is the default size for the text in the rich text editor. It must be one of the values form the list in the other customization *fontSizeValues*.
- **fontSizeValues**
  Only relevant for TincyMCE. This is the list of values shown in the font size selector of the rich text editor. It is a comma separates list of legitimate font size values including their unit like “6pt,10pt,12pt”. The values can be prepended by a label which is shown in the selector instead of the value itself: “tiny=6pt,regular size=10pt,large=12pt”.

Using a Script to Configure Settings

Instead of providing the values for the attributes directly in the page customization, you can reference a script which returns the desired values. This allows you to add logic when determining the editor settings. You need to create the script and reference it in the *configuration script* field of the page customization.

Example:

The default font size should depend on the customer group of the ticket’s main customer. You can reference a script with the following code.

```
ticket = ticketService.getById(ticketId);
def mainContact = ticket.mainContact
def cgName = mainContact.customerGroup.name
switch(cgName) {
   case "DirectCustomers": [fontSizeValue:"14pt"];
   break;
   case "Reseller": [fontSizeValue:"12pt"];
   break;
   default: [fontSizeValue:"10pt"];
   break;
}
```

Code example 25: *Script to set the default font for the rich text editor depending on the customer group of the ticket's main contact*

customerGroupSelector

Available directly under the root node in the Definition Section.

Attributes:

- **hiddenCustomerGroups**
  List of the technical names of customer groups which should not appear in Customer Groups Filter in the main menu. Comma-separated list of names of customer groups. Default: empty

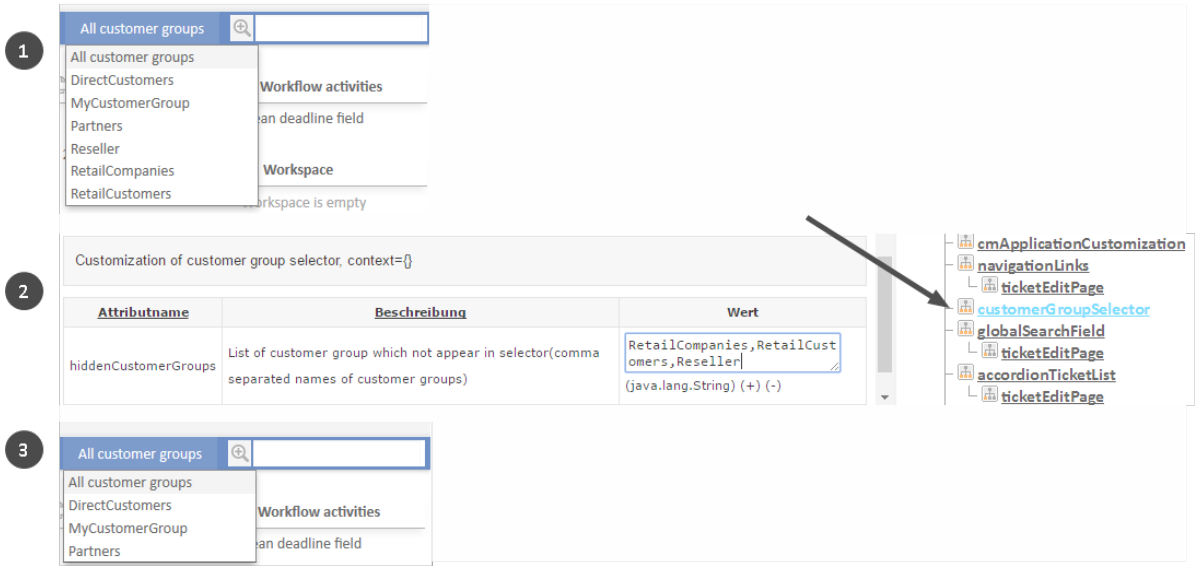


Figure 236: *ConSol CM Web Client - hiddenCustomerGroups*

customerSectionPanel

Here you can define whether the menu item *edit* should be displayed in the context menu for companies. Please note that the engineer must also have the according access rights (see section Roles) to edit company data.

Attributes:

- **additionalCustomersSortStrategy**
  The sort order of additional customers for a ticket can be defined using this attribute. If no value is set, the additional customers are sorted in the order they have been added to the ticket.

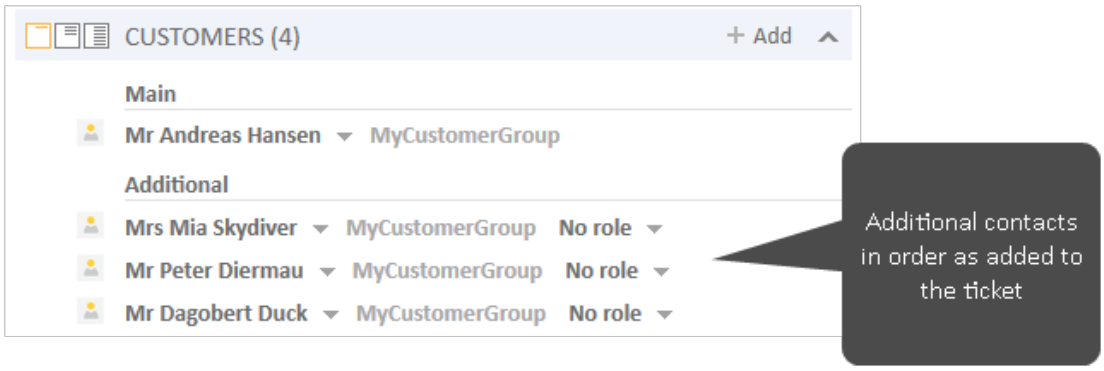


Figure 237: *ConSol CM Web Client - Page customization for sort order of additional customers (1)*

The following values are possible:

- **COMPANY_OF_MAIN_CUSTOMER**
  Customers are sorted by the company description with the company of the main customer first.
- **COMPANY**
  Customers are sorted by the company description.
- **CONTACT**
  Customers are sorted by the contact description.
- **ROLE**
  Customers are sorted by customer roles.

Multiple values can be provided as a comma-separated list. The default sort order (no value) works as before: customers are sorted as previously in the order of their addition.

The customer object descriptions used for the display of contacts and companies are taken from the template *<Customer object>.Ticket page*, see section Templates for Customer Data.

In the following example, the additional customers should be ordered by the name of the customer using the value *CONTACT*.

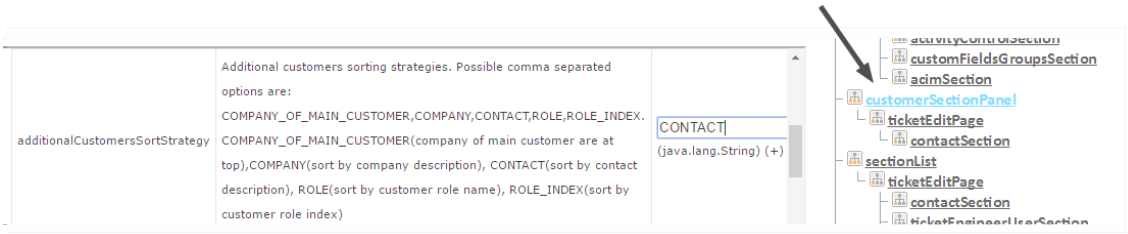


Figure 238: *ConSol CM Web Client - Page customization for sort order of additional customers (2)*

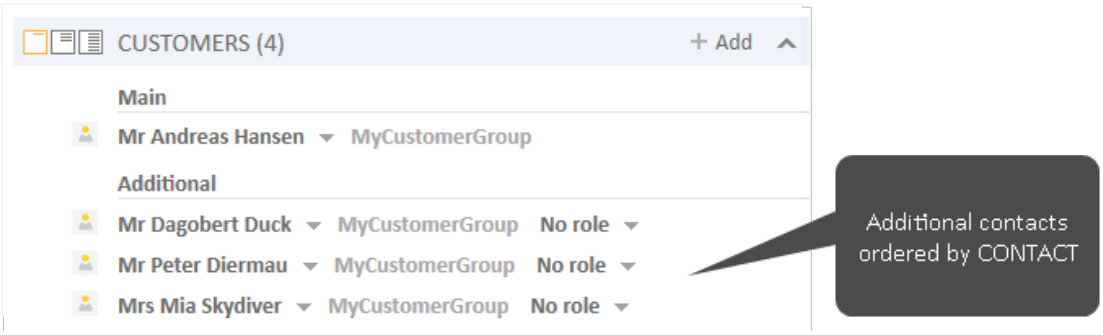


Figure 239: *ConSol CM Web Client - Page customization for sort order of additional customers (3)*

- **companyEditLinkVisible**
  (available in *customerSectionPanel* in ticket and on *companyEditPage* and *contactEditPage*)
  The visibility of the link for editing a company, default is “true”.

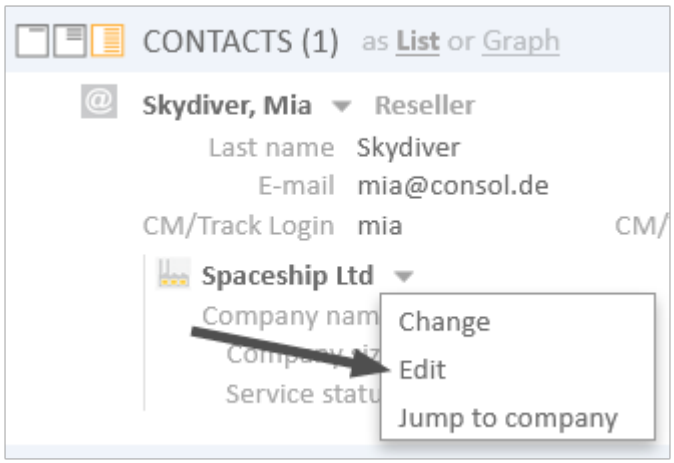


Figure 240: *ConSol CM Web Client - Visibility of company Edit link (companyEditLinkVisible = true)*

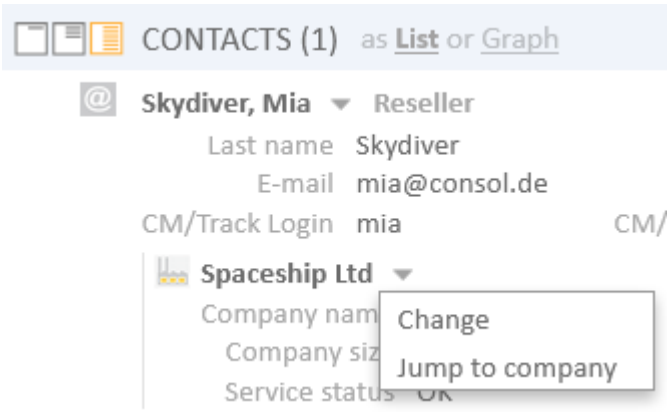


Figure 241: *ConSol CM Web Client - Visibility of company Edit link (companyEditLinkVisible = false)*

- **suggestionsDisabled**
  Determines whether a search field is displayed when selecting a contact for a ticket.

  If this attribute is set to “true”, the search field is displayed. The user sees suggested contacts when typing in the search field.

  To create a new contact, he needs to click *Create* to display the contact data fields. There are no suggestions when entering information in the data fields.

  If this attribute is set to “false”, the contact data fields are displayed directly and the user sees suggestions while typing.

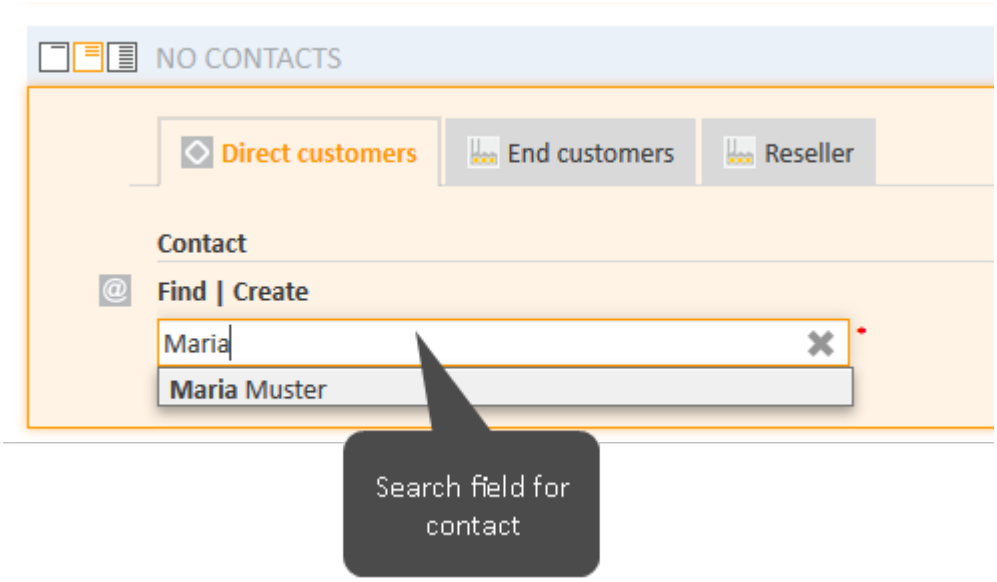


Figure 242: *ConSol CM Web Client - Contact search field (suggestionsDisabled = true)*

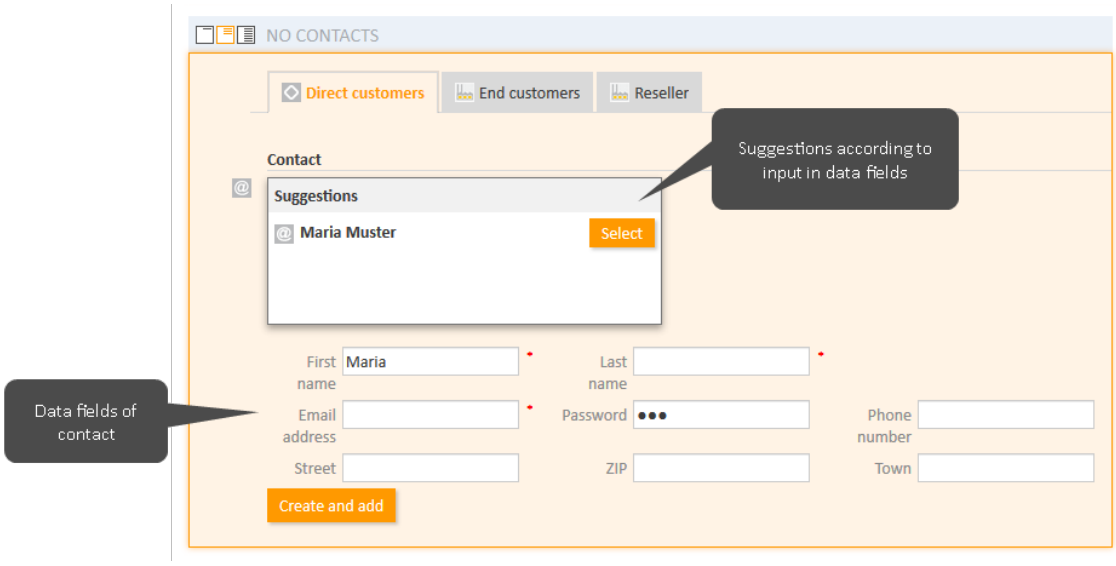


Figure 243: *ConSol CM Web Client - Contact data fields (suggestionsDisabled = false)*

The default value of this attribute is “false”.

customerTickets

Using the attributes of this type, you can configure the behavior of the ticket list on customer (i.e., contact and/or company) pages.

Attributes:

- **enabled**

  Boolean. Defines if the engineer and queue filter are available in the ticket list on customer (i.e., company and/or contact) pages. Default value: “false”.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

This will only apply if the number of tickets in the list exceeds the number which is set for the attribute *compactViewLimit*.

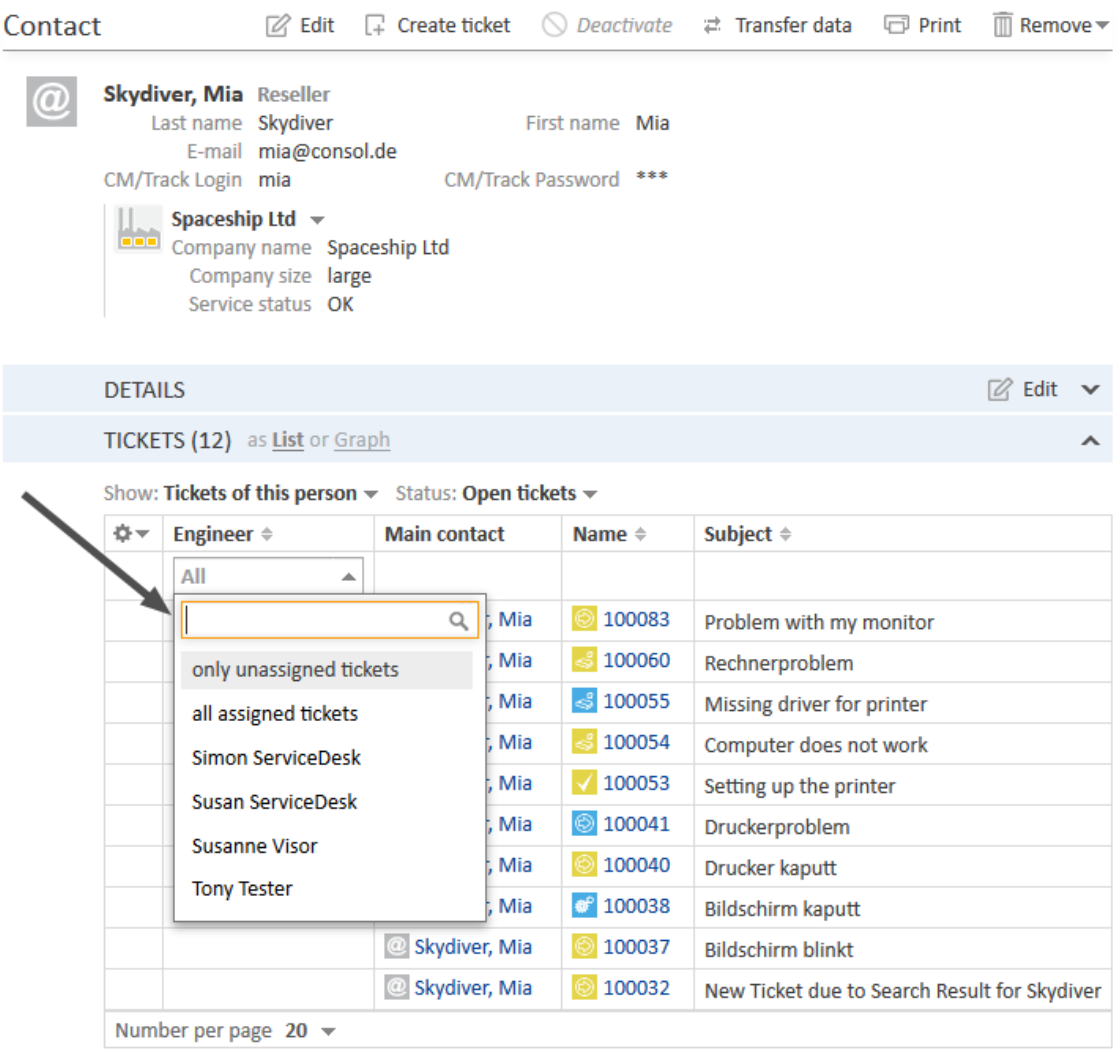


Figure 244: *List of customer tickets on a customer (here: contact) page, engineer and queue filters available*

The filter is applied to all types of customer pages, i.e., to contact and company pages if the attribute *enabled* is set for the type *customerTickets* (for an example see following figure).

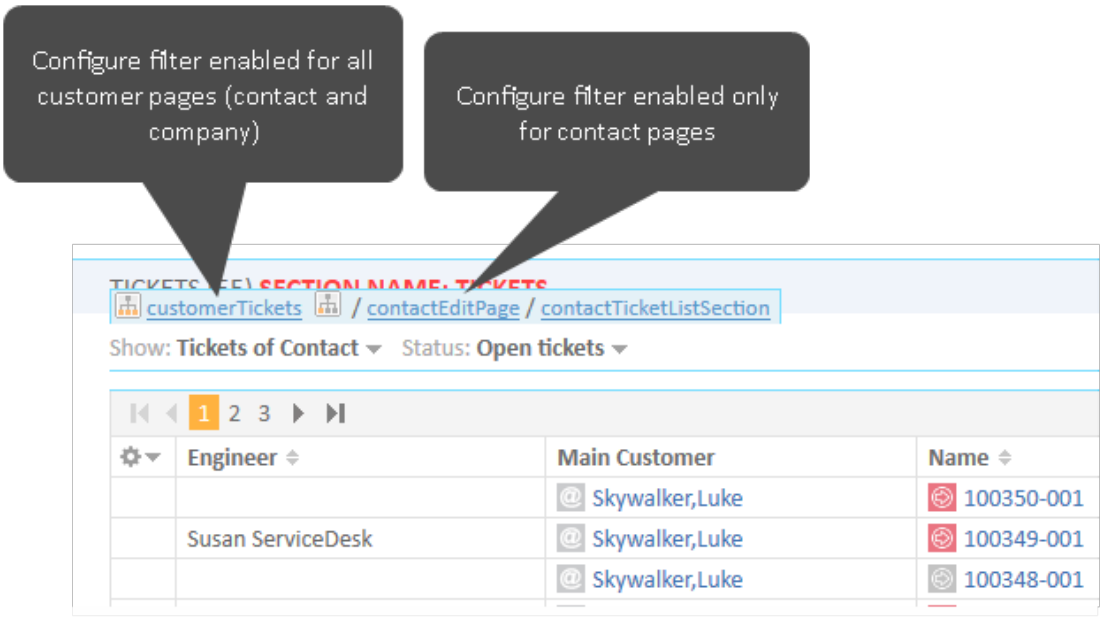


Figure 245: *Page customization type and scopes used to enable engineer and queue filer in customer ticket list*

The filter is applied only to contact pages when the attribute *enabled* is set for the scope *contactEditPage* or *ContactTicketListSection*, and the filter is applied only to company pages when the attribute *enabled* is set for the scope *companyEditPage* or *CompanyTicketsSection*.

> You need to re-index the tickets before using this feature. Otherwise, the filters are empty and a warning is written to the log files.

- **compactViewLimit**

  Integer. If the limit is exceeded the results are presented along with filtering feature. The default limit is “10”.

detailSearch (Type)

Available for the following scopes:

- searchDetailPage

Attributes:

- **closeDateTicketSearchParameterEnabled**
  Boolean field. When set to “true” (default), the search criterion *Close date* is available and the engineers can search for tickets which were technically closed in the provided date range. When set to “false”, the criterion is not available.

> Only tickets whose close date is indexed are found in the search. On updated systems, this means that tickets closed in a ConSol CM version prior to 6.11.2.0 are only found if the index was fully synchronized afterwards.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **criteriaForAllTypeOfContacts**
  Boolean field. When set to “true”, the search will include the main customers and the additional customers of tickets. When set to “false” (default), the search will apply the search criteria only to the main customers of tickets.
- **duplicatedCustomFieldLabelFormat**
  It allows to customize the label format used when several ticket fields have the same localized name. There are four values which can be used to create a unique label:
  - fieldName
  - groupName
  - fieldTechnicalName
  - groupTechnicalName

  The values “fieldName” and “groupName” are localized.
  Default format: ${fieldName} (${groupName}) (java.lang.String)
  This helps distinguish different ticket fields or customer fields with the same name, e.g., when they are offered in the drop-down menu in the detail search.
  Example:
  SalesProcess - Priority
  HelpdeskProcess - Priority
- **enableRowSelection**
  Boolean. Switches on the row selection functionality in the result tables, i.e. a checkbox will be available for each line, one checkbox to (de-)select all lines is available. Default value “true”. The checkboxes are only displayed if search actions are defined for the respective object type. The search action script will only be executed for the rows which have been selected (by the engineer) using the checkboxes.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

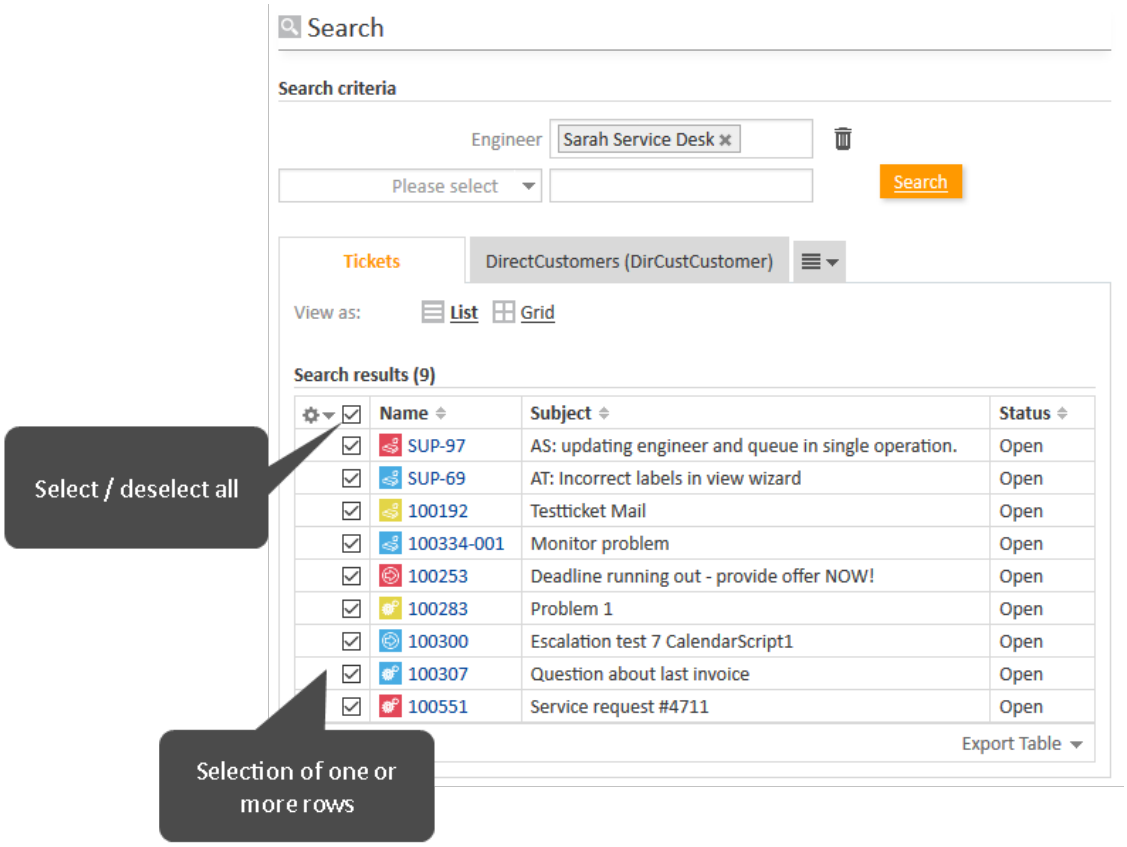


Figure 246: *ConSol CM Web Client - Search result table with detailSearch/searchDetailPage, enableRowSelection = true*

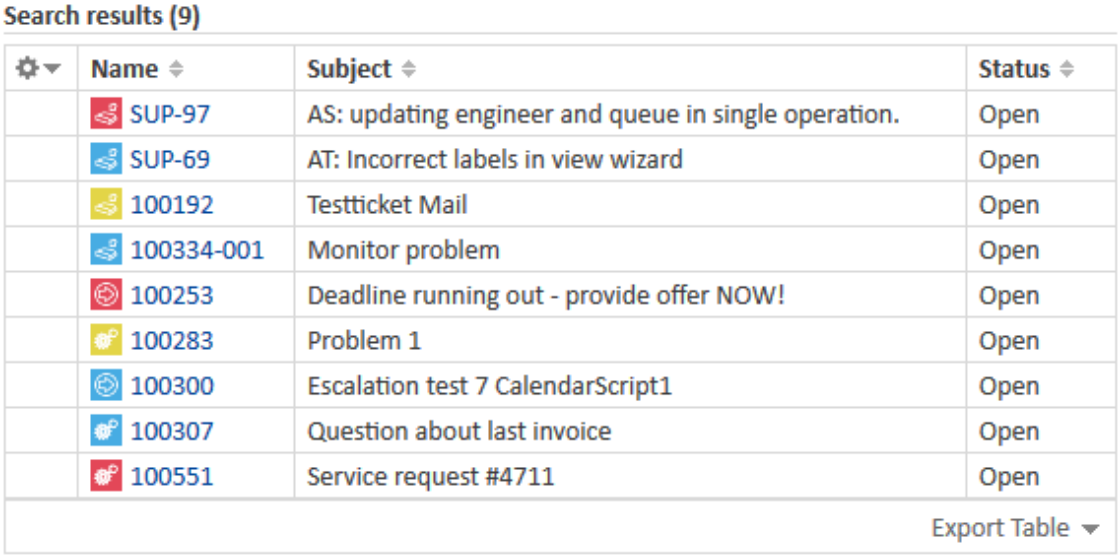


Figure 247: *ConSol CM Web Client - Search result table with detailSearch /searchDetailPage, enableRowSelection = false*

- **enableSearchForDeactivatedEnums**

Boolean. If “true”, deactivated enum and MLA values will be accessible in the drop-down menu in Detailed Search.

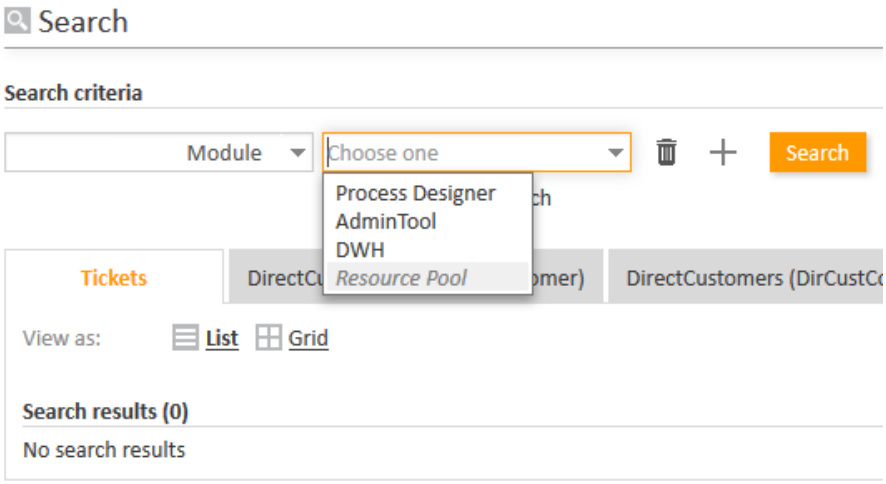


Figure 248: *'enableSearchForDeactivatedEnums' set to true in searchDetailPage: deactivated enum value is displayed*

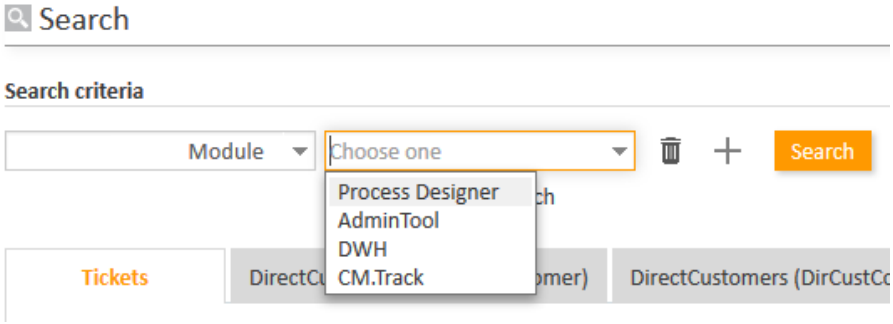


Figure 249: *'enableSearchForDeactivatedEnums' set to false in searchDetailPage: deactivated enum value is not displayed, but next active value is displayed in list*

- **enableSearchForOtherUsersTickets**
  Boolean. Decides if an engineer can search CM for all tickets (“true”) or only for tickets which are assigned to himself (“false”). Default “true”.
  This attribute has to be set on the (sub-)scope level, i.e., for the (sub-)scope *detailSearch/searchDetailPage*.
  The value of this attribute only influences the layout of the drop-down menu in the Detailed Search (see figures below). It does not have any influence on the results of the Quick Search.

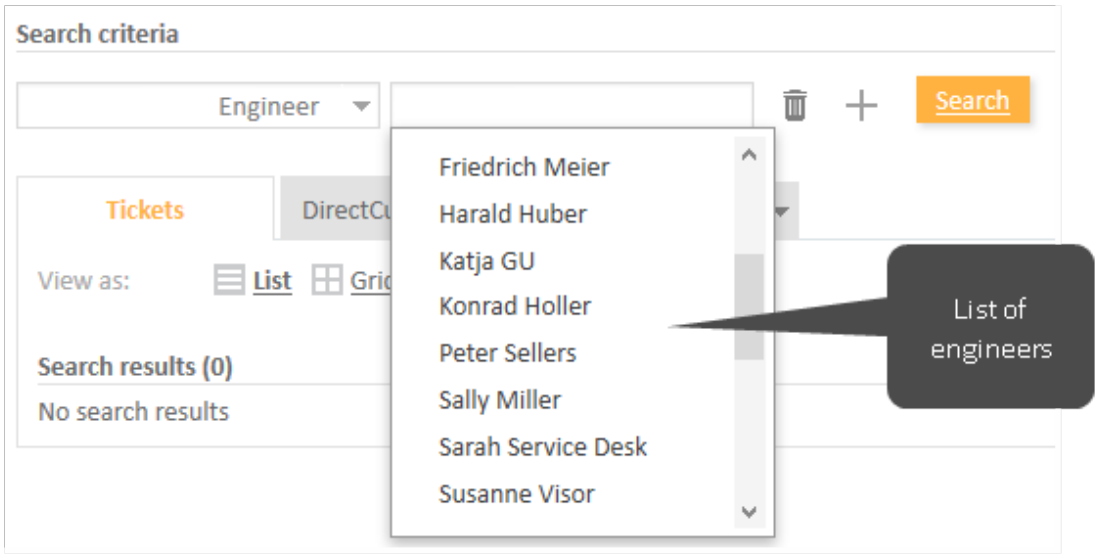


Figure 250: *ConSol CM Web Client - Detailed Search for engineers with enableSearchForOtherUsersTickets = true*

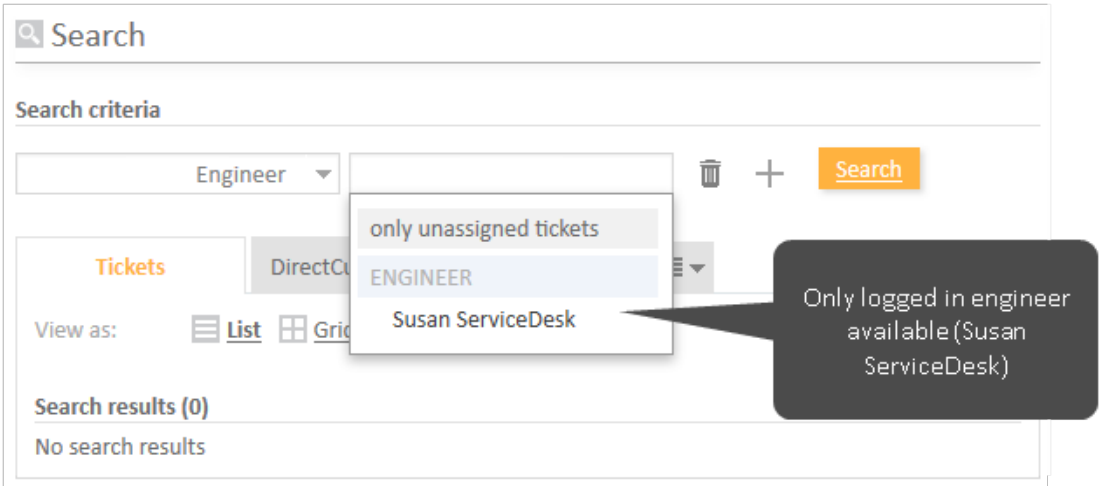


Figure 251: *ConSol CM Web Client - Detailed Search for engineers with enableSearchForOtherUsersTickets = false*

- **excludedUserNames**

  (usually has to be set for searchDetailPage)
  Login names provided here will be filtered out in engineers' criteria selector

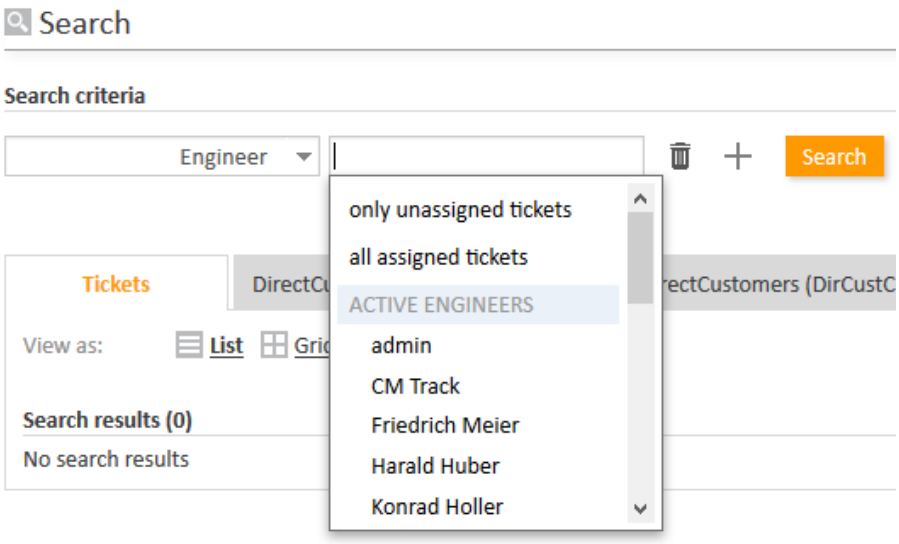


Figure 252: *Drop-down menu in engineer detail search before excluding user names*



Figure 253: *Using the page customization attribute excludedUserNames for two user names*

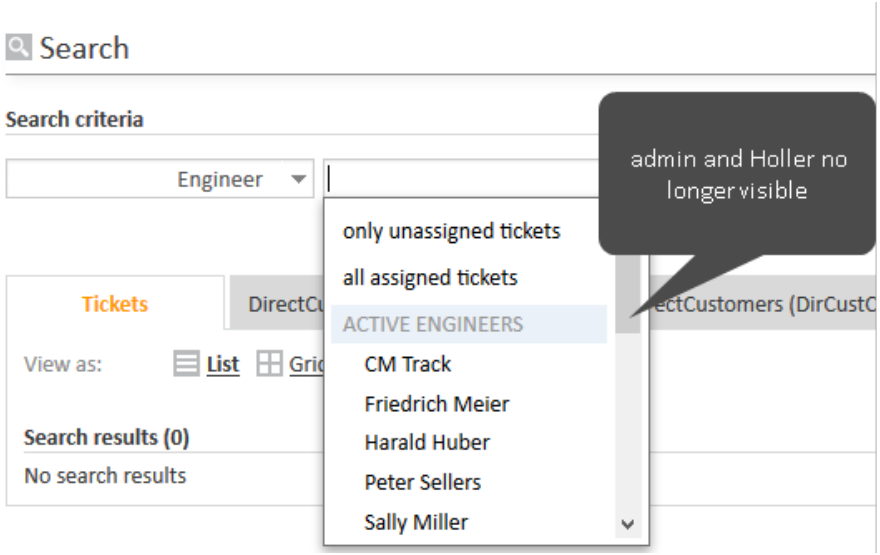


Figure 254: *Drop-down menu in engineer detail search after two user names have been excluded from the list*

- **maxGridTicketsNumber**
  Maximum number of tickets shown in grid view, i.e., in the entire grid, not in one column. The default value is “120”.

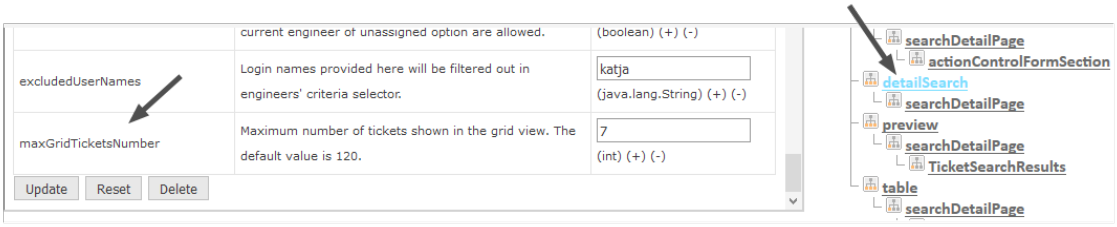


Figure 255: *ConSol CM Web Client - Using page customization to adapt maximum number of tickets shown in grid*

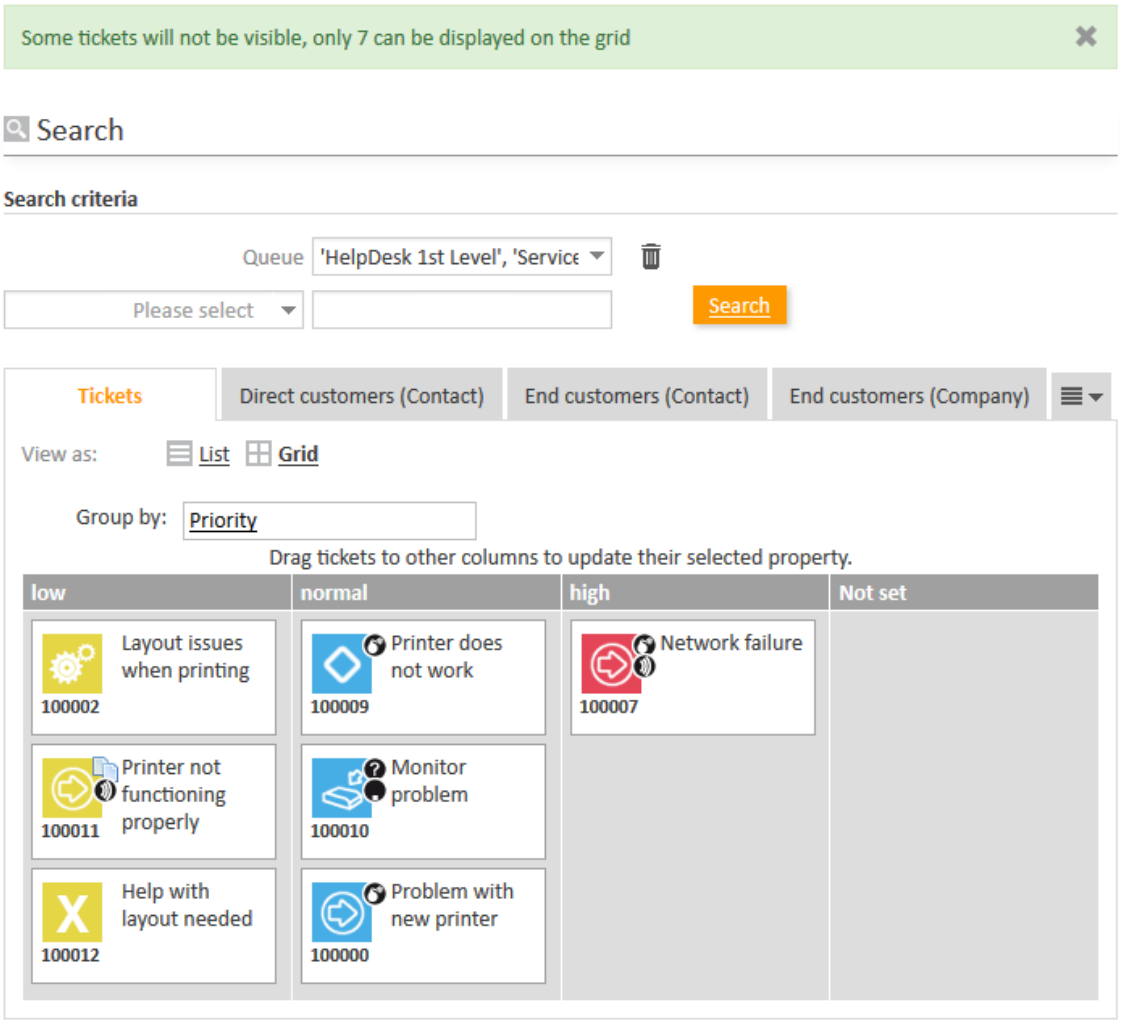


Figure 256: *ConSol CM Web Client - Maximum number in ticket grid with maximum 7*

The maximum number of tickets displayed in the ticket grid view can also be role-specific. For example, an engineer with the role *Teamlead* would see 100 tickets, whereas an engineer with the role *Helpdesk* would see 50 tickets. In the following simple example, all engineers with the role *ServiceDesk* see ten tickets, all others see 5. The configuration uses a script which is defined for the page customization type. The script is stored in the *Script and Template Administration* of the Admin Tool.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

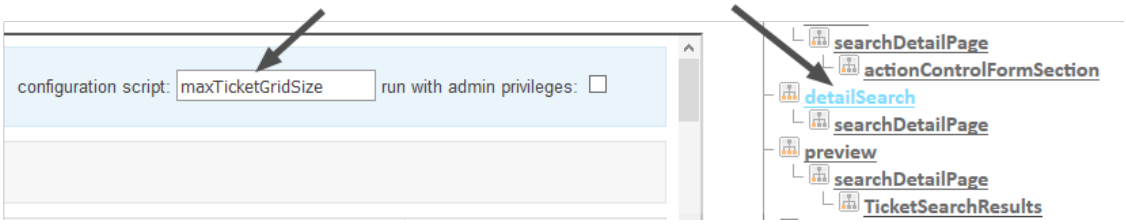


Figure 257: *ConSol CM Web Client - Defining a script for maximum number of tickets in grid view using page customization*

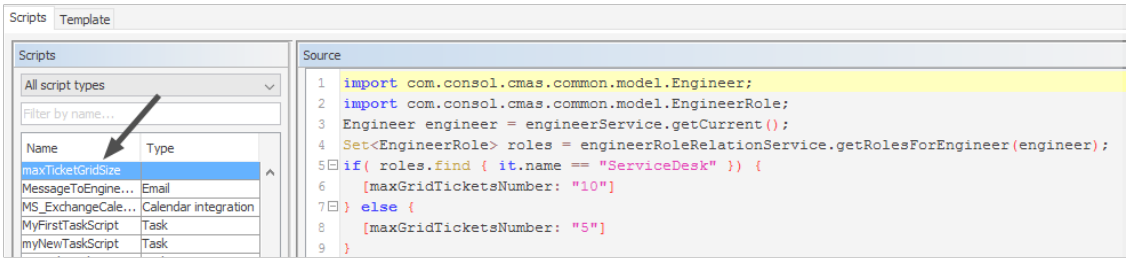


Figure 258: *ConSol CM Admin Tool - System, Scripts and Templates: Script for maximum number of tickets in grid*

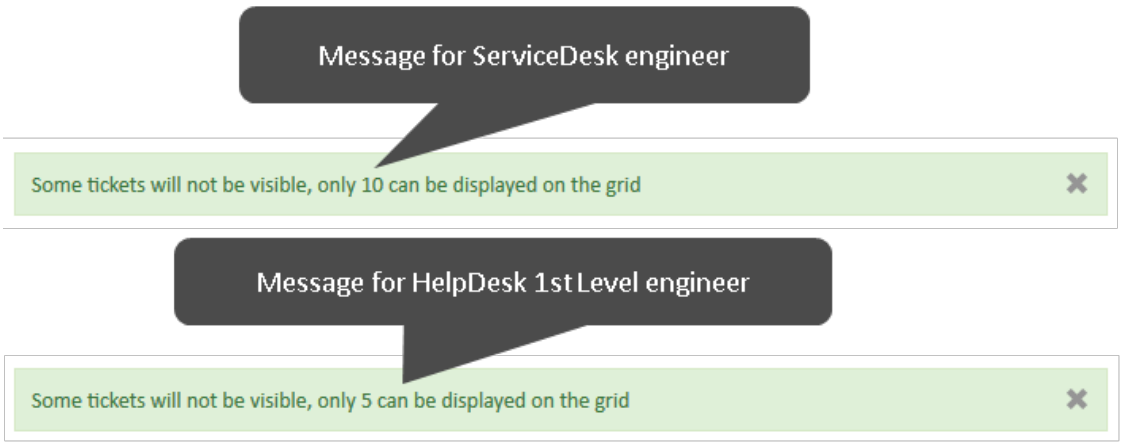


Figure 259: *ConSol CM Web Client - Number of tickets in ticket grid for different engineer roles*

engineerAutocomplete

Available for

- ticketCreatePage
- ticketEditPage
- userProfilePage
- contactEditPage

Attributes:

- **maxHints**
  Defines the maximum number of suggestions which is displayed. When set to “0”, all suggestions are displayed, no limit.
- **suffixCharactersToRemove**
  Occurrence of any of these characters will be removed from the tail of each search pattern word. (string, default: empty)

enumAutocomplete

(available on *ticketEditPage* for enums with annotation `enum type` = “autocomplete”)

Attributes:

- **maxHints**
  Defines the maximum number of suggestions which is displayed. When set to “0”, all suggestions are displayed, with no limit.
- **suffixCharactersToRemove**
  Occurrence of any of these characters will be removed from the tail of each search pattern word. (string, default: empty)

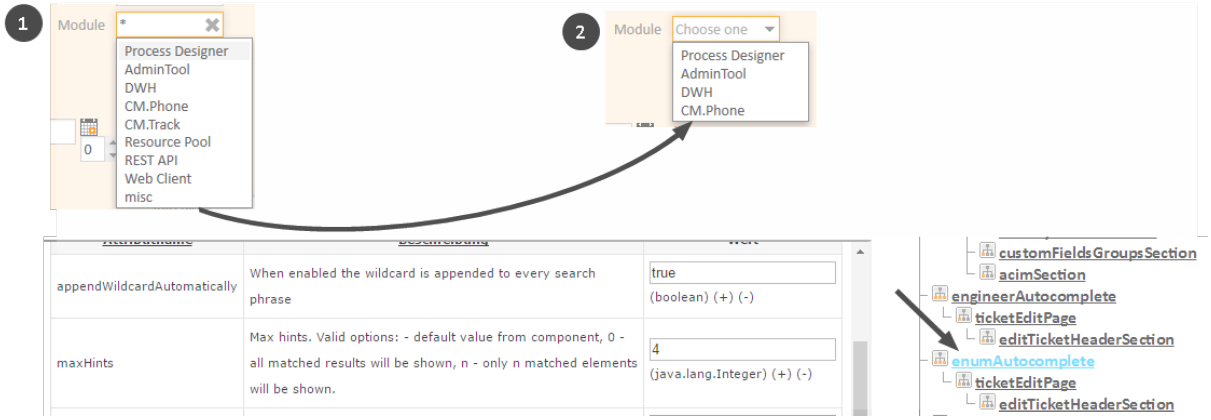


Figure 260: *ConSol CM Web Client - Using enumAutocomplete parameter to reduce number of hints*

froala (Type)

Available if the page (ticket, customer or resource in edit mode) contains a rich text field.

Attributes:

- **configuration**
  The configuration of the rich text field in JSON format. You can use the options provided by the Froala Editor (see https://www.froala.com/wysiwyg-editor/v2.0/docs/options).

Example:

Set the placeholder text displayed in the editor and define the selectable font sizes:

```
{"placeholderText": "Add a picture of the device","fontSize": ["8", "10", "12",
 "14", "18"]}
```

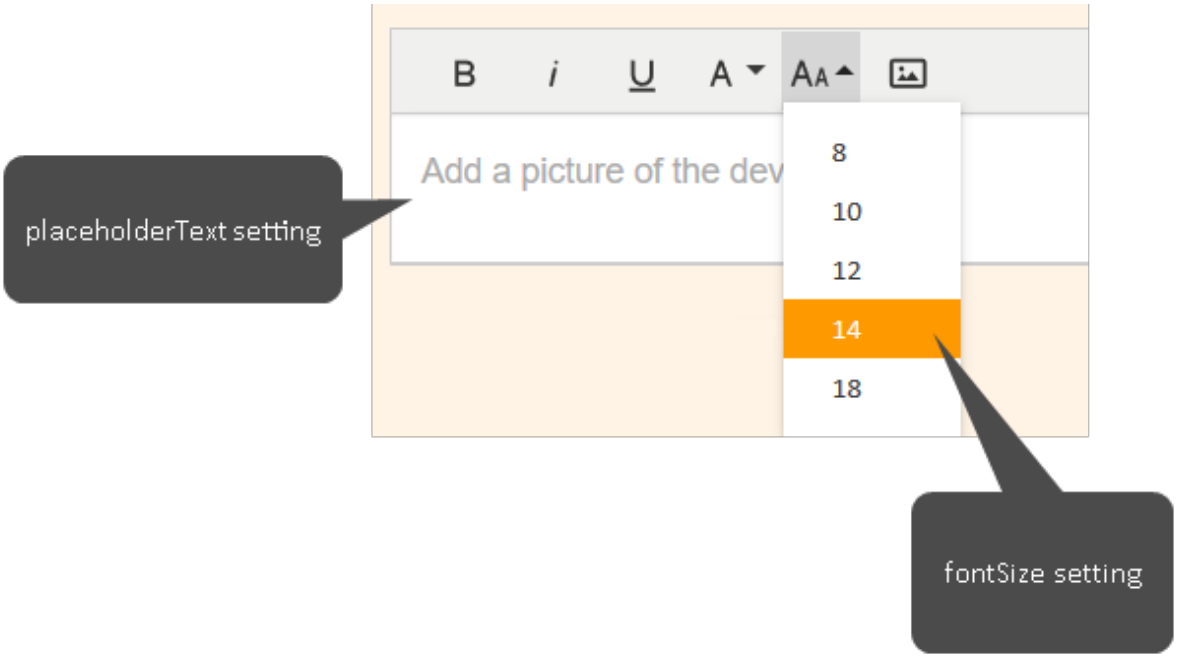


Figure 261: *ConSol CM Web Client - Adapted rich text field editor*

generalFeedback

This type is used to manipulate log entries in a clustered environment and to set the timeout for success messages.

Attributes:

- **appendHostInfo**
  Configures whether the cluster node information is shown together with the error message.
- **appendHostInfoSystemProperty**
  Identifies which system property will be used to provide the value shown in *appendHostInfo.*
  There are two properties which can be used:
  - cmas.http.host.port (default)
  - cmas.clusternode.id
- **successMessageTimeout**
  Integer. Defines the time in seconds before a success message (e.g. “Ticket XY has been created”) in the Web Client will be faded out. If set to “0”, the message will never fade out automatically but will have to be closed by the engineer. The default value is “0”.

globalSearchField (Type)

Here you can define the layout and the behavior of the *Quick Search* input field.



Figure 262: *ConSol CM Web Client - Quick Search input field*

Attributes:

- **appendWildcardAutomatically**
  Boolean. When enabled the wildcard ("*") is appended to every search phrase consisting of more than one character or number automatically, i.e., even when you have entered only part of a word it will be found. Default “true”.

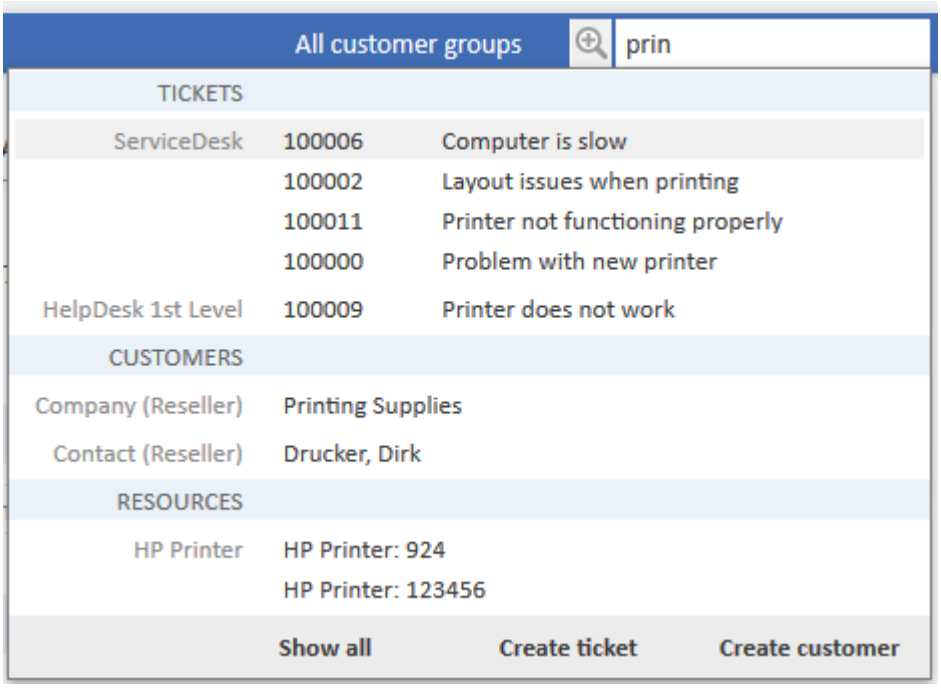


Figure 263: *Results of Quick Search with wildcard appended automatically (appendWildcardAutomatically = true)*

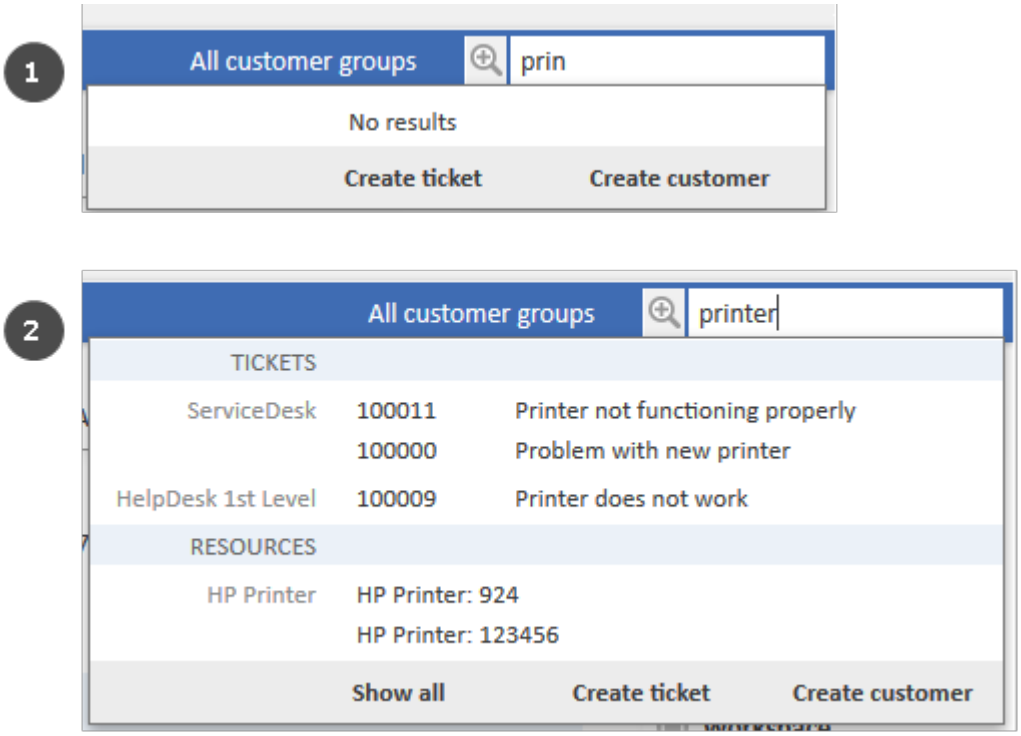


Figure 264: *Results of Quick Search with wild card not appended automatically (appendWildcardAutomatically = false)*

- **maxHints**
  Defines the maximum number of suggestions which is displayed. When set to "0", all suggestions are displayed, with no limit.
- **searchResultItemsOrder**
  Comma-separated values defining order and visibility of search result items. (java.lang.String)
  Possible values: CONTACT, COMPANY, TICKET
- **suffixCharactersToRemove**
  Occurrence of any of these characters will be removed from the tail of each search pattern word. (java.lang.String)

mailTemplate (Type)

(available on *ticketEditPage/acimSection*)

Attributes:

- **addingManyTemplateEnabled**
  Boolean. Makes it possible to compose an email by using more than one template.
  Obsolete with CM version 6.10.5.0, since starting with this CM version, a new Text Template Manager has been implemented which offers (besides other features) to insert more than one template into an email in the Ticket Email Editor or in a comment in the Ticket Comment Editor.
- **emailDisplayMode**
  String. Defines the format of the email address which is displayed in the To/Cc/From/Bcc fields in the Ticket Email Editor. Possible values: "FULL" (name and email address), "NAME", "EMAIL". Default "FULL".

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **engineerPersonalMailsIncluded**
  Boolean. Enable appending personal email feature. If the value of *engineerPersonalMailsIncluded* is “false”, new email addresses typed in the *To*, *Cc*, *Bcc* fields are not remembered and not present as suggestions next time.
- **includeTextBlocksInEmailTemplate**
  Boolean. Whether text blocks from email template will be included by default.
- **mailBodyLockedAfterTemplateSelection**
  Boolean. Indicates whether email body will be locked after template selection.
- **mailEncryptionAvailable**
  Boolean. Makes email encryption option available.
- **mailSelectionComponentWidth**
  The width of the email selection component (in pixels). (java.lang.Integer)
- **mailTemplateSortStrategy**
  Email template list sorting strategies. (java.lang.String)
  Default value: USAGE, NAME. Possible comma-separated options are: USAGE, NAME
- **maxElementLength**
  The maximum length of a single element. If variable's value is set to “0”, elements will not be trimmed. (java.lang.Integer)
- **minMailInputLength**
  Minimum input length that triggers email suggestion drop-down when engineer starts typing email recipients. Integer. Default “1”.
- **quotingFeature**
  Activate the quoting function in email content. (boolean)
- **selectionComponentWidth**
  The width of the mail/comment selection component (in pixels). Integer.
- **showBcc**
  Boolean. Defines if the *Bcc* field in the Ticket Email Editor is displayed. Default “true”.
- **showCc**
  Boolean. Defines if the *Cc* field in the Ticket Email Editor is displayed. Default “true”.
- **showReplyTo**
  Boolean. Defines if the *ReplyTo* field in the Ticket Email Editor is displayed. Default “true”.

  > Please note that the attribute *showReplyTo* is not the only parameter which influences ConSol CM behavior concerning display and use of the email Reply-To address! See section Scripts of Type Email for a detailed explanation!

- **showUniqueEmails**
  Boolean. Results in autocomplete email fields will be compared by email, but only first email will be used in results. If set to “true” then results will be compared by whole email description.
- **templateSortStrategy**
  String. Template list sorting strategies. Default value is: USAGE, NAME. Possible comma separated options are: USAGE, NAME. Default USAGE,NAME.

markersLibrary (Type)

This page customization type is available on the following page:

- templateEditPage (in the Text Template Manager)

Attributes:

- **excludedFields**
  The list of excluded ticket fields, e.g.: helpdesk.module,sales.priority. Fields which are listed here will not be offered in the *Select binding* menu any longer. In huge environments, it might save loading time of the templateEditPage to exclude some fields. However, it has to be absolutely clear that they will never be needed for template assignment or binding!
  The attribute can be set on type level or for the subscope templateEditPage/templateSection/bindings.

navigationLinks (Type)

Here you can define the display for several hyperlinks that are displayed in the main menu of the Web Client.

Attributes:

- **createContactLinkVisible**
  Whether the *createContact* (*Create customer*) link (menu item in the main menu) can be shown (boolean, default value is “true”).

  > In addition to this attribute being set to “true”, engineers must have appropriate permissions to see the *Create customer* menu item.

- **createTicketLinkVisible**
  Whether the *createTicket* link (menu item in the main menu) can be shown (boolean, default value is “true”).

  > In addition to this attribute being set to “true”, engineers must have appropriate permissions to see the *Create ticket* menu item.

- **externalLinks**
  External main menu items which will be appended to the main menu. This attribute may configure more than one external link (the order is significant, no characters between the link statements).
  Format (compatible with wiki): [http://link description] or [http://link description]@target
  This attribute may be used to integrate hyperlinks to the company's web site, to a reporting application, to a help page or to any other valid URL.
  Example: [http://www.consol.com ConSol][http://www.somewhere.com Somewhere]@

  > Links are opened in a new tab by default. If the link should be opened in the same tab, add @ after the link, e.g. [http://www.consol.com ConSol]@

- **formsLinkVisible**
  Whether the menu item *Forms* to access the web forms can be shown in the main menu (boolean, default value is “true”). See also section CM/Forms.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

> In addition to this attribute being set to “true”, engineers must have administrator permissions to see the link.

- **manageTemplateLinkVisible**
  Whether the menu item *Text template* (for the start of the ConSol CM Text Template Manager) can be shown (boolean, default value is “true”) in the main menu. See also section Text Templates.

> In addition to this attribute being set to “true”, engineers must have appropriate permissions (*Global Permissions - Write template*) to see the menu item *Text templates*.

- **officeTemplateLinkVisible**
  Whether the menu item *Document templates* (for the start of the Document Template Manager) can be shown (boolean, default value is “true”) in the main menu.

> In addition to this attribute being set to “true”, engineers must have appropriate permissions (*Global Permissions - Write template*) to see the menu item *Document templates*. CM/Doc has to be enabled in the system (see section CM/Doc).

- **overviewLinkVisible**
  Defines whether the menu item *Overview* can be shown in the main menu.

preview (Type)

The attributes define the behavior of the CM Web Client concerning the preview feature for result tables where tickets are listed.

Available on

- the Search Detail Page (tab *Tickets*): preview**/searchDetailPage/TicketSearchResults**
- the resource page (ticket section): preview**/resource/ticketRelationsSection/ResourceTicketRelationSearchResults**

Set the attributes for the subscope printed in bold font.

Attributes:

- **enabled**
  Boolean. Enable/disable the inline preview feature. Default “true”.
- **initialImagesHeight**
  Integer. Height of images in the preview in pixels. This applies to both image attachments and inline images within comments or emails. Default “150”.
- **previewHeight**
  Integer. Height of the attachment preview in pixels. Default “200”.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

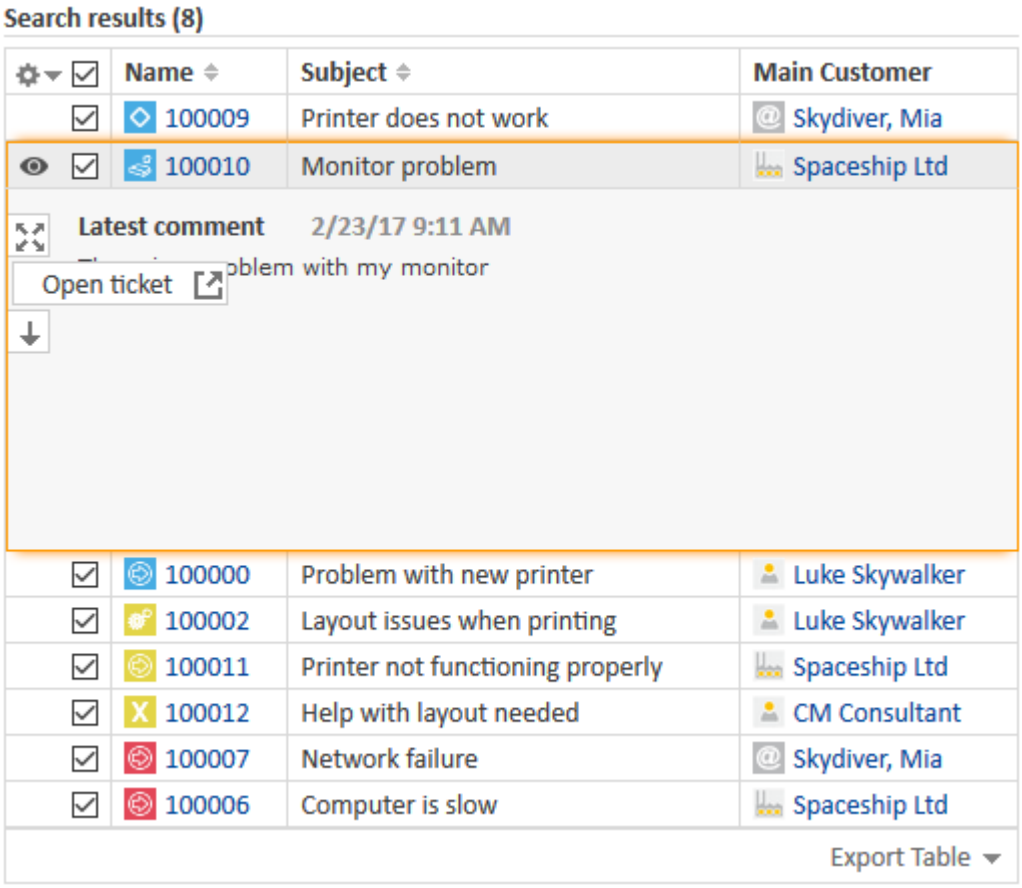


Figure 265: *ConSol CM Web Client - Preview with /searchDetailPage/TicketSearchResults, enabled = true*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

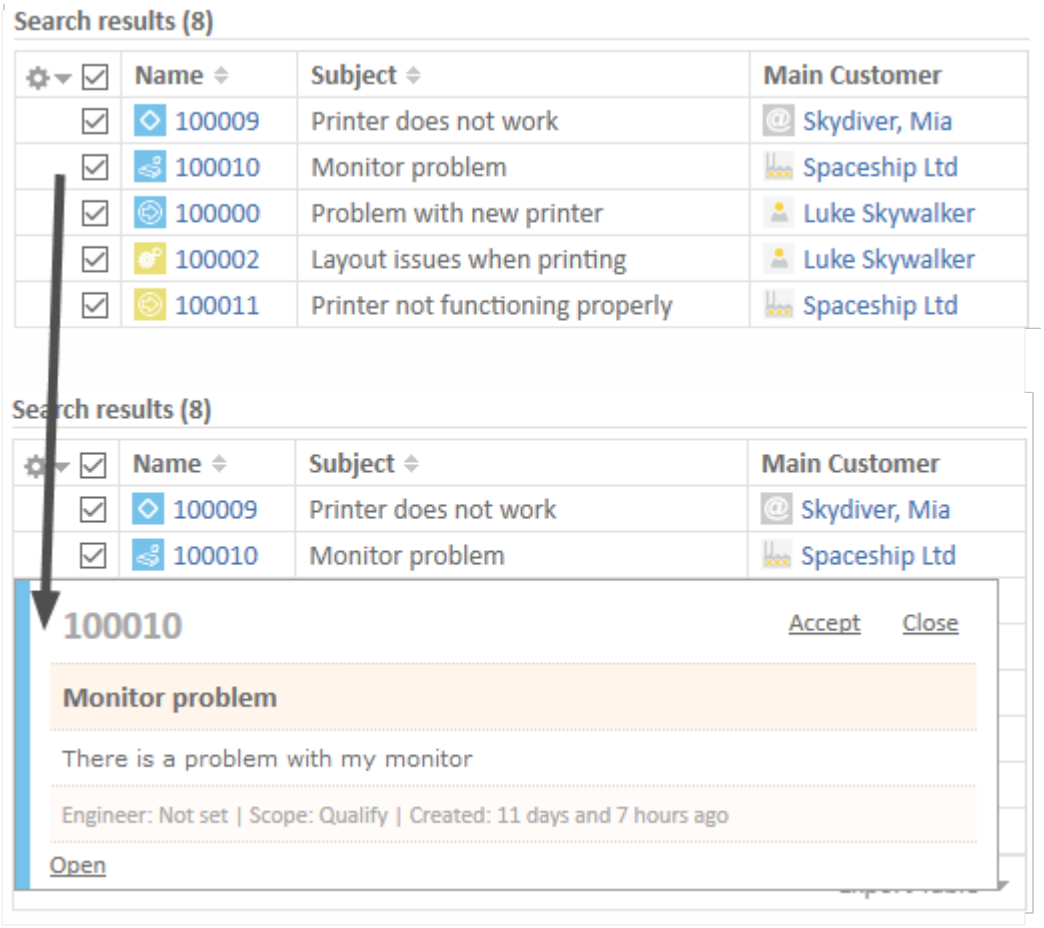


Figure 266: *ConSol CM Web Client - Preview with /searchDetailPage/TicketSearchResults, enabled = false*

For a detailed description of the preview functionality, please refer to the *ConSol CM User Manual*.

resourceRelations

Available e.g., on ticketEditPage.

Attributes:

- **counterVisible**
  Whether the counter should be shown in the section header. The default value is “true”.
- **defaultRelationSortStrategy**
  Additional resource relation sorting strategies. Possible options are: CREATION_DATE, DESCRIPTION, Default CREATION_DATE.
- **state**
  The visibility mode of the section, possible values are [expanded, collapsed, collapsed_and_pre-load, hidden], default: “expanded”.

resourceRelationsPanel

Available, e.g., on the TicketEditPage.

Attributes:

- **preserveOrder**
  Whether to preserve order of elements. By default elements will be placed in an optimal position based on available vertical space. Default is “false”.

resourceTypes

Available, e.g., on the Resource Dashboard

Attributes:

- **preserveOrder**
  Whether to preserve the order of elements. By default elements will be placed in an optimal position based on available vertical space. Default is “false”.

section (Type)

Available in the following scopes:

- ticketEditPage
- contactEditPage
- companyEditPage
- userProfilePage

Attributes:

- **state**
  The visibility mode of the section. A detailed explanation is provided under *sectionList*

sectionList (Type)

Available in the following scopes:

- ticketEditPage
- contactEditPage
- companyEditPage

Attributes:

- **counterVisible**
  Whether the counter should be shown in the section header. The default value is “true”.

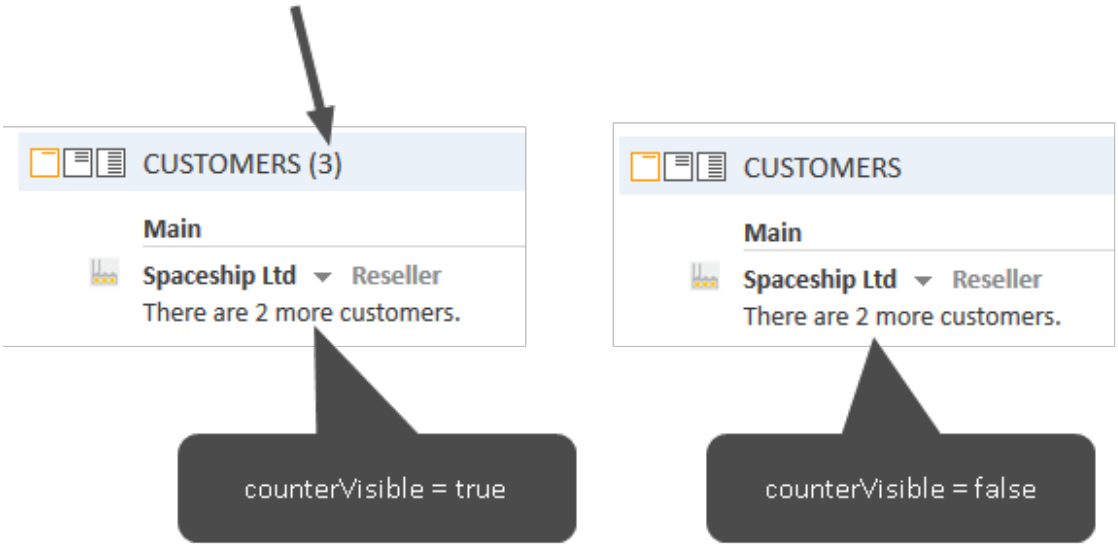


Figure 267: *ConSol CM Web Client - Attribute counterVisible*

- **state**
  The visibility mode of the section. Possible values are [expanded, collapsed, collapsed_and_preload, hidden]. The default value is “expanded”. This mode defines the initial visibility mode after start of a session. The engineer can change the visibility of a section afterwards, but this will not be saved. When the next session is started (at the next login), the initial visibility mode will be applied again.
  - **expanded**
    Default, data are shown initially
  - **collapsed**
    Data are not shown initially and will be loaded only on demand, can provide some performance improvement
  - **collapsed_and_preload**
    Data are not shown initially, but will be loaded
  - **hidden**
    The section is completely hidden and cannot be made visible. Can only be reversed using tools like the application server admin console.

> **Attribute HIDDEN will hide all customer data!**
>
> When the attribute *state* = “hidden” has been set for a section/scope, this scope will not be displayed in the page customization tree and thus will not be available in the page customization anymore. You can reverse this setting only using tools which directly access the Java beans, e.g., ***JConsole*** for JBoss. So please think twice before applying this value.

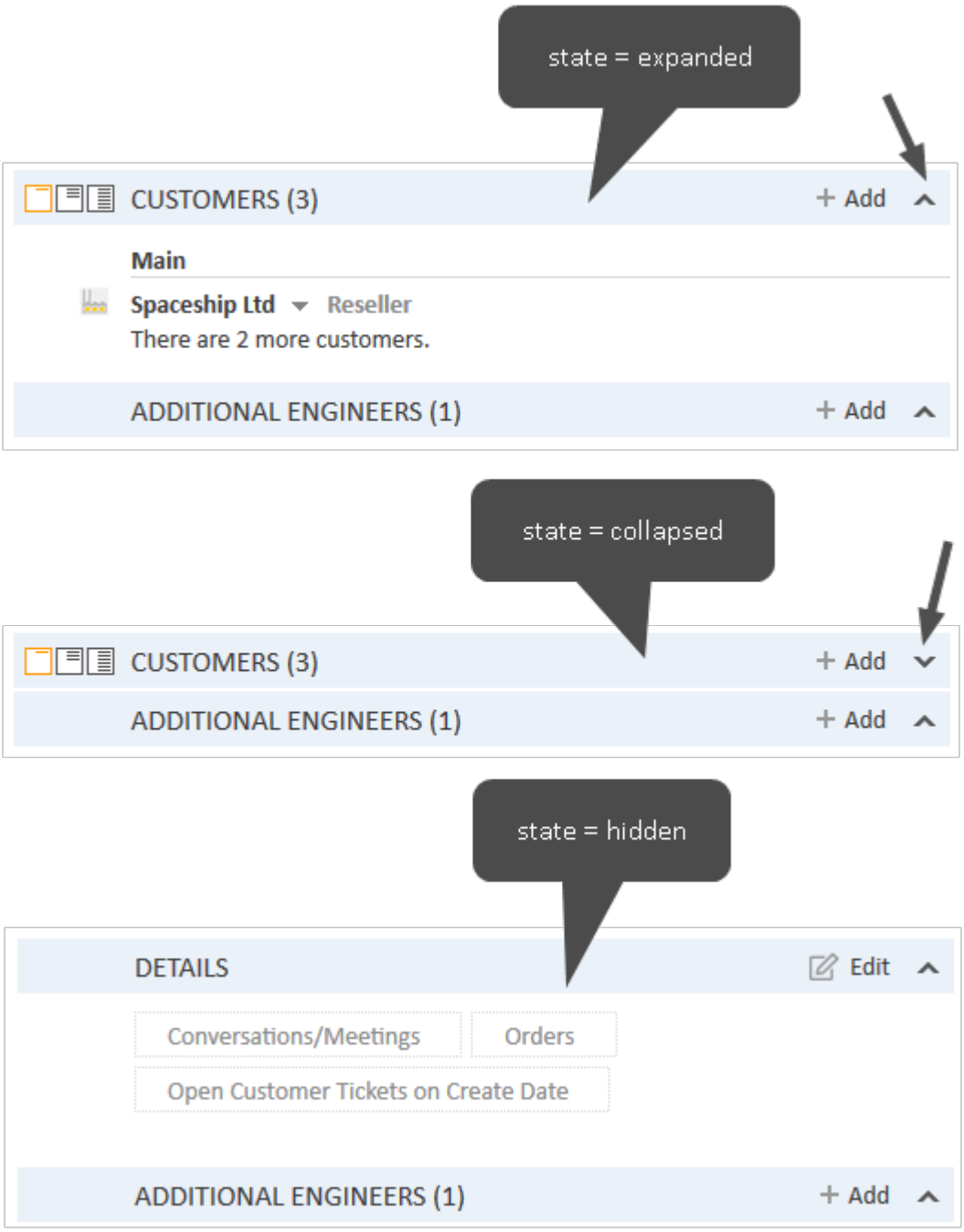


Figure 268: *ConSol CM Web Client - Attribute state (example: Customers section of a ticket)*

table

This is used to configure the csv export functionality in search result pages. Please refer to section CSV Export of Search Results for an explanation.

template

Available on all pages which feature the Rich Text Editor with text template insertion. The format is always defined on the lowest scope for which the attribute value is available.

Attributes:

- **includeTextBlocksInTemplate**
  Used to define whether the text blocks which are included in a template are selected by default, e.g. whether they are inserted together with the template or whether the user has to select them explicitly. The default value is “true”.
- **templateNumberFormat**
  Used to define the format of numeric variables, i.e., of types integer (number) or decimal (fixed-point number) for the placeholders (markers) in text templates.

  The default value is “number”. It represents the default format and can be used to return to the default after having a dedicated format configuration previously. The format definition syntax is explained in the Java documentation for the class DecimalFormat.

  The most important elements are:

  - “0” - shows a digit in this place always, will show “0”, i f no digit is in this position.
  - “#” - shows only, if the number has a digit in this position.
  - “.” - decimal separator
  - “,” - internal grouping separator for integer/decimal positions

  For example “00000.#####” as an attribute value will always show five digits before the decimal separator and show up to five digits after the decimal separator, depending on how many are present for decimal numbers. If the number has less than five digits before the decimal separator, the positions are filled with zeros, e.g. 125 is displayed as 00125.

ticketCreatePanel

Attributes:

- **queuesSortOrder**

  Determines the order of the queue tabs. Enter the technical names of the queues separated by comma to display the queues in this order. The queues which are not configured in this attribute are displayed in the default sorting (alphabetically) after the ones which are configured.

  > If the logged in user has selected a default queue in the user profile, this queue is always displayed first, independently from the configured sort order.

- **scrollSpeed**
  Scroll speed in milliseconds. The attribute is used to determine the speed of page scrolling when you click on a handler on the right side of the main page section.

  The value will determine how long the animation will run. The default value is “200”, meaning that it takes 200ms to scroll the page.
- **topBottomPageButtonVisible**

  Determines if the handler to scroll the page is visible (“true”) or not (“false”). The default value is

"false".

ticketList (Subscope)

see accordionTicketList (Type).

ticketPanel

Attributes:

- **scrollSpeed**
  Scroll speed in milliseconds. The attribute is used to determine the speed of page scrolling when you click on a handler on the right side of the main page section.

  The value will determine how long the animation will run. The default value is "200", meaning that it takes 200ms to scroll the page.

- **topBottomPageButtonVisible**

  Determines if the handler to scroll the page is visible ("true") or not ("false"). The default value is "false".

TicketRelation

Available in the following scopes:

- TicketRelationSection (e.g., on the resource page)

Attributes:

- **compactViewLimit**
  If the limit is exceeded, the table is presented along with a filtering feature. The default limit is "10".
- **defaultRelationSortStrategy**
  Additional resource relation sorting strategies. Possible options are: CREATION_DATE (default), FIRST_SORTABLE_COLUMN (sort by the first column which is sortable)

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

ticketsAutocomplete

(available on the relations addition form)

Attributes:

- **maxHints**
  Defines the maximum number of suggestions to display. When set to “0”, all suggestions are displayed, with no limit.

ticketsBookingAutocomplete

(available on the time booking addition form of the *userProfilePage*)

Attributes:

- **maxHints**
  Defines the maximum number of suggestions to display. When set to “0”, all suggestions are displayed, with no limit.

timeBookingSection

(available on *userProfilePage*)

Attributes:

- **state**
  Possible values are:
  - **expanded**
    Default, data are shown initially
  - **collapsed**
    Data are not shown initially and will be loaded only on demand.
  - **collapsed_and_preload**
    Data are not shown initially, but will be loaded.
  - **hidden**
    The section is completely hidden and cannot be made visible). Can only be reversed using tools like the application server admin console.

    Attribute **hidden** will hide the time booking section permanently.

    When the attribute *state* = “hidden” has been set for a section/scope, this scope will not be displayed in the page customization tree and thus will not be available in the page customization anymore. You can reverse this setting only using tools which directly access the Java beans, e.g., ***JConsole*** for JBoss. So please think twice before applying this value.

    Please keep in mind that the visibility of the time booking section on the ticket page is configured via the acimSection, attribute *timeBookingFeature*.

unitAutocomplete

(available on the customer addition and creation forms)

Attributes:

- **maxHints**
  Defines the maximum number of suggestions which is displayed. When set to “0”, all suggestions are displayed, with no limit.

unitFormPanel

Available on:

- contactCreatePage
- ticketEditPage
- contactEditPage, e.g., unitFormPanel / ticketCreatePage / contactSection

Attributes:

- **maxSuggestions**
  This refers to the customer section which is displayed when you create a new ticket. Here, suggestions for customers are displayed if matching hits are found in the database. The number of suggestions which are displayed can be configured using this attribute.

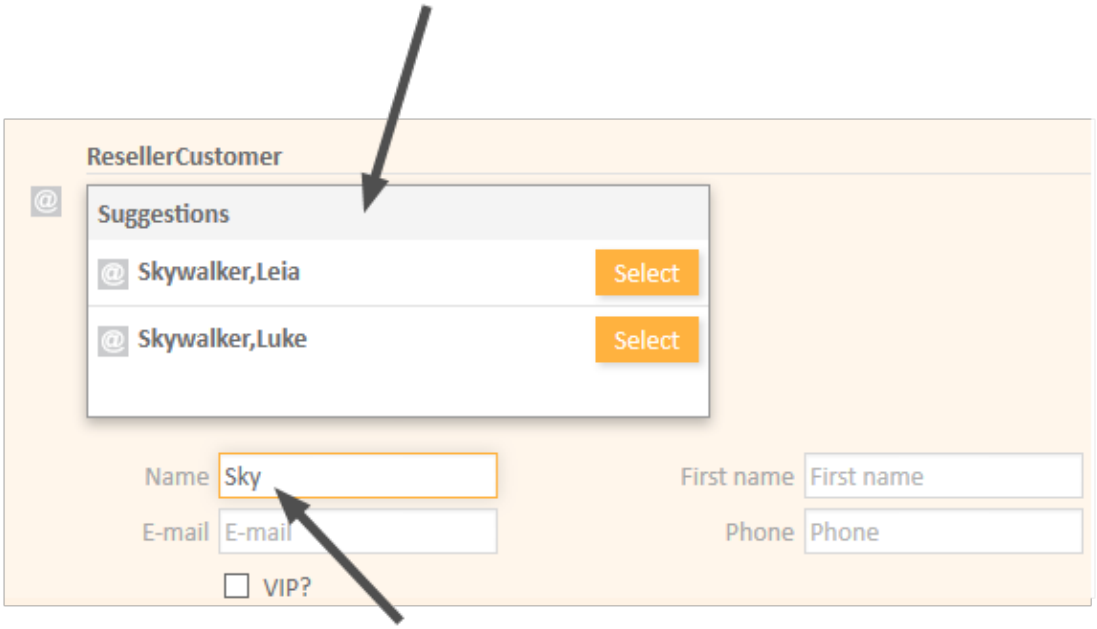


Figure 269: *ConSol CM Web Client - Suggestions for the ticket contact*

unitRelationSection (Type UnitSection)

Available on:

- contactEditPage
- companyEditPage

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Attributes:

- **compactViewLimit**
  If the limit is exceeded, the relations are presented along a with filtering feature. The default limit is “10”.

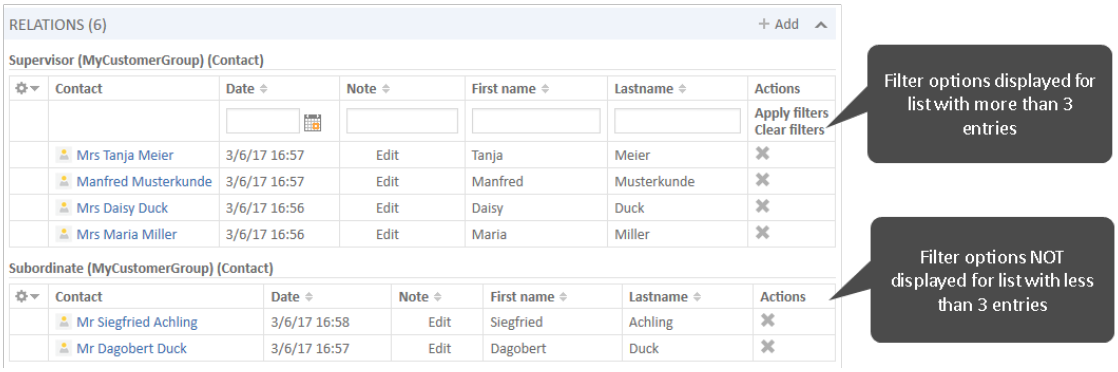


Figure 270: *ConSol CM Web Client - Effect of attribute compactViewLimit = 3 (contactEditPage / unitRelationSection)*

- **numberOfRelations**
  The default number of relations to display. The default value is “10”.
- **unitPreviewLayout**
  The unit preview configuration, this is used for the box which is displayed when you click on the name of a customer or company in the relation section. Similar to the box which is displayed when you click on a ticket name in a list which contains tickets.
  A JSON object has to be returned, as demonstrated in the next code example.

```
{ "layout": [ ["firstname", "lastname", "lastname"], ["email", "email", null]
 ]}
```

Code example 26: *JSon object for customer data format in box preview*

For different unit definitions in the customer data model use the following syntax:

```
{"unit_definition_1": "<json description>", "unit_definition_2": "component_
 name_used_to_display_preview"}.
```

Code example 27: *Example value for customer data preview box layout*

In JSON we set the configuration for a particular unit definition. The JSON can contain a unit preview or just the name of a component (spring bean) used to render the preview.

UnitResourceRelation

Available in unitRelationsSection (e.g., on the resource page).

Attributes:

- **compactViewLimit**
  If the limit is exceeded, the relations are presented along with a filtering feature. The default limit is 10.

- **defaultRelationSortStrategy**
  Additional resource relation sorting strategies. Possible options are: CREATION_DATE (default), FIRST_SORTABLE_COLUMN (sort by the first column which is sortable)
- **numberOfRelations**
  The default number of relations to display. The default value is “10”.

unitSearch

(available on the *ticketCreatePage* in the company section)

Attributes:

- **aidLevel**
  Beginner-friendly help level:
  - NONE
  - BASIC (wider search field with more descriptive text)
  - EXTENDED (as in BASIC plus additional help icon with tool tip)

  (java.lang.String, default value = “BASIC”)

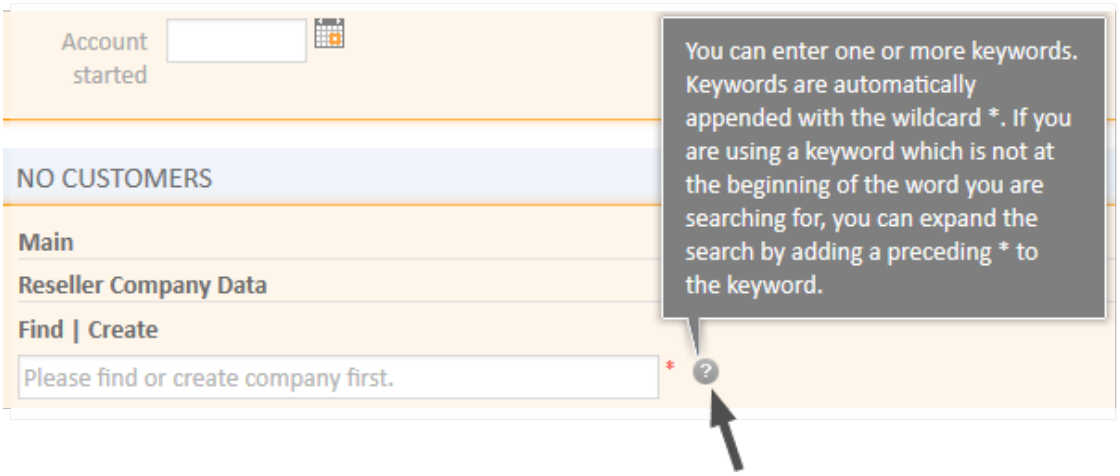


Figure 271: *ConSol CM Web Client - EXTENDED aidLevel in the unit search*

unitSearchHeader

(available on *ticketCreatePage* in the company section)

Attributes:

- **companyCreateLinkVisible**
  Boolean. The visibility of the link for referenced company creation.

viewDiscriminatorsSection (Type)

(available, e.g., on *userProfilePage*)

Attributes:

- **visibilityFlag**
  Boolean. The visibility of the *View criteria* section (section for attribute settings for dynamic views on *userProfilePage*). (default value = “true”)

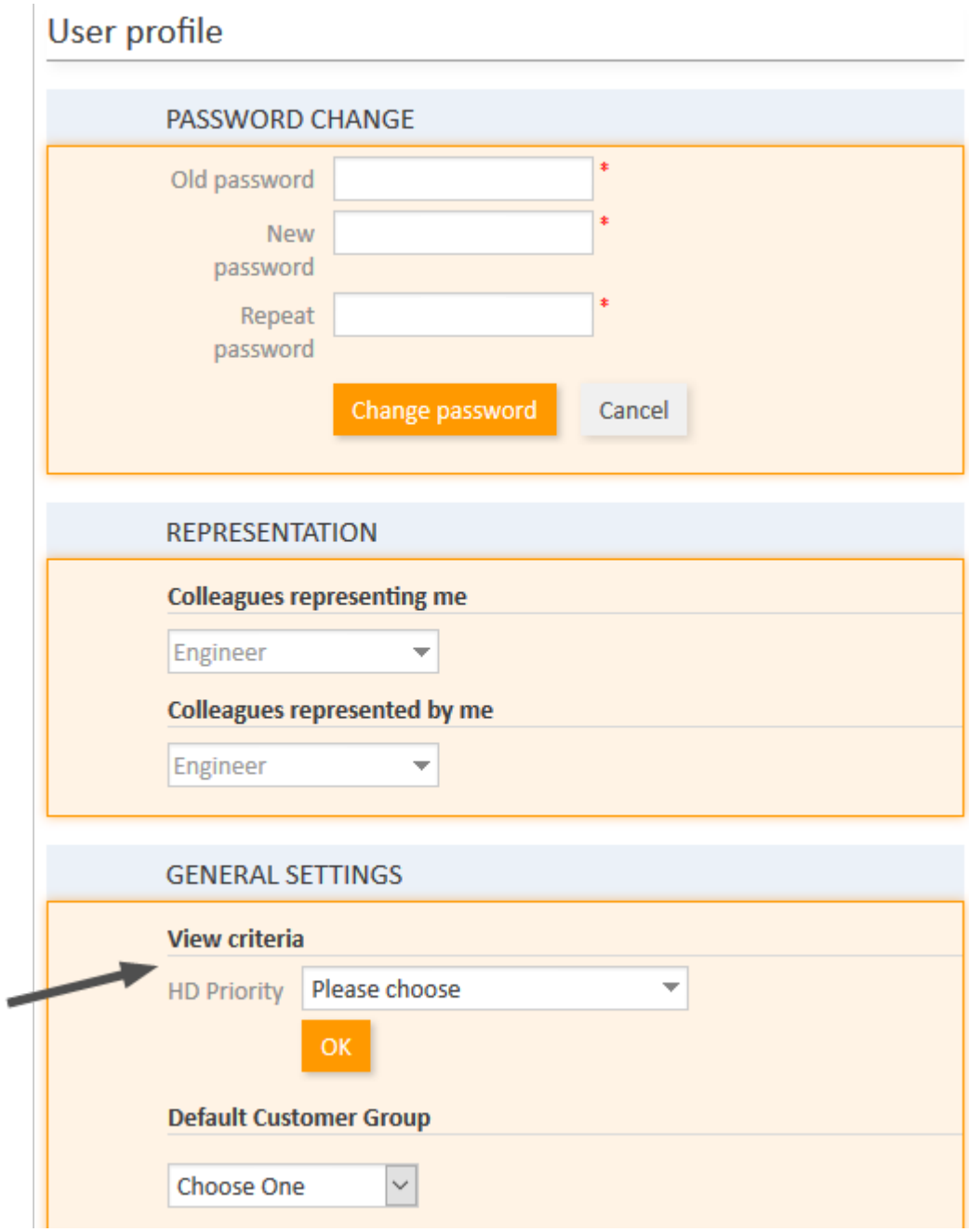


Figure 272: *ConSol CM Web Client - Visibility of View criteria section (visibilityFlag=true)*

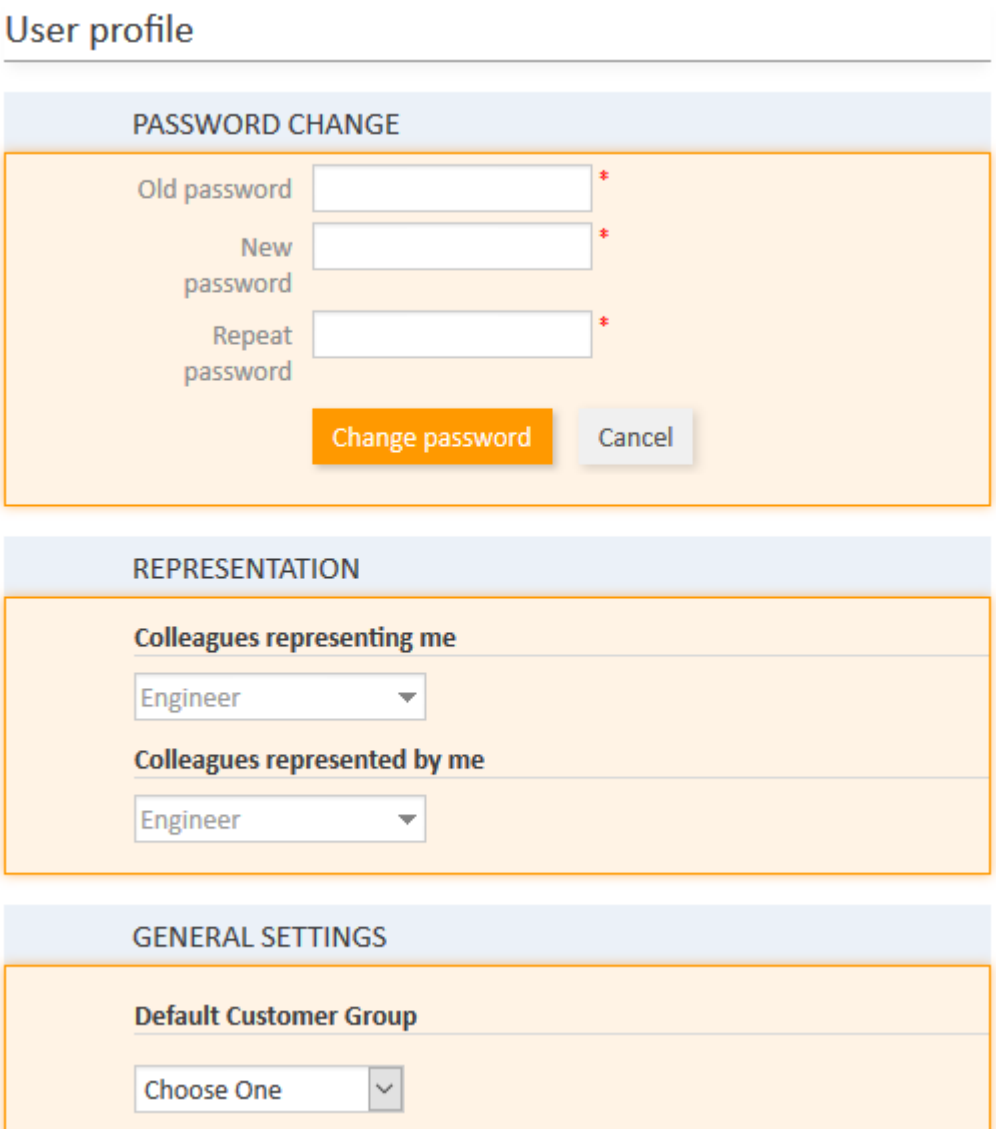


Figure 273: *ConSol CM Web Client - Visibility of View criteria section (visibilityFlag=false)*

welcomePage

The most important configuration on the *welcomePage* (overview page) is the dashboard configuration, see Dashboards.

E.3.6 Page Customization for the Graphical Representation of Relations

The representation of relations as graphs is based on page customization. Two modes are available:

The **standard mode** allows to configure the graph display for relation sections of tickets and on customer and resource pages. The graph display is then offered as link *Graph* and the engineer can switch between list and graph mode. For a detailed explanation, see section Page Customizations for the Visualization of Relations, Standard.

The **expert mode** allows to fade-in a new section on ticket, customer, or resource pages where an entirely custom-specific graph, based on a script, is displayed. For a detailed explanation, see section Page Customizations for the Visualization of Relations, Expert.

The **basic configuration principle** is identical in both modes and is explained in section Configuration of the Graph Display of Relations .

E.3.6.1 Page Customizations for the Visualization of Relations, Standard

Introduction

In all sections which represent relations (see section Configuration of the Graphical Display in Standard Mode), it is possible to configure the ConSol CM Web Client to offer - besides the list view - a graph view of the relations. This is done using page customization.

You can only switch on the graph view and use the default settings, this is a very simple step. Or you can go ahead and configure a more complex configuration.

The following example shows how to configure the display of relation graphs in a ticket page. Since CM/Resource Pool is active, the example comprises the configuration of ticket-resource relations as graph. The same principle as shown for a

- ticket page

applies to

- a company page
- a contact page
- a resource page (available if CM/Resource Pool is active)

Page Customization Example (Standard): contactSection in Ticket Page

Start with Page Customization of Contact Section in Ticket Page

In a ticket page, there are three sections where relations can be displayed.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

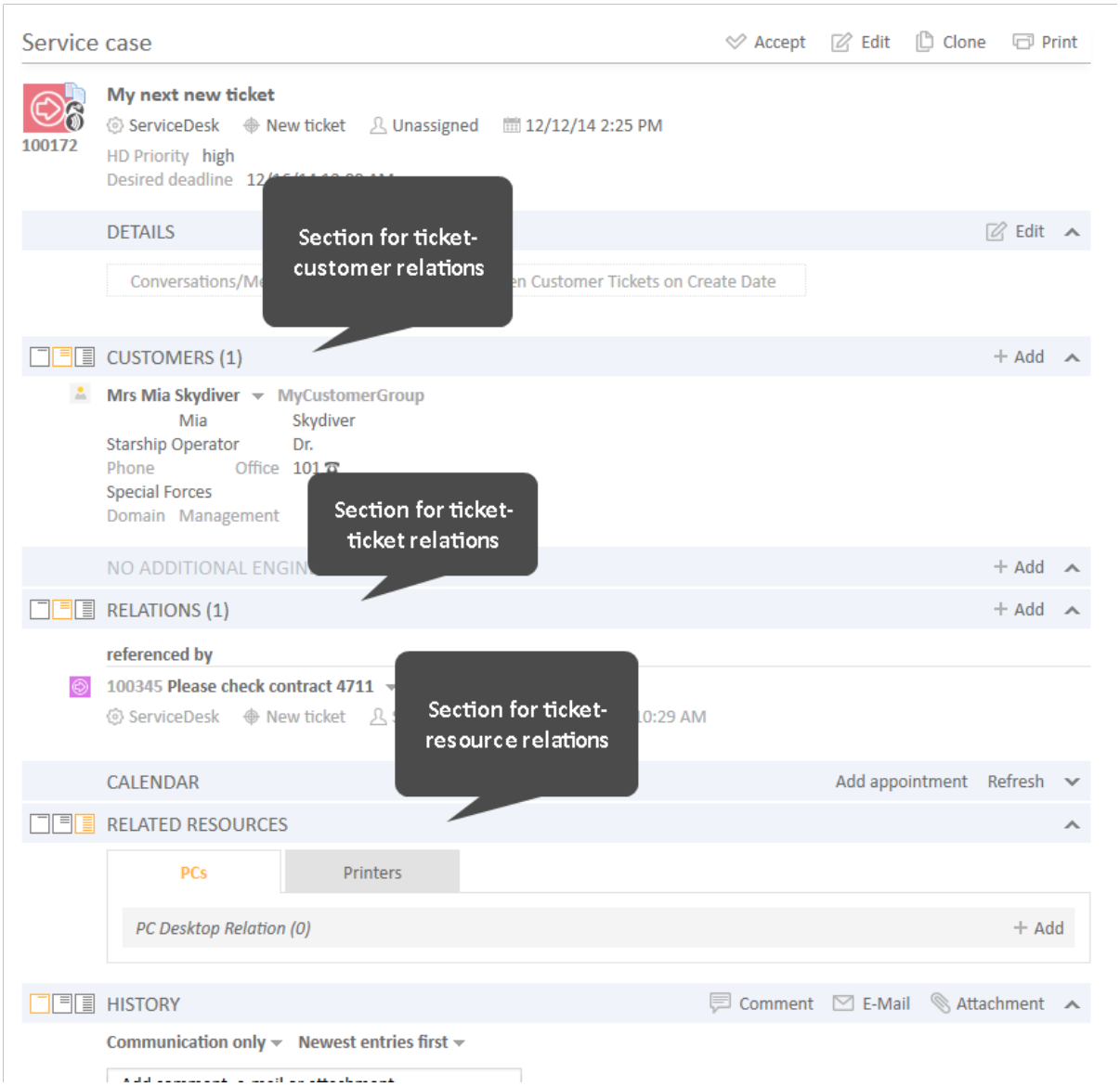


Figure 274: *ConSol CM Web Client - Ticket page: Sections where relations are displayed*

For each of the sections, the graph display can be configured separately. For example, start with the customer relations section. Log in as engineer with administrator privileges and *Enable page Customization* (main menu). For the customer relations sections several page customization scopes and subscopes are offered. Use the section-specific subscope to configure the graph. For the customer relations section, this is the *contactSection* subscope.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

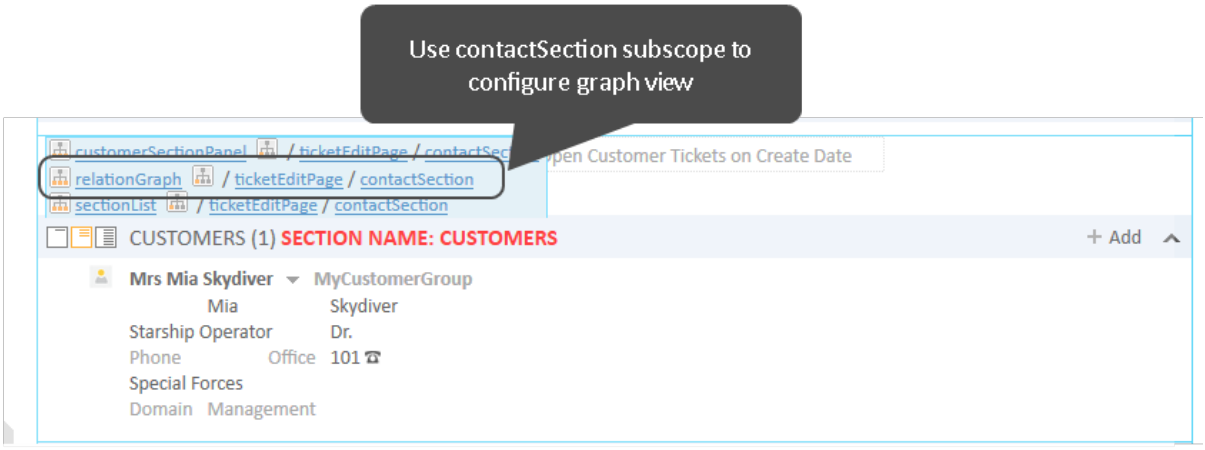


Figure 275: *ConSol CM Web Client - Selection of page customization scope for relationGraph in contactSection*

You can also select the subscope *contactSection* section in the Page Customization tree. In either case, the *relationGraph* configuration page for the contact section is opened (see following figure).

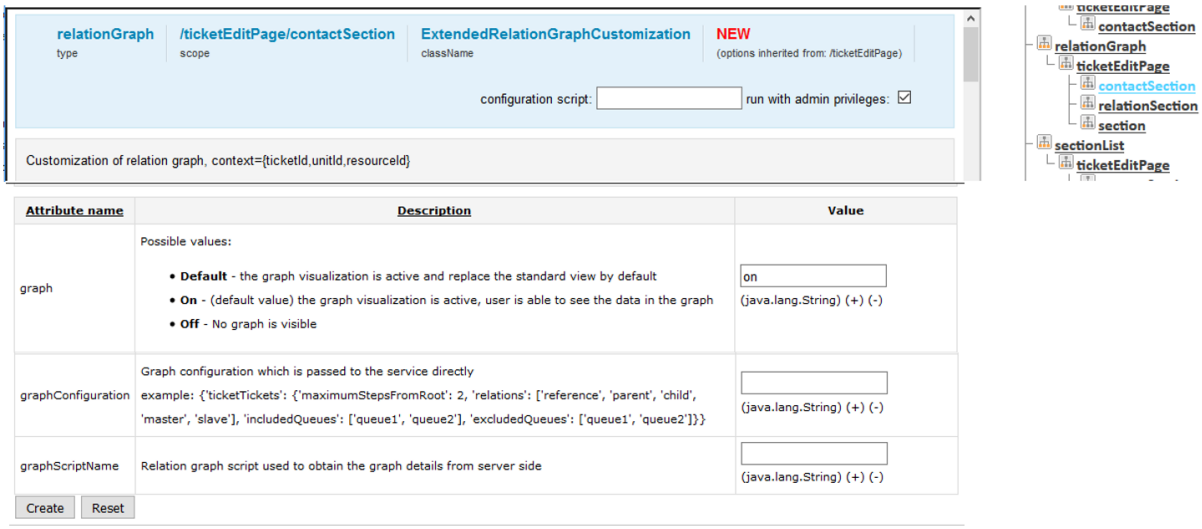


Figure 276: *Page customization of contact section in ticket edit page*

Attributes for Page Customization of Standard Relations Graph View

The following attributes are available:

- **graph** - String. Possible values:
  - Default - the graph visualization is active and replace the standard view by default
  - On - (default value in new installations) the graph visualization is active, user is able to see the data in the graph
  - Off - no graph is visible

- **graphConfiguration** - String, JSON statement

  With this JSON statement, you can configure the appearance of the graph. A default value is defined implicitly in the system, so when you just switch on the graph display, you do not have to define a *graphConfiguration*. But you can use it, if the default configuration does not cover the required display. For a detailed explanation, please see section Configuring the Graph Display of Relations Using JSON Statements.

- **graphScriptName** - String

  Refers to a script which is stored in the Admin Tool *Scripts and Templates* section. In the script, the relations graph can be built based on complex information. For example, it is possible to include data from other CM objects and/or from other systems which might provide CM-relevant data. The script has to be of script type *Relation graph* and always has to return an object of class `RelationGraph`. For a detailed explanation of this type of scripts, please refer to section Configuring the Graph Display of Relations Using Scripts.

- **sectionHeight** - Integer

  Enter the desired height (in pixels) for the relation graph section. The value "0" means that the standard height is used.

Please provide either a *graphConfiguration* or a *graphScriptName*. In case, both are provided, the script is the one which will be used. The *graphConfiguration* is not taken into account in this case.

Simple Example Using *graphConfiguration*

The following example configuration is used for a first explanation of how the *graphConfiguration* attribute works.

In the *contactSection* (see figures above), the following values are set for the attributes:

- *graph*: "on"
- *graphConfiguration*:

```
{"ticketCustomers" :
 {"maximumStepsFromRoot" : 2,
  "relations":["main","default","end customer"],
  "includedCustomerGroups":["Reseller","MyCustomerGroup"]}
 }
```

This results in the following graph display:

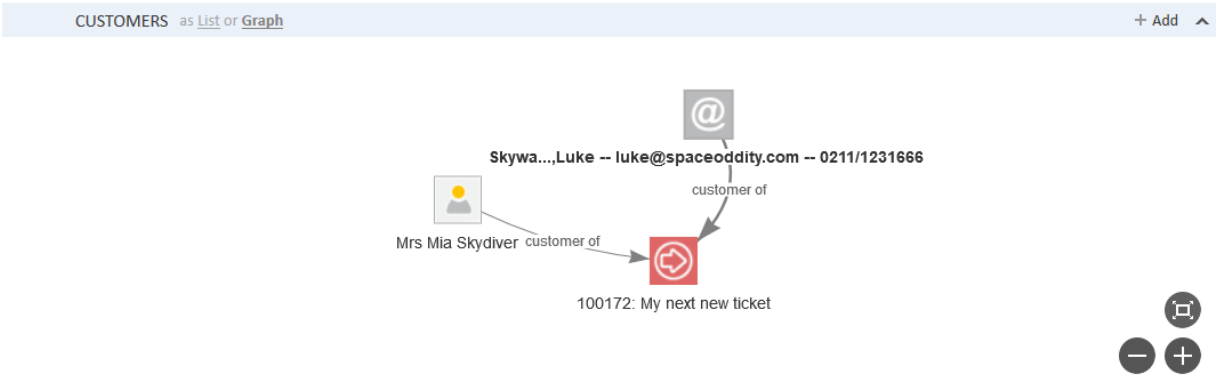


Figure 277: *ConSol CM Web Client - Ticket page: Graph display of customer relations to the ticket*

If you change the *graphConfiguration* as follows

```
{"ticketCustomers" :
 {"maximumStepsFromRoot" : 2,
  "relations":["main","default","end customer"],
  "includedCustomerGroups":["DirectCustomers","MyCustomerGroup"]}
 }
```

the following graph is displayed:

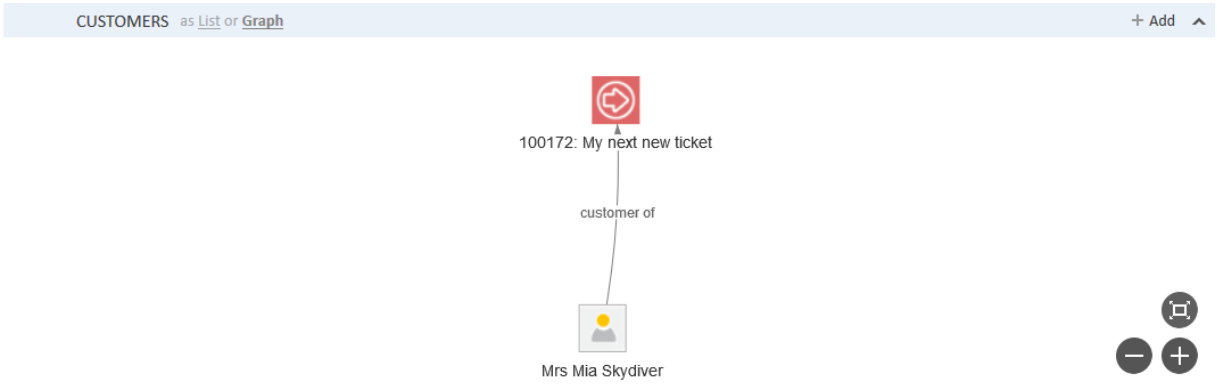


Figure 278: *ConSol CM Web Client - Ticket page: Graph display of customer relations to the ticket, variant*

The customer (contact) Luke Skywalker is in the customer group *Reseller*, and when the customer group *Reseller* has not been explicitly included, this customer is not displayed in the graph.

This simple example hopefully helps you understand the general principle of the *configurationGraph*. A detailed explanation of all parameters is provided in section Configuring the Graph Display of Relations Using JSON Statements.

Simple Example Using *graphScript*

A *graphScript* always has to return an object of class `RelationGraph`.

The following simple script will show only ticket-customer relations and could be used in the customer section of the ticket page. In this case, it would be easier to work with the *configurationGraph*, however, we want to show you how a script works ...

```
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 example 28: *Simple graphScript*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

The following graph is displayed:

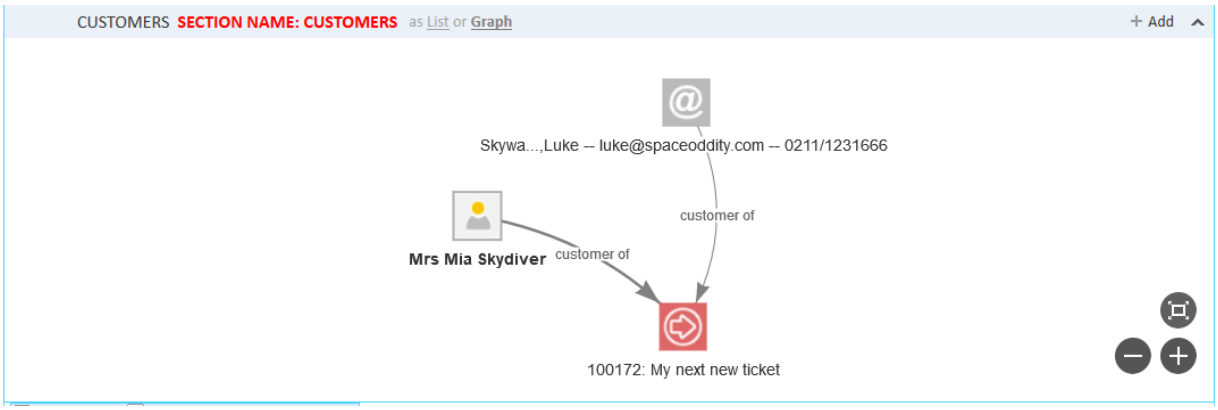


Figure 279: *ConSol CM Web Client - Ticket page: Graph display in the customers section based on the script shown above*

If the layout should be changed, the script could be modified as follows.

```
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 example 29: *Simple graphScript with simple node layout definition*

The following graph is displayed:

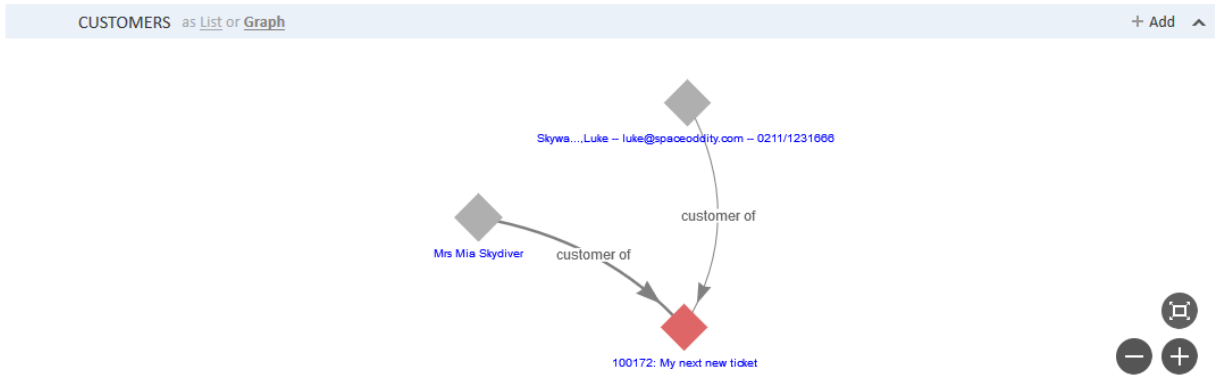


Figure 280: *ConSol CM Web Client - Ticket page: Slightly modified graph in customer relations section, based on the script shown above*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

In the script shown above, only the appearance of the nodes has been changed. The appearance of the edges also can be modified. Furthermore, there are numerous configuration parameters which can be set. For a detailed explanation, please refer to section Configuring the Graph Display of Relations Using Scripts.

E.3.6.2 Page Customizations for the Visualization of Relations, Expert

Introduction

The Standard configuration of graph displays for relations (see section Page Customizations for the Visualization of Relations, Standard) offers the possibility to configure graph displays in relations sections of ticket, customer, and resource pages, i.e., it extends the functionality of sections which are already present in the respective pages and adds the option to have the graph view displayed.

In addition to that, a whole new section can be included in:

- ticket pages
- contact pages
- company pages
- resource pages (if CM/Resource Pool is active)

This section is hidden in standard installations and can be used to display an individual relation graph which is completely system-specific. The graph is based on a script. There are no other possibilities to configure the content and appearance of the graph! It has to be done in an Admin Tool script.

> Please note that that means:
>
> You, as a script developer, are free to include everything which is required, since a script can make use of the entire ConSol CM Java/Groovy API.
>
> You, as a script developer, are fully responsible for the content which is displayed.

The following example will show you how to implement a system-specific relations graph in the resource page. The same principle of cause applies to ticket, contact, and company pages.

Page Customization Example (Expert): New Section in Resource Page

The following steps have to be performed in order to implement a system-specific relations graph in the resource page:

- Step 1: Make the required section visible in the page
- Step 2: Configure the page customization attributes for the newly displayed section
- Step 3: Write the script
- Step 4: Check the result

Step 1: Make the required section visible in the page

The section which has to be displayed is named *customRelationGraphSection*. Since the section is not yet displayed, you cannot select it in page customization mode directly on the page, but you have to use the Page Customization tree.

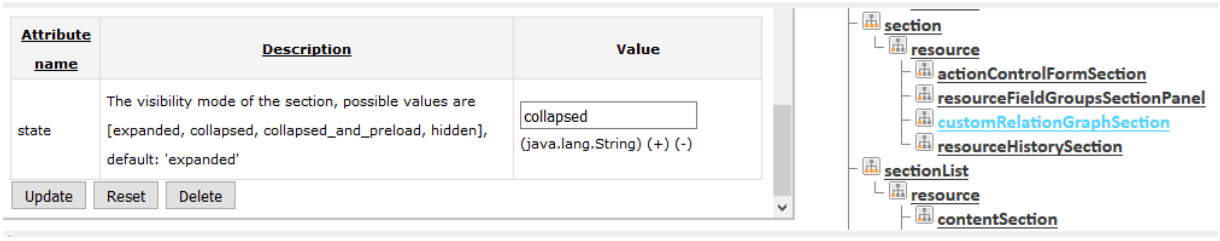


Figure 281: *One set of page customization attributes for the customRelationGraphSection*

Set the attribute *state* to either “collapsed” or “expanded”. Then the section will be available on the page, in the example on the resource page. The default name/header of the new section is *Custom relation graph*. The section is displayed above the history section. No graph is displayed yet - if you look at the section in the Web Client, an error will be displayed, because there is no script yet which defines the respective graph.

Step 2: Configure the page customization attributes for the newly displayed section

In order to configure the content and appearance of the graph, you first have to set the page customization attributes for the new section. Select the section either on the page or in the Page Customization tree.

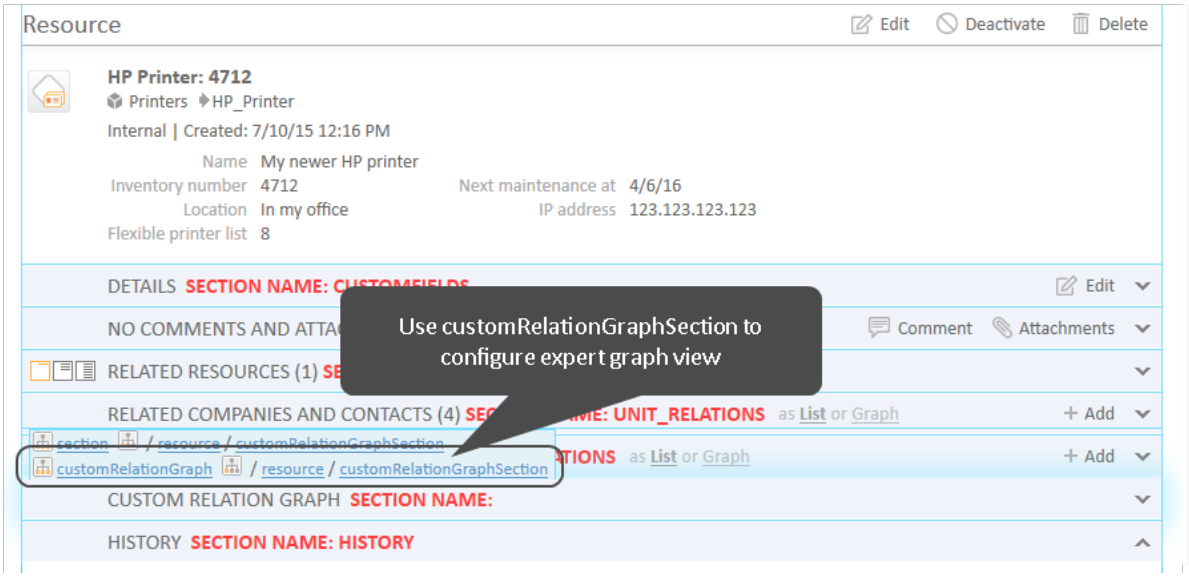


Figure 282: *ConSol CM Web Client - Selection of page customization scope for customRelationGraph in resource page*

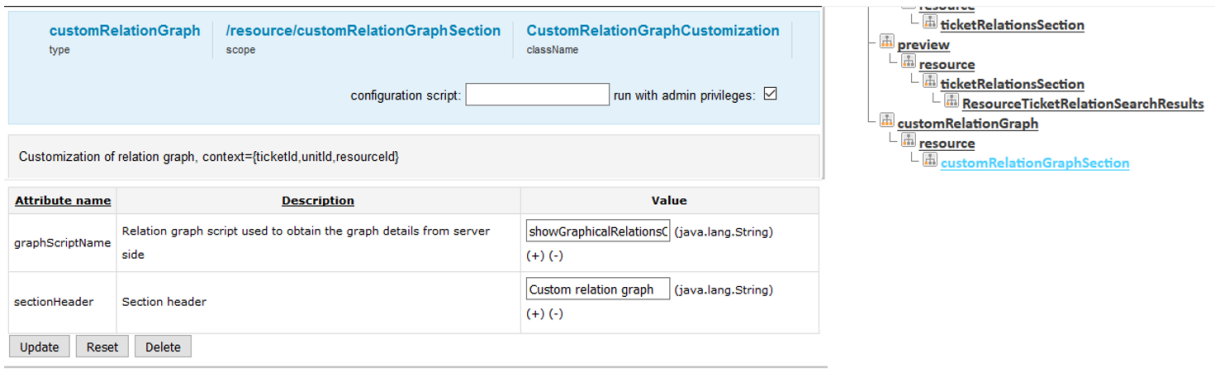


Figure 283: *Page customization tree and attributes for customRelationGraphSection*

Two attributes are available:

- **graphScriptName** - String. Refers to a script which is stored in the Admin Tool *Scripts and Templates* section. In the script, the relations graph can be built based on complex information. For example, it is possible to include data from other CM objects and/or from other systems which might provide CM-relevant data. The script has to be of script type *Relation graph* and always has to return an object of class `RelationGraph`. For a detailed explanation of this type of scripts, please refer to section Configuring the Graph Display of Relations Using Scripts.
- **sectionHeader** - String. Will be displayed in the Web Client as header of the section. Use any String.
- **sectionHeight** - Integer. Enter the desired height (in pixels) for the relation graph section. The value "0" means that the standard height is used.

Step 3: Write the script

The *graphScriptName* has to refer to a script of type *Relation graph* which is stored in the *Scripts and Templates* section of the Admin Tool. For a detailed explanation of this type of scripts, please refer to section Configuring the Graph Display of Relations Using Scripts.

In the current example, the following values were set for the attributes:

- sectionHeader = "This device within the entire infrastructure"
- graphScriptName = "showGraphicalRelationsOfResource2.groovy"

The following script is used:

```
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 example 30: *Script for customRelationGraph , showGraphicalRelationsOfResource2.groovy*

Step 4: Check the result

The following graph is displayed in the Web Client:

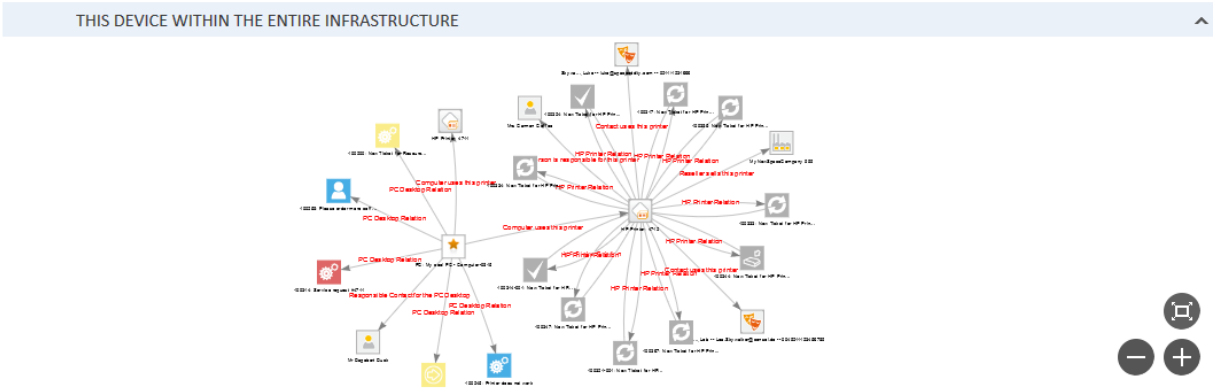


Figure 284: *ConSol CM Web Client - Display of expert relation graph in the resource page*

E.3.6.3 Configuration of the Graph Display of Relations

Introduction to Configuration of the Graph Display of Relations

The graph display of relations is based on vis.js, namely on the Network library. Refer to the vis.js Network detailed documentation to get to know all available configuration parameters. In the following sections, the most important parameters will be explained using some examples.

As explained in section Attributes for Page Customization of Standard Relations Graph View, either a JSON statement or a script can be used to define the layout of the graph.

Configuring the Graph Display of Relations Using JSON Statements

Introduction

There are two ways of providing a JSON statement for configuring a relation graph:

- as value of a page customization attribute:
  provide the statement as value of the page customization attribute `graphConfiguration`, see section Attributes for Page Customization of Standard Relations Graph View
- in a *graphScript*
  provide the statement as JSON string which is then used by the class.method `RelationCriteria.parseJson(JSONstring)`

The first case is the common way of configuring standard relation graphs. The second can be used in special cases, however, usually other methods would be better in this case.

> Please note that only the scope can be defined in a JSON statement! If you want to modify the graph layout (e.g., color of edges, font style), you have to work with scripts, see section Configuring the Graph Display of Relations Using Scripts.

Syntax for the Configuration of the Graph Display

In the JSON statement, the **scope** of the graph can be defined.

For **tickets**, you can define the following parameters:

- **For ticket-ticket relations:**

```
"ticketTickets" : {
 "maximumStepsFromRoot" : 2,
 "relations" : [ "reference", "parent", "child", "master", "slave" ],
 "includedQueues" : [ "queue1", "queue2" ],
 "excludedQueues" :[ "queue1", "queue2" ]
 }
```

  Use either `includedQueues` or `excludedQueues`!

  The following relation types are possible:

  - “reference”
    ticket references the current ticket
  - “parent”
    ticket is parent ticket (fixed) of the current ticket

- “child”
  ticket is child ticket (fixed) of the current ticket
- “master”
  ticket is parent ticket of the current ticket
- “slave”
  ticket is child ticket of the current ticket

- **For ticket-customer relations:**

```
"ticketCustomers" : {
 "maximumStepsFromRoot" : 2,
 "relations" : [ "main", "default", "contactTicketRoleName" ],
 "includedCustomerGroups" : [ "customerGroup1", "customerGroup2" ],
 "excludedCustomerGroups" : [ "customerGroup1", "customerGroup2" ]
 }
```

Use either `includedCustomerGroups` or `excludedCustomerGroups`!

The following relation types are possible:

- “main”
  customer is the main customer of the ticket
- “default”
  customer is an additional customer without role
- the name of a customer role

- **For ticket-resource relations:**

```
"ticketResources" : {
 "maximumStepsFromRoot" : 2,
 "relations" : [ "related", "resourceRelationDefinition" ],
 "includedResourceTypes" : [ "resourceType1", "resourceType2" ],
 "excludedResourceTypes" : [ "resourceType1", "resourceType2" ]
 }
```

For **customers = units (i.e., contacts and companies)**, you can define the following parameters:

- **For customer-ticket relations:**

```
"customerTickets" : {
 "maximumStepsFromRoot" : 2,
 "relations" : [ "main", "default", "contactTicketRoleName" ],
 "includedQueues" : [ "queue1", "queue2" ],
 "excludedQueues" : [ "queue1", "queue2" ]
 }
```

The following relation types are possible:

- “main”
  the customer is the main customer of the ticket
- “default”

customer is an additional customer without role)

- the name of a customer role

- **For unit-unit relations:**

```
"unitUnits" : {
 "maximumStepsFromRoot" : 2,
 "relations" : [ "unitRelationDefinition1", "unitRelationDefinition2" ],
 "includedCustomerGroups" : [ "customerGroup1", "customerGroup2" ],
 "excludedCustomerGroups" : [ "customerGroup1", "customerGroup2" ]
 }
```

Use either `includedCustomerGroups` or `excludedCustomerGroups`!

For directional unit-unit relations, it is possible to specify the direction, i.e., whether the objects linked in the graph should be source or target of the relation. The syntax is as follows.

Case 1. The linked customers are the source of the relation and the root node is the target:

```
"relations" : [ "unitRelationDefinition:source"]
```

Case 2. The linked customers are the target of the relation and the root node is the source:

```
"relations" : ["unitRelationDefinition:target"]
```

- **For contact-company relations:**

```
"contactCompany" : {
 "maximumStepsFromRoot" : 2
 }
```

- **For company-contact relations:**

```
"companyContacts" : {
 "maximumStepsFromRoot" : 2
 }
```

- **For unit-resource relations:**

```
"unitResources" : {
 "maximumStepsFromRoot" : 2,
 "relations" : [ "related", "resourceRelationDefinition" ],
 "includedResourceTypes" : [ "resourceType1", "resourceType2" ],
 "excludedResourceTypes" : [ "resourceType1", "resourceType2" ]
 }
```

Use either `includedResourceTypes` or `excludedResourceTypes`!

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

For resources, you can define the following parameters:

- **For resource-resource relations:**

```
"resourceResources" : {
 "maximumStepsFromRoot" : 2,
 "relations" : [ "related", "resourceRelationDefinition" ],
 "includedResourceTypes" : [ "resourceType1","resourceType2" ],
 "excludedResourceTypes" : [ "resourceType1", "resourceType2" ]
 }
```

Use either `includedResourceTypes` or `excludedResourceTypes`!

For directional resource-resource relations, it is possible to specify the direction, i.e., whether the objects linked in the graph should be source or target of the relation. The syntax is as follows:

Case 1. The linked resources are the source of the relation and the root node is the target:

```
"relations" : ["resourceRelationDefinition:source"]
```

Case 2. The linked resources are the target of the relation and the root node is the source:

```
"relations" : ["resourceRelationDefinition:target"]
```

- **For resource-unit relations:**

```
resourceUnits" : {
 "maximumStepsFromRoot" : 2,
 "relations" : [ "related", "resourceRelationDefinition" ],
 "includedCustomerGroups" : [ "customerGroup1", "customerGroup2" ],
 "excludedCustomerGroups" : [ "customerGroup1", "customerGroup2" ]
 }
```

Use either `includedCustomerGroups` or `excludedCustomerGroups`!

Configuring the Graph Display of Relations Using Scripts

Scripts might be used in two locations to define graph displays of relations:

- in the standard display mode as value of the `graphScriptName` attribute
- in the customized expert display mode as value of the `graphScriptName` attribute



Please note that in a script, every relation can be included into the display! So the person who implements the script is responsible for the graph display! Please make sure the correct relations are displayed, especially for scripts in the standard sections. For example, it might not make much sense to display ticket-resource relations in a customer section of a ticket, even if it is technically possible. If complex networks of relations of different types should be displayed, we recommend to place those in a customized / expert relations graph and set an informative header for this section!

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

A script which defines the graph display of relations

- has to be of type “Relation graph”
- has to return an object of class `RelationGraph`
- can define
  - the scope of the graph
  - the layout of the graph
- can make use of the following objects which are implicitly present in the script, depending on the location of the script
  - ticketId (in ticket pages)
  - unitId (in unit pages)
  - resourceId (in resource pages)

In order to define the **scope** of a relation graph in a script, use the class `RelationCriteria` and the static classes which represent the different flavors of `RelationCriteria`, e.g., `RelationCriteria.ResourceTickets` (see ConSol CM API documentation!)

Example script:

```
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;
```

Display:

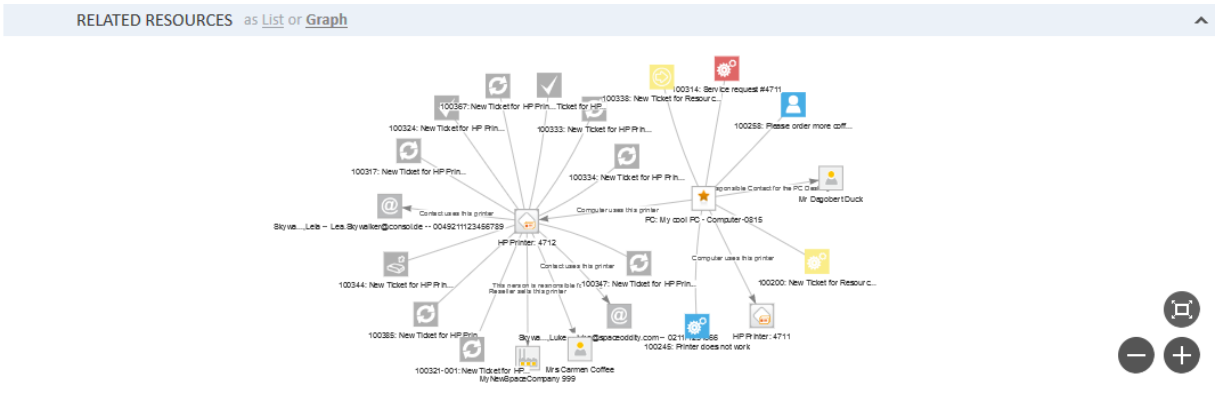


Figure 285: *ConSol CM Web Client - Resource page: Graph display*

In order to define the **layout** of the graph, use `RelationNode` and `RelationEdge` objects.

Example script (script as above, extended with layout configuration):

```
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;
```

Display:

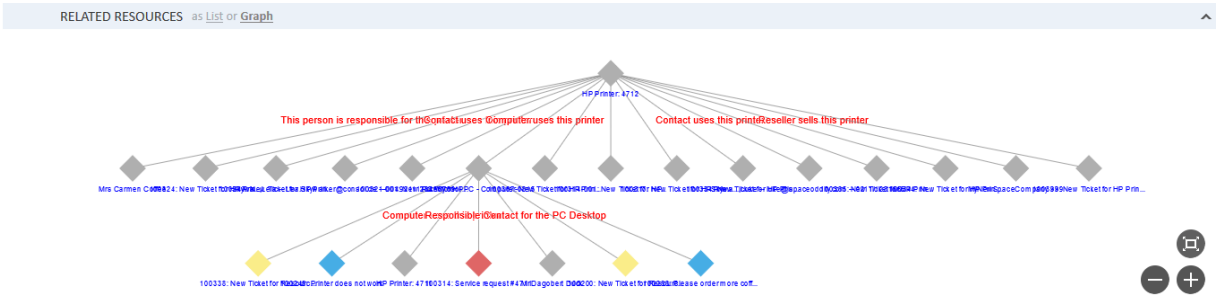


Figure 286: *ConSol CM Web Client - Resource page: Graph display with modified layout options*

The font size has been set to 12 pt, the font color has been changed to blue, the shape has been changed to diamond, and the layout has been changed to a hierarchical display.

This example gives an impression of how the layout of relation graphs can be modified. Use this principle (with the help of the vis.js Network documentation to adapt the relation graphs in your CM system to your company's requirements.

E.4 Dashboards

E.4.1 Introduction to dashboards in ConSol CM

Dashboards allow to display graphical information in the Web Client. This can be statistics, highlighted system data or any other information which is helpful for the users in their daily work. The content of a dashboard is configured using widgets. Dashboards are available on the general welcome page and the resource overview page.

E.4.1.1 Concepts, terms and definitions

Concept	Other terms	Definition
dashboard		Web Client screen which shows one or several widgets to provide an overview of important data
widget		Dashboard item which shows data or provides a specific functionality

E.4.1.2 Purpose and usage

Dashboards are used to show important information in a graphical way to the users. They are used in two places of the Web Client:

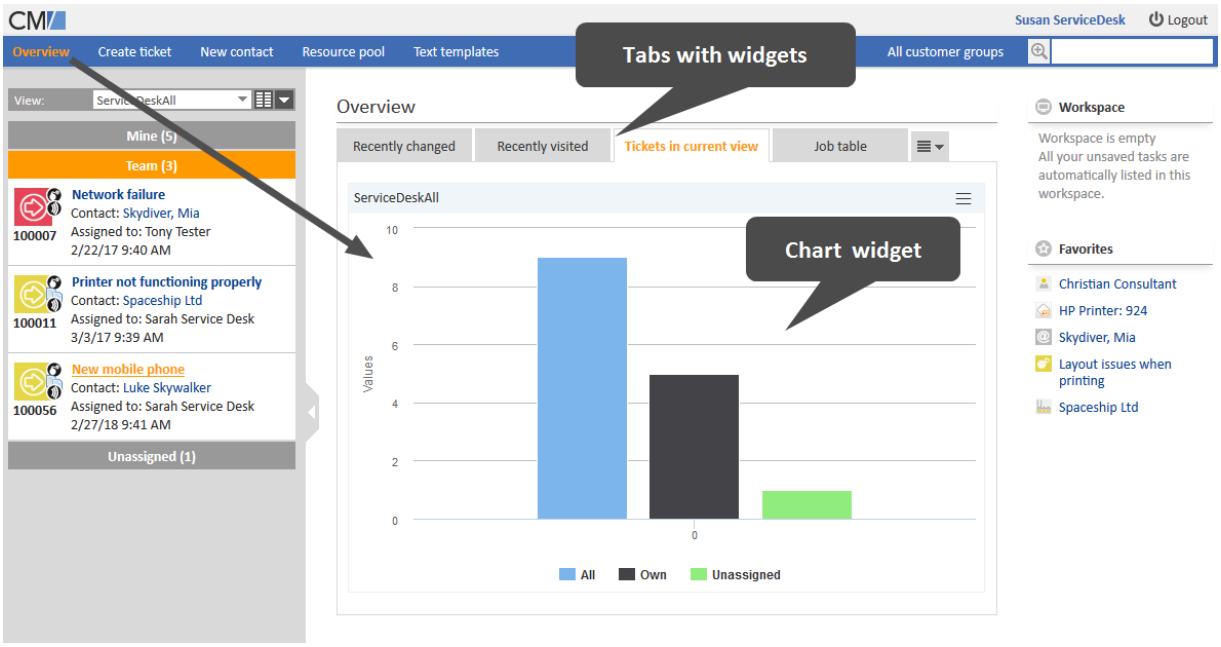
- **Welcome page:**
  This dashboard is displayed right after logging in to the Web Client. By default, it shows a bar chart with the number of cases in the current view.
- **Resource overview page:**
  This dashboard is displayed after clicking the *Resource pool* link in the main menu. It shows a bar chart with the number of resources of the different resource types in each resource category.

The dashboards can be customized using widgets. There are several widgets types which allow displaying different kinds of information, see Available widgets. Widgets can contain interactive elements. Most of the widgets require a widget script which retrieves the required data, see Scripts of the Type Widget.

By default, the dashboards are displayed to all users. If you want to adapt the dashboard content to the user's roles, you need to add the corresponding logic to the dashboard and / or widget scripts, see Adapting the dashboard to the user's role.

The following figure shows a dashboard which contains several widgets in tabs. The currently selected tab shows the default chart widget with the cases in the current view.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH



CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.4.2 Available widgets

The following widget types are available:

Name	Type	Description	Script required
Calendar	Calendar	Displays an integrated Microsoft Outlook calendar, see Calendar widgets and Microsoft Exchange Calendar Integration.	Yes
Chart	Chart	Displays a chart, see Chart widgets	Yes
HTML	Generic	Displays HTML content, see HTML widgets	Yes
KPI	kpi	Displays KPIs in colored boxes with a trend indicator, see KPI widgets	Yes
News	News	Displays news, see News widgets	Yes
Recent changes	RecentChanges	Displays a list of objects which have been changed recently, see Recent changes widgets	No
Recently visited	RecentlyVisited	Displays a list of the objects which the current user has viewed recently, see Recently visited widgets	No
Table	Table	Displays a table, see Table widgets	Yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.4.3 Basic tasks

E.4.3.1 Defining the dashboard layout

The overall layout of the dashboards is defined using the page customization, see Page Customization. The following attributes are used:

- **Welcome page**: `widgetsGrid / welcomePage`, accessible when the menu item *Overview* has been opened
- **Resource overview page**: `widgetsGrid / resourceDashboard`, accessible when the menu item *Resource pool* has been opened

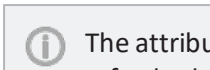
The following general principles apply when defining the dashboard layout:

- Each row of the dashboard grid is represented as an array of elements. The rows are separated by a coma. The grid starts with the upper left corner.
  Example: `[a,b,c],[d,e,f]`
- A widget is described by its name and its type, separated by a colon. The name for a specific widget must be unique. See Available widgets for a list of the widget types.
  Example: `ticketsInView:Chart`
- The widget can occupy multiple adjacent rows and columns. The type of a widget has to be indicated only at the first appearance of the widget's name. Afterwards, it can be omitted.
  Example: `[ticketsInView:Chart, ticketsInView, ticketsInView]`
- `null` is a reserved key word for an empty cell.
- You can distribute the widgets in several tabs. There is one array for each tab.
  Example: `[tabName:'Tickets in current view',widgets:[ticketsInView:Chart]]`
- The tab name can be localized using the `i18n` attribute. It is then displayed in the language which the user has set in the browser.
  Example: `[tabName:'Tickets in current view',i18n:{en:'Tickets in current view',de:'Tickets in der aktuellen Sicht'},widgets:[ticketsInView:Chart]]`
- The dashboard can be completely disabled by removing the value from the `layout` attribute.

Please proceed as follows to define the overall layout of a dashboard.

1. Log in as an administrator.
2. Open the page with the dashboard which you want to edit in the Web Client.
3. Open the page customization by clicking the *Page customization* link in the menu bar.
4. Locate the page customization scope `widgetsGrid / welcomePage` or `widgetsGrid / resourceDashboard`.
5. Edit the value of the `layout` attribute.
   - If all the widgets should be displayed on the same page, add one array for each row and fill the arrays with the widget references.
   - If you use tabs, create one array for each tab. The `widgets` attribute within the tab contains one array for each row. Add the localized tab name to the `i18n` attribute if needed.
6. Save your changes by clicking the *Create* or *Update* button in the page customization.
7. Reload the page.
8. Open the page customization again to configure the widgets, see Configuring widgets.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

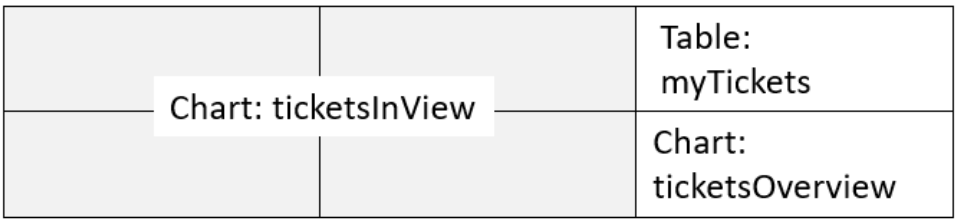


The attribute `refreshOnViewChange` allows to determine whether the dashboard should be refreshed when the user changes the view of the case list. The default value is “true”, i.e. the dashboard is refreshed when the view changes.

The following examples show different layouts.

Layout example 1 - Widget covering several cells

Three widgets are displayed, two charts and one table.

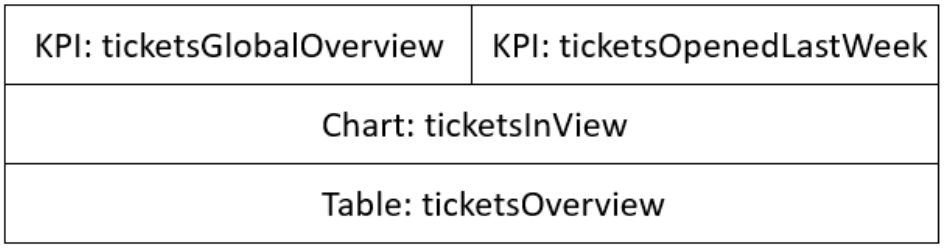


Value of the `layout` attribute:

```
[ticketsInView:Chart, ticketsInView, myTickets:Table], [ticketsInView,
 ticketsInView, ticketsOverview:Chart]
```

Layout example 2 - Different number of widgets per row

Four widgets are displayed, two KPI widgets, one chart and one table.



Value of the `layout` attribute:

```
[ticketsGlobalOverview:KPI,ticketsOpenedLastWeek:KPI],[ticketsInView:Chart],
 [ticketsOverview:Table]
```

Layout example 3 - Tabs with localized names

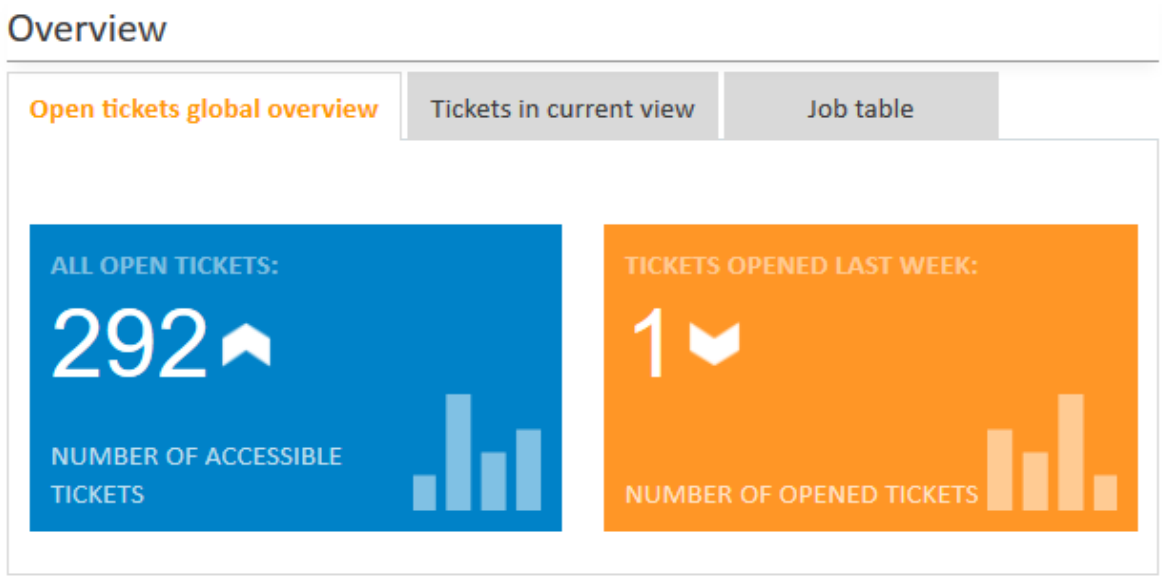
The following example shows a dashboard which consists of three tabs with localized names. The first tab contains two KPI widgets, the second tab a chart widget and the third tab a table widget.

Value of the `layout` attribute:

```
[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]]
```

Display in the Web Client, with English browser locale:



E.4.3.2 Configuring widgets

The following general principles apply when configuring individual widgets:

- You can only edit the widgets which are shown on the page when the page customization is opened. If the dashboard has several tabs, you need to open the tab with the desired widget before starting the page customization.
- Each widget has a node in the subtree of its widget type.
  Example: `chartWidget / welcomePage / ticketsInView`.
- The attributes for each widget type can be either defined for the individual widget (e.g., `chartWidget / welcomePage / ticketsInView`) or for all widgets of the widget type (e.g., `chartWidget` or `chartWidget / welcomePage`). Settings on a deeper level overwrite the settings on an upper level, i.e., settings for individual widgets overwrite the settings for the widget type.

Please proceed as follows to configure a widget:

1. Add the (empty) widget to the dashboard, see Defining the dashboard layout.
2. Open the page with the dashboard which includes the widget in the Web Client. If your dashboard uses tabs, you need to select the tab where the widget is placed.
3. Open the page customization by clicking the *Page customization* link in the menu bar.
4. Locate the page customization scope of the widget, e.g. `chartWidget/ welcomePage / ticketsInView`.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

5. If the widget type requires a script, create the script and reference it in the page customization. The place where the script is referenced depends on the widget type:
   - **Calendar**: attribute `calendarInitializationScript`, script type *Calendar integration*, see Microsoft Exchange Calendar Integration
   - **Chart, KPI, table**: *configuration script* field at the top of the page customization, script type *Widget*, see Scripts of the Type Widget
   - **HTML**: attribute `visualizationScript`, script type *Widget visualization*, see Scripts of Type Widget Visualization
   - **News**: attribute `scriptName`, script type *News publisher*, see Scripts of Type News Publisher
6. Set the required values in the attributes of the page customization. You need to set the attribute `visible` to “true” for the widget to be displayed.

   ⚠ You can set attributes both in the widget script and in the page customization. The values set in the script overwrite the values entered in the page customization.

   Please see the section of the respective widget type for details about the available attributes.

E.4.3.3 Localizing widget text

You can localize the text displayed in the widget using the `localization` attribute of the widget configuration. It can be set either in the page customization or in the widget script. The `localization` attribute can hold the localized values for all string attributes used in the widget.

Please proceed as follows to define localized strings in the page customization:

1. Enter a placeholder in the following syntax as a value for the attribute which you want to localize.

   Example: attribute: `title`, value: `_('titlestring')`
2. Provide localizations in the following syntax in the `localization` attribute.

   Example: attribute: `localization`, value: `de:{titlestring:'Offene Vorgänge'}, en:{titlestring:'Open cases'}`

   You can add localizations for different attributes as a comma-separated list within the language tag: `de:{titlestring:'Offene Vorgänge',footerstring:'Anzahl bearbeitbare Vorgänge'}`

Please proceed as follows to define localized strings in the widget script:

1. Use placeholders in the following syntax when aggregating data for the widget (highlighted in red):

```
data.add("{name: _('all'), data:[${allTickets}]}" as String)
data.add("{name: _('own'), data:[${ownTickets}]}"as String)
data.add("{name: _('unassigned'), data:[${unassignedTickets}]}"as String)
```

2. Provide localizations in the return statement using the following syntax (highlighted in red):

```
localization:"de: {all:'Alle',own:'Eigene',unassigned:'Unzugewiesene'},"+
 "en: {all:'All', own:'Own', unassigned: 'Unassigned'}"];
```

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.4.4 Advanced tasks

E.4.4.1 Adapting the dashboard to the user's role

There are two options to adapt the dashboard for different users:

- Show or hide individual widgets
- Show or hide tabs

If you want to display different widgets for different kinds of users, you need to set the `visibility` attribute of the widget within the widget script. This allows you to display the widget only to the users who have a certain role.

The following example shows how to hide the widget if the current user does not have the *consultant* role.

```
def role = roleService.getById('consultant');
def engineer = engineerService.getById(engineer_id);
if(!getRolesForEngineer(engineer).contains(role))
{
   return {"visible": false}
}
```

Code example 31: *Widget visibility depending on the user's roles*

If you want to display different tabs for different kinds of users, you need to modify the `layout` attribute of the dashboard in a script. This is done by creating a page customization script and referencing it in the *configuration script* field at the top of the page customization.

The following example shows how to display a different set of tabs if the current user has the *sales* role.

```
def role = roleService.getById('sales');
def engineer = engineerService.getById(engineer_id);

if (getRolesForEngineer(engineer).contains(role)) {
   configuration.put("layout",
      "[tabName:'Sales performance', widgets:[sales:Chart]]," +
      "[tabName:'Most important leads', widgets:[leads:Table]]");

} else {
   configuration.put("layout",
      "[tabName:'Service summary', widgets:[service:Chart]]" +
      "[tabName:'HelpDesk performance', widgets:[hd:Chart]]");
}

return configuration;
```

Code example 32: *Dashboard layout depending on the user's roles*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.4.5 Calendar widgets

Calendar widgets are configured in the type *calendarWidget* or one of its subitems. They can be used to display a Microsoft Exchange calendar on the dashboard. Calendar widgets use the FullCalendar library.

General attributes:

- **localization**
  String. The localized title of the widget. See Localizing widget text.
- **title**
  String. The title string shown in the header bar.
- **visible**
  Boolean. Determines whether the widget is shown (“true”) or not (“false”).

FullCalendar attributes:

- **allDaySlot**
  Boolean. Determines if the all-day slot is displayed at the top of the calendar. Default “true”.
- **appointmentBackgroundColor**
  String. Sets the background color for all appointments in the calendar. You can use any CSS color format, such as “#f00”, “#ff0000”, “rgb(255,0,0)”, or “red”.
- **appointmentBorderColor**
  String. Sets the border color for all appointments in the calendar. You can use any CSS color format, such as “#f00”, “#ff0000”, “rgb(255,0,0)”, or “red”.
- **appointmentColor**
  String. Sets the background and border colors for all appointments in the calendar. You can use any CSS color format, such as “#f00”, “#ff0000”, “rgb(255,0,0)”, or “red”.
- **appointmentConstraint**
  String. Limits appointment dragging and resizing to certain windows of time.

  Possible values:

  - **<appointment_ID>**
    Appointments that are being dragged or resized must be fully contained by at least one of the appointments linked to by the given appointment ID.
  - **businessHours**
    Appointments being dragged or resized must be fully contained within the week's business hours (default: Monday-Friday 9am-5pm), see *businessHours* attribute.
  - **<start-time>-<end_time>;<days_of_week>**
    A custom time window in the same format as the *businessHours* attribute. Days of week are optional.
    Examples: “10:00-18:00; 1,2,3,4” or “10:00-18:00”
- **appointmentDurationEditable**
  Boolean. Allows appointments’ durations to be editable through resizing. Default “true”.
- **appointmentOverlap**
  Boolean. Determines if appointments in the calendar, when dragged and resized, are allowed to overlap each other. Default “true”.
- **appointmentStartEditable**
  Boolean. Allows appointments’ start times to be editable through dragging. Default “true”.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **appointmentTextColor**
  String. Sets the text color for all appointments in the calendar. You can use any CSS color format, such as “#f00”, “#ff0000”, “rgb(255,0,0)”, or “red”.
- **aspectRatio**
  String. Determines the width-to-height aspect ratio of the calendar. Default value “2.8”. If empty, the calendar component will use the internal default value 1.35.
- **businessHours**
  String. Emphasizes certain time slots in the calendar.
  Format: <start-time>-<end_time>;<days_of_week>
  Example: 10:00-18:00; 1,2,3,4 (from 10am to 6pm, Monday-Thursday)
- **calendarEventHandlerScript**
  String. Name of the script which handles calendar events. Besides standard context variables like `ticket`, there are additional ones:
  - **eventType**
    enum (values: CREATE, UPDATE, DELETE)
  - **appointment**
    with appointment data (uid, subject, location, etc.).
- **calendarInitializationScript**
  String. Name of the script which produces the calendar configuration. If the value is empty or the script returns “null” the calendar is not shown.
- **contentHeight**
  String. Makes the calendar’s content area this many pixels tall. By default, this option is not set and the calendar’s height is calculated based on the *aspectRatio* attribute.
- **defaultAllDayAppointmentDuration**
  String. A fallback duration for all-day appointments without a specified *end time* value. Default value “1” (one day).
- **defaultDate**
  String. The initial date displayed when the calendar first loads. Accepts an ISO8601 date string like “2020-02-01”.
- **defaultTimedAppointmentDuration**
  String. A fallback duration for timed appointments without a specified *end time* value. If not set, the default value is “02:00:00” (2 hours). This attribute also affects the default duration of appointments during creation.
- **DefaultView**
  String. Default calendar view.
  Possible values: month, basicWeek, basicDay, agendaWeek, agendaDay. Default “agendaWeek”.
  See Available Views for view examples.
- **editable**
  Boolean. Whether the appointments can be created, dragged and resized. This value over-writes the source configuration. Default “false”.
- **firstDay**
  String. The day that each week begins with. (Sunday=0, Monday=1, Tuesday=2, etc.). If empty, the value is based on browser’s locale.
- **forceAppointmentDuration**
  Boolean. A flag to force calculation of an appointment’s end if it is unspecified. Default “false”.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **handleWindowResize**
  Boolean. Whether to resize the calendar automatically when the browser window resizes. Default “true”.
- **headerCenter**
  String. Defines the buttons and title at the top/center of the calendar. See *headerLeft* description for details. Default “title”.
- **headerLeft**
  String. Defines the buttons and title at the top/left of the calendar. Comma- or space-separated list values (values separated by a comma are displayed adjacently). Default “prev,next today”.

  Possible values:

  - **title**
    Text for the current month/week/day.
  - **prev**
    Button for moving the calendar back one month/week/day.
  - **next**
    Button for moving the calendar forward one month/week/day.
  - **prevYear**
    Button for moving the calendar back one year.
  - **nextYear**
    Button for moving the calendar forward one year.
  - **today**
    Button for moving the calendar to the current month/week/day.
  - **<view name>**
    Button that will switch the calendar to any of the available views (see *defaultView* description for available views).

  The header is not shown if all three header attributes (*headerCenter*, *headerLeft*, *headerRight*) are empty.
- **headerRight**
  String. Defines the buttons and title at the top/right of the calendar. See *headerLeft* attribute description for details. Default “month,agendaWeek”.
- **height**
  String. Sets the height, in pixels, of the entire calendar (including header). By default, this option is not set and the calendar’s height is calculated based on the *aspectRatio* attribute.
- **hiddenDays**
  String. Excludes certain days of the week from being displayed. Comma separated list of day-of-week indices (Example: “1,3,5”). Each index is zero-based (Sunday=0) and ranges from 0-6.
- **lazyFetching**
  Boolean. Determines when appointment fetching should occur. Setting this attribute to “false” makes sense when there are a lot of external changes to the user’s calendar. Default “true”.
- **maxTime**
  String. Determines the end time (exclusively) which will be displayed, even if the scrollbars have been scrolled all the way down. Default value is “24:00:00”.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **minTime**
  String. Determines the starting time that will be displayed, even if the scrollbars have been scrolled all the way up. Default value is “00:00:00”.
- **nextDayThreshold**
  String. When an appointment’s end time spans into another day, the minimum time it must be in order for it to render as if it were on that day. Default: “09:00:00” (9am). Only affects timed appointments that appear on whole days. Whole day cells appear in *month view*, *basicDay*, *basicWeek* and the *all-day slots* in the agenda views.
- **readable**
  Boolean. Determines if the user can see the calendar entries. Set “true” to make the calendar entries visible and “false” if the user should not see the calendar entries. In the latter case, the calendar widget is still displayed. Only available if the calendar is displayed as a calendar widget on the dashboard.
- **rightToLeftMode**
  Boolean. If enabled, displays the calendar in right-to-left mode. Default “false”.
- **scrollTime**
  String. Determines how far down the scroll pane is initially scrolled. Default is “06:00:00” (6am).
- **slotAppointmentOverlap**
  Boolean. Determines whether timed appointments in agenda view should visually overlap. Default “true”.
- **slotDuration**
  String. The frequency for displaying time slots. Default is “00:30:00” (30 minutes).
- **snapDuration**
  String. The time interval at which a dragged appointment will snap to the agenda view time grid. Also affects the granularity at which selections can be made. Default value is set to *slotDuration*.
- **state**
  String. The visibility mode of the section, possible values are [*expanded, collapsed, collapsed_and_preload, hidden*]. Default: “expanded”. Only available if the calendar is displayed on the case, contact, or resource page.
- **weekNumbers**
  Boolean. Determines if week numbers should be displayed in the calendar. Default “true”.
- **weekends**
  Boolean. Whether to include Saturday/Sunday (i.e., weekend) columns in any of the calendar views. Default “true”.

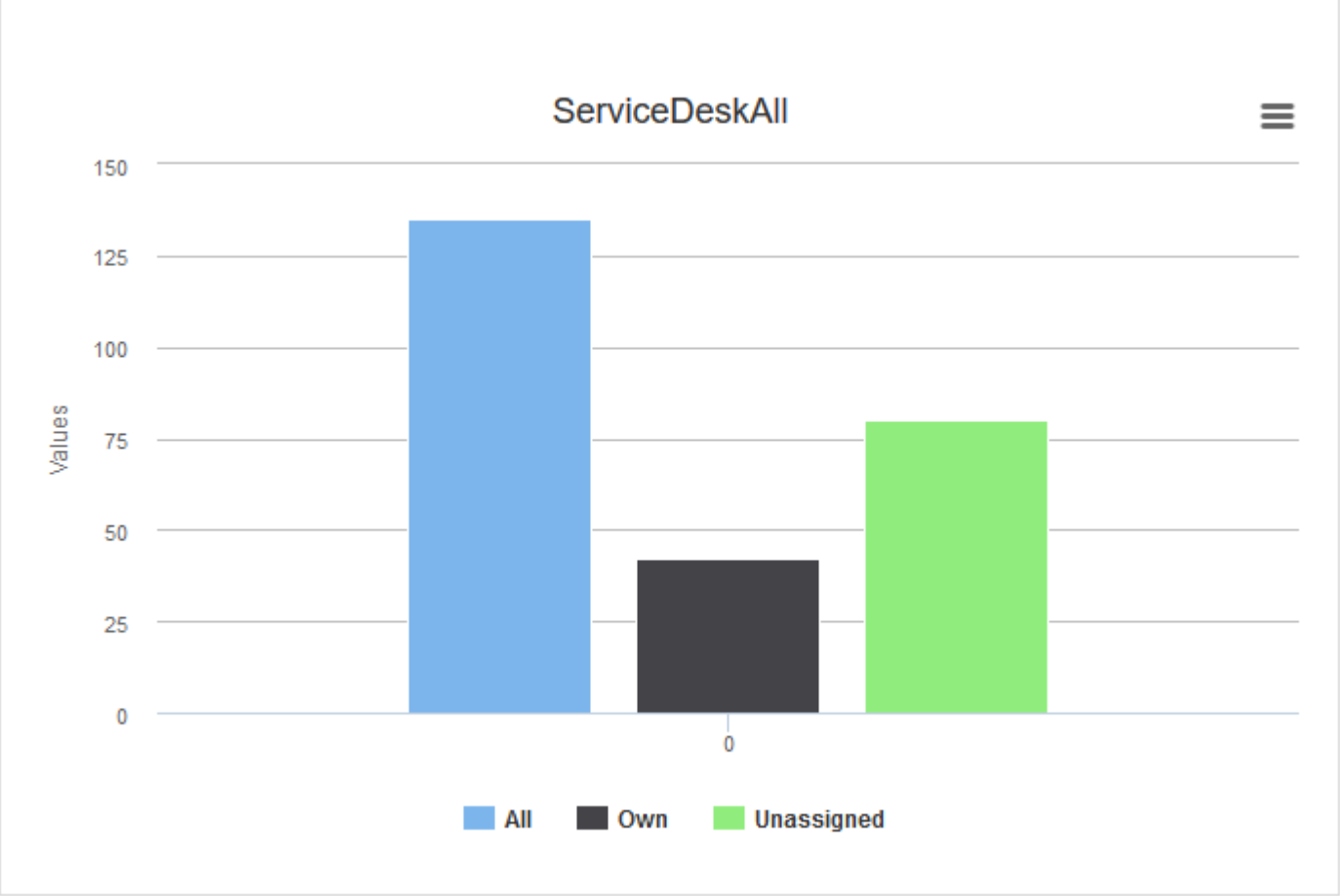
See http://fullcalendar.io/docs for further details.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.4.6 Chart widgets

Chart widgets are configured in the type *chartWidget* or one of its subitems. They use the Highcharts library. All attributes are JSON objects.

The following figure shows the default chart widget with the number of cases in the current view:



⚠ You can set attributes both in the widget script and in the page customization. The values set in the script overwrite the values entered in the page customization.

General attributes:

- **developmentMode**
  Boolean. Enables the development mode for the widget. If this attribute is set to “true”, the widget is displayed with a red border and a JSLint validation is performed on the widget script. If there are errors in the widget script, the error messages are displayed instead of the widget.
- **highcharts3dEnabled**
  Boolean. Determines whether the 3D display module should be loaded. The default value is “false”. Set this attribute to “true” to load the 3D module.

  ⚠ 3D rendering requires a lot of browser resources.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **highchartsModulesEnabled**
  String. Determines which additional modules should be loaded. The available modules are:
  - accessibility.js
  - annotations.js
  - boost.js
  - broken-axis.js
  - bullet.js
  - current-date-indicator.js
  - cylinder.js
  - data.js
  - debugger.js
  - drilldown.js
  - export-data.js
  - exporting.js
  - full-screen.js
  - funnel.js
  - networkgraph.js
  - no-data-to-display.js
  - offline-exporting.js
  - pareto.js
  - series-label.js
  - solid-gauge.js
  - sonification.js
  - static-scale.js
  - sunburst.js
  - tilemap.js
  - timeline.js
  - treemap.js
  - variable-pie.js
  - variwide.js
  - vector.js
  - venn.js
  - windbarb.js
  - wordcloud.js
  - xrange.js

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **localization**
  String. Localized values, i.e., `de: {subject:'Thema', yes:'Ja'}, en: {subject:'Subject', yes:'Yes'}`. See Localizing widget text.
- **visible**
  Boolean. Determines whether the widget is shown (“true”) or not (“false”).

Highcharts attributes:

- **chart**
  String. Options regarding the chart area and plot area as well as general chart options. Example:

```
chart = "type:'column', pltShadow:false, backgroundColor:'#4dc245',
 height:300";"items: [{html:'sometext', style: { left: '100px'; }}]"
```

  Code example 33: *Chart object*

- **colors**
  String. An array containing the default colors for the chart’s series. When all colors are used, new colors are pulled from the start again.
- **credits**
  String. Highchart, by default, puts a credits label in the lower right corner of the chart.
- **drilldown**
  String. Options for drill down, the concept of inspecting increasingly high resolution data through clicking on chart items like columns or pie slices.

  > In addition to setting the drilldown options, you need to extend the widget script. The data returned by the script must contain the data shown in the drilldown which needs to be linked to the data it extends.

- **exporting**
  String. Options for the exporting module.
- **global**
  String. Global options that apply to all charts. Can only be set for a type, i.e., for *chartWidget*, not for scopes or single widgets.
- **labels**
  String. HTML labels that can be positioned anywhere in the chart area.

```
labels = "items: [{html:'sometext', style: { left: '100px'; }}]"
```

  Code example 34: *Labels object*

- **lang**
  String. Language object. The language object is global and applies to all charts. Can only be set for a type, i.e., for *chartWidget*, not for scopes or single widgets.
- **legend**
  String. The legend is a box containing a symbol and name for each series item or point item in the chart.

- **loading**
  String. The loading options control the appearance of the loading screen that covers the plot area on chart operations.
- **navigation**
  String. A collection of options for buttons and menus appearing in the exporting module.
- **noData**
  String. Options for displaying a message like “No data to display”.
- **pane**
  String. Applies only to polar charts and angular gauges. This configuration object holds general options for the combined X and Y axes set.
- **plotOptions**
  String. *plotOptions* is a wrapper object for configuration objects for each series type.
- **series**
  String. The data displayed in the chart.
- **subtitle**
  String. The chart’s subtitle..
- **title**
  String. The chart’s main title.
- **tooltip**
  String. Options for the tooltip that appears when hovering over a data series or point.
- **xAxis**
  String. The X axis or category axis.
- **yAxis**
  String. The Y axis or value axis.

See https://api.highcharts.com/highcharts/ for details about the available Highcharts attributes.

E.4.6.1 Coding example

The following example shows the script which is used for the default chart widget showing the number of cases in the current view. The script has to return a HashMap containing the attributes which should be set.

```
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 = []

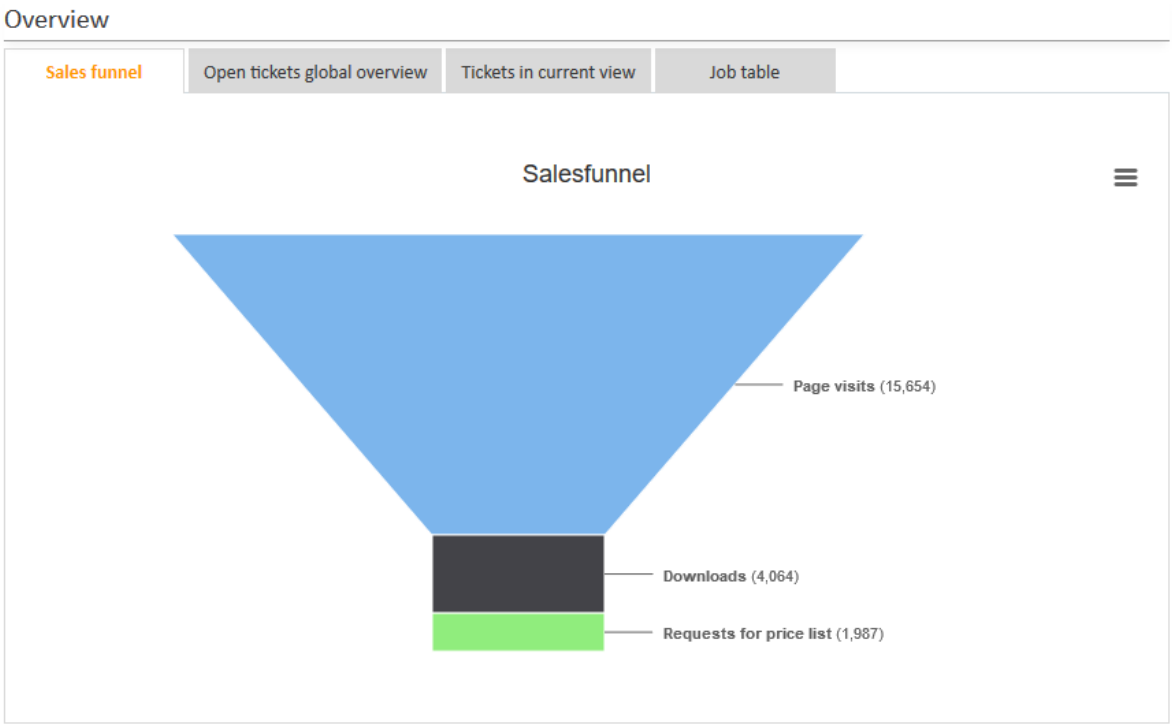
data.add("{name: _('all'), data:[${allTickets}]}" as String)
data.add("{name: _('own'), data:[${ownTickets}]}"as String)
data.add("{name: _('unassigned'), data:[${unassignedTickets}]}"as String)


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 example 35: *Script for the default chart widget ticketsInViewDataWidget.groovy*

The type of chart can be modified in the `type` sub-attribute of the `chart` attribute. There is a large number of available chart types, see https://api.highcharts.com/highcharts/plotOptions.

The following figure shows a chart of the type `funnel`:



The following code example demonstrates the basic principle of the implementation of a funnel widget. Some static data is used for demo purposes.

```
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'}"
]
```

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Code example 36: *Script for a sales funnel chart with demo data*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.4.7 HTML widgets

HTML widgets are configured in the type *htmlContentWidget* or one of its subitems. They can be used to display any kind of content on a dashboard.

Available attributes:

- **localization**
  String. Enter the localized values, e.g., for the widget title, using the following syntax: `de: {title:'Wetter'}, en: {title:'Weather'}`. See Localizing widget text.
- **parameters**
  String. Enter any additional parameters for the widget. These parameters can be used in the widget visualization script, where they are available as String parameters of the `render()` and `resource()` methods.
- **title**
  String. Enter the widget title. If the title should be localized, use the following syntax: `_('title')`
- **visible**
  Boolean. Determines whether the widget is shown (“true”) or not (“false”).
- **visualizationScript**
  String. Enter the name of the script of type *Widget visualization* used to configure the widget.

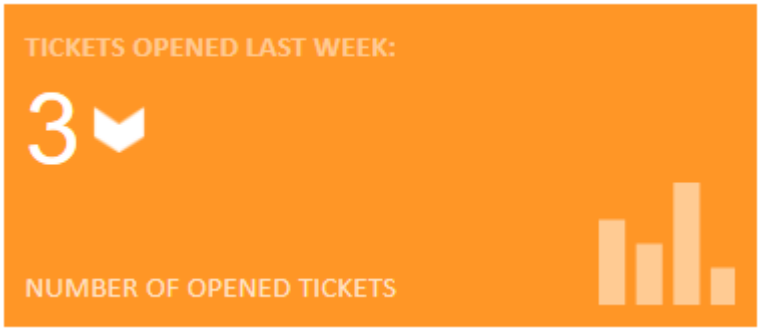
Please see Scripts of Type Widget Visualization for an example which implements a weather widget.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.4.8 KPI widgets

KPI widgets are configured in the type *kpiWidget* or one of its subitems. They use the `jquery-kpiwidget` library.

The following figure shows a KPI widget:



> ⚠ You can set attributes both in the widget script and in the page customization. The values set in the script overwrite the values entered in the page customization.

Available attributes:

- **color**
  String. Background color for the rectangle widget area, for example “#A0B0C0”.
- **footer**
  String. Text to show at the bottom of the widget, can be localized. See Localizing widget text.
- **height**
  String. Widget rectangle height in pixels.
- **localization**
  String. Localized values for string attributes. See Localizing widget text.
- **maxValueForSize**
  String. The numeric widget value above which the font size will be reduced in order to fit the value in the line inside the rectangle, default is “100000”.
- **symbol**
  String. Character symbol or string to show as prefix for the numeric value.
- **title**
  String. Text to show in the top line of the widget, can be localized. See Localizing widget text.
- **trend**
  String. Identifier for the trending symbol to be shown after the numeric value, valid identifiers are “up”, “down”, and “flat”.
- **value**
  String. The numeric value to be shown. This attribute should normally by determined by the widget script, otherwise, a fixed number is displayed.
- **visible**
  Boolean. Determines whether the widget is shown (“true”) or not (“false”).

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.4.8.1 Coding example

KPI widgets are based on the `jquery-kpiwidget` library. The script has to return a HashMap containing the attributes which define the KPI representation.

The following code provides an example of a KPI widget script:

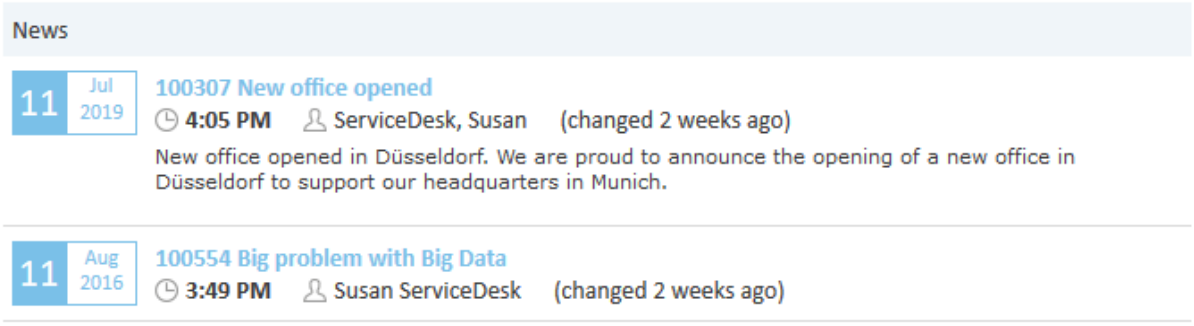
```
import com.consol.cmas.common.model.ticket.*
import java.util.*
import com.consol.cmas.common.model.DateRange
TicketCriteria crt = new TicketCriteria()
def to_date = new Date()
def from_date = to_date - 7
def range = new DateRange(from_date,to_date)
crt.setCreationDateRange(range)
// crt.setStatus(TicketCriteria.Status.OPEN)
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 example 37: *Script for a KPI widget which counts the cases which have been opened during the last week*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.4.9 News widgets

News widgets are configured in the type *newsWidget* or one of its subitems. They can be used to display news on dashboards in the Web Client. The following figure shows an example of a news widget on the dashboard.



Available attributes:

- **author**
  String. Enter “true” if the author who created the news item should be displayed, or “false” if he should not be displayed.
- **color**
  String. Enter the color for the news symbol and headline in hex syntax, e.g., #000000 for black.
- **createDate**
  Boolean. Enter “true” if the date when a news item was created should be displayed, or “false” if it should not be displayed.
- **height**
  Integer. Enter the height of the widget in px.
- **localization**
  String. Enter the localized values, e.g., for the widget title, using the following syntax: `de: {title:'Nachrichten'}, en: {title:'News'}`. See Localizing widget text.
- **scriptName**
  String. Enter the name of the script (type *News publisher*) which contains the logic for retrieving news items.
- **title**
  String. Enter the widget title. If the title should be localized, use the following syntax: `_('title')`
- **updateDate**
  Boolean. Enter “true” if the date when a news item was updated should be displayed, or “false” if it should not be displayed.
- **visible**
  Boolean. Determines whether the widget is shown (“true”) or not (“false”).

Please see Scripts of Type News Publisher for an example script which implements a news widget.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.4.10 Recent changes widgets

Widgets showing recent changes are configured in the type *recentChangesWidget* or one of its subitems. The list of the `excludedHistoryTypes` shows which actions qualify as a recent change.

Available attributes:

- **excludedHistoryTypes**:
  String. The history entry types which should be excluded from the listing as a change. The value is entered as a comma-separated list of the following entry type identifiers:
  - "OPEN": The case has been created.
  - "REOPEN": The case has been re-opened.
  - "UPDATE": The case has been updated.
  - "SUBJECT_CHANGE": The case subject has been updated.
  - "ENGINEER_CHANGE": The assigned user has been changed.
  - "TICKET_USER_ADDED": An additional user has been added.
  - "TICKET_USER_REMOVED": An additional user has been removed.
  - "TICKET_USER_MOVED": An additional user function has been changed from one function to another.
  - "QUEUE_CHANGE": The case's queue has been changed.
  - "ACTIVITY_CHANGE": The case status has changed.
  - "TRANSFERED_TO_ACTIVITY": The case has been transferred to another activity.
  - "CONTENT_ADD": New case content has been added.
  - "ATTACHMENT_DELETE": Case attachment has been deleted.
  - "ATTACHMENT_UNDO_DELETION": The case attachment deletion has been undone.
  - "EMAIL_ATTACHMENT_DELETE": An attachment was deleted from an email.
  - "EMAIL_ATTACHMENT_UNDO_DELETION": An email attachment deletion has been undone.
  - "BOOKING": A time booking has been added.
  - "CONTACT_ADD": A contact has been added to the case .
  - "CONTACT_REMOVE": A contact has been removed from the case .
  - "CONTACTROLE_SWAP": The contact role in the case has been changed.
  - "CONTACTROLE_UPDATE": The contact role in the case has been updated.
  - "TRANSFERRED_TO_CONTACT": The case has been transferred to a new contact.
  - "TICKETRELATION_CREATE": A case relation has been created.
  - "TICKETRELATION_REMOVE": A case relation has been removed.
  - "TICKETRELATION_UPDATE": A case relation has been updated.
  - "TICKETRELATION_MOVE": A case relation has been changed from one relation type to another
  - "RESOURCE_RELATION_CREATED": A resource relation has been created.
  - "RESOURCE_RELATION_UPDATED": A resource relation has been updated.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

  - “RESOURCE_RELATION_REMOVED”: A resource relation has been removed.
  - “RESOURCE_ON_THE_FLY_RELATION_CREATED”: An on-the-fly resource relation has been created.
  - “RESOURCE_ON_THE_FLY_RELATION_UPDATED”: An on-the-fly resource relation has been updated.
  - “RESOURCE_ON_THE_FLY_RELATION_REMOVED”: An on-the-fly resource relation has been removed.
- **excludedQueues**:
  String. A comma-separated list of queues identified by their technical names. The named queues and their cases will be excluded from the listing in the widget.
- **localization**:
  String. The localized title of the widget.See Localizing widget text.
- **maxItems**:
  Integer. The maximum number of items shown in the widget's main view.
- **title**:
  String. The title shown in the header bar.
- **visible**:
  Boolean. Determines whether the widget is shown (“true”) or not (“false”).

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.4.11 Recently visited widgets

Widgets showing recently visited objects are configured in the type *recentlyVisitedWidget* or one of its subitems.

E.4.11.1 Page customization attributes

Available attributes:

- **excludedCustomerGroups**:
  String. A comma-separated list of customer groups identified by their technical names. The named customer groups and their members will be excluded from the listing in the widget.
- **excludedItemTypes**:
  String. A comma-separated list of base object types which will be excluded. The types are identified by the following keywords:
  - TICKET: All cases will be excluded from the listing in the widget.
  - UNIT: All contacts (companies and persons) will be excluded from the listing in the widget.
  - RESOURCE: All resources will be excluded from the listing in the widget.
- **excludedQueues**:
  String. A comma-separated list of queues identified by their technical names. The named queues and their cases will be excluded from the listing in the widget.
- **excludedResources**:
  String. A comma-separated list of resource types identified by their technical names. The named resource types and their resources will be excluded from the listing in the widget.
- **localization**:
  String. The localized title of the widget. See Localizing widget text.
- **maxItems**:
  Integer. The maximum number of items shown in the widget main view.
- **title**:
  String. The title string shown in the header bar.
- **visible**:
  Boolean. Determines whether the widget is shown (“true”) or not (“false”).

E.4.11.2 System properties

The ConSol CM system behavior concerning data gathering for the recently visited widget can be configured using system properties.

Depending on the use of the widget, the data gathering which is the basis for the widget’s display can be switched on or off. If the widget is not displayed, it does not make sense to have the data gathering thread running in the background using system resources for nothing. In this case, you can switch off the data gathering process using the following system property:

- cmas-core-server, recent.items.persistence.enabled

If the data gathering has been switched on, you can configure its behavior using the following system properties:

- recent.items.max.per.engineer
- recent.items.cleanup.interval.minutes

- recent.items.cleanup.cluster.node.id

E.4.12 Table widgets

Table widgets are configured in the type *tableWidget* or one of its subitems. They use the DataTables library.

The following figure shows a table widget with some demo data:

Job table

Show 10 entries Search:

First name	Last name	Job title	Expertise level	Hire date
Homer	Simpson	Nuclear disaster	3	25.03.1989
Zaphod	Beeblebrox	President of the Galaxy	0	12.09.1979
Sheldon	Cooper	Mad scientist	321	01.04.2006
Robin	Scherbatsky	Anchorwoman	25	10.09.2004
Elmer	Fudd	Duck hunter	1	15.12.1962
Eric	Cartman	Pupil	10	23.02.1995
Mickey	Mouse	Private investigator	111	04.11.1932
Wilma	Flintstone	Housewife	64	07.01.1964
Charlie	Harper	Composer	12	16.07.2001
Daenerys	Targaryen	Mother of dragons	238	08.05.2010

Showing 1 to 10 of 12 entries Previous 1 2 Next



You can set attributes both in the widget script and in the page customization. The values set in the script overwrite the values entered in the page customization.

General attributes:

- **developmentMode**
  Boolean. Enables the development mode for the widget. If this attribute is set to “true”, the widget is displayed with a red border and a JSLint validation is performed on the widget script. If there are errors in the widget script, the error messages are displayed instead of the widget.
- **localization**
  String. Localized values, i.e., `de: {subject:'Thema', yes:'Ja'}, en: {subject:'Subject', yes:'Yes'}`. See Localizing widget text.
- **visible**
  Boolean. Determines whether the widget is shown (“true”) or not (“false”).

DataTables attributes:

- **columns**
  String. Options which you can apply to the columns objects.
- **data**
  String. Data displayed in the widget.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **options**
  String. Options which you can apply to the table.

See https://datatables.net/reference/option/ for details about the DataTables attributes.

E.4.12.1 Coding example

The following example demonstrates the basic principle of the implementation of a table widget. The script has to return a HashMap containing the attributes which should be set. The displayed data is returned in the `data` attribute. For this example, some static data is used for demo purposes.

```
// 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 example 38: *Script for a table widget*

E.5 Layout Customization Using Skinning

The layout of the Web Client can be customized using skinning. The required CSS files, images, and libraries can be saved on the file system.

Please proceed as follows to create a customized skin:

1. Create a folder called `design` in the ConSol CM data directory (value of the system property cmas-core-shared, data.directory).
2. Create a folder within the `design` directory and set the system property cmweb-server-adapter, themeOverlay to the name of this folder.
3. Add the customized layout files. There are two possibilities:
   - You can place everything in the theme folder. The theme folder can contain one or several files with the custom styling.
   - You can create a file called `theme_<folder name>.css`. Its content is merged with the default CSS file.
4. The changes are loaded automatically.

> If you do not want to set the system property, you can create a subfolder called `default` and use it for your skin files.

The existing default styling is only overwritten for the aspects which are explicitly configured in the custom skin. It is therefore possible to create, for example, a custom CSS file which only contains the settings which should be changed with respect to the default styling. This facilitates the maintenance of the custom CSS file after ConSol CM updates.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.5.1 Example

The following example shows how to change a simple layout aspect, in this case the color of the menu bar, using skinning.

The following figure shows the default layout of the Web Client:

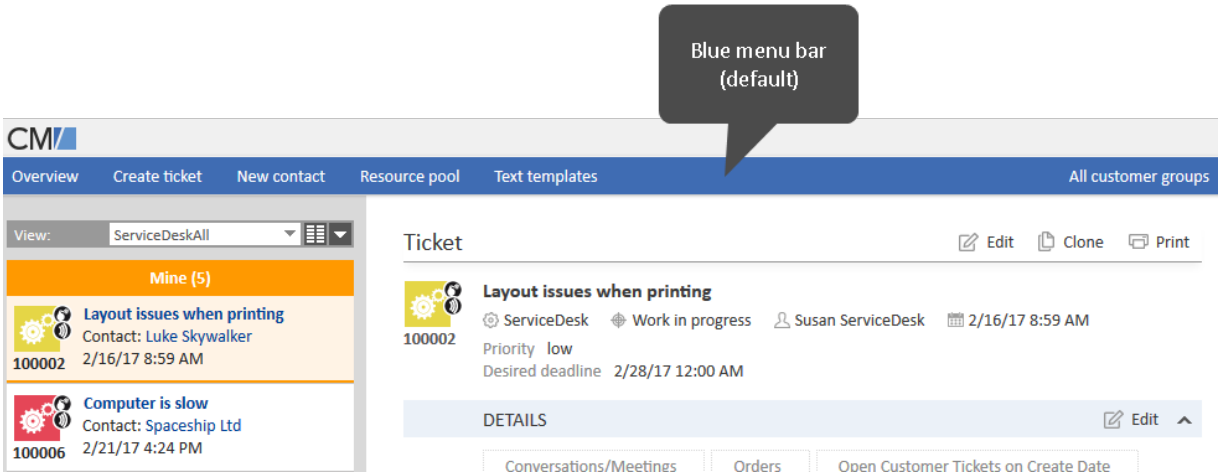


Figure 287: *Web Client with default skin*

The following figure shows the new layout with the customized color of the menu bar:

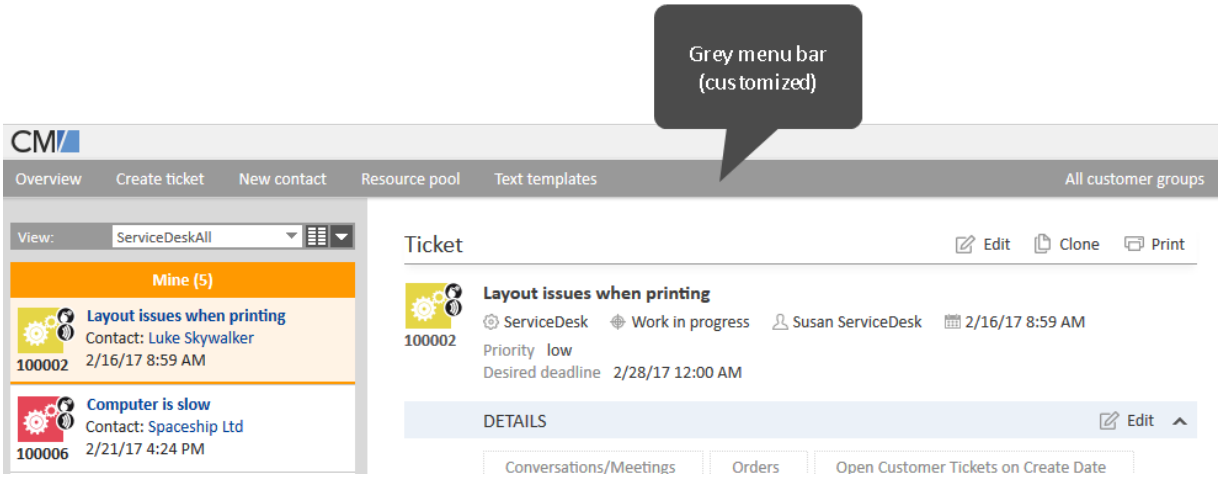


Figure 288: *Web Client with customized skin (menu bar)*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.6 Design and Configuration of REST-based ConSol CM Client GUIs

E.6.1 Introduction

In order to provide a comfortable configuration of the user interface for REST clients, the Admin Tool contains the navigation group *Clients*.

It can be used to configure the user interface of

- CM/Track
- customer-specific CM REST clients

Configuration in this context means:

- displayed components
- layout of the page
- localization for various languages

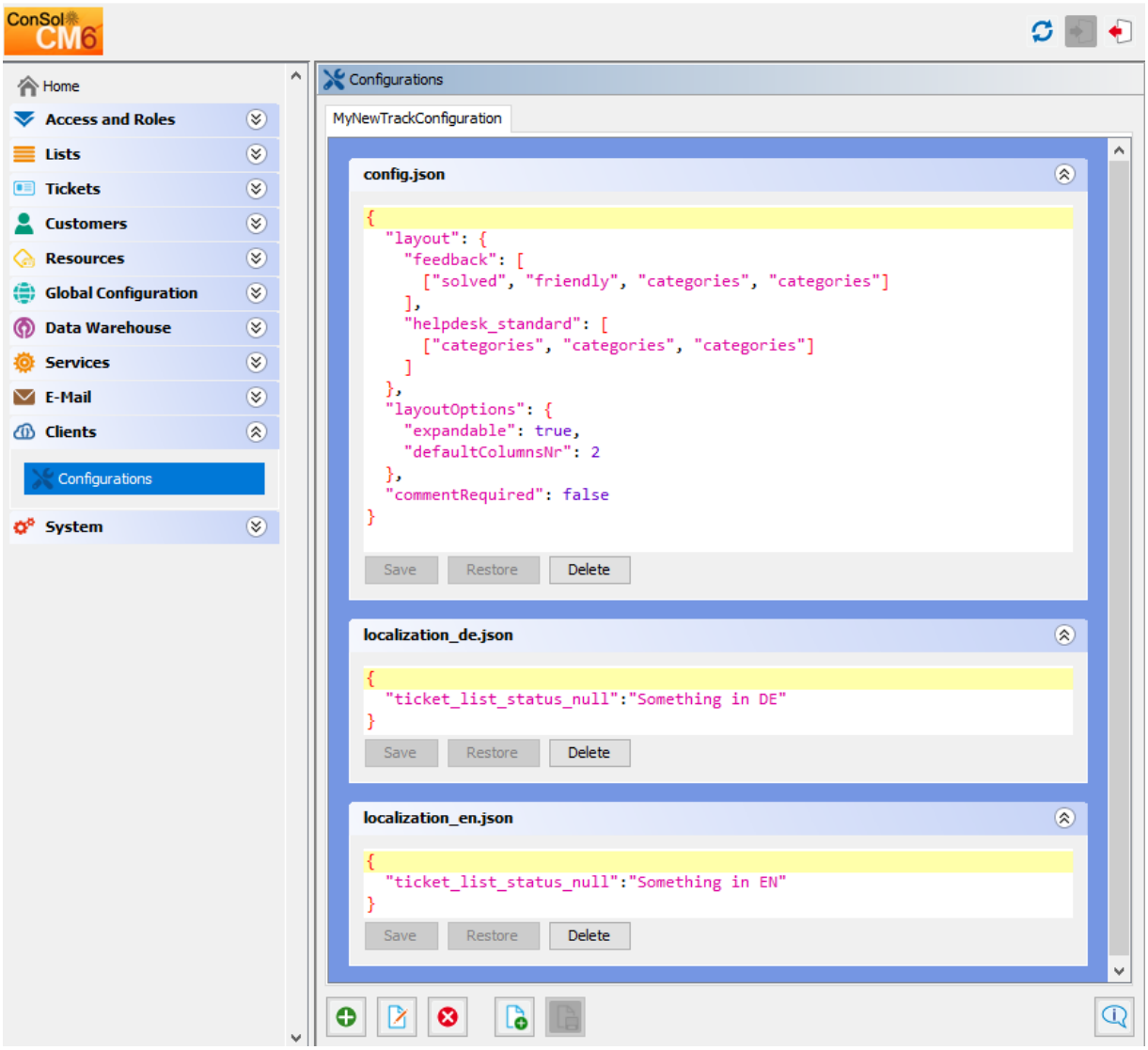


Figure 289: *ConSol CM Admin Tool - REST client configuration (example)*

The configuration panel features a tab for each existing configuration, labeled with the configuration name. A configuration consists of a number of files in the JSON (JavaScript Object Notation) format, which will be provided to the client upon request. In the tab, the files are shown each one after another to be edited. On the bottom of the panel are buttons to organize the configurations. These can be used to achieve the tasks addressing a configuration as a whole:

- **Create**
  new configuration opens a dialog to name the configuration and choose a template to base it on.
- **Edit**
  configuration opens a dialog to rename the selected configuration and choose the template.
- **Delete**
  configuration removes the configuration as a whole and its tab will not be present anymore.
- **Create new file to the current configuration**
  opens dialog to set the file name and extends the configuration with an empty file with this name.
- **Save**
  all files saves all unsaved changes in all files in the currently selected configuration.
- **Examples**
  opens a window with examples from templates with each template showing in a tab. Currently there is only one template, named *TrackV2*, available for selection of the client type in the *Create* and *Edit* dialogs as well as in the *Examples* window. The following example shows a newly created configuration, based on the example track configuration.

Each file section offers file-oriented functions. The respective buttons are located below the file content editing section:

- **Save**
  stores the changes made to the file to deliver them to the client.
- **Restore**
  sets the file content back to the state after the last time it was saved.
- **Delete**
  removes the whole file from the configuration.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.6.2 Configuration Principle

The configuration which is relevant for the client depends on the URL the client has called. The specific configuration for the client is delivered by the Angular app within ConSol CM.

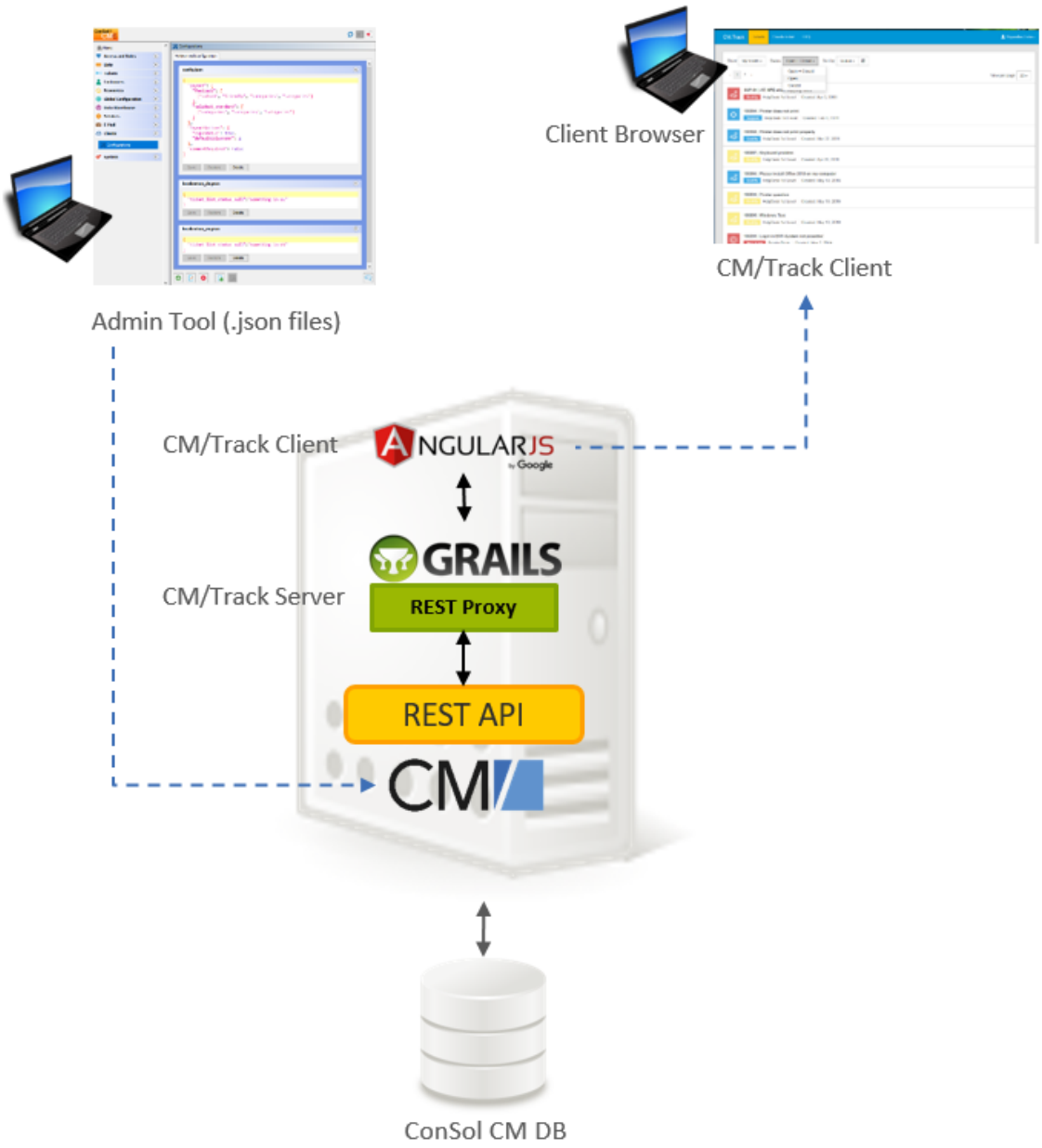


Figure 290: *Principle of REST client GUI configuration with the Admin Tool*

E.6.2.1 Mapping of the Configuration to the CM/Track Instance

The mapping between the specific domain and the required configuration is configured using the following CM system property:

`cmas-restapi-core, domain.map.for.client.config.<configName>`

The value has to be the domain without the protocol, target port (if required), and path of the respective CM/Track instance.

Example: `cmas-restapi-core, domain.map.for.client.config.trackConfig1` = “myserver-:8080/track”

If two CM/Track instances should work with the same configuration, use the following syntax:

`cmas-restapi-core, domain.map.for.client.config.<configName>` = <comma-separated list of domains with ports>

Example: `cmas-restapi-core, domain.map.for.client.config.trackConfig2` = “myOther-Server:8080/track, myNewServer:8180/track”

The CM system property has to be added manually to the system configuration.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.6.3 Configuration of Specific Pages in CM/Track

The configuration of specific pages in CM/Track is done using configuration and localization files.

The file `config.json` is used to configure aspects of CM/Track after the user logged in. It includes both layout settings for the pages to create and display tickets, and settings for the welcome page. The localizations for the non-public pages of CM/Track can be found in the so called localization files, e.g., `localization_en.json` or `localization_de.json`.

The file `public.json` is used to configure the login page. It includes both settings and localizations.

E.6.3.1 Configuration of the Ticket Pages

The configuration of the layout of the ticket pages, i.e., the page to create a ticket and the page to view a ticket, in CM/Track is based on the following files:

- `config.json`
- localization file (e.g., `localization_en` or `localization_de`)

> Please note that each ticket field which should be displayed in CM/Track has to be configured using the annotations concerning customer exposure. Only fields which are customer-exposed will be displayed. Please read section CM/Track: Data Availability For Customers for a detailed explanation.

config.json

In this file, the layout/order of the ticket fields is defined, and general layout and configuration parameters are set.

There are three `layout` objects, which allow you to set a different layout for creating and for viewing tickets:

- `layout`
  General layout, applies to both creating and viewing tickets
- `editLayout`
  Layout for creating tickets
- `viewLayout`
  Layout for viewing tickets

The structure of the three `layout` objects is identical:

```
"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]
   ]
}
```

If both the general layout and the specific layout are used, the settings for the specific layout (view or edit) overwrite the settings of the general layout.

Rules for writing a JSON layout object:

- Each line (array) which contains ticket fields must have the same number of elements. Use the placeholder EMPTY for empty cells.
- Do not use comments within the JSON statement.

Example:

```
"layout": {
   "helpdesk_standard": [
      ["module", "categories","categories"],
      ["reactiontime","priority","EMPTY"]
   ]
},
```

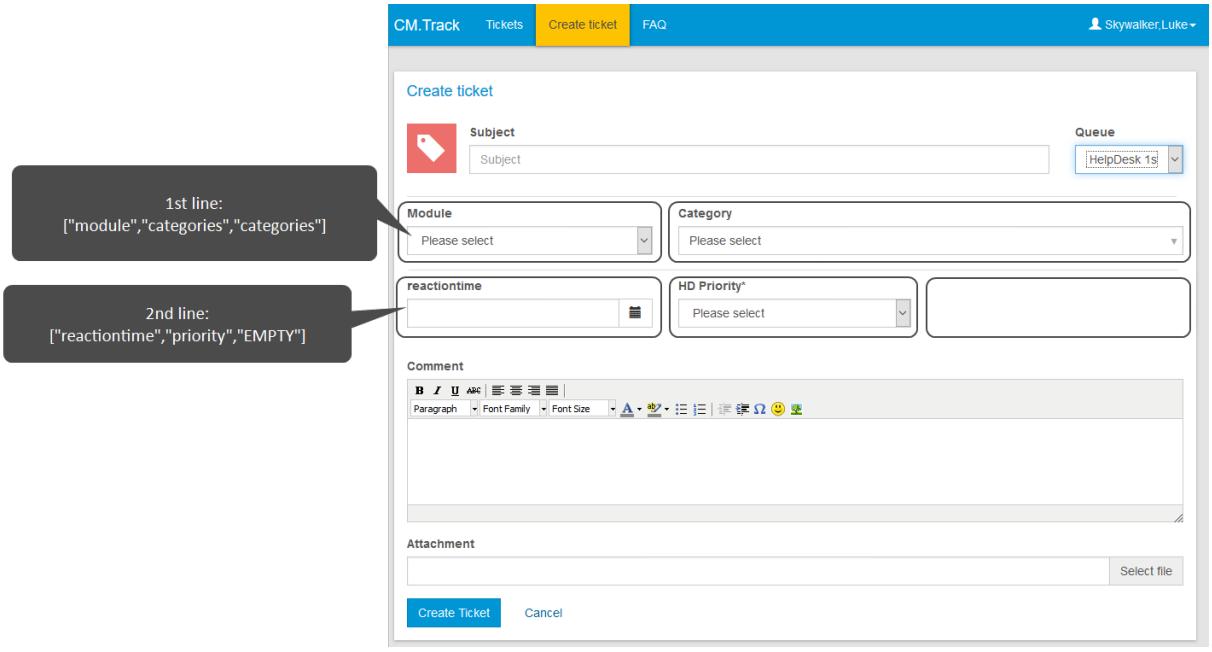


Figure 291: *CM/Track client - Display of the ticket fields of the example layout shown above*

There are three `layoutOptions` objects, which allow you to set different layout options for creating and for viewing tickets:

- `layoutOptions`
  General layout options, apply to both creating and viewing tickets
- `editLayoutOptions`
  Layout options for creating tickets
- `viewLayoutOptions`
  Layout options for viewing tickets

The structure of the three `layoutOptions` objects is identical:

```
"layoutOptions": {
   "expandable": false,
   "defaultColumnsNr": 3
}
```

If both the general layout options and the specific layout options are set, the settings for the specific layout options (view or edit) overwrite the settings of the general layout options. The following settings can be made in these objects:

- `expandable` (see example below)
  - “true”: Ticket fields which are not explicitly mentioned in the JSON layout object but which are annotated as customer exposed will be displayed.
  - “false”: Only ticket fields which are explicitly mentioned in the JSON layout object are displayed.
- `defaultColumnsNr`: This is only used if `expandable` = “true” and a ticket field group is not explicitly layouted. Then the `defaultColumns` are used for display.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

The `acf` object allows to determine the layout of ACFs in CM/Track independently from their layout in the Web Client. For each ACF, the order and size of the fields can be determined. The configuration works in the same way as the settings for the `layout` attribute, i.e. there is one line / array for each row and the names of the fields which should span more than one column are repeated. Fields which are not configured explicitly are displayed below the configured fields.

The following example shows a configuration for an ACF called *qualify* with the field *comment_medium* spanning two columns:

```
"acf" : {
  "qualify": [
    ["impact", "severity"],
    ["categories", "comment_short"],
    ["comment_medium", "comment_medium"]
  ]
}
```



If the field names are not unique because the ACF contains fields of several ticket field groups, you need to use the syntax *groupName.fieldName* to reference the fields.



Use 3 columns in the ACF if it contains tables which require lots of space.

If a **comment** is required to create a ticket, make the following setting:

```
"commentRequired" : true
```

If a **prefill** script should be executed to set default values when creating a ticket, make the following setting:

```
"usePrefillScript" : true
```

The prefill script is configured in the same way as for the Web Client, please see Scripts of Type Default Values.

You can determine the **order** in which the ticket **history entries** are displayed in CM/Track using the attribute `historyOrder`. It can have two values: “descending” for displaying the newest entries first (default value) and “ascending” for displaying the oldest entries first.

Example:

```
"historyOrder": "ascending"
```

Example:

Only the first two lines are layouted, but more fields (e.g., *infotext*) are displayed, because expandable is “true”.

```
{
   "layout": {
      "helpdesk_standard": [
         ["module", "categories","categories"],
         ["reactiontime","priority","EMPTY"]
      ]
   },
   "layoutOptions": {
      "expandable": true,
      "defaultColumnsNr": 2
   },
   "commentRequired": false
}
```

Code example 39: *Example config.json file*

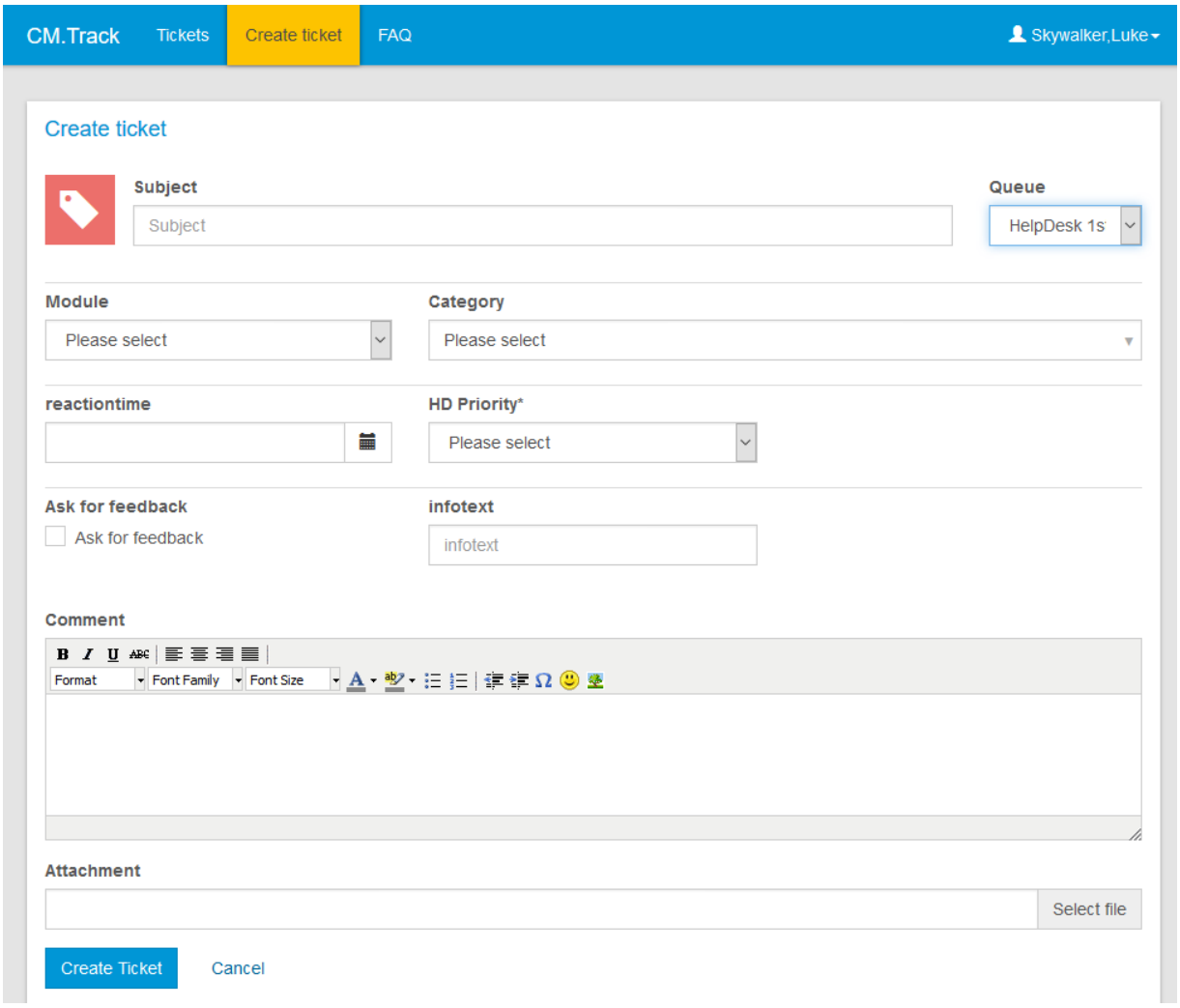


Figure 292: *CM/Track layout using layout of code example above*

Same example using `expandable` = “false”: only the explicitly layouted ticket fields will be displayed.

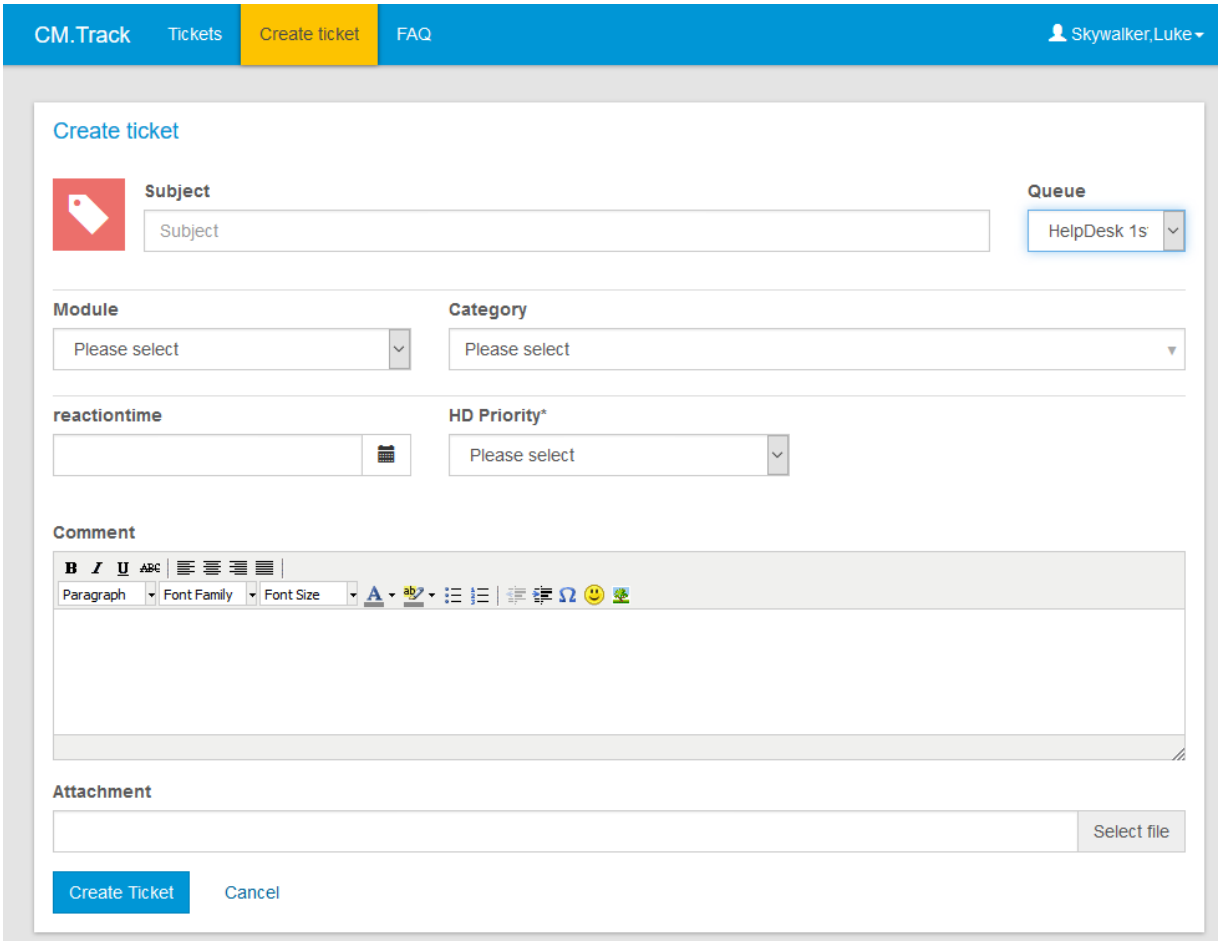


Figure 293: *CM/Track layout using layout of code example above, but expandable = false*

Localization File

The localization files, e.g., `localization_en.json` and `localization_de.json`, contain simple mappings of properties to values. The values are the terms which should be displayed in the GUI in the respective language. You can view the template files with the localizations used in your version of ConSol CM by clicking the *Examples* icon.

You can adapt the delimiters which are used for fixed point numbers by adding the attributes `number_decimal_separator` and `number_group_separator` with the desired delimiters to the localization file of each language.

The following example shows the attributes with the default values for English:

- **Decimal separator:**

```
'number_decimal_separator': '.'
```

- **Thousands separator:**

```
'number_group_separator': ','
```

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.6.3.2 Configuration of the Welcome Page

The welcome page is displayed after the user has logged in to CM/Track. It provides the following elements:

- A customizable welcome message
- A customizable sub-header
- A search field
- Colored boxes with links to create tickets in the queues for which the user has the required permissions. If the number of queues is greater than four, then the fourth box is a placeholder offering a list with all remaining queues. Each regular box shows a button *Create*, which directly leads to the ticket create page for this particular queue.
- A box displaying news, if the news feature is configured.

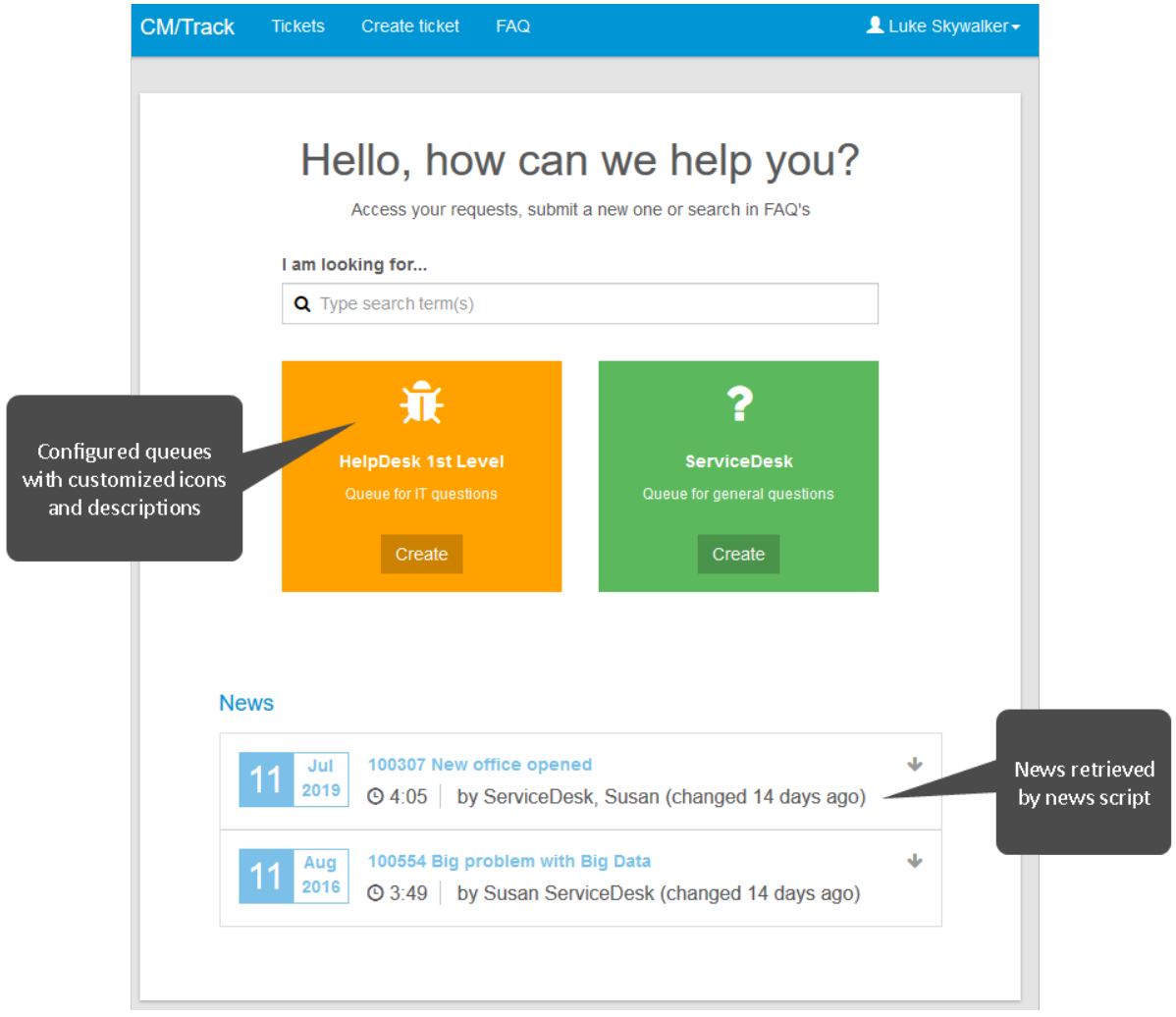


Figure 294: *Welcome page in CM/Track*

In order to configure the welcome page of your customer portal, CM/Track, you have to adapt the `config.json` file. You can view the template file used for your version of ConSol CM by clicking the *Examples* icon.

The following code shows an example of the welcome page section of a `config.json` file:

```
"welcomePage": {
   "queuePanels": [
      {
         "@name": "HelpDesk_1st_Level",
         "description": "custom_HD_description",
         "icon": "fa-bug"
      },
      {
         "@name": "ServiceDesk",
         "description": "custom_SD_description",
         "icon": "fa-question"
      }
   ]
}
```

Code example 40: *welcomePage section of config.json file*

The following parameters are available:

- `@name`: technical name of the queue
- `description`: description of the queue, can either be a fixed string without localization or a key from the `localization_en.json` file or another localization file. (From the example above: *custom_HD_description* and *custom_SD_description* are keys which have to be present in the localization files).
- `icon`: name of an icon from the FontAwesome icon set (see http://fontawesome.io/icons/). Use the following syntax: fa-<name of the icon>.

If no explicit configuration is provided, the default values (description of the queue in the Admin Tool and folder icon) are used. All queues for which the user has permissions to create a ticket are displayed.

If more than one queue is available, the order of the queues is the order in which the queues are configured in the `config.json` file. The queues which are not explicitly configured are displayed at the end.

The localization for the welcome page beyond the queue configuration is done by a small set of keys in the localization files for each language configured, e.g., `localization_en.json`.

```
{
   "custom_HD_description": "Queue for IT questions",
   "custom_SD_description": "Queue for general questions"
}
```

Code example 41: *Excerpt from localization_en.json showing the custom labels created for the queue description*

The above configuration results in the above welcome page for a user with access to two queues. Note that all queues to which the user has access are displayed on the welcome page, even if they have not been configured explicitly. Queues without a customized configuration are shown with the default settings, i.e., folder icon and description from the queue configuration in the Admin Tool.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Displaying News on the Welcome Page

The welcome page of CM/Track can be extended by including news. For this purpose, a script of the type *News Publisher* (see Scripts of Type News Publisher) is created in the navigation item *Scripts and Templates* of the Admin Tool, and the display configuration of the news is added to the `config.json` file. The news which are returned by the script can be retrieved from several sources, for example, ticket comments or RSS feeds, or created directly in the script.

The following code shows an example of the news configuration in the `config.json` file. It configures the display of news items as created by the script `news.groovy`. Please see News widgets for a description of the other settings.

```
"newsConfiguration": {
   "scriptName": "news.groovy",
   "autoscroll": "smooth",
   "title": "News",
   "height": 350,
   "createDate": true,
   "updateDate": true,
   "author": true,
   "color": "#488ca1"
},
```

Code example 42: *newsConfiguration section of config.json file*

The localized title of the news section is added to the attribute `newsTitle` of the `localization_en.json`/`localization_de.json` file. If no localized title is specified, the default title “News” is displayed.

E.6.3.3 Configuration of the Public Pages

To customize a public CM/Track page (the sign in page, the password change pages) you need to add a `public.json` file to your track clients configuration. The `public.json` file contains localizations and the setting regarding the password reset functionality. The localizations are all to be found in the JSON object `signin`, `i18n`.

You can view the template file for your version of ConSol CM by clicking the *Examples* icon.

In order to change words and/or phrases which are displayed on public pages, just edit the `public.json` file in the Admin Tool and save it. The change will be visible on the GUI just in time.

E.6.3.4 Configuration of the Password Reset and Change Functionalities

The functionality for password reset on the login page and password change after logging in can be enabled and disabled in the respective configuration files.

config.json

The `config.json` file allows you to determine whether the *Change password* button should be displayed after logging in to CM/Track. If `disablePasswordChange` is set to “true”, the user cannot change the password. If it is set to “false”, changing the password is possible.

```
"disablePasswordChange": true
```

Code example 43: *Excerpt from config.json to disable the functionality to change the password*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Per default, i.e., when the `disablePasswordChange` parameter has not been defined explicitly, the menu item *Change password* is available.

public.json

The `public.json` file allows you to determine whether the *Forgot your password?* link should be displayed on the login page. If `disablePasswordReset` is set to “true”, the user cannot reset his password. If it is set to “false”, resetting the password is possible.

```
"disablePasswordReset": true,
```

Code example 22: Excerpt from public.json to disable the functionality to reset the password

Per default, i.e. when the `disablePasswordReset` parameter has not been defined explicitly, the link *Forgot your password?* is displayed.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

E.6.4 Creating New Configuration Files

You can add a new JSON file to the configuration using the button *Create new file to the current configuration*. There are two use cases for this functionality.

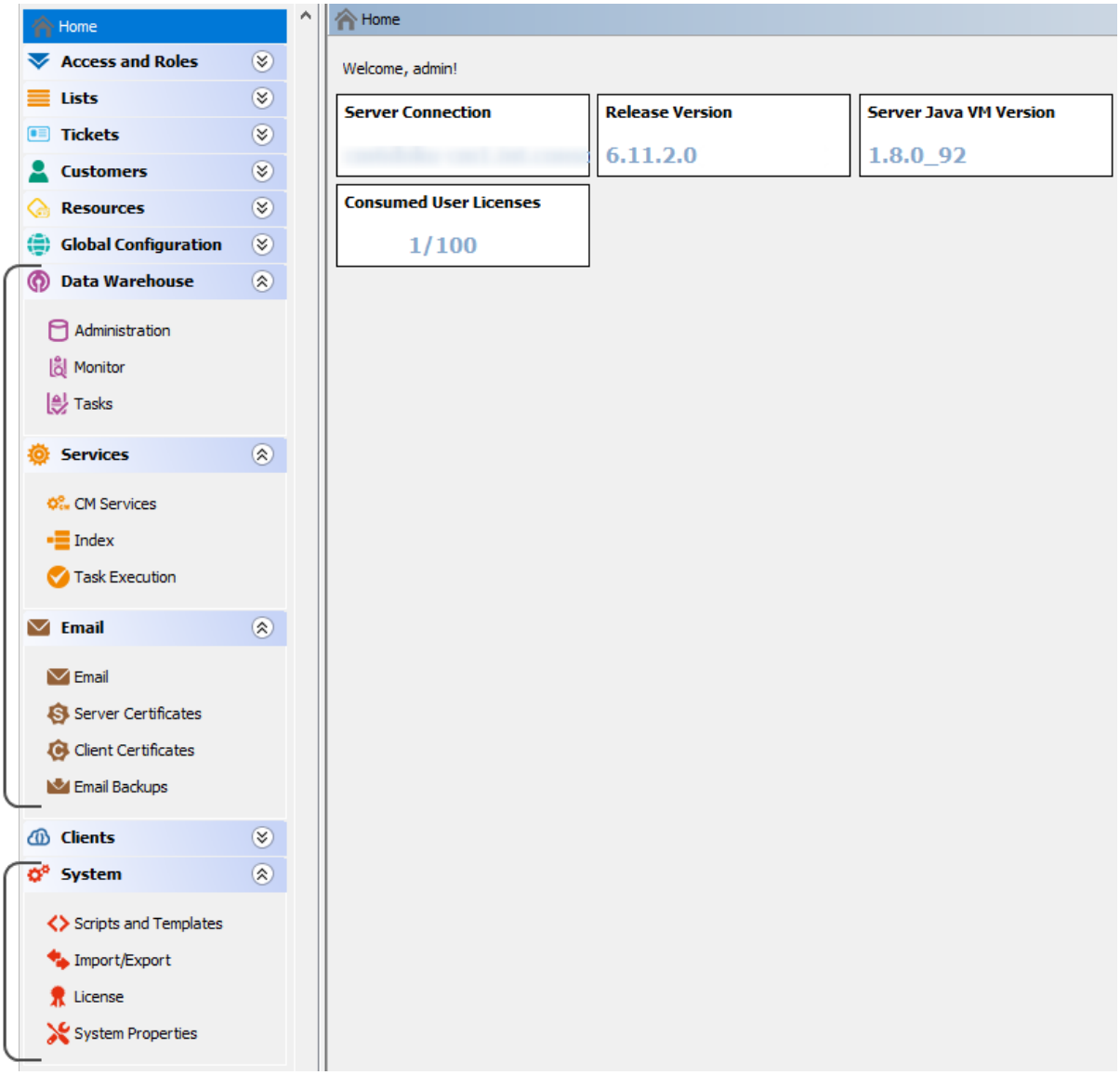
- You want to add a new localization file, because you have added another language to your system configuration. Use `localization_<browser locale>` as name for such a file. You could, for example, add a file named `localization_fr` for French translations. Build the new file based on the existing example localization files.
- You want to add a new file which is required in a customer-specific project. This is a case for experienced CM consultants and developers.

E.6.5 Behavior Concerning System Installations and Updates

The templates/examples for the REST client GUI configuration are added during the installation or update of ConSol CM. You can create a new configuration based on the templates as required. When you update ConSol CM to a newer version, the files of any existing configurations are left as is. This means that new configuration options and localizations are not added automatically. You can add them manually after the update based on the template files.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F - Expert



In this section, you will learn how to configure some advanced settings in ConSol CM:

The following topics are covered:

- Ticket Administration: You will learn how to delete or re-open tickets.
- Data Warehouse (DWH) Management: You will learn how to configure CM to initialize and work with the Data Warehouse, the basis for ConSol CM Reporting.
- CM Services: You will learn which services work within the ConSol CM application.
- Search in ConSol CM: You will learn all about searching in ConSol CM, e.g., how to manage the indexer or how to implement search actions for the Action Framework.
- The Task Execution Framework (TEF): You will learn what the TEF is and how you can configure it in your ConSol CM system.
- The ConSol CM Action Framework: You will learn how to configure actions for customers, resources or search results.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- Email Configuration: You will learn all about the email functionality in ConSol CM and how to configure the respective modules.
- Scripts and Admin Tool Templates: You will learn which script and template types are managed with the Admin Tool and how to code these scripts and templates yourself.
- Deployment (Import/Export): You will learn all about ConSol CM scenarios which are used for ConSol CM system export and import.
- License Management: You will learn how to install a new license in your ConSol CM system.
- System Properties: You will learn what system properties are and what you can do with them to configure basic settings of ConSol CM.
- Working with Templates: You will learn all about text and document templates which can be used as integral part of your ConSol CM system.
- Time Booking Using ConSol CM: You will learn what you have to do to prepare your ConSol CM system for time booking.
- Authentication Methods in ConSol CM: You will learn various configurations for the engineer authentication in the Web Client.
- ConSol CM External Interfaces: You will learn about the different external interfaces of ConSol CM.
- System Architecture: You will get a first glimpse of the ConSol CM system architecture. More information on this topic is provided in the *ConSol CM Setup Manual* and the *ConSol CM Operations Manual*.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.1 Ticket Administration

This chapter discusses the following:

F.1.1 Introduction to Ticket Administration 461

F.1.2 Ticket Administration Using the Admin Tool 461

F.1.1 Introduction to Ticket Administration

Ticket administration allows you to:

- **Delete tickets**
  e.g., if a ticket was created by mistake.
- **Reopen tickets**
  e.g., if a ticket has been closed too early.

> ⚠ Please keep in mind that a ticket which is reopened starts the process at the start node of the respective workflow. So when a ticket has passed through nodes where events are triggered that should be performed only once (e.g., the ticket is passed to an approver) it might be better to open a new ticket. An alternative is to modify the workflow to contain a shortcut which allows such tickets to bypass steps to avoid that these steps are run multiple times for that ticket.

F.1.2 Ticket Administration Using the Admin Tool

In the Admin Tool, you can manage tickets using the navigation item *Administration* in the navigation group *Tickets*.

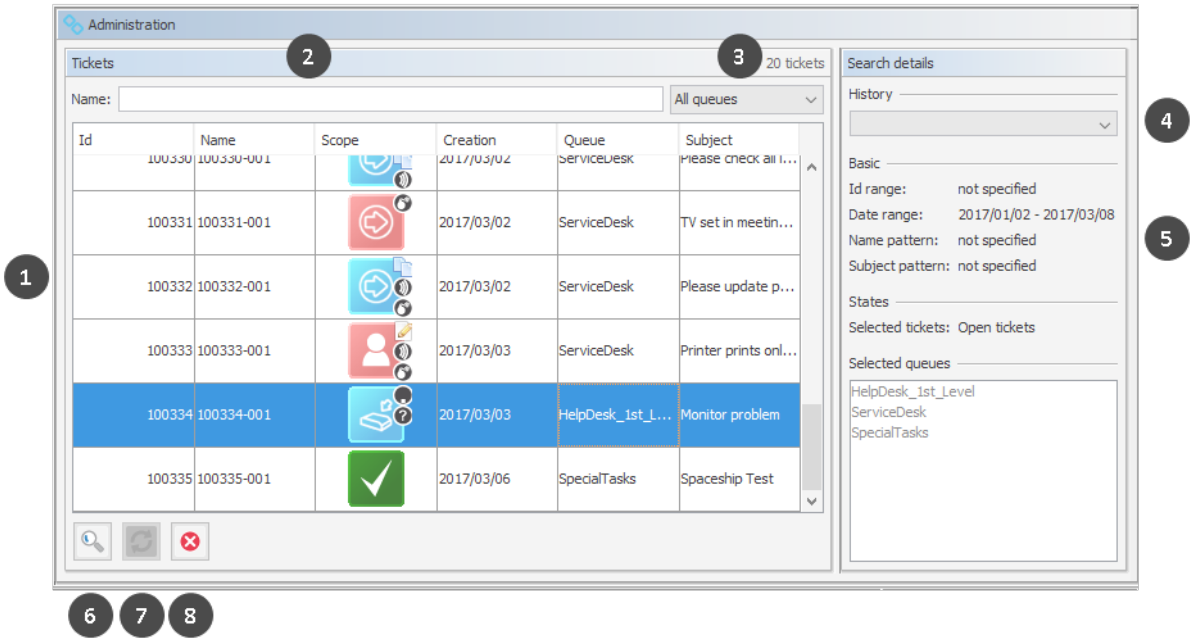


Figure 295: *ConSol CM Admin Tool - Tickets, Administration: List of tickets after search*

The ticket administration screen consists of the following elements:

- List of all tickets which match the search criteria (1)
- Filter ticket by name (2)
- Filter tickets by queue (3)
- List of all recently executed searched (4)
- Details of the current search (5)
- Button to define the search criteria to start a new search (6)
- Button to reopen the selected tickets (7)
- Button to delete the selected tickets (8) - only present if enabled in the CM system

F.1.2.1 Deleting or Reopening Tickets

For these operations you can either use the buttons below the list or you can use the context (right-click) menu.

- **Buttons:**
  Select the desired tickets in the list and click the *Delete* button to delete tickets or click the *Reopen* button to reopen tickets. If you confirm the following dialog with *Yes*, the corresponding action will be executed.
- **Context menu:**
  Select the desired tickets in the list and use the right mouse button to open the context menu. Select the desired operation.

F.1.2.2 Switching off the Delete Functionality Using a System Property

The delete functionality can be disabled system-wide using the system property cmas-app-admin-tool, delete.ticket.enabled. This is a boolean property. When it is set to “false” the *Delete* button is no longer displayed and the delete functionality is no longer available in the context menu.

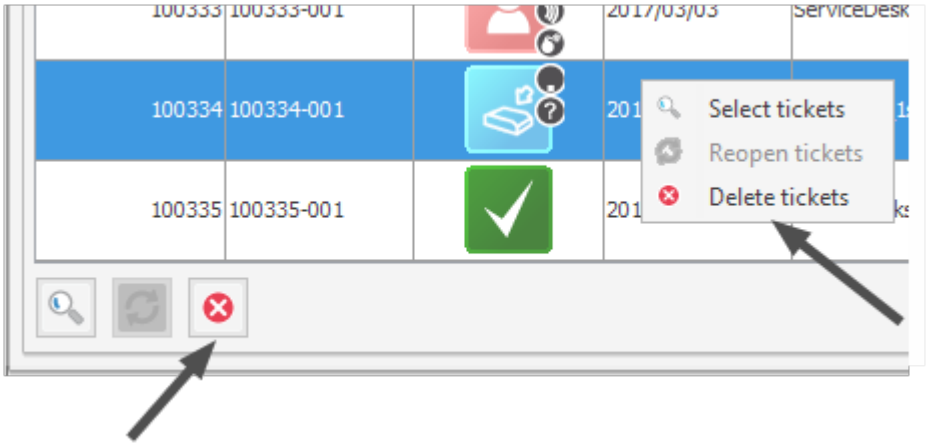


Figure 296: *ConSol CM Admin Tool - Tickets, Administration: cmas-app-admin-tool, delete.ticket.enabled = true*

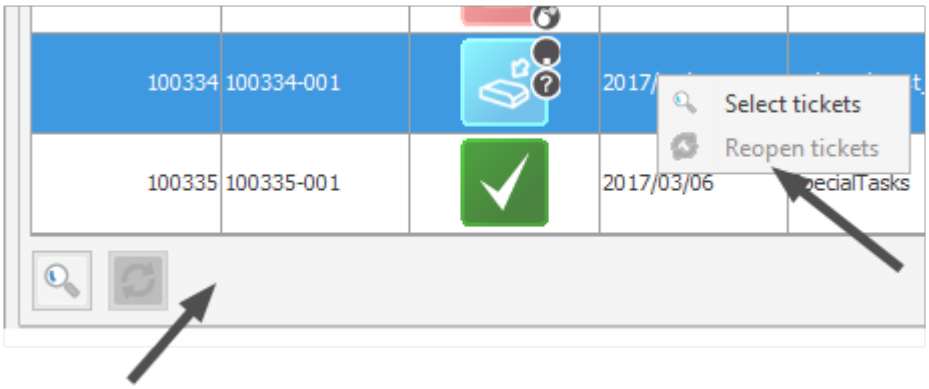


Figure 297: *ConSol CM Admin Tool - Tickets, Administration: cmas-app-admin-tool, delete.ticket.enabled = false*

F.1.2.3 Searching for Tickets

To search for tickets you want to delete or reopen click the *Search* button in the bottom left corner of the page or use the context menu. A pop-up window appears where you can enter the search criteria.

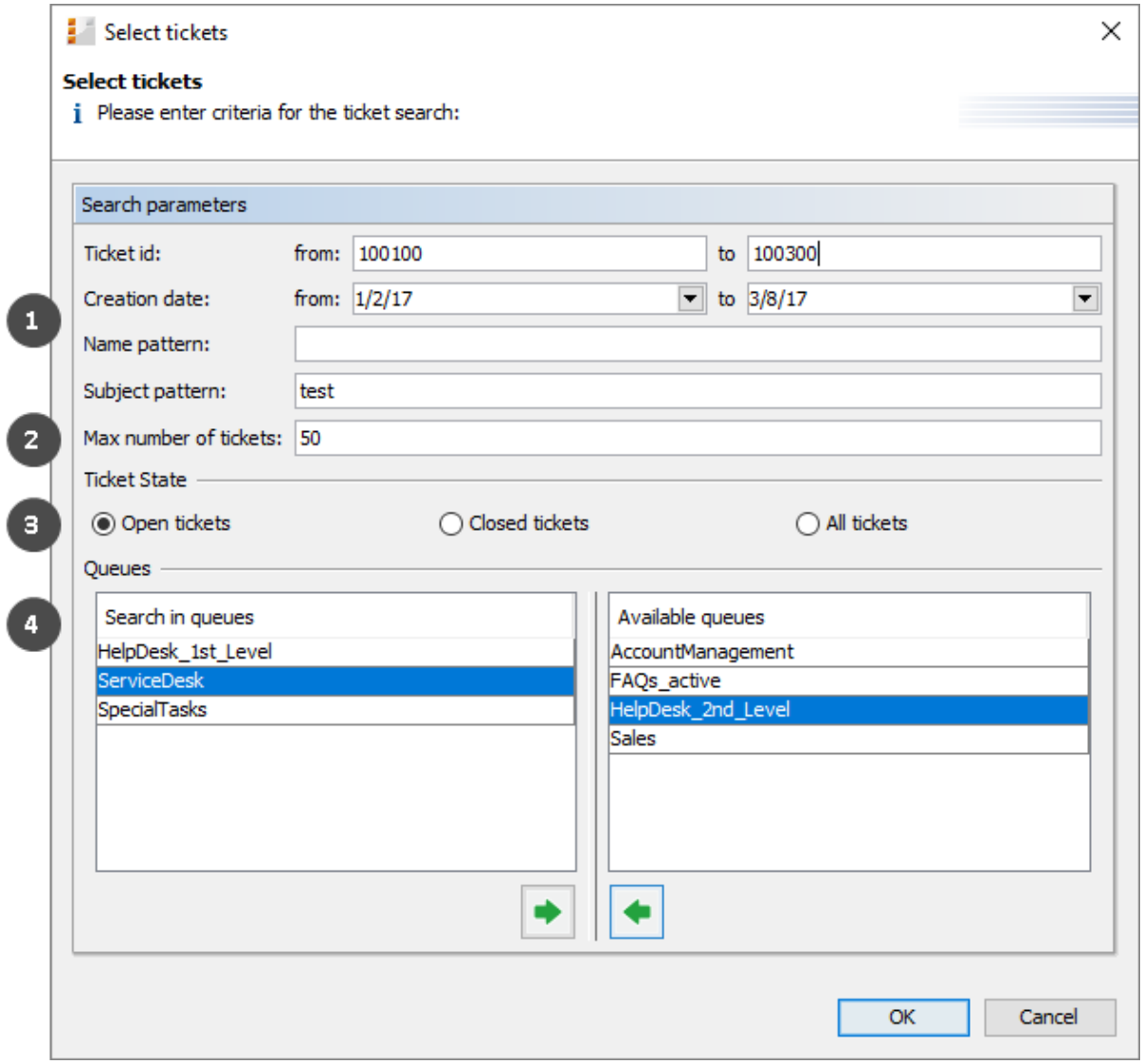


Figure 298: *ConSol CM Admin Tool - Tickets, Administration: Ticket search*

The following parameters can be used for searching:

- **Search criteria (1)**
  - **Ticket id**
    You can enter an ID range for the tickets here.
  - **Creation date**
    Via calendars you can restrict the search to tickets opened within a given period of time.
  - **Name pattern**
    Here you can enter keywords or search patterns for the ticket name.
  - **Subject pattern**
    In this field you can enter keywords or search patterns for the ticket subject.
- **Max number of tickets (2)**
  Here you can specify the maximum number of tickets to be displayed in the list.
- **Ticket State (3)**
  Using the radio buttons you can determine if you want to search for *open*, *closed*, or *all* tickets.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Queues (4)**
  The list on the right shows the available queues. Select the queues to search in and click the *Assign* button to move them to the search list on the left. If you do not choose any queues the search will be extended to all available queues.



Please note that there is a difference between the ticket id and the ticket name! The ticket id is an internal ID in the CM database which is usually not displayed on the GUI. The ticket name is displayed below the ticket icon and is - in everyday life - often called the ticket id, even though this is not correct. The ticket name might contain a prefix (e.g., SUP), depending on the queue-specific configuration. The ID will never contain a prefix!

See the example in the figure above:

The ticket id is *100103*, the ticket name is *SUP-116*. To search the CM database for this ticket you could either use a criterion like *ticket id* is between *10.000* and *200.000*, or a criterion like *ticket name pattern* should be *SUP-*1**.

Click *OK* to start the search. The result will be displayed on the *Ticket Administration* page. If the list is too long, you can limit the display using the name and queue filters above the list.

In the area next to the ticket list on the right you can find an overview of the search criteria you have chosen. The list box *History* above this area contains your most recent searches. If you click on an entry in this list a pop-up window with the criteria of the selected search will open. You can modify the search here or run it as-is.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.2 Data Warehouse (DWH) Management

This chapter discusses the following:

F.2.1 Introduction 467
F.2.2 DWH Management Using the Admin Tool 468
F.2.3 DWH-Related System Properties 481
F.2.4 Transfer Mode 481
F.2.5 Expert DWH Information 483



To set up a DWH, a running ConSol CM Reporting Framework (CMRF) instance is required. If your system does not include a CMRF yet, please consult your ConSol CM manager or contact ConSol Software.

F.2.1 Introduction

F.2.1.1 Data Warehouse

A data warehouse (commonly known as a ***DWH***) is a collection of data from one or more systems and/or databases that provides a basis for reporting and data analysis. Often, imported data is combined or rearranged (integrated) to make it more suitable for reporting and analysis purposes. A more detailed introduction to the data warehouse principle and to the ConSol CM data warehouse is provided in the *ConSol CM DWH Manual*.

F.2.1.2 ConSol CM Data Warehouse and ConSol CM Reporting Framework

A ConSol CM default installation comprises all modules that are required to build a CM data warehouse. One of the two core components is the ***ConSol CM Reporting Framework (CMRF)***.

This is a Java EE application which synchronizes the data between a ConSol CM database and a DWH database. The following picture shows an overview of the system architecture of a typical DWH/CMRF installation. We recommend that you use separate servers for the ConSol CM and CMRF instances. Please refer to the current *System Requirements* for information about the supported application servers and RDBMS.

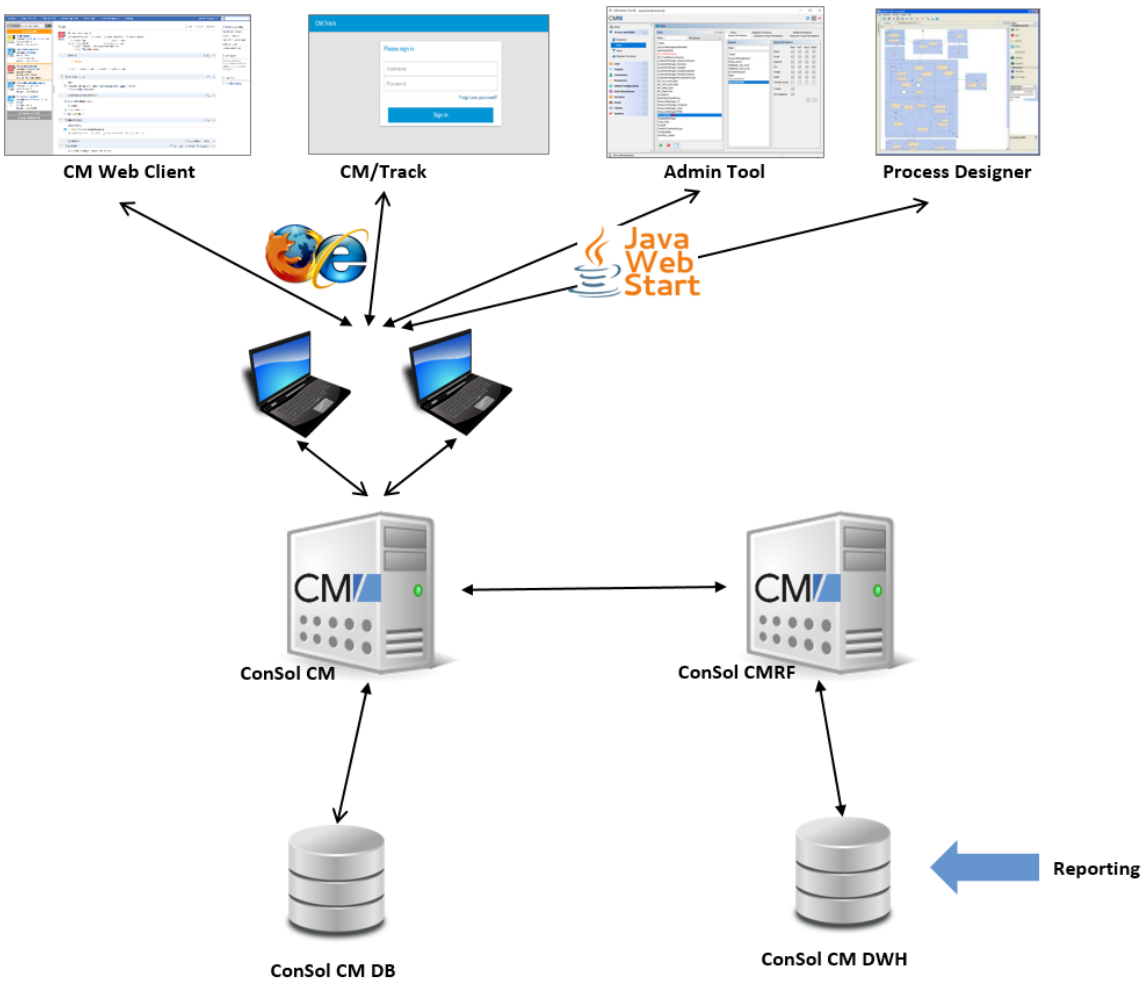


Figure 299: *ConSol CM - System architecture with DWH and CMRF (2 servers)*

There are two different synchronization modes for transferring data from ConSol CM to a DWH database:

- **LIVE mode**
  In this mode, every change that is submitted to the ConSol CM database is immediately synchronized with the DWH.
- **ADMIN mode**
  In this mode, the administrator has to trigger the synchronization manually.

⚠ Only data from ticket fields, customer fields, and resource fields with the annotation `reportable` = “true” will be synchronized with the DWH. In addition, all data which will be transferred as default will be transferred to the DWH. For a detailed explanation, please refer to the *ConSol CM DWH Manual*.

F.2.2 DWH Management Using the Admin Tool

To manage the DWH, use the navigation items in the navigation group *Data Warehouse*:

- Administration
- Tasks

F.2.2.1 Administration

For the administration of all DWH operations, open the navigation group *Data Warehouse*, navigation item *Administration*.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

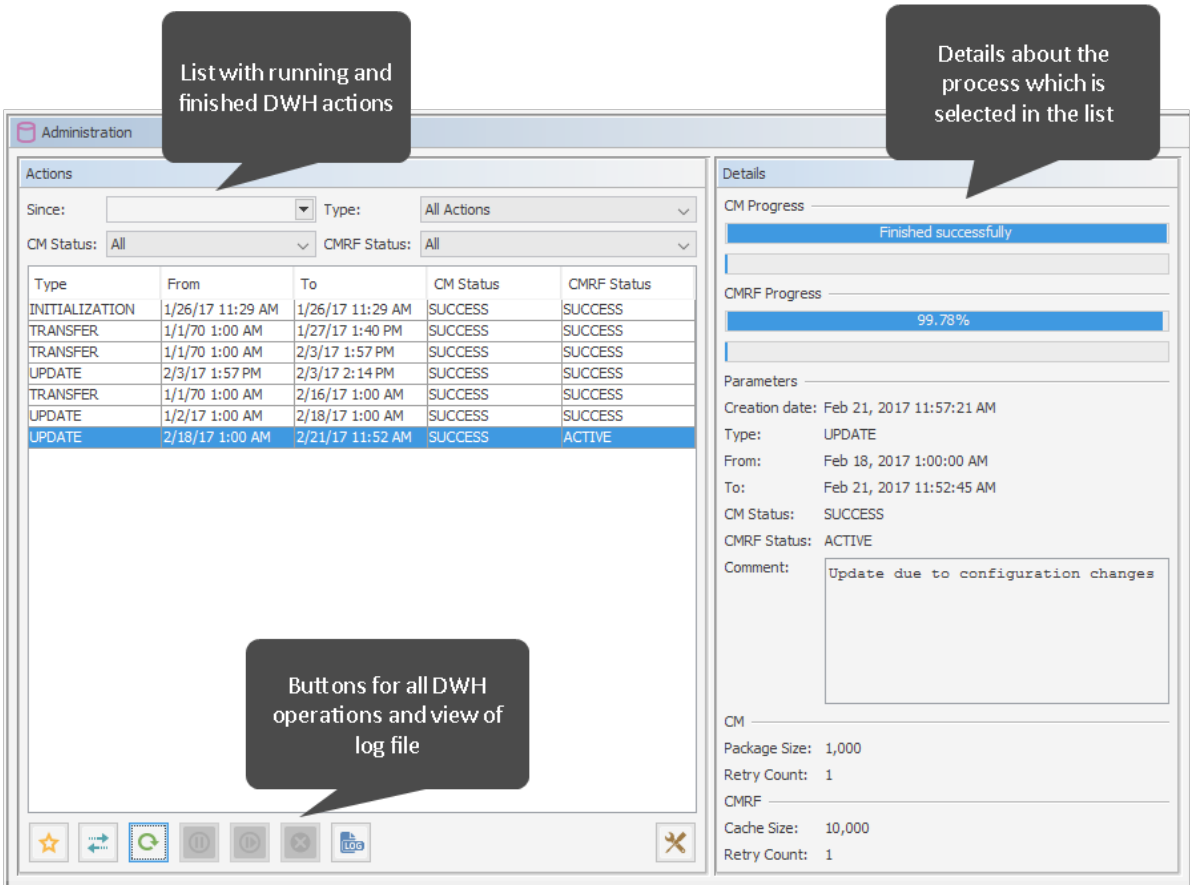


Figure 300: *ConSol CM Admin Tool - DWH, Tasks: Configuration and logs*

On the left hand side, the table *Actions* is displayed which lists all DWH operations which have been performed or which are still running.

You might want to filter the list by one or more of the filters above the list. The filter criteria will be combined by an AND.

- **Since**
  offers a date picker menu. Only the DWH actions which were started after the selected date will be displayed.
- **Type**
  only the DWH actions of the selected type will be displayed
  - **All actions**
    all available actions
  - **Already executed**
    Actions which are finished. The CM and CMRF status is SUCCESS or ERROR
  - **Currently executing**
    Actions which are currently running. The CM and/or CMRF status is ACTIVE.
  - **Planned to execute**
    Actions which have been configured and saved and entered into the list but which are not yet executed. The CM and/or CMRF status is NEW.

- **CM Status**
  only the DWH actions which have the selected CM status will be displayed.
  - All
  - NEW - first status after creation
  - ACTIVE - data is sent to CMRF
  - PAUSED - active state is stopped until the *Resume* button is clicked
  - SUCCESS - sending of data to CMRF has been finished successfully
  - ERROR - sending of data to CMRF has been finished unsuccessfully (see log files for information)
- **CMRF status**
  only the DWH actions which have the selected CMRF status will be displayed
  - ALL
  - NEW - first status after creation
  - ACTIVE - data is processed in CMRF
  - PAUSED - active state is stopped until the *Resume* button is clicked
  - SUCCESS - processing of data has been finished successfully
  - ERROR - processing of data has been finished unsuccessfully



A very elaborated explanation of the ConSol CM DWH transfer processes is provided in the *ConSol CM Operations Manual*, section *CMDB / CMRF/ Data Warehouse Synchronization Process*. In the following section in the current manual, only short information will be provided.

The list contains columns and values which are retrieved from the CM database table `cmas_dwh_synchronization`.

- **Type**
  - **INITIALIZATION**
    The DWH is new/empty. In the initialization step, the database structure is built.
  - **REINITIALIZATION**
    A new initialization of an already existing DWH
  - **TRANSFER**
    Initial data transfer after set-up. For a detailed explanation, please see section First DWH Synchronization (Transfer).
  - **UPDATE**
    Only new data is transferred to an already existing DWH. For a detailed explanation, please see section DWH Synchronization During System Operation (Update).
- **From**
  The start date of the DWH action
- **To**
  The end date of the DWH action

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **CM status**
  During a DWH action, data is transferred from the CM database to the DWH. This column indicates the status of the CM part of the action, i.e. the status of the transfer from CM to the DWH control tables. For an explanation of the status, please refer to the list above.
- **CMRF status**
  During a DWH action, data is transferred from the CM database to the DWH. This column indicates the status of the CMRF part of the action. For an explanation of the status, please refer to the list above.

Below the list, the following buttons provide all possible DWH actions:

- **Initialize**
  - case a) Create tables during DWH set-up. See Initialization of the DWH (Initialize).
  - case b) If the DWH already exists, you can here start a re-initialization. In order to do this, select the *Delete existing data* option. The database will the be rebuilt from scratch.
- **Transfer**
  Start initial data transfer after set-up. For a detailed explanation, please see section First DWH Synchronization (Transfer).
- **Update**
  Transfer new/additional data to the DWH. For a detailed explanation, please see section DWH Synchronization During System Operation (Update).
- **Pause**
  Pause a running DWH action
- **Resume**
  Continue a DWH action which has been paused
- **Delete**
  Deletes an unfinished data warehouse operation from the list, works for scheduled unstarted and previously paused operations. The most common case for this happens when queuing an initialize, transfer and update operation in a row without waiting.
- **Log**
  Open a panel which displays the `cmrf.log` file. This always provides the entries from the log file, the selection in the list does not have any influence.
- **Configuration**
  Opens the DWH configuration pop-up menu. See Basic DWH Configuration.



IMPORTANT BACKGROUND INFORMATION ABOUT DWH OPERATIONS

Please be aware of the ConSol CM/CMRF behavior with regards to operations for the DWH!

When you click one of the buttons *Initialize*, *Transfer* or *Update*, an operation with this type will be created as an entry in the ConSol CM database table `cmas_dwh_synchronization`. Each click on a button creates a new operation of the respective type. The operations are then executed one after another in the order of their creation, i.e., in FIFO (first-in-first-out) order! You can monitor all these actions in the *Administration* panel.

On the right hand side of the *Administration* panel, the *Details* section is displayed which lists all details of the action which is selected in the list.

- **CM Progress**
  Progress bars which indicate the progress of the transfer of CM data to the DWH control tables:
  - upper row: the overall progress for the operation on the sending side
  - lower row: the progress for the current sub-operation
- **CMRF progress**
  Progress bars which indicate the progress of CMRF updating the DWH database.
  - upper row: the overall progress for the operation on the receiving side
  - lower row: the progress for the current sub-operation
- **Parameters**
  All parameters of the selected DWH action:
  - **Creation date**
    Set automatically. Timestamp of the creation of the DWH action
  - **Type**
    DWH action type (INITIALIZATION |UPDATE etc.), see list above.
  - **From**
    The start date of the DWH transfer action. Might differ from the creation date when an action has been planned for a time in the future.
  - **To**
    The end date of the DWH action. Only set for finished actions.
  - **CM status**
    see above
  - **CMRF status**
    see above
  - **Comment**
    The comment which has been set when the DWH action was defined. Might also be empty.
  - **CM**
    - **Package size**
      Number of transferred objects per transaction from CM to the DWH control tables
    - **Retry count**
      Number of retries which are performed if an error occurs. After this number of retries has been reached, the operation is finished with the CM status ERROR.
  - **CMRF**
    - **Package size**
      Number of transferred objects per transaction from the DWH control tables to the DWH tables
    - **Retry count**
      Number of retries which are performed if an error occurs. After this number of retries has been reached, the operation is finished with the CMRF status ERROR.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Basic DWH Configuration

Before you can set up a ConSol CM DWH you have to prepare a database (or database schema) which will contain the DWH data. The respective database server has to be available for the CMRF server. For a detailed description of those topics, please refer to the *ConSol CM Setup Manual*. Once the database (or database schema) for the DWH has been prepared and the CMRF is up and running, you can continue with the following steps.

In order to configure the DWH synchronization mode and the DWH-relevant notifications, open the navigation group *Data Warehouse*, navigation item *Administration*. Click on the *Configuration* button and enter all required values.

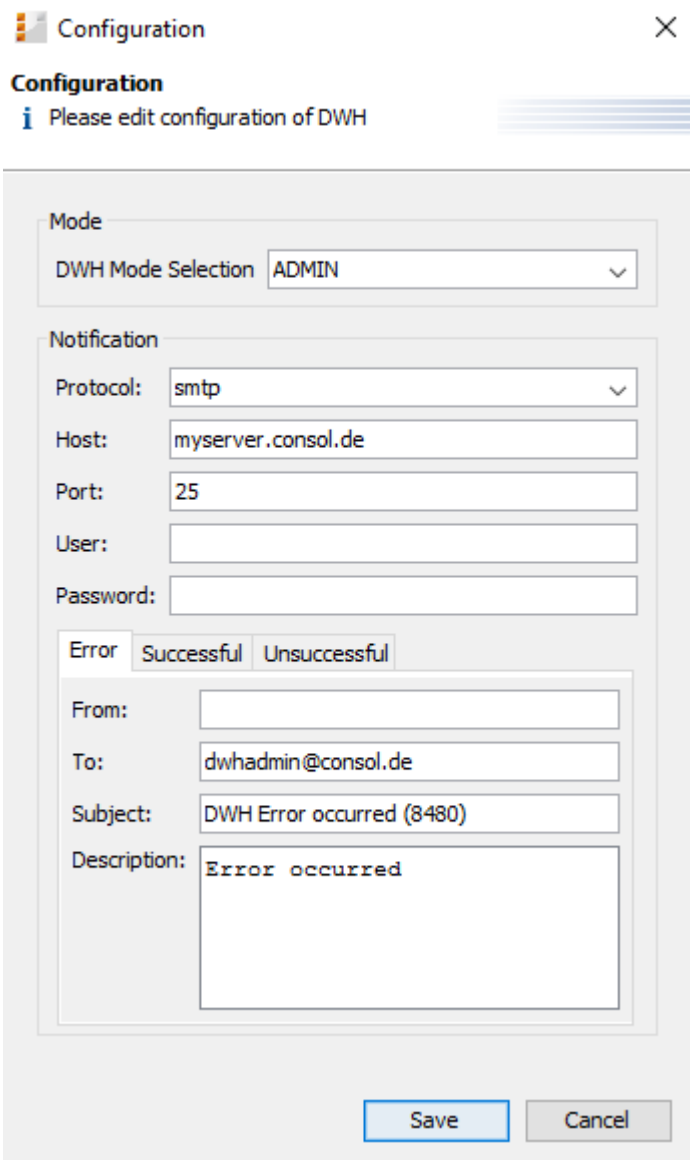


Figure 301: *ConSol CM Admin Tool - Data Warehouse, Administration: DWH configuration*

- **DWH Mode Selection**:
  - **LIVE**
    In this mode, every change that has been submitted to the ConSol CM database is immediately synchronized to the DWH.
  - **ADMIN**

In this mode, the administrator has to trigger the synchronization manually.

- **OFF**
  No data is transferred to the DWH.

You can also see the current DWH mode by looking at the corresponding DWH system property cmas-dwh-server, dwh.mode(see System Properties).

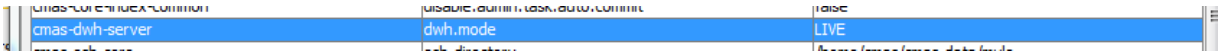


Figure 302: *ConSol CM Admin Tool - System property for DWH mode*

- **Notification**
  Here, the parameters for the emails which are sent due to DWH events, are configured:
  - **Protocol**
    Required - The protocol that is used to send the message. This is usually SMTP.
  - **Host**
    Required - The email server. You can enter a name (DNS-resolvable) or an IP address.
  - **Port**
    Required - The port on the email server where the mail daemon is listening.
  - **User**
    Optional - User name, if user authentication is required by the email server.
  - **Password**
    Optional - Password of the email user if user authentication is required by the email server.
  - Tabs **Error/Successful/Unsuccessful**
    Here the email parameters for emails that are sent by the system regarding DWH operations can be configured. There are three types of messages: in case of an error, in case of a successful operation, and in case of an unsuccessful operation.
    - **From**
      The From email address for messages (this may differ from the From address used for emails to customers and to engineers).
    - **To**
      The email address of the recipient of the DWH messages. Initially this will be the ConSol CM administrator's email address (CM system property cmas-core-security, admin.email).
    - **Subject**
      The (email) subject of the error/success/unsuccessful message.
    - **Description**
      The body (text) of the message.

Initialization of the DWH (Initialize)

When the basic configuration has been performed, the DWH initialization can be started. Click the *Initialize* button. A new DWH action of type INITIALIZATION will be entered in the list.

During this step, the database structure in the DWH is created with all tables and relations. No data will be transferred yet.

If the DWH has been in operation and has to be set-up a second time, a reinitialization has to be performed. Check the *Delete existing data* option in order to delete the old database structure and create a new one.

First DWH Synchronization (Transfer)

To fill the data warehouse with the ConSol CM data for the first time, click *Transfer*. The initial transfer is started. Depending on the number of tables, this might take some time (even several hours). You can follow the log entries by opening the `cmrf.log` file using the *Log* button.

For each Transfer operation, several parameters can be set, see section Parameters for Transfer and Update Operations.

> ⚠ The action *Transfer* should only be used directly after initialization, as this deletes all existing data from the standard tables and deletes all custom tables! For large databases this (transfer) can lead to problems concerning the available space on the database server due to a very large transaction log volume.

DWH Synchronization During System Operation (Update)

For each Update operation, several parameters can be set, see next section (Parameters for Transfer and Update Operations).

System running in ADMIN mode

If the DWH is running in ADMIN mode, the DWH administrator has to start the update manually by clicking *Update*. When the *Update* button is clicked, a new task of type “Update ”will be created for the operation. This task will appear in the list in the *Tasks* panel (navigation item *Tasks*). Then all data that are supposed to be transferred, i.e., data from fields with the `reportable` = “true” annotation that have been added or changed since the end date of the last Transfer or Update action, are transferred.

If a ticket field, customer field or resource field did not have the `reportable` annotation at the time of the last transfer and has it now, the corresponding content of the field from all tickets, customers and resources is transferred.

System running in LIVE mode

In case the system is running in DWH live mode, the task of type “Update ”is created and managed as mentioned for the ADMIN mode. Additionally, the LIVE mode will be suspended for the time of the update and will be reactivated automatically when the update is finished.

Notes for ADMIN and LIVE mode

> ⚠ Do not remove the annotation `reportable` = “true” for any field without being absolutely sure that the data is not required in reports any longer! If you remove a field that is used in reports and/or data cubes, the reporting will fail at run-time!

The DWH Update can also be started via command line or script. In this case you have to use a tool which can access the ConSol CM MBeans via command line, e.g. ***Twiddle*** for JBoss. The MBean to use is `consol.cmas.global.dwh.synchronizationService`. The command (method) is `update`.

The following command line shows an example command with Twiddle. Twiddle as 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
```

Please note that the CM package size, the CMRF cache size, and the retry counts cannot be set using the MBean. The default values will be used.

Parameters for Transfer and Update Operations

For each *Transfer* or *Update* operation, several parameters can be set in the pop-up menu which is opened when the *Transfer* or the *Update* button has been clicked.

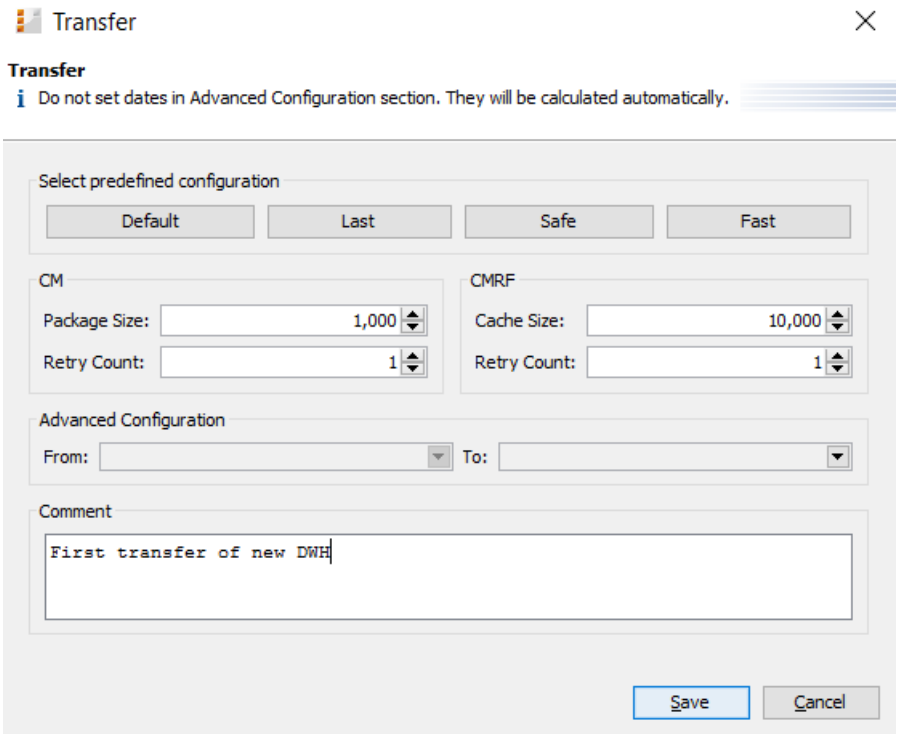


Figure 303: *ConSol CM Admin Tool - Data Warehouse, Administration: Configuration of a transfer operation*

Select predefined configuration for the individual job to start:

- **Default**
  run the job with the default settings, just like without any modification in the dialog.
- **Last**
  run the job with the same configuration like the previous run.

- **Safe**
  run the job with cautious settings, which may cause it to run longer, however, the settings ensure that no `OutOfMemory` exception will occur.
- **Fast**
  run the job with settings optimized for fast completion, however, `OutOfMemory` exceptions can occur, if the Java Virtual Machine is not well-tuned.

When you click one of the buttons, the parameters for the other fields (e.g. *package size*) are loaded into the fields from the respective configuration. No operation is started yet. You can further modify the values for the planned operation.

The following fields are available:

- **CM** (sending side)
  - **Package Size**
    Number of objects sent in one DWH message. Higher values mean better transfer performance and bigger memory usage. If the system has enough RAM, the value can be 1000 or even more. (In CM versions lower than 6.11, this was covered by the CM system property cmas-dwh-server, batch-commit-interval, but this was a system-wide parameter. Starting with CM version 6.11, it is possible to set this for every DWH operation individually.)
  - **Retry Count**
    Number of retries until an ERROR is thrown.
- **CMRF** (receiving side)
  - Cache Size
  - **Retry Count**
    Number of retries until an ERROR is thrown.
- **Advanced Configuration**
  - From
  - To

> ⚠ **Please do not use the Advanced Configuration options unless clearly instructed by a ConSol representative to do so!**
>
> The parameters offered in the section *Advanced Configuration* for setting a date interval to be covered by the job should not be used without very clear understanding of the consequences. It could lead to an inconsistent data warehouse. They should be only used when specifically advised with detailed instructions by ConSol support or ConSol CM consulting.

- **Comment**
  Optional. A comment which characterizes the DWH action. Will be written into the database and displayed in the *Details* section.

F.2.2.2 DWH Tasks

If there are due or active DWH tasks, this will be indicated in the Admin Tool, see following figure. The navigation group *Data Warehouse* shows an exclamation mark so that even when this navigation group is closed you will know that there are active tasks. The number of active tasks is indicated at the navigation item *Tasks*.

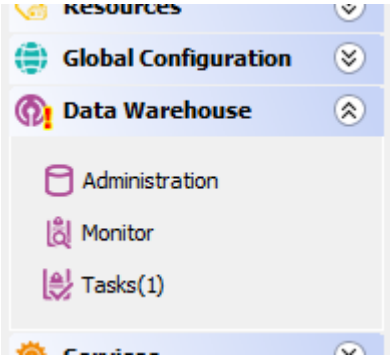


Figure 304: *Indication of active DWH tasks in the Admin Tool*

The list *Current tasks* contains columns and values which are retrieved from the CM database table `cmas_dwh_task`.

In the list of *Current tasks*, you will find entries (one entry per task) if ...

- the DWH is running in ADMIN mode and the administrator has started an update: all tasks that have to be performed are listed.
- the DWH is running in LIVE mode but the check box *Automatic commit of administrative changes* has not been checked.
- Ticket field, customer field or resource field annotations have been changed to `reportable` = “true” and the checkbox *Automatic commit of administrative changes* has not been checked.
- Ticket field, customer field or resource field annotations have been added and changed to `reportable` = “true” and the checkbox *Automatic commit of administrative changes* has not been checked.
- Ticket field, customer field or resource field annotations have been changed from `reportable` = “true” to “false” and the checkbox *Automatic commit of administrative changes* has not been checked.

If the checkbox *Automatic commit of administrative changes* has not been checked, the tasks will remain in the status "ready for execution".

You can mark one or more tasks in the list and execute them manually (*Run tasks* button).

If the checkbox *Automatic commit of administrative changes* has been checked, the tasks will be run automatically by the system. The execution can take several minutes.

There are some special cases to consider:

- Removing the annotation `reportable` = “true” or `reportable group` = “true” does not create a DWH task. These column and the respective DWH tables are only removed by the next DWH update.
- In ADMIN mode, a task for changing `reportable` to “false” is ignored! A task is created and is removed instantly. The DWH table is not changed! This change is performed by the next DWH update.
- In LIVE mode, the task is executed immediately, even when the checkbox *Automatic commit of administrative changes* is not checked.
- With the DWH update action the table structure is extended by the newly annotated field from the open tasks, but the data is only initially filled in during the execution of the respective DWH task!

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.2.2.3 DWH Monitor

To open the DWH Monitor, use the navigation group *Data Warehouse*, navigation item *Monitor*.

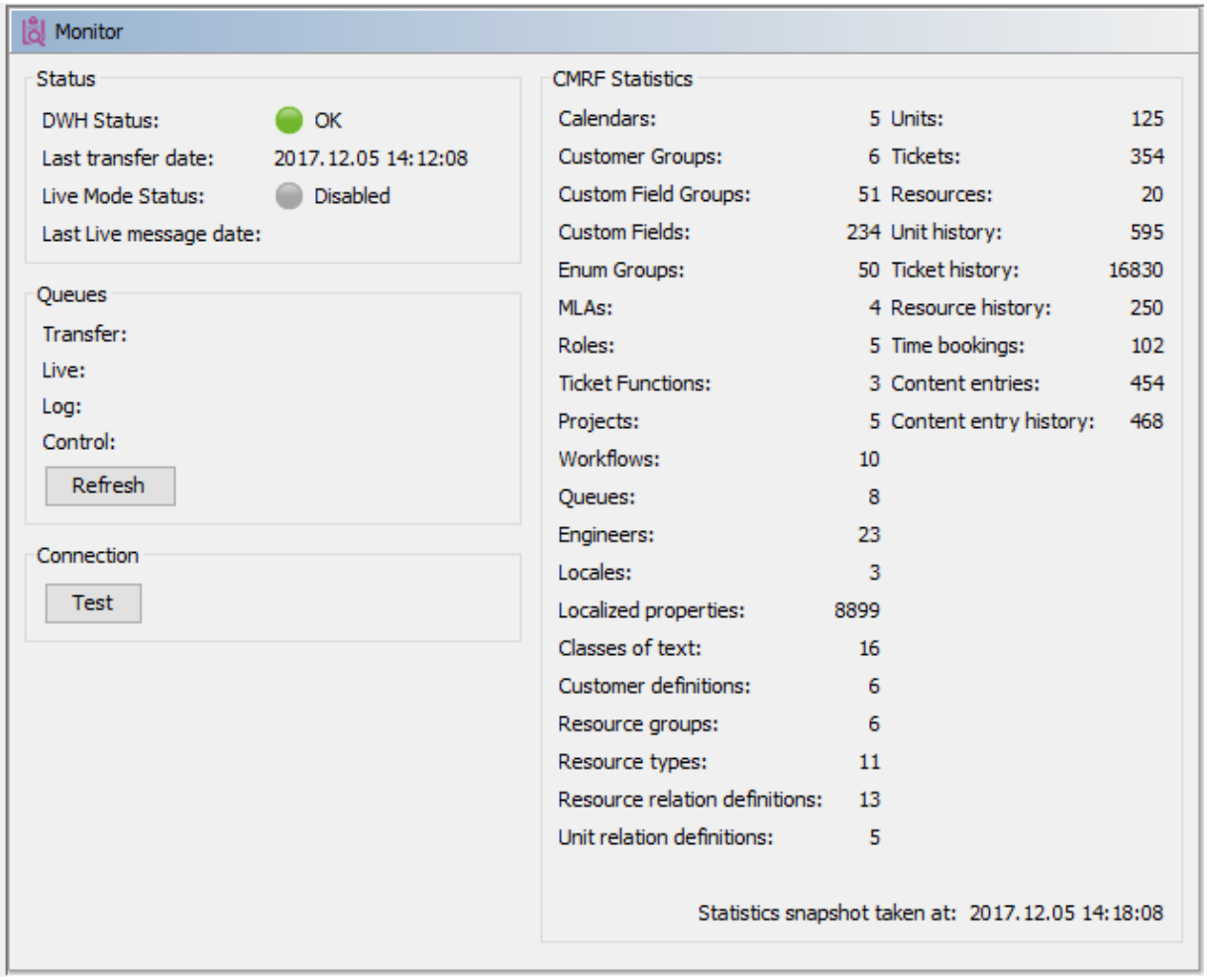


Figure 305: *Consol CM Admin Tool - Data Warehouse, Monitor: DWH Monitor*

The DWH monitor provides a quick overview of all DWH parameters. The date of the current snapshot is indicated at the bottom right of the screen.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

The following parameters are displayed:

- **Status**
  A quick overview of the most important parameters.
  - **DWH status**

Color of status symbol	DWH status message
Green	OK
Yellow	Uninitialized
	Action is processed
	Action is paused
	Action is planned
Red	Error
Grey	Disabled

  - **Last transfer date**
    Date of the last transfer or update operation
  - **Live mode status**

Color of status symbol	Live mode status message
Green	OK
Yellow	Uninitialized
	Action is processed
	Action is paused
	Action is planned
	Started (update is needed)
Red	Error
	Error (update is needed)
Grey	Disabled

- **Queues**
  Informs about the number of message entries in the internal data warehouse message queue tables. Use the *Refresh* button to update the values.
  - Transfer: Number of messages in the data warehouse transfer table (INT_TRANSFER_QUEUE)
  - Live: Number of messages in the LIVE messages table (INT_LIVE_QUEUE)

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

  - Log: Number of messages in the log messages table (INT_LOG_QUEUE)
  - Control: Number of messages in the control messages table (INT_CONTROL_QUEUE)
- **Connection**
  Test the connection to the DWH database using the *Test* button.
- **CMRF statistics**
  Provides a list of the number of objects which have been transferred.

F.2.2.4 DWH Troubleshooting and Repair

If any errors have occurred during initialization, transfer, or update, the log entries are displayed in the log panel which is opened with the *Log* button .

You can also check the original log file in `<JBOSS_HOME>/standalone/log/cmrf.log`.

Please note that these are the standard paths. They may be configured to use different paths in the file `cm6-cmrf.xml` file. A detailed description of CM logging and log files is provided in the *ConSol CM Setup Manual* and the *ConSol CM Operations Manual*.

Usually the log file and/or log panel entries give good hints regarding the initial reason for a transfer failure. If you run into a problem you cannot resolve and you have a maintenance contract with ConSol, please contact our support team.

> For more details on Troubleshooting and Repair see section *CM / CMRF / DWH Synchronization: FAQs and Tips for Troubleshooting* in *ConSol CM Operations Manual*

F.2.3 DWH-Related System Properties

A list of all system properties which are relevant for a specific DWH configuration can be found in section CMRF & DWH Configuration of the System Properties chapter.

F.2.4 Transfer Mode

All data which has been annotated as `reportable` = “true” has to be transferred to the Data Warehouse. This is performed by ConSol CM and the CMRF.

In ConSol CM versions 6.11. and up, there is one transfer mode:

- DIRECT mode

The mode is set using the system property cmas-dwh-server, communication.channel. Possible values are:

- “DIRECT”: database communication channel, only available value since version 6.11.

In DIRECT mode, the CM server sends messages to the CMRF by using direct access to database tables in the DWH database:

- INT_CONTROL_QUEUE
- INT_LIVE_QUEUE
- INT_TRANSFER_QUEUE

CM is able to do that because in DIRECT mode, the CMRF datasource is made accessible for ConSol CM as well, i.e.

- in overlay mode:
  ConSol CM and CMRF use the same application server

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- in standalone mode:
  the CMRF database configuration file is also located on (copied to) the ConSol CM application server

The CMRF server reads the entries from the tables and writes the data into the DWH.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.2.5 Expert DWH Information

F.2.5.1 Creating DWH Cubes

The ConSol CM API includes methods to create, update and delete cubes on the DWH database using task scripts. The created cube contains the following information:

- Basic ticket data (ID, name, subject, queue, scope, creation date, close date)
- Assigned engineer (first name, last name, login)
- Ticket fields which are annotated as `reportable`, except for lists and structs
- Main customer, this can be either a contact or a company (ID and customer group)
- Company of the main customer (ID)

Optionally, the cube can be configured to contain:

- Customer fields of the main customer which are annotated as `reportable`, except for lists and structs
- Customer fields of the main customer's company which are annotated as `reportable`, except for lists and structs



The IDs used in the cube are the IDs from CMRF, not the IDs which the objects have in ConSol CM.

The following settings can be made for the cube:

- **Name:**
  The name of the cube, i.e. the name of the view in the DWH database.
- **Queues:**
  One or several queues whose tickets should be included in the cube.
- **Language:**
  Language which should be used for the data fields and the enum values.
- **Customer fields:**
  Whether the customer data fields should be included in the cube.
- **Company fields:**
  Whether the data fields of the customer's company should be included in the cube.

The methods belong to the classes `DwhCubeService` and `DwhCube`. The following example shows a task script to create a cube.

```
def onInitialize(taskDescriptor) {}

def onExecute(taskDescriptor) {
   DwhCube cube = new DwhCube()
   cube.setViewName("cube_all")
   cube.setQueues(queueService.getAll())
   cube.setContactCustomFields(true)
   cube.setCompanyCustomFields(true)
   cube.setLocale(Locale.GERMAN)
   dwhCubeService.create(cube)
}

def onError(taskDescriptor) {}
def onCancel(taskDescriptor) {}
```

The cube must be updated using the `dwhCubeService.update(cube)` method if there are changes to `reportable` annotations or localized labels of data fields / enum values. Otherwise, there changes are not reflected in the cube.

F.2.5.2 Localization of Static DWH Tables

The static DWH table can include database columns for the localized values of some entities. This means that the localized descriptions of the entities can also be transferred to the DWH. The following example shows the fields in the Admin Tool and in the DWH for projects in ConSol CM.

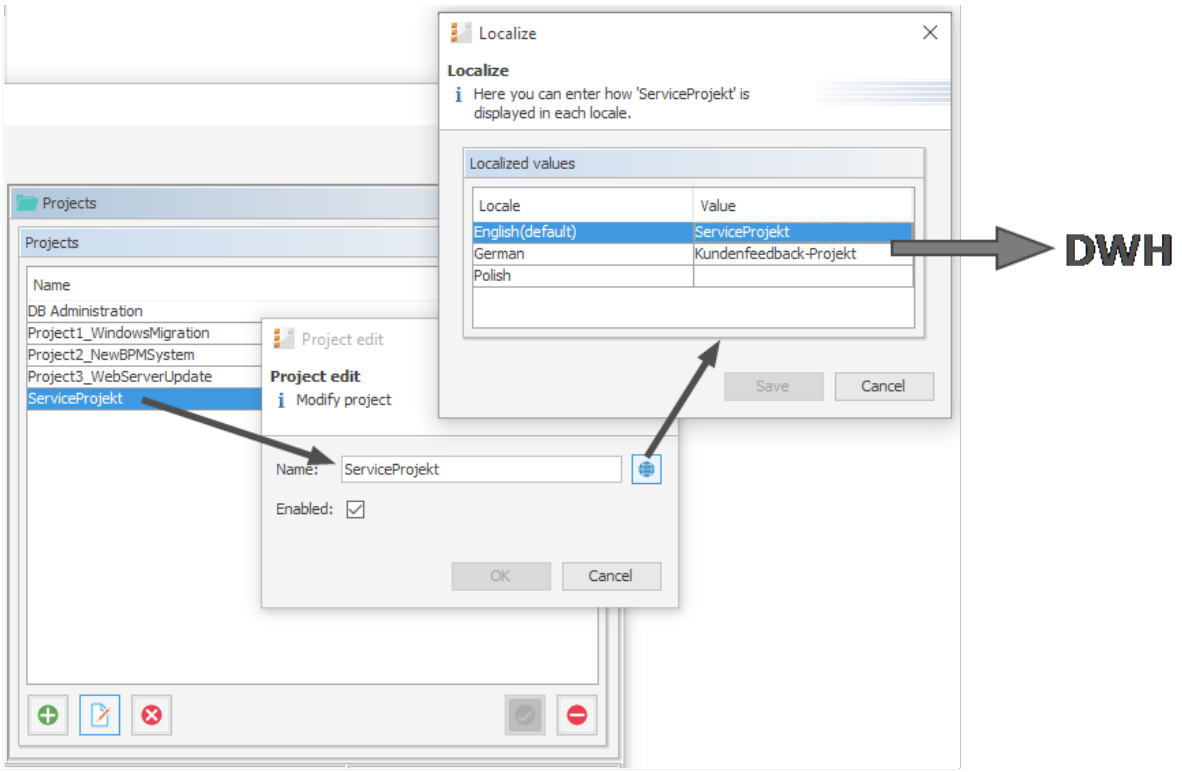


Figure 306: *ConSol CM Admin Tool - Global Configuration, Projects: Localized values of a project*

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

Figure 307: *DWH table with localized values*

By default, the transfer of the localized values is disabled, as it can significantly increase the duration of the DWH updates. You can enable the transfer of the localized values using the Java system property `cmrf.localization.enabled`.

Add the parameter `-Dcmrf.localization.enabled=true` to the CMRF start command.
A DWH update is required to fill the corresponding columns.

Example start command with localization enabled:

```
nohup $JBOSS_HOME/bin/standalone.sh --server-config=cm6-cmrf.xml -b=0.0.0.0 -
 Dcmrf.localization.enabled=true
```

⚠ Setting `cmrf.localization.enabled` to `true` can increase the runtime of DWH updates significantly.

For a detailed explanation, please see also the *ConSol CM DWH Manual*.

F.2.5.3 MBeans for DWH Management

The MBean `dwh.admin.service`, in `consol.cmas.admin.global`, provides access to the DWH mode (OFF |ADMIN | LIVE) . The two methods `getMode()` and `setMode()` are available.

The MBean can be accessed using graphic tools, e.g., JConsole for JBoss.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.3 CM Services

CM Services are managed on the navigation item *CM Services* in the navigation group *Services*. This navigation item contains the list of the available CM services. Stopped or disabled services are displayed in italics. You can start a service by clicking the *Start service* button (green triangle) and stop a service by clicking the *Stop service* button (red square) below the list of services.

> ⚠ The status of a service should only be changed by an experienced ConSol CM consultant or by a member of the ConSol CM support team! ConSol CM core functionalities might not work when a service is not running!

List of services:

- **Archive GDPR service**
  Sends information about anonymized and deleted customers to CM/Archive if the system property cmas-archive-core-server, archive.enabled is set to “true”.
- **DWH live service**
  Controls just-in-time DWH update in LIVE mode.
- **DWH log service**
  Reads and processes CMRF/DWH log messages for the Admin Tool and stores them in the ConSol CM database. The entries are used for the log protocol in the Admin Tool. See section Data Warehouse (DWH) Management.
- **DWH transfer service**
  Controls DWH transfers.
- **Index changes notifier**
  Collects changes which concern the index and saves them to the database.
- **Index changes receiver**
  Reads the database table where the pending index updates are stored and starts the update of the index.
- **Job Executor**
  Controls escalations for processes resp. workflows.
- **Kerberos v5 authentication provider**
  Required if Kerberos authentication is enabled.
- **NIMH**
  Retrieves incoming emails when the server is running in NIMH mode.
- **recent items cleanup**
  Removes recent items according to the configuration in the system properties cmas-core-server, recent.items.cleanup.interval.minutes and cmas-core-server, recent.items.max.per.engineer. If ConSol CM runs in a cluster, the service is only enabled on the cluster node specified in cmas-core-server, recent.items.cleanup.cluster.node.id.
- **Remote client pooling**
  Controls that Web Clients get changes from the Admin Tool.
- **Rest API service**
  Activates or deactivates the REST (*Representational State Transfer*) interface.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Server Session Service**
  Checks sessions and stops sessions when the lifetime of a client or Admin Tool session has expired. See, for example system, properties admin.tool.session.check.interval and server.session.timeout.
- **TaskExecutorService**
  The engine for task execution in the Task Execution Framework. It comprises a main processing thread (with watchdog attached) which scans the database for tasks with the status *NEW*, and a second component which controls a dedicated thread pool used for tasks execution.
- **Unused content remover**
  Removes attachments and comments which have been marked as *deleted* in the Web Client (in the protocol section of a ticket).

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.4 Search in ConSol CM

ConSol CM offers a powerful search engine.

Learn how to configure ConSol CM for your search requirements in section Search Configuration. Section Index contains information about the index, which is needed for search.

Starting with ConSol CM version 6.10.1, it is possible to perform actions on search results. This feature is part of the ConSol CM Action Framework and is described in the Action Framework - Search Actions section.

Result sets retrieved via the Web Client can be exported to a CSV list. This feature was added in ConSol CM version 6.10.1 and is explained in section CSV Export of Search Results.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.4.1 Search Configuration

ConSol CM provides a powerful search mechanism for all objects involved in the business processes (customers, tickets and resources). Technically, the search is based on the **Indexer**, a module of ConSol CM.

The following paragraphs explain the entire topic *Search in ConSol CM* from an administrative point of view. Please refer to the *ConSol CM User Manual* for a detailed explanation about how to use the search as an engineer.

F.4.1.1 Search Modes

A ConSol CM engineer can use several search modes:

Quick Search

This is performed using the Quick Search field in the upper right-hand corner of the Web Client. The display of the results (i.e., the fields and the order of the fields in the result list) can be formatted using templates, as detailed in sections Templates for Customer Data and CM/Resource Pool - Templates for Resource Data. Please keep in mind that you can adapt the length of the result set using the system property cmweb-server-adapter, globalSearchResultSizeLimit. The result set will also be influenced by the Page Customization attribute *appendWildcardAutomatically*, see section Page Customization, appendWildcardAutomatically for details.

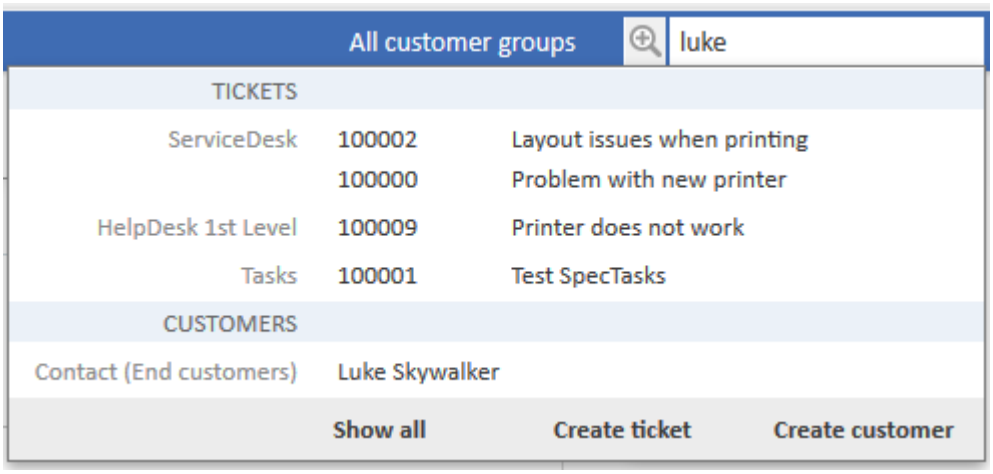


Figure 308: *ConSol CM Web Client - Quick Search*

Phonetic Search

For the Quick Search, a phonetic search can be configured for String fields. Please see section Configuring the Phonetic Search for the explanation of how to prepare your ConSol CM system for this feature.

With a phonetic search, the engineer cannot only find exact matches in the search results but also results which sound similar even if the spelling differs from the entered search term. The implementation is based on the Apache Commons Codec libraries.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

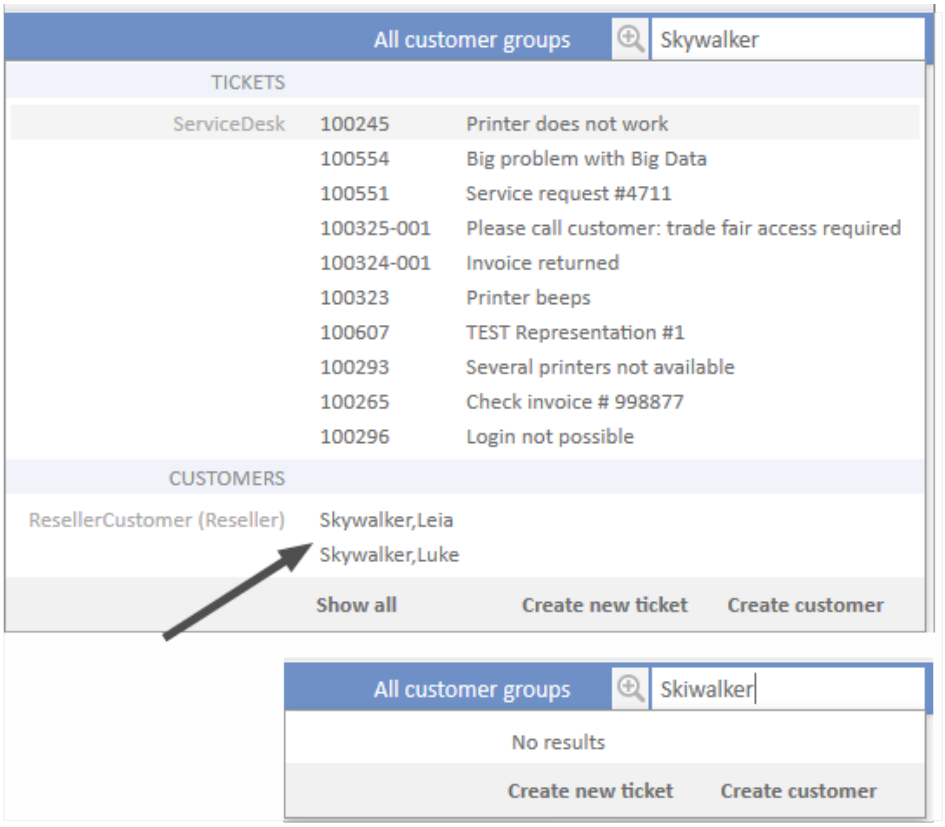


Figure 309: *ConSol CM Web Client - Search results without phonetic search*

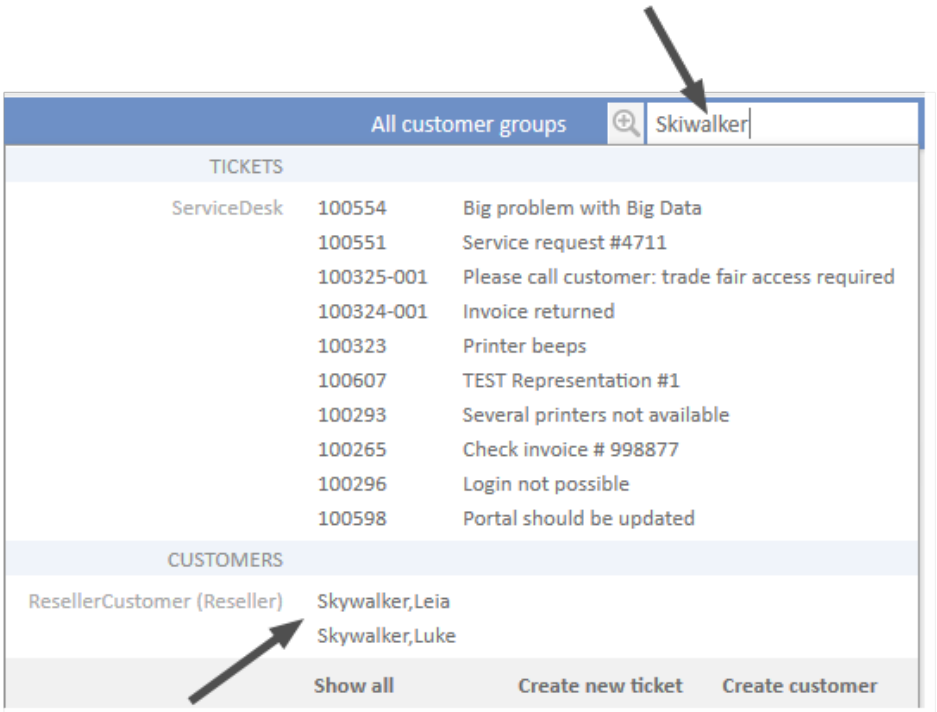


Figure 310: *ConSol CM Web Client - Search results with phonetic search*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Detailed Search

This is performed using the *Detailed Search* page. To open this page, click on the magnifier icon next to the Quick Search entry field.

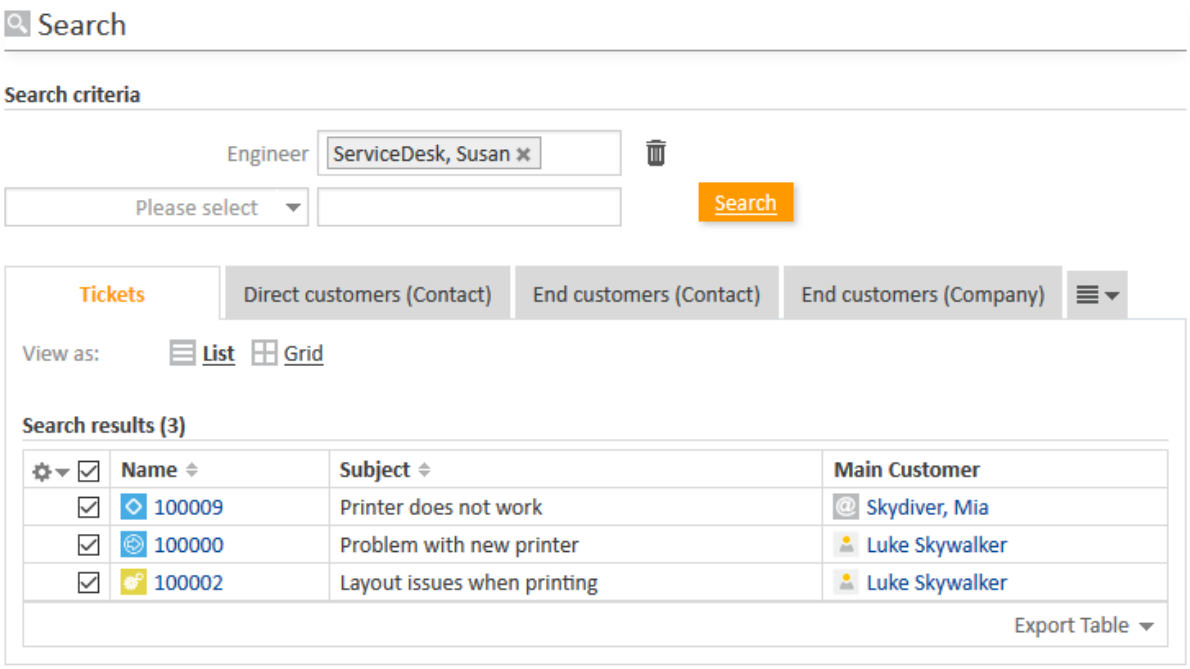


Figure 311: *ConSol CM Web Client - Detailed Search*

Please keep in mind that the result list length and paging of the result list for the Detailed Search can be configured using the system properties cmweb-server-adapter, searchPageSize and cmweb-server-adapter, searchPageSizeOptions.

For the engineer search, you, as an administrator, can define engineer names which should not be offered in the drop-down menu in the search criteria. This is configured using page customization. Please see section excludedUserNames for details.

Please refer to the *ConSol CM User Manual*, section *Searching for Tickets, Customers and Resources* to learn how to use the search functionality.

Important note about search with fields of lists of structs

Please note that in ConSol CM versions 6.10.6.x and older, two fields which are part of the same list of structs (i.e. of a table) are combined by OR in the Detail Search. In CM versions 6.10.7.0 and up, the two fields are combined by AND. This will lead to different search results in the different CM versions!

Please refer to the *ConSol CM User Manual* for a detailed example.

Configuring the Search Result List

You, as an administrator, can configure the layout of the search result list using the annotation `order-in-result`. This annotation influences the columns of object-specific search result sets. For example, the customer field group *ResellerCompanyData*, belonging to the customer data model *Reseller-Model*, contains the following customer fields:

- *company_name*: order-in-result = 1
- *company_number*: order-in-result = 2

Thus, in a search result list, e.g., a Detailed Search which shows a result set with companies of a customer group which uses the ResellerModel will contain the two columns *company_name* and *company_number*.

For a field (i.e., column in the result table) which should not be displayed in the Web Client (i.e., in the field selector for the result list and as column in the table) at all, set the value of the `order-in-result` annotation to "0" (zero).

The following figure shows the `order-in-result` annotation of a customer field. You reach this screen by opening the navigation item *Data Models* in the navigation group *Customers*.

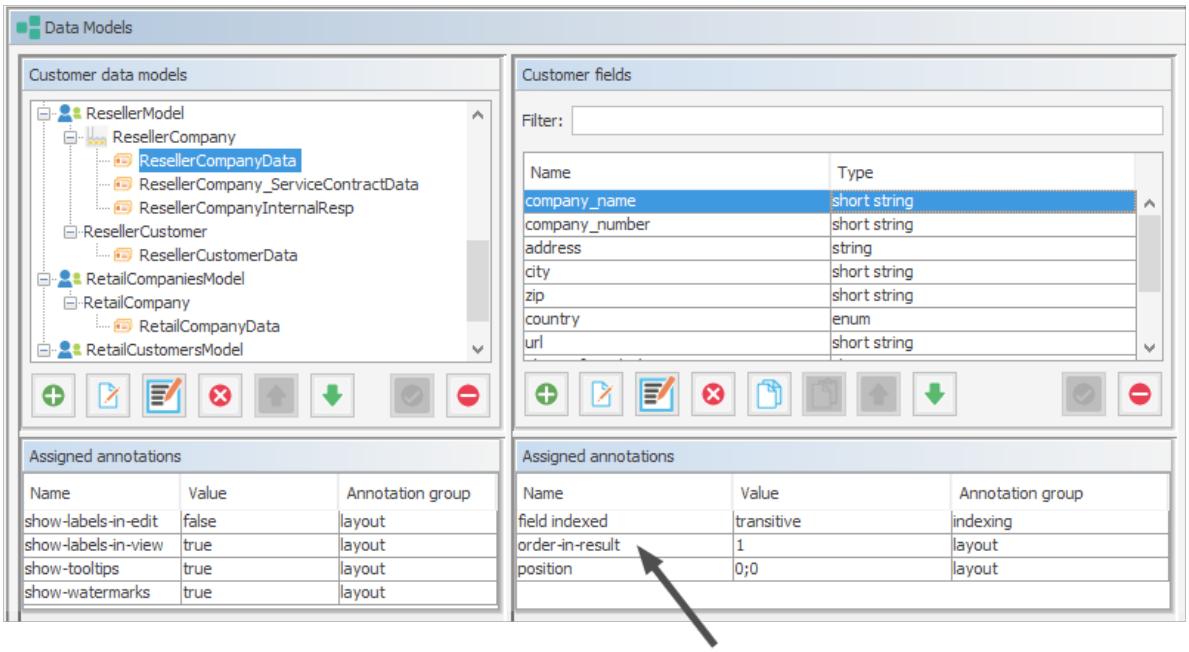


Figure 312: *ConSol CM Admin Tool - Customers, Data Models: Setting the annotation order-in-result for a customer field*

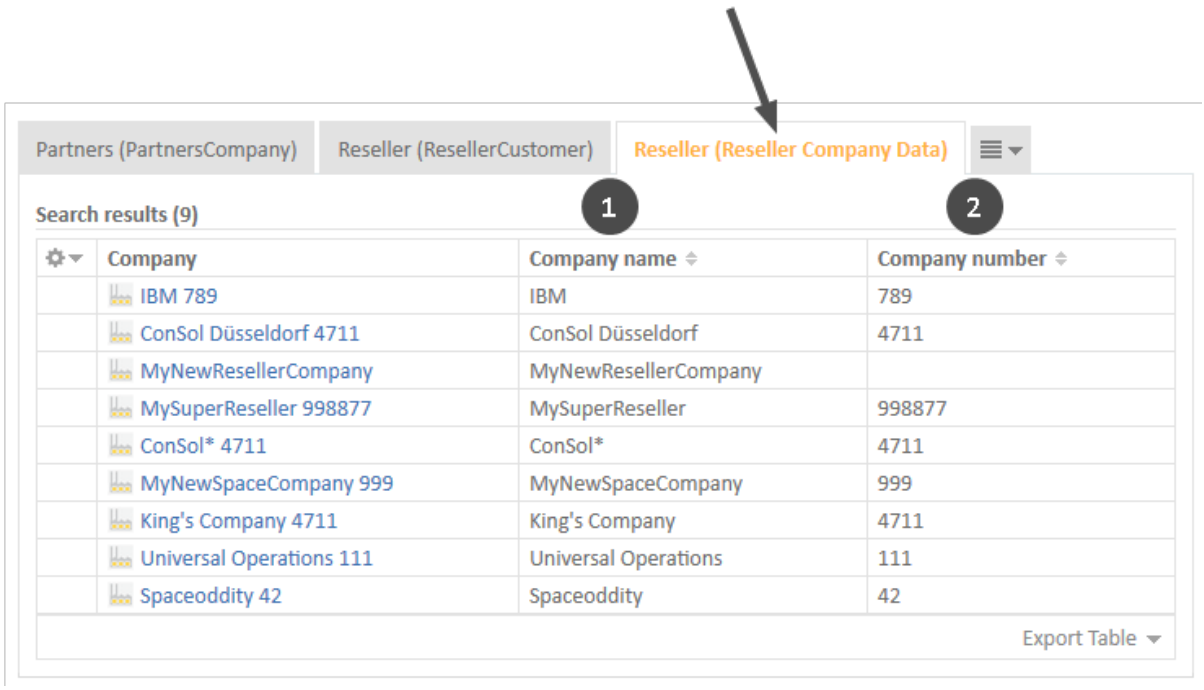


Figure 313: *ConSol CM Web Client - Search result set with the two annotated columns (order-in-result)*

Notes about the Order of Search Result Columns

Since the annotation `order-in-result` is not the only parameter which influences the order of columns in search results (tables), please find here a detailed explanation.

In the following section, *user preferences* means the layout of a search result table which the engineer has configured manually, e.g., the columns which should be displayed, the order of those columns or the sorting.

Rules:

1. The columns of a search result table can display different fields for each of the CM object types:
   a. for tickets: ticket fields as well as internal fields like queue, creation date, engineers
   b. for customers (contacts or companies): customer fields as well as internal fields like object name of a customer
   c. for resources: resource fields as well as internal fields like object name of a resource
2. The `order-in-result` annotation organizes only the order of the ticket fields, customer fields, and resource fields, not of internal fields.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

3. The internal field types have the following visibility and order by default:
    a. for ticket lists:
        i. Engineer
        ii. Main customer
        iii. Ticket name
        iv. Ticket subject
    b. for customer (contact/company) lists:
        i. object (i.e. contact or company) name
    c. for resources lists:
        i. object (=resource) name
4. The data fields (ticket field, customer fields, and resource fields) are primarily ordered by the user preferences. Those will overwrite any other configuration.
   When, e.g., as default configuration, a ticket field has the annotation `order-in-result = 5` and is visible, it will (case a) not be displayed if the engineer has deselected the column, and (case b) it will be displayed as column #2 if the engineer has placed it there.
5. If no order is defined by the user preferences, the order is the ascending order of numbers from the data field annotation values `order-in-result`. Those numbers are evaluated globally across all data fields within all ticket field groups, all customer field groups, and all resource field groups. Identical values of the annotation `order-in-result` of different data fields are theoretically possible.
   For an exactly defined order, use some convention for an entire CM installation and apply unique numbers, e.g., use high four digit values.
6. If there are still two fields with identical values for `order-in-result` which should both be displayed, the data fields are alphabetically ordered based on their localized names. The locale of the engineer's browser is used.

Configuring Detail Search Behavior for Enum and MLA Values

The values within an enum (a sorted list) or an MLA might change over time, e.g. a new software version is added to a list, an old one should no longer be available. In this case, the deprecated/old value will be deactivated. This means that there might be tickets which contain enum or MLA values which are deactivated in the current system. Of course it is indispensable that engineers can still search for tickets which contain this value. Therefore deactivated enum or MLA values are offered in the Detailed Search as shown in the following figure for the deactivated enum value *Resource Pool*.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

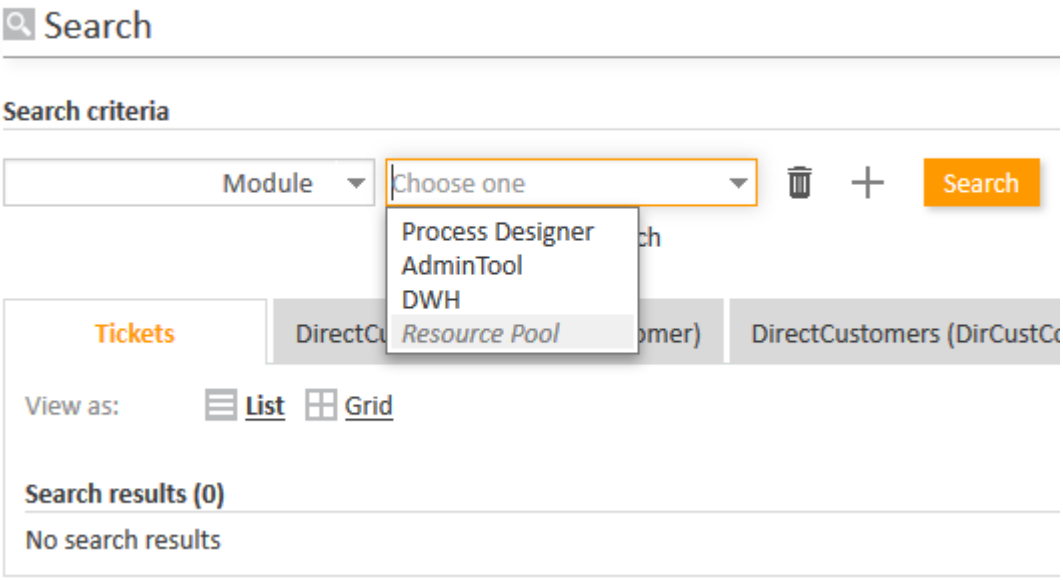


Figure 314: *ConSol CM Web Client - Detailed Search for modules, one enum value deactivated*

The ConSol CM behavior concerning the Detailed Search of deactivated enum and MLA values can be configured using a Page Customization attribute: enableSearchForDeactivatedEnums .

Autocomplete Search (Search by Using Intelligent Fields)

The Autocomplete Search is performed implicitly when you start entering a word in an Autocomplete field, e.g., in company data or customer data when you create a ticket (see figures below).

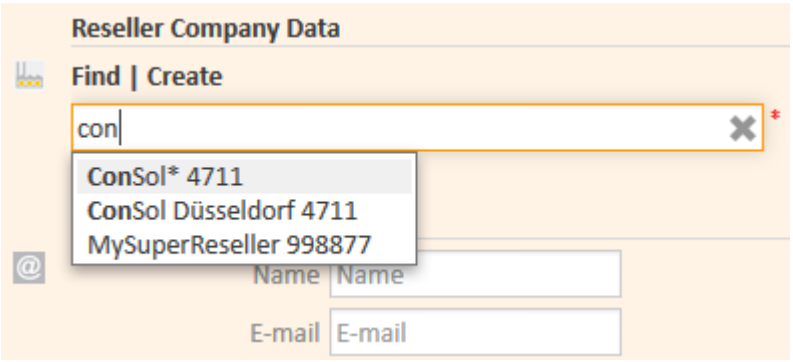


Figure 315: *ConSol CM Web Client - Autocomplete Search*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

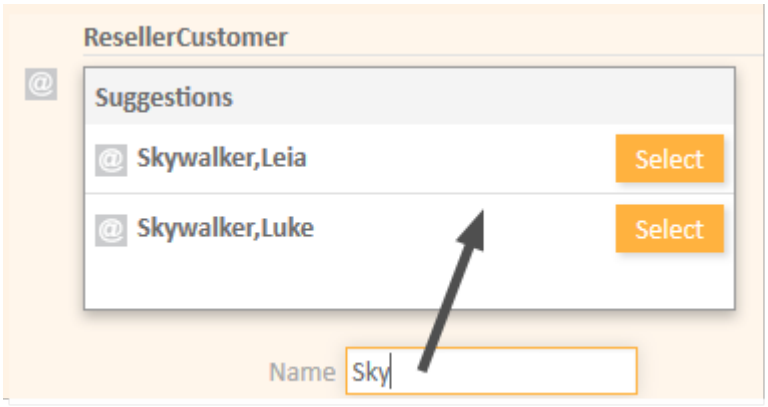


Figure 316: *ConSol CM Web Client - Suggestions for an Autocomplete Search*

Starting with ConSol CM version 6.10.2, a search in all contacts of the ConSol CM database is also performed in the Ticket Email Editor when an engineer starts typing in the To, Cc or Bcc field:

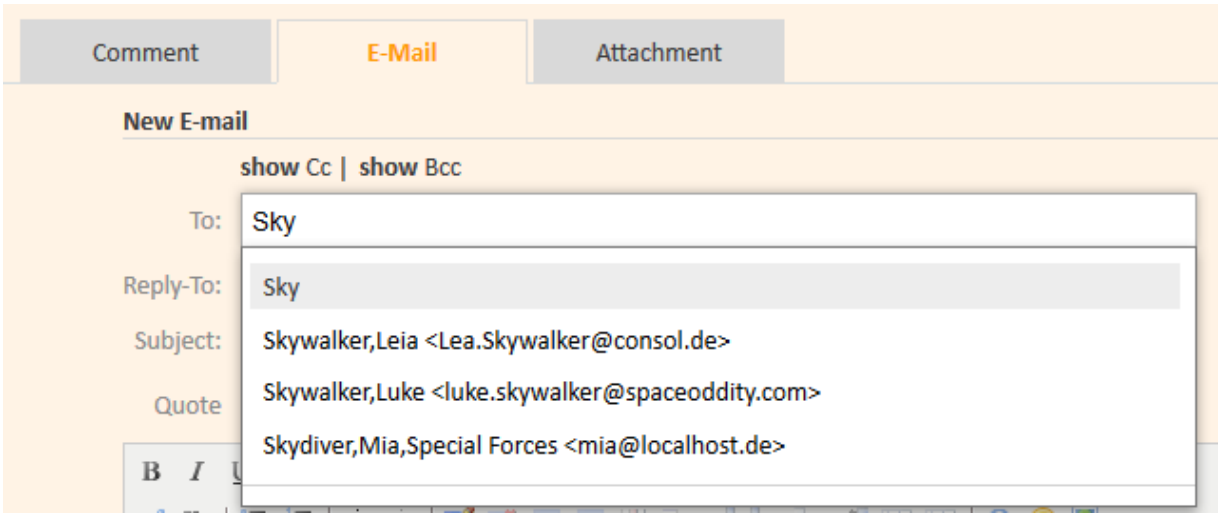


Figure 317: *ConSol CM Web Client - Ticket Email Editor: Automatic search for contacts as email receivers*

The number of characters which must be typed before automatic search is activated can be configured using the page customization attribute *minMailInputLength* (Integer, Default 1) in the scope *mailTemplate*.

F.4.1.2 Fields Which Can Be Searched

For ticket fields, customer fields and resource fields which should be searched, the annotation `field indexed` has to be set. See section Search Configuration. This makes the field available for the Quick Search and for the Detailed Search.

Two types of fields are processed by the search engine:

1. data fields and content which are/is indexed by default
2. data fields which are annotated using the annotation `field indexed`

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Data Fields and Content Which Are Indexed by Default

- Engineer data
  - Email
  - First name
  - Last name
  - ID
- Ticket data
  - Attachment content (this is controlled by the CM system property cmas-core-index-common, index.attachment which is “true” per default)
  - Creation date
  - Engineer
  - History
  - ID (technically the ticket name)
  - Queue
  - Referenced engineers
  - Subject
  - View

Data Fields Which Are Annotated Using the Annotation field indexed

For all data fields which have been added and configured by the ConSol CM administrators, the annotation `field indexed` has to be set to make the fields searchable. A field which is indexed is available in

- the Quick Search
- the Detailed Search
- all searches in workflow and Admin Tool scripts which use, for example, criteria classes/objects (e.g., `TicketCriteria`, `UnitCriteria`, `ResourceCriteria`). This includes
  - scripts which have been manually added by the administrator
  - scripts which are present by default (e.g., the script `createTicket.groovy`. In this script, the correct contact for a new ticket can only be set if the customer field which is used to retrieve the correct contact or company is indexed)



Please note that if values of a table, i.e. a list of structs, should be available for the search, all elements of the data structure have to be annotated with `field-indexed` = “true”. This means that the list, the struct and every single data field within the struct have to be annotated that way!

You, as an administrator, can define the system's behavior with regards to search results. Depending on the values of the `field indexed` annotation, the search results for a contact might include all the tickets of the contact as well. The following table shows the implications of all possible values of the `field indexed` annotation for the different object types. The `field indexed` annotation has to be set for each individual data field, e.g., *name* and *email* for a contact, *zip* and *address* for a company, *priority* and *software module* for tickets, or *name* and *model* for a resource type.

Please note that two distinct perspectives are addressed here:

1. the **search criterion**, i.e., the field for which you start the search, e.g., all companies that start with ConS*. A field is made available in the search by setting the `field-indexed` annotation.
2. the **result set**, i.e., the objects which are displayed in the result lists. The objects in the result set are defined by the value of the `field-indexed` annotation. Depending on this value, you might see only the objects you have searched for (e.g., only contacts) or also objects which are related to the objects you have searched for (e.g., companies of those contacts). Please see the following table for a detailed explanation.

Object type/ value of field indexed annotation	transitive	unit	local	not indexed
TICKET	• no difference between the three values • the ticket data will be available for searching • no other objects which are related to a ticket in any way will be retrieved • we recommend that you use the default value “transitive”			• the ticket data will be NOT available for searching
CONTACT	• the contact data will be available for searching • the tickets of the contact will also be found when you search for the contact • searching by company by its contact field is NOT possible	• the contact data will be available for searching • searching for a company by its contact field is NOT possible	• the contact data will be available for searching • no other objects which are related to a contact in any way will be retrieved	• the contact data will be NOT available for searching

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Object type/ value of field indexed annotation	transitive	unit	local	not indexed
COMPANY	• the company data will be available for searching • the tickets of the company will also be found when you search for the company • the contacts of the company will also be found when you search for the company	• the company data will be available for searching • the contacts of the company will also be found when you search for the company	• the company data will be available for searching • no other objects which are related to a company in any way will be retrieved	• the company data will be NOT available for searching
RESOURCE	• no difference between the three values • the resource data will be available for searching • no other objects which are related to a resource in any way will be retrieved • we recommend to use the default value “transitive”			• the resource data will be NOT available for searching

Please note that if it should be possible to sort result tables (in the Web Client) according to a column (by clicking on the column header), the respective field has to be indexed!

Configuring the Phonetic Search

In order to activate the phonetic search for a data field, set the annotation `phonetic` to “true” for this field. The annotation can only be used for String fields.

F.4.2 Index

F.4.2.1 Introduction to the index in ConSol CM

ConSol CM stores most of its data in a relational database. In order to improve the performance of search operations, ***Apache Lucene*** indexes are used. For each data field that should serve as search criterion, an index is created. The indexes are stored on the file system, in a subdirectory of the data directory.

Concepts, terms and definitions

Concept	Other terms	Definition
indexer		ConSol CM component for index management
quick search		Simple free-text search using a search field in the upper right corner of the Web Client
detailed search		Advanced search on a specific page which allows combining several search criteria
autocomplete search		Context-dependent free-text search integrated into specific pages

Purpose and usage

The index is required to be able to search for objects in ConSol CM. The scope of the index is the entire ticket text, the attachments (can be configured in the system property cmas-core-index-common, index.attachment) and the data fields which are annotated with `field-indexed`. The data fields configured for the index are available as search criteria in the detailed search, and they can trigger an autocomplete search on customer and resource pages. The quick search shows the objects which contain the search term in an indexed field.

F.4.2.2 Available settings for the index

The configuration of the index is done in a number of different places. The items described in the following sections need to be configured for the index to work correctly. In addition, you can fine-tune the indexer in the system properties, see Fine-tuning the indexer.

Data directory

The indexes are stored in the file system, in a subdirectory of the data directory indicated during system setup. The path of the data directory is stored in the system property cmas-core-shared, data.-directory.

The following requirements need to be fulfilled for the index to work correctly:

- The data directory is always available for the ConSol CM server, also if it was created on another server and is linked to or mounted on the application server.
- The data directory provides enough space for storing indexes.

Annotations

The data fields which should be available as search criteria need to be specified by assigning the annotation `field-indexed` to the respective field. This is done in the data field administration fo the Admin Tool, see Ticket Fields (Setting Up the Ticket Data Model) for tickets, Customer Field

Management and GUI Design for Customer Data for customer and CM/Resource Pool - Setting Up the Basic Resource Model for resources.

The `field-indexed` annotation can have four possible values:

- **transitive**: Only relevant for customers. An index is created for the field, so the customer can be searched by this field. For contacts, the tickets of the contact are also found. For companies, the tickets of the company and the contacts of the company are found as well.
- **unit:** Only relevant for customers. An index is created for the field, so the customer can be searched by this field. For contacts, only the contact itself is found. For companies, the contacts of the company are found as well.
- **local**: An index is created for the field, so the object (ticket, customer or resource) can be searched by this field. Only the object itself is found.
- **not indexed**: No index is created for the field, so it cannot be searched.

⚠ Using the value “transitive” for customer fields can have a negative impact on the indexer performance if the company or contact has many tickets or the company has many contacts. This is due to the fact that all tickets or contacts have to be reindexed when there are changes to the annotated field.

Nested fields, as used in lists or tables, all have to have the same indexing type, otherwise they cannot be searched. For example, when you work with a table, the data fields of the type *list*, *struct*, and all data fields within the table which should be searchable, need to have the value “local” for the annotation `field-indexed`.

A **phonetic search** can be enabled for string fields. The phonetic search results do not include only exact matches but also results which sound similar even if the spelling differs from the entered search term, e.g. “Claire” and “Clare”. In order to activate the phonetic search for a data field, assign the annotation `phonetic` with the value “true” to this field.

Update mode

There are two kinds of changes which require index updates:

- **Operative changes:** Changes to tickets, customers, resources and engineers
- **Administrative changes:** Changes to the configuration

Operative changes are always processed directly. For the processing of administrative changes, there are two modes:

- **Directly start processing:** Administrative changes are processed directly. Deactivate the checkbox *No automatic commit of administrative changes*.
- **Start processing manually:** Administrative changes are processed when the user clicks the *Commit administrative changes* button on the *Index* page. Select the checkbox *No automatic commit of administrative changes*.

⚠ Direct processing of administrative changes can have a negative impact on the indexer performance if there are changes which require reindexing lots of objects, e.g. if a data field used in all tickets is added to the index.

See Technical background of the index for further details.

F.4.2.3 Basic tasks

There are two types of tasks related to the index:

- **Configuration-related tasks:** These tasks are performed when configuring the ConSol CM system.
- **Operation-related tasks:** These tasks are performed when operating a ConSol CM system.

Configuration-related tasks

You can perform the following configuration-related tasks on the index:

- Decide which data fields should be indexed, i.e. available for search, see Annotations.
- Decide if attachments should be indexed, see system property cmas-core-index-common, index.attachment. The default is `true`, meaning that attachments are indexed.
- Decide if administrative changes should be applied automatically, see Update mode.

Operation-related tasks

You can perform the following operation-related tasks on the index:

- **Check the current index tasks:**
  The current index tasks are displayed in the table on the *Index* page. You can see the ID, the type of task (indicates how the task was originated), the status (open or running), the creation date, the progress, and the details (which information is updated in the index).
- **Check the index status:**
  The index status is shown below the header buttons. In addition, it is saved to the system property cmas-core-index-common, index.status. There are three status:
  - **GREEN**
    All index tasks have run correctly. At the beginning of the synchronization process, the index status is set to GREEN. If there are any problems, it will change to YELLOW or RED.
  - **YELLOW**
    Fixable problems were identified. This status is set when there are pending administrative changes (*No automatic commit of administrative changes* not selected) or a retry task was created.
  - **RED**
    Errors have occurred. The index needs to be rebuilt.
- **Rebuild the index:**
  If the index status is RED, you need to rebuild the index by clicking the *Synchronize index* button. All open index tasks are discarded and the index is rebuilt according to the options selected in the modal window:
  - **Tickets**
    Determines the order in which the tickets are indexed. You can add queues to the list of *Prioritized queues* and sort them to determines the indexing order. The tickets from the *Remaining queues* are indexed afterwards. The options above the lists allow to decide how to handle closed tickets:
    - **Index open tickets of all queues, then closed tickets of all queues**: Indexes first the open tickets according to the queue priorization and then the closed tickets.
    - **Index open and closed tickets of each queue**: Indexes open and closed tickets according to the queue priorization.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Attachments**
  Determines when to index attachments:
  - **Index tickets along with attachments**: The attachments are indexed together with the tickets which they belong to. With this option, it takes longer until all tickets are indexed.
  - **Index tickets without attachments, then again with attachments**: First the tickets are indexed without attachments. The attachments are indexed afterwards. With this option, the tickets are available for search more quickly, because indexing the attachments takes more time.

  Click *OK* to synchronize the index.

  > The existing index is **not** deleted during the rebuild process. Therefore, the search features are available while the process runs. Changes made after the rebuild was started are immediately reflected in the index, because these data have a higher priority than the data which is synchronized due to a manually triggered index update.

- **Rebuild the index for a certain period:**
  You can rebuild the index for a certain period if there were known indexer problems during this time. Click the button *Recover index* to select the time range. All the changes (creation and update) which were made in the selected time period are reindexed. Deletions and annotation changes are not considered.
- **Repair the index:**
  You can repair the index by clicking the *Repair index* button. All the pending index tasks of the type *Retry tasks* are executed.
- **Apply administrative changes:**
  If the checkbox *No automatic commit of administrative changes* is not selected, you need to click the button *Commit administrative changes* to update the index after configuration changes.

F.4.2.4 Advanced tasks

Fine-tuning the indexer

The behavior of the indexer can be fine-tuned using system properties, see System Properties. The relevant properties are located in the module `cmas-core-index-common`. These properties allow to adjust parameters to improve the indexer performance, e.g. the number of threads, or whether attachments should be indexed or not. The most important indexer properties are described in Indexer and Search Configuration.

Technical background of the index

Which changes require index updates?

There are two kind of changes relevant for the index:

- **Operative changes**

  These are changes to the runtime data:

- **Data field values:** values of fields (ticket, customer and resource fields) which are configured for indexing using the `field-indexed` annotation
- **Engineer data:** email, first name, last name
- **Ticket data:** attachments (unless configured otherwise), creation date, assigned engineer, history, name, queue, additional engineers, subject, view

The index is automatically updated when these changes are carried out.

- **Administrative changes**

  These are changes of certain configuration data:

  - scopes
  - queues
  - enum values
  - engineer functions
  - supported locales
  - roles
  - `field-indexed` annotation of ticket, customer or resource fields

  The processing of administrative changes depends on the setting *No automatic commit of administrative changes*:

  - If the checkbox is selected, the changes are processed automatically.
  - If the checkbox is not selected, the changes are stored in the database for deferred processing. Processing can be started by clicking the *Commit administrative changes* button.

How are index updates processed?

Two services, see CM Services, are involved in index updates:

- **Index changes notifier:**
  Stores the entities which require an index update in the `cmas_index_update_serialized` table.
- **Index changes receiver:**
  Polls the `cmas_index_update_serialized` table for new entries and creates index tasks in the tables `cmas_index_update_task` and `cmas_index_update_part`. Administrative tasks are directly created in the database table `cmas_index_administrative_task`. The indexing server then executes the tasks from these tables and updates the Lucene files accordingly. If the execution of an update task fails, a new update task with the type *Retry task* is created and the index status is set to YELLOW. You need to execute the task by clicking the *Repair index* button in the Admin Tool.

> On clustered systems, the index is built on the master indexing server. The slave indexing servers download the index files from the master indexing server.

> Do **not** stop the *Index changes notifier* service. If it is stopped and there is a change which requires an index update, the index status is set to RED and a rebuild is necessary. You can stop the *Index changes receiver*. After restart it will pick up all the missing changes from the respective table

F.4.3 Action Framework - Search Actions

F.4.3.1 Introduction to Search Actions

It is possible to define actions for search results. These actions are presented like workflow activities for result lists of

- tickets
- customer
- resources

Search (result) actions operate on the result set of a search. You can configure search (result) actions to offer different bulk operations to the ConSol CM engineers, e.g., assigning all tickets in the result list to the current engineer (see the following figure), sending a specific email to all customers in the result list, or creating a new maintenance ticket for all PCs (resources) in the result list.

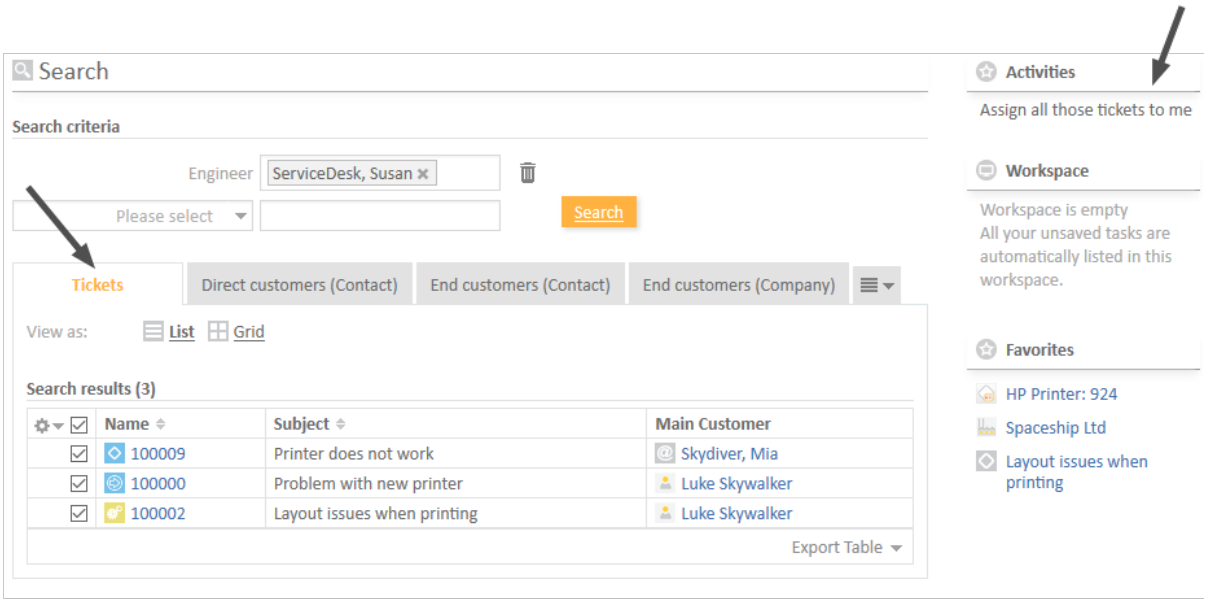


Figure 318: *ConSol CM Web Client - Search actions for tickets*

> A search (result) action will be displayed in the Web Client only if
>
> - the engineer has ACT permissions for all elements of the result set (if any)
> - the condition script (if any) has returned “true”

Each search action is implemented based on Admin Tool scripts of the following types.

- **Used for tickets:**
  - Search action for tickets
  - Search condition for tickets
- **Used for resources:**
  - Search action for resources
  - Search condition for resources

- **Used for customers:**
  - Search action for customers
  - Search condition for customers



Please note that the search action can be performed on

a) only the part of the result list which is displayed, e.g., on only 20 of 100 hits

OR

b) on the entire result list, e.g., on all 100 hits

The behavior of the search action depends on

- the implementation of the Admin Tool script. Please see the detailed explanation in section The Result Set in Search Action Admin Tool Scripts.
- the selection of rows. See Page Customization, enableRowSelection.

F.4.3.2 Configuring Search Actions Using the Admin Tool

Search actions are defined in the Admin Tool. You have to perform two or three steps to implement a search action, depending on the type of the search action.

Steps to Perform to Implement a Search Action for Tickets

1. Define and write the Admin Tool script (navigation group *System*, navigation item *Scripts and Templates*)
   - mandatory: a script of type *search action for tickets* (see following figure)
   - optional: a script of type *search condition for tickets*

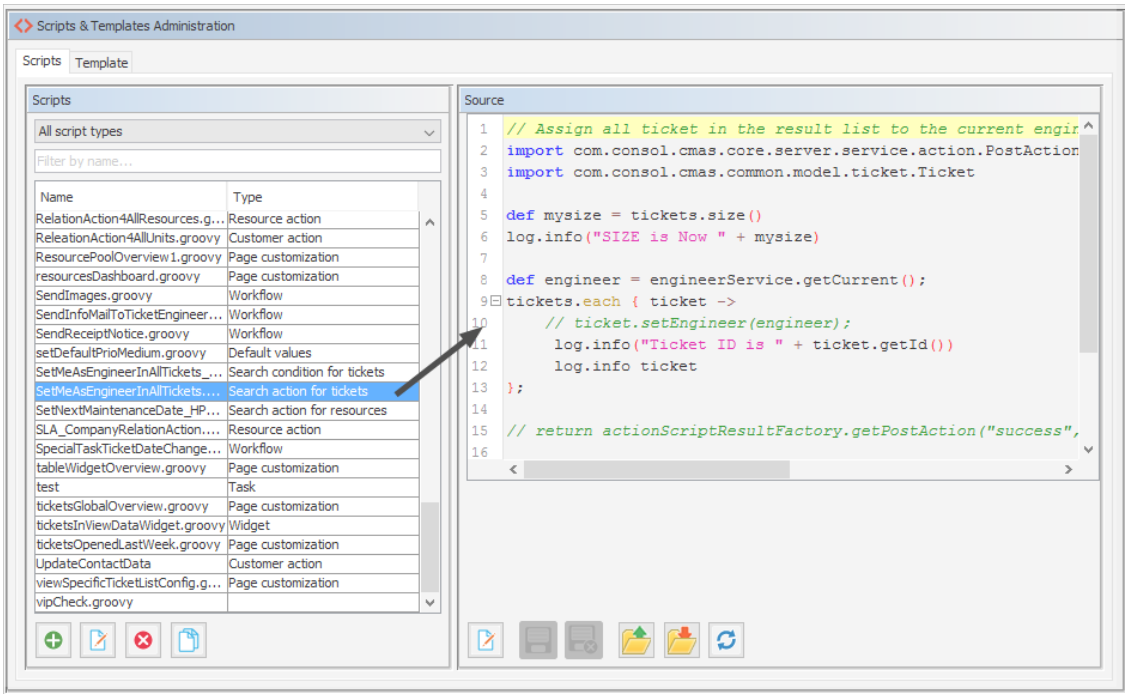


Figure 319: *ConSol CM Admin Tool - System, Scripts and Templates: Script for a search action for tickets*

Example script:

```
// Assign all tickets in the result list to the current engineer

def engineer = engineerService.getCurrent();
tickets.each { ticket ->;
   ticket.setEngineer(engineer);
}
client.showInfoMessage("cmweb.search.assigned").withRefreshContent();
```

Code example 44: *Search action script for tickets*

> For a detailed explanation about how to write scripts for the ConSol CM Action Framework, please read section Scripts for the Action Framework.

2. Define the search action in the navigation item *Search Actions*, navigation group *Tickets*. During this step, the action script is assigned to the search action. A condition script might also be assigned, but this is optional. You can add the localized term for the Ticket Action using the *Localize* button. For a detailed explanation of the localization mechanism, please refer to section Localization of Objects in General, Type 1.

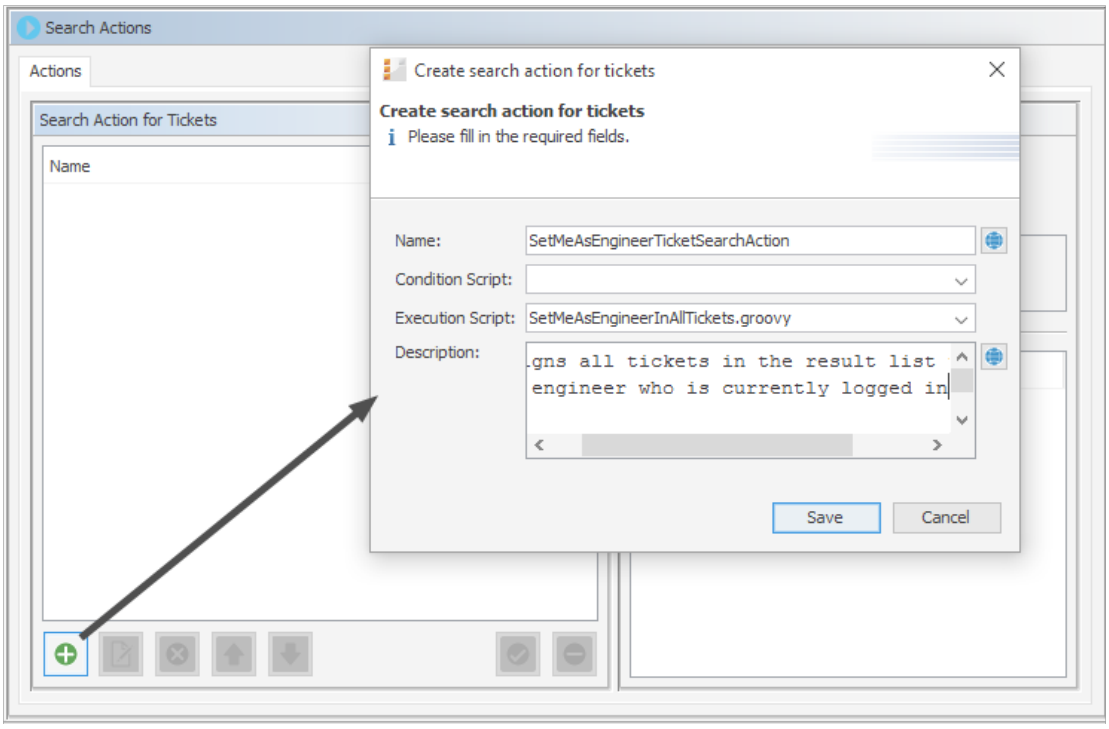


Figure 320: *ConSol CM Admin Tool - Tickets, Search Actions: Defining a new ticket search action*

Steps to Perform to Implement a Search Action for Resources

1. Define and write the Admin Tool script
   - mandatory: a script of type *search action for resources*
   - optional: a script of type *search condition for resources*

   Example script which sets the next maintenance date of all retrieved HP printers to a date in two weeks:

```
// Schedule maintenance date for all selected HP printers to a date in two
 weeks from today
import groovy.time.TimeCategory

def now = new Date();
def nextMaintenanceDate;
use(TimeCategory) {
   nextMaintenanceDate = now + 2.weeks
}

for(Resource r : resources){
   r.set("HP_Printer_Fields_basic.NextMaintenaceDate", nextMaintenanceDate)
}

client.showInfoMessage("maintenance.date.updated").withRefreshContent();
```

Code example 45: *Search action script for resources*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

> For a detailed explanation about how to write scripts for the ConSol CM Action Framework, please read section Scripts for the Action Framework.

2. Define the search action in the navigation item *Actions*, navigation group *Resources*. During this step, the action script is assigned to the search action. A condition script might also be assigned, but this is optional. You can add the localized term for the resource action using the *Localize* button. For a detailed explanation of the localization mechanism, please refer to section Localization of Objects in General, Type 1.

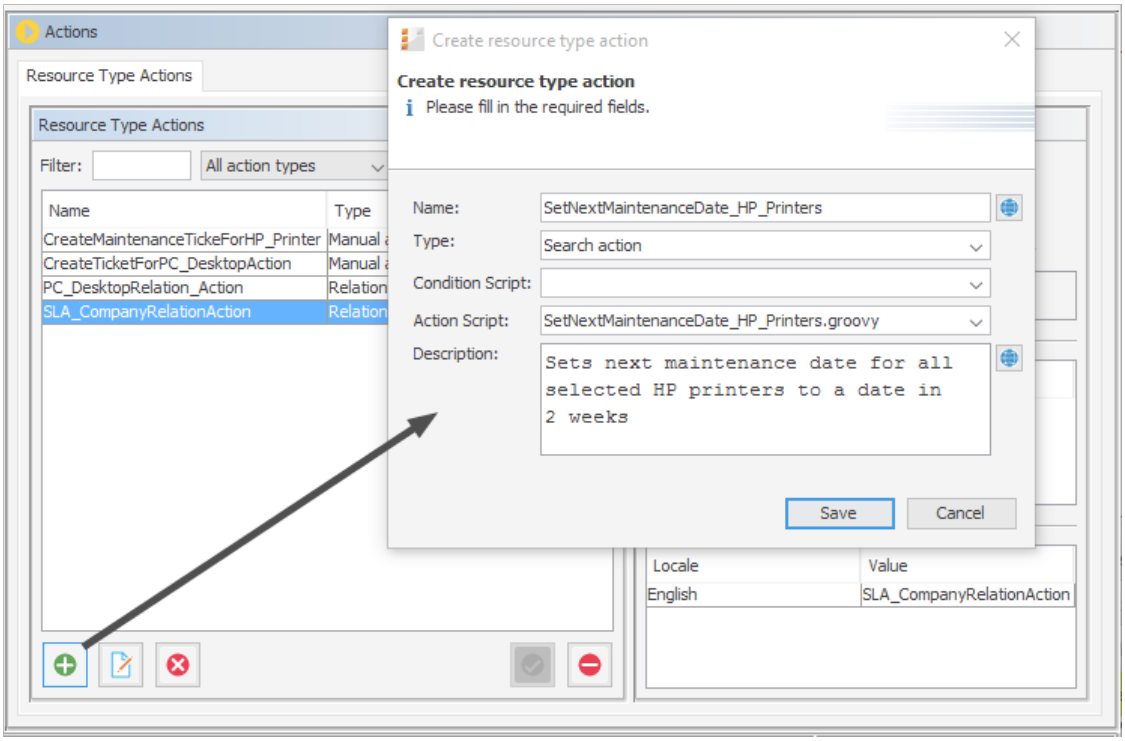


Figure 321: *ConSol CM Admin Tool - Resources, Actions: Defining a resource search action*

3. Assign the action to one or more resource type(s), navigation group *Resources*, navigation item *Data Models*, tab *Search Actions* for each resource type.

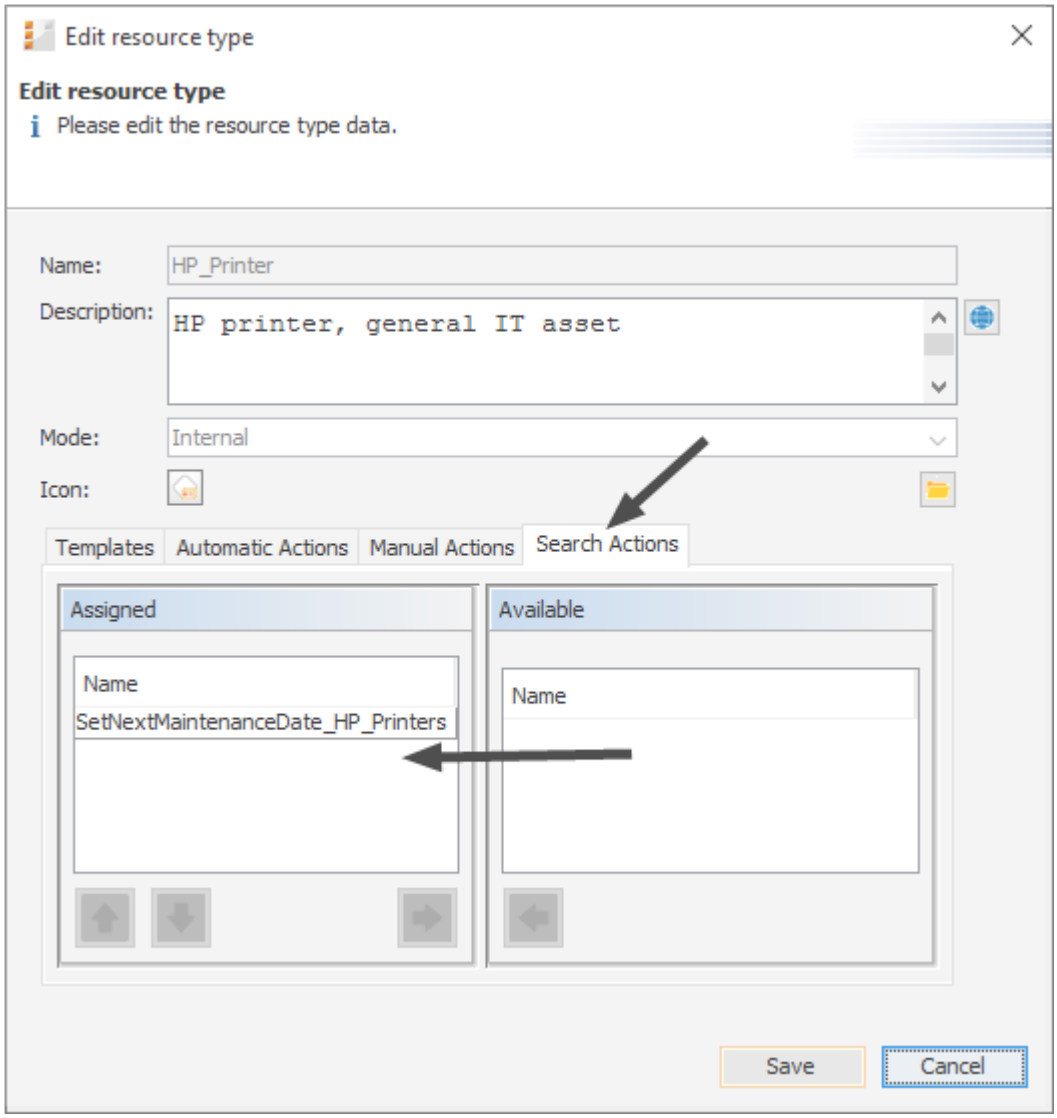


Figure 322: *ConSol CM Admin Tool - Resources, Actions: Assigning a search action to a resource type*

4. Ensure that it works in the Web Client.

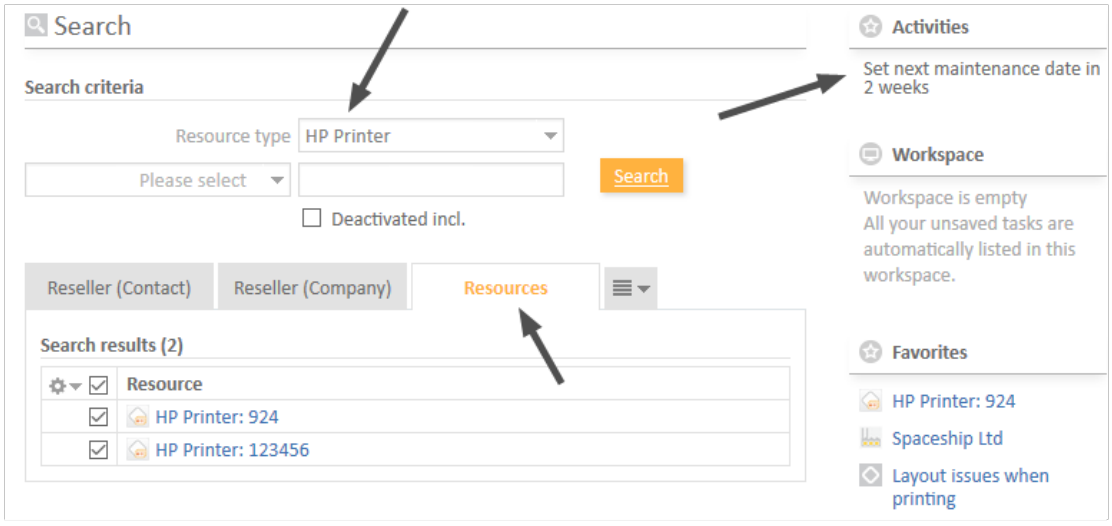


Figure 323: *ConSol CM Web Client - Search action on resources, 1*

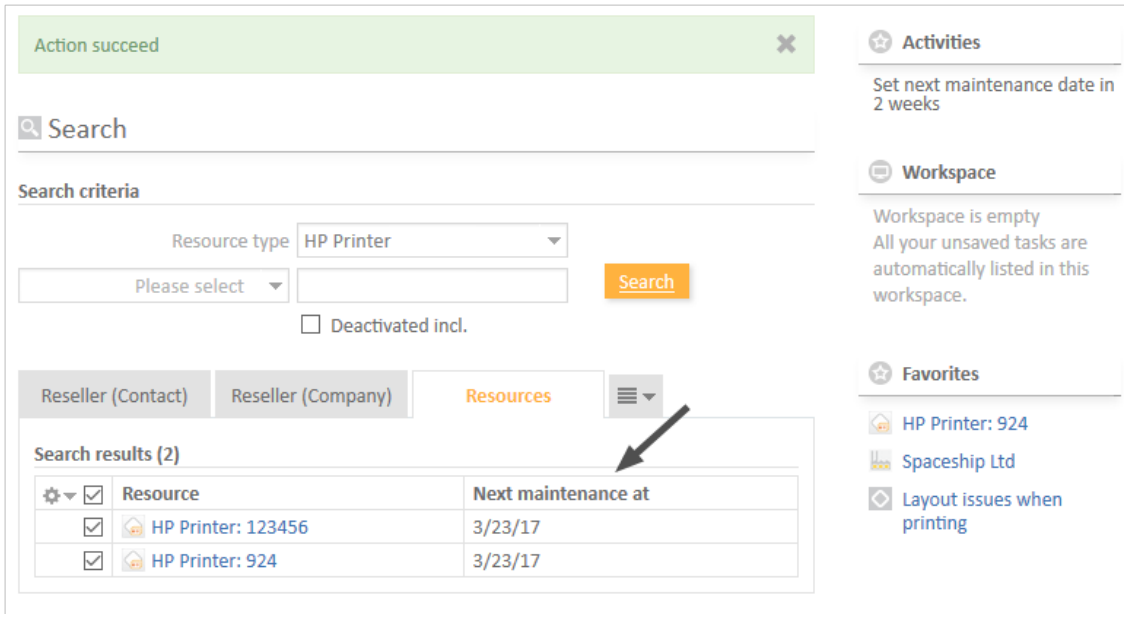


Figure 324: *ConSol CM Web Client - Search (Result) Action on resources, 2*

Steps to Perform to Implement a Search Action for Customers

1. Define and write the Admin Tool script
   - mandatory: a script of type *search action for customers*
   - optional: a script of type *search condition for customers*, in this example we will work with a condition script

Example action script which creates a ticket for each contact in the result list:

```
// deadline is a required field in Service Desk tickets!
import groovy.time.TimeCategory

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
   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)
}

client.showInfoMessage("cmweb.search.assigned").withRefreshContent();
```

Code example 46: *Search action script for customers*

Please make sure that

- you only use customer fields in the script which are present in the customer data model of the customer group where the Search (Result) Action script will be used! For example: To reference the name of a contact, the exact field name has to be used (*customer_name* in the example above)! Hence, before implementing a Search (Result) Action script, it is recommended that you check the customer data model for the required fields.
- the customer group has been assigned to the queue where new tickets should be created.

Example condition script which causes the search action to only be displayed if the list has more than five entries.

```
if (units.size() >= 5){
   return true
} else {
   return false
}
```

Code example 47: *Search condition script for units*

> For a detailed explanation about how to write scripts for the ConSol CM Action Framework, please read section Scripts for the Action Framework.

2. Define the search action in the navigation item *Actions*, navigation group *Customers*. During this step, the action script is assigned to the search action. A condition script might also be assigned, but this is optional. You can add the localized term for the customer action using the *Localize* button. For a detailed explanation of the localization mechanism, please refer to section Localization of Objects in General, Type 1.

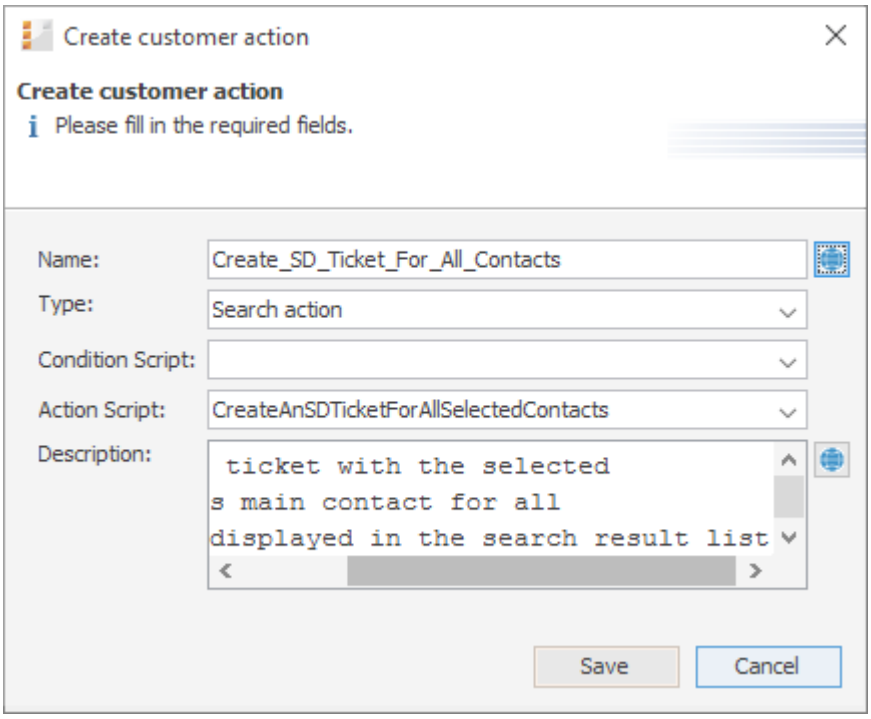


Figure 325: *ConSol CM Admin Tool - Customers, Actions: Defining a customer search action*

3. Add the condition script.

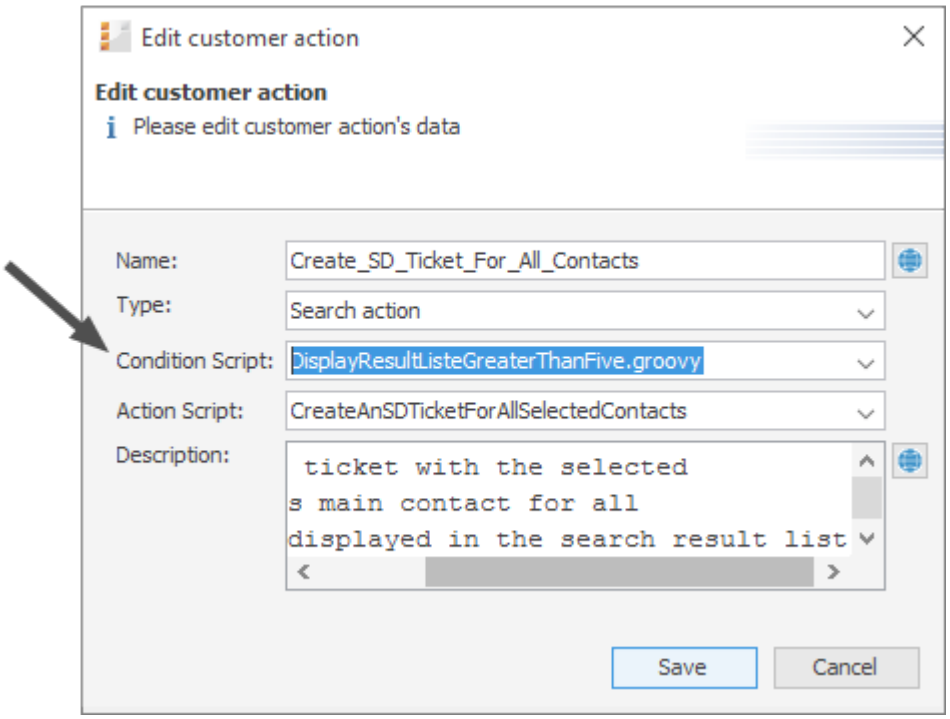


Figure 326: *ConSol CM Admin Tool - Customers, Actions: Defining a search script condition*

4. Assign the action to one or more customer groups, navigation group *Customers*, navigation item *Customer Groups*, tab *Search Actions* for each customer group. Assign the action(s) to the contacts (*Contact Search Actions*) and/or the company/companies (*Company Search Actions*) of this customer group.
   Note that the action and condition scripts are listed together in one table.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

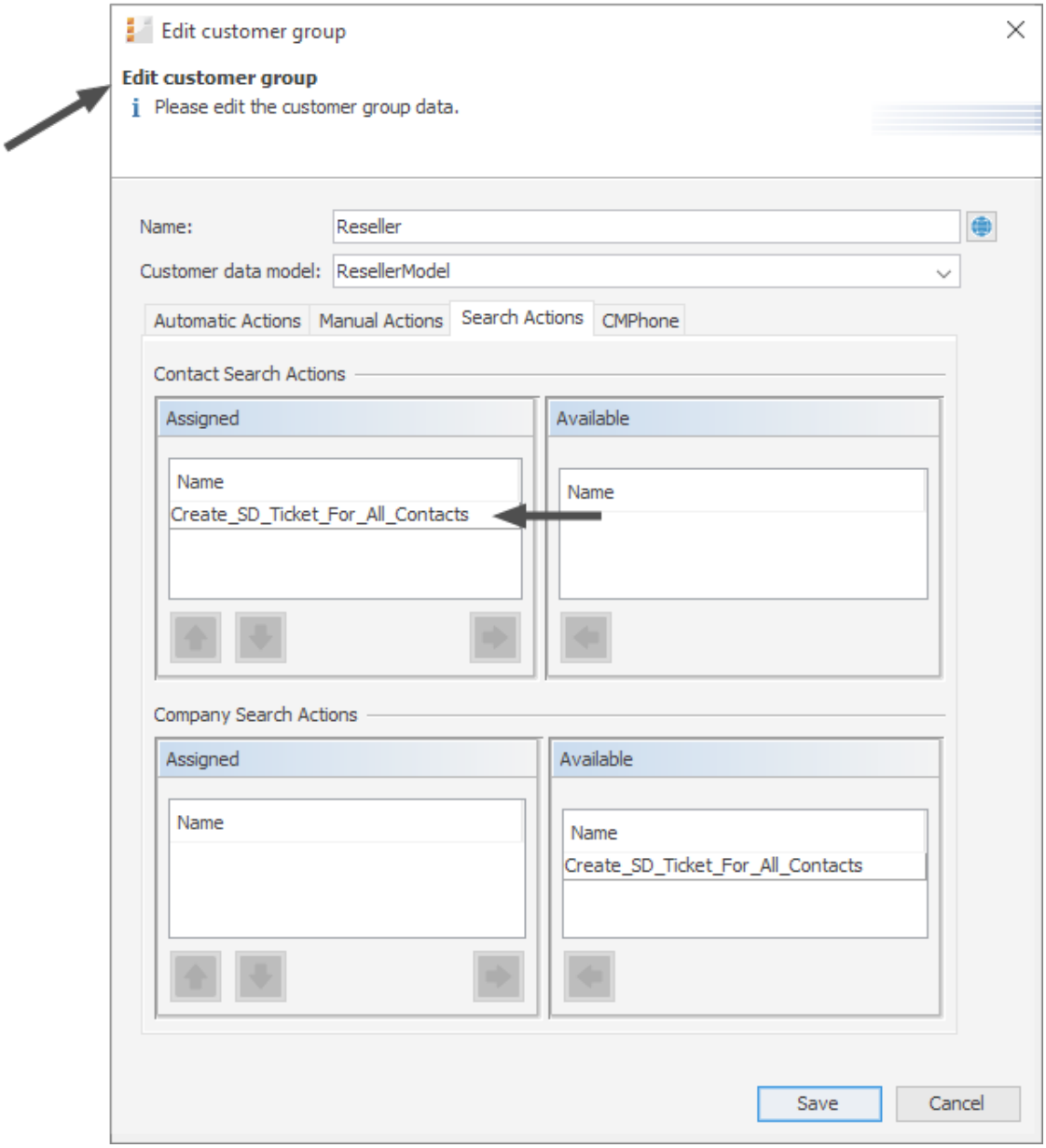


Figure 327: *ConSol CM Admin Tool - Customers, Customer Groups: Assigning a search action to a contact object within a customer group*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

5. Ensure that it works in the Web Client.

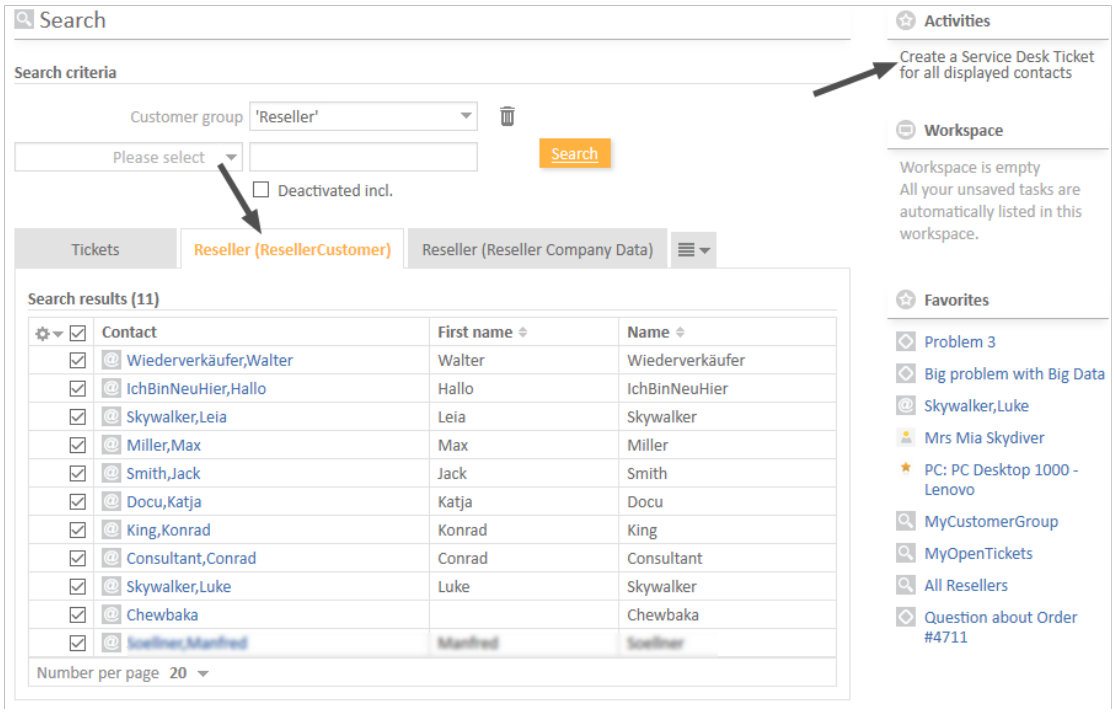


Figure 328: *ConSol CM Web Client - Search action on customers (here: contacts), 1*

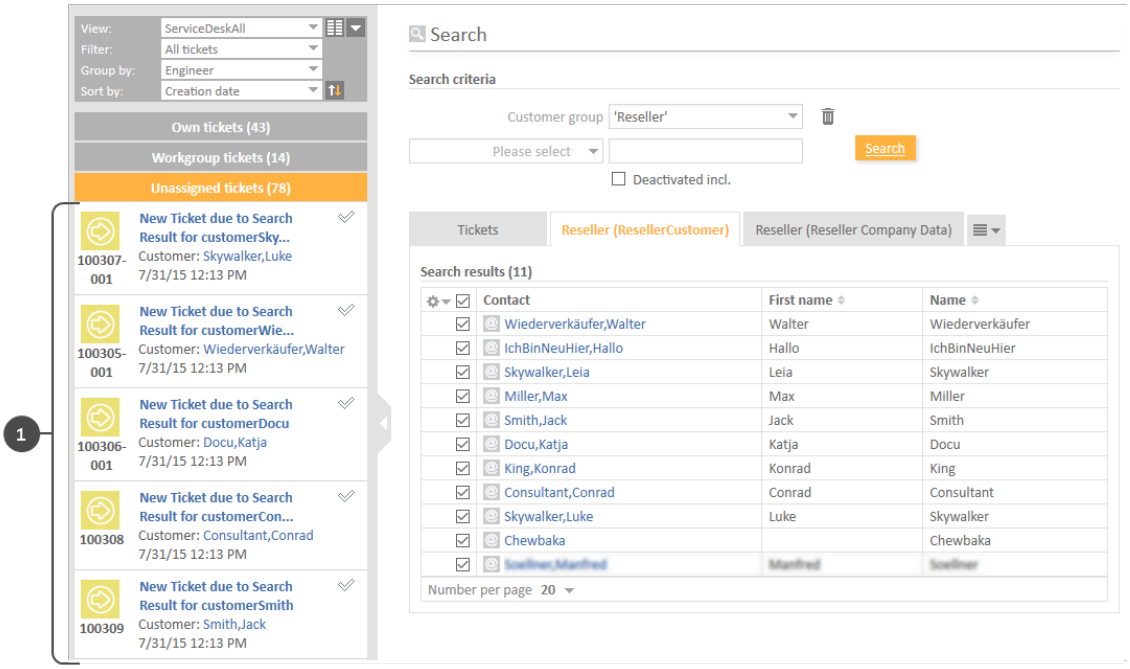


Figure 329: *ConSol CM Web Client - Search action on customers (here: contacts), 2*

The ticket list shows the new ServiceDesk tickets (1) which have been created in the Search Action script.

The following figure shows that the activity is not offered if the result list contains less than five entries (as defined in the condition script).

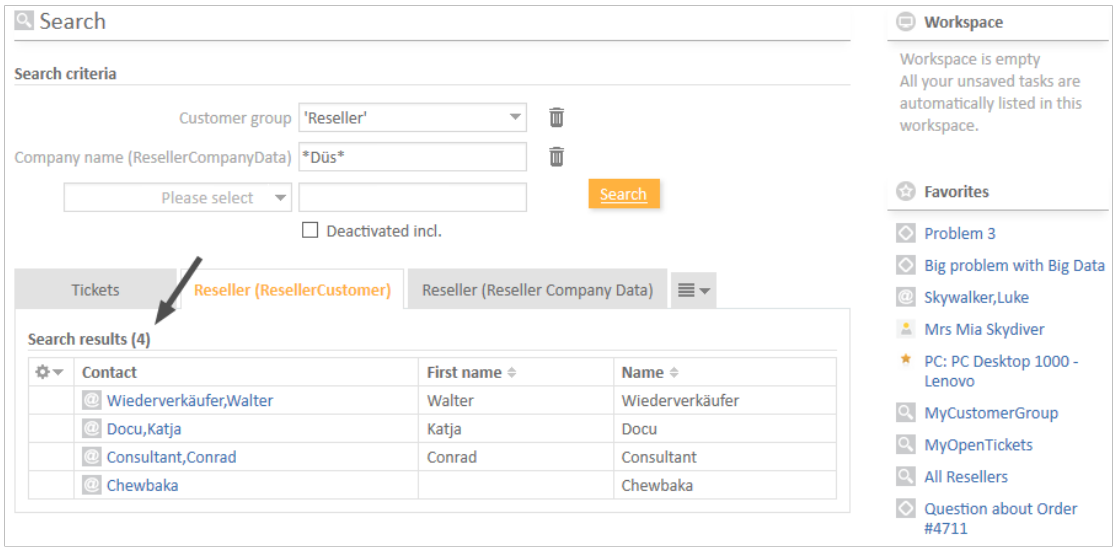


Figure 330: *ConSol CM Web Client - Search action on customers (here: contacts), 3*

F.4.3.3 Tips and Tricks for Search Action Admin Tool Scripts

Objects Implicitly Available for Search Action Admin Tool Scripts

Some objects are implicitly available in the Admin Tool scripts, i.e., you do not have to import specific Groovy classes, nor do you have to instantiate such objects from a class.

Objects which are implicitly (without an explicit import or instantiation) available in Admin Tool scripts:

All search action scripts

- pageSize (Integer)
- pageNumber (Integer)

Script types *search action for tickets* and *search condition for tickets*

- criteria (TicketCriteria)
- tickets (List<Ticket> for ticket search action)

Script types *search action for resources* and *search condition for resources*

- criteria (ResourceCriteria)
- resources (List<Resource> for resource search action)

Script types *search action for customers* and *search conditions for customers*

- criteria (UnitCriteria)
- units (List<Unit> for unit search action)

The Result Set in Search Action Admin Tool Scripts

The search action can be performed on different interpretations of the result set. The behavior of the search action depends on the implementation of the Admin Tool script.

The search action can be performed on

a) only the part of the result list which is displayed in the Web Client, e.g., on only 20 of 100 hits. So this depends on the paging the engineer has selected and on the rows the engineer has selected if the row selection is switched on (see section Page Customization, enableRowSelection for details).

In the Admin Tool script, this is represented by the list of objects which is implicitly available, i.e., `resources`, `tickets` or `units`.

OR

b) on the entire result list, e.g., on all 100 hits. This can be achieved by using the `criteria` object.

Display a Message After Executing a Search Action Script

Use the methods of the class `OperationResponseBuilder` to display feedback messages after a search action script is executed. Two values of `OperationType` are available:

- SUCCESS - green message
- FAILURE - red message

Example for a positive feedback:

```
client.showMessage("action.info.executionOK", OperationMessage.Severity.INFO)
```

Code example 48: *Positive feedback*

For a detailed explanation about how to write scripts for the ConSol CM Action Framework, please read section Scripts for the Action Framework.

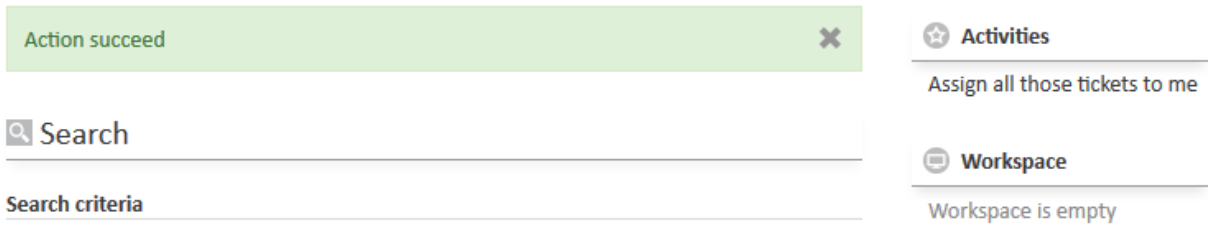


Figure 331: *ConSol CM Web Client - SUCCESS message after search action*

Example of a negative feedback:

```
client.showMessage("action.info.execution.nok", OperationMessage.Severity.WARN)
```

Code example 49: *Negative feedback*

For a detailed explanation about how to write scripts for the ConSol CM Action Framework, please read section Scripts for the Action Framework.

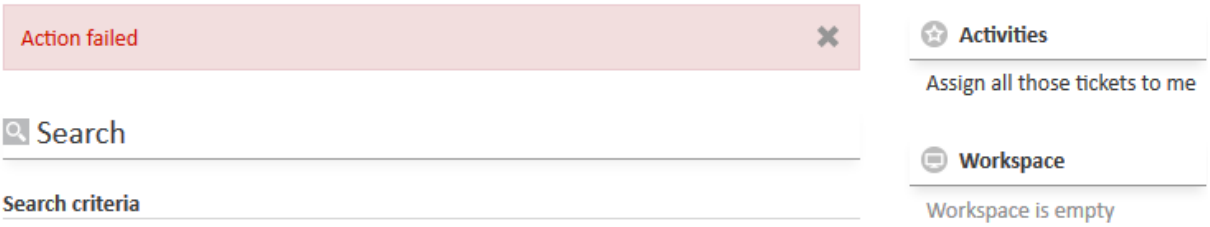


Figure 332: *ConSol CM Web Client - FAILURE message after search action*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.4.4 CSV Export of Search Results

F.4.4.1 Introduction

The search result lists can be exported in CSV format. This functionality is enabled by default and can be configured using page customization.

The engineers can use the *Export table* option which is displayed on search result pages for lists of

- tickets
- contacts
- companies
- resources

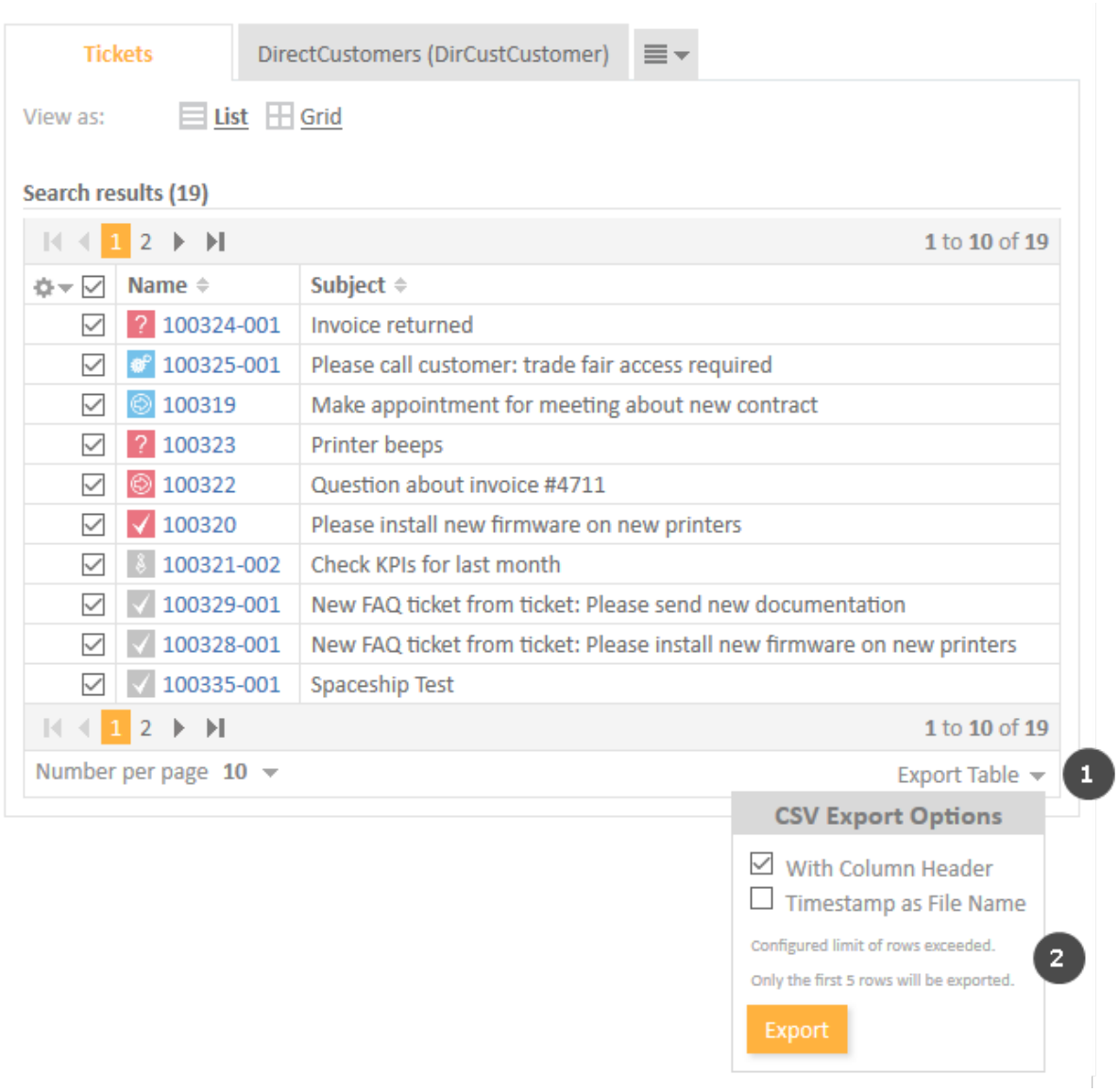


Figure 333: *ConSol CM Web Client - CSV export option for a ticket search result list*

- CSV export activated using the page customization attribute *enabled* (1)
- CSV export sub menu (2). In the example, the export would contain more rows than configured using the page customization attribute *exportRowsLimit.*

Available CSV export options in the Web Client:

- **With column header**
  This adds the column labels of the result table in the first line, as a content description.
- **Timestamp as File Name**
  If this is selected, the default file name will be the current date and time instead of the standard name “export.csv”.

Clicking on *Export* will automatically offer the option to open the newly created CSV file with the standard application configured to open CSV files on the client machine. In the CSV file there will be no pagination - the complete result set up to the limit configured in *exportRowsLimit* is exported. Icons in the result display are not included in the CSV. Ticket field string values are quoted by standard double quotes (ASCII value 34 decimal resp. 0x22 hex).

F.4.4.2 Configuring the CSV Export Functionality

The CSV export functionality is enabled by default. It can be disabled either globally in the scope `table` or for individual search result tables:

Use the following sub-scopes:

- **for ticket search results** (see figure above)
  table/searchDetailPage/TicketSearchResults
- **for contact or company search results**
  table/searchDetailPage/<customer group name>_<customer object name>
  This has to be defined for each customer group!.
- **for resource search results**
  table/searchDetailPage/ResourceSearchResults

The following attributes are available:

- **enabled**
  Enable/Disable the CSV export functionality for this search result page. The default value is “true”.
- **exportRowsLimit**
  Maximum number of rows to export in the CSV export. The default value is “500”.

> ⚠ The value “0”, which could be used to allow the export of an unlimited number of rows, is not supported anymore.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.5 The Task Execution Framework (TEF)

This chapter discusses the following:

F.5.1 Introduction 522
F.5.2 Admin Tool Scripts of Type Task 524
F.5.3 Programming with Tasks 527
F.5.4 System Properties Relevant for the TEF 532

F.5.1 Introduction

Using the ***Task Execution Framework*** (TEF), ConSol CM can perform various tasks which are not directly tied to or embedded in another script (like a workflow script, unit action, resource action, search action or another type of Admin Tool script) and which can be executed asynchronously. This can be used, e.g., for long-term system tasks which might cause a timeout when started within a regular ConSol CM script. The TEF tasks can be executed in an asynchronous manner. TEF scripts can be started (i.e., the TEF API is available in):

- manually in the Admin Tool
- from workflow activity scripts
- from email scripts
- from action scripts (search actions, unit actions, resource actions)

A task is stored as Admin Tool script of type *Task* (see Admin Tool Scripts). This script type is explained in the respective section below.

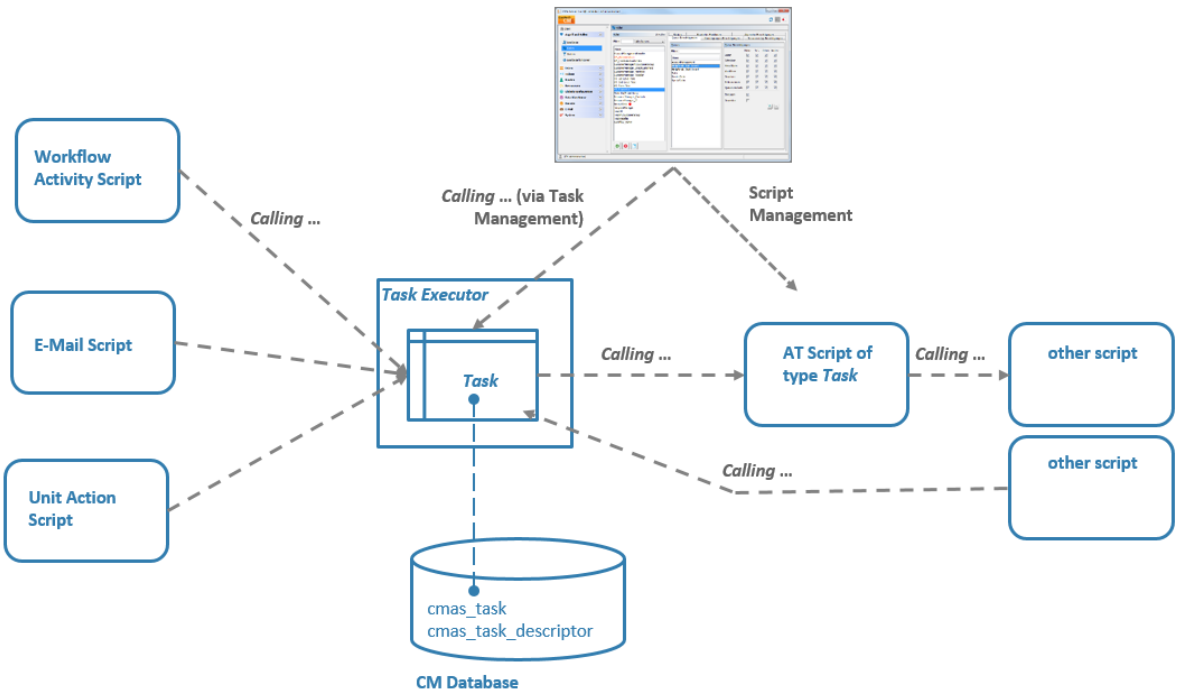


Figure 334: *ConSol CM Task Execution Framework*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

The **Task Executor** is a ConSol CM module (a singleton with watchdog functionalities) which controls the execution of the tasks. The Task Executor scans the database for new scheduled tasks and uses its thread pool for the execution of the tasks which have to be executed at a certain point in time.

The **task definition** is stored in an Admin Tool script. Thus, for one task definition, usually one Admin Tool script is used.

A (scheduled) task, i.e., one single run of the task, can be started ...

- using the Admin Tool, navigation item *Task execution*, see section Execution of a Task Using the Admin Tool. Here, a task can be started immediately, it cannot be scheduled for another point in time.
- implementing the calling of the task script in another script of one of the following types. Here, a script can be executed immediately or can be scheduled for a later point in time.
    - workflow activity scripts
    - email scripts
    - action scripts (as part of the Action Framework, i.e., in unit action scripts, resource action scripts or search action scripts)

In scripts, for every execution of a task, a **task descriptor**, i.e., an object of the class `TaskDescriptor`, is available. This task descriptor provides information like the task's progress.

The start date of a task can be set using the **task execution specification** (object of the class `ExecutionSpecification`).

Please see section Programming with Tasks for programming details.

F.5.2 Admin Tool Scripts of Type *Task*

F.5.2.1 Introduction to Admin Tool Scripts of Type *Task*

Every Admin Tool script of type *Task* has to implement the following methods. The method signatures are inserted automatically when a script of this type is created.

```
def onInitialize(taskDescriptor) {}
def onExecute(taskDescriptor) {}
def onError(taskDescriptor) {}
def onCancel(taskDescriptor) {}
```

F.5.2.2 Example Admin Tool Script of Type *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 example 50: *Admin Tool Script of Type Task*

F.5.2.3 Execution of a Task Using the Admin Tool

In the Admin Tool, navigation group *Services*, navigation item *Task Execution*, you can start tasks which have been defined as Admin Tool scripts before.



To be able to execute tasks, the system property cmas-app-admin-tool, start.-groovy.task.enabled has to be set to the value “true”. This property is not present in a default installation and has to be added manually.

⚠ Please note that if the system property cmas-app-admin-tool, start.groovy.task.enabled is set to “true”, and thus if you, as an administrator, can execute tasks using the Admin Tool, you have to be absolutely sure what the task will be doing! Be aware of the risks involved with tasks which, for example, delete customer data or tickets and should only be executed via a workflow or Admin Tool script!

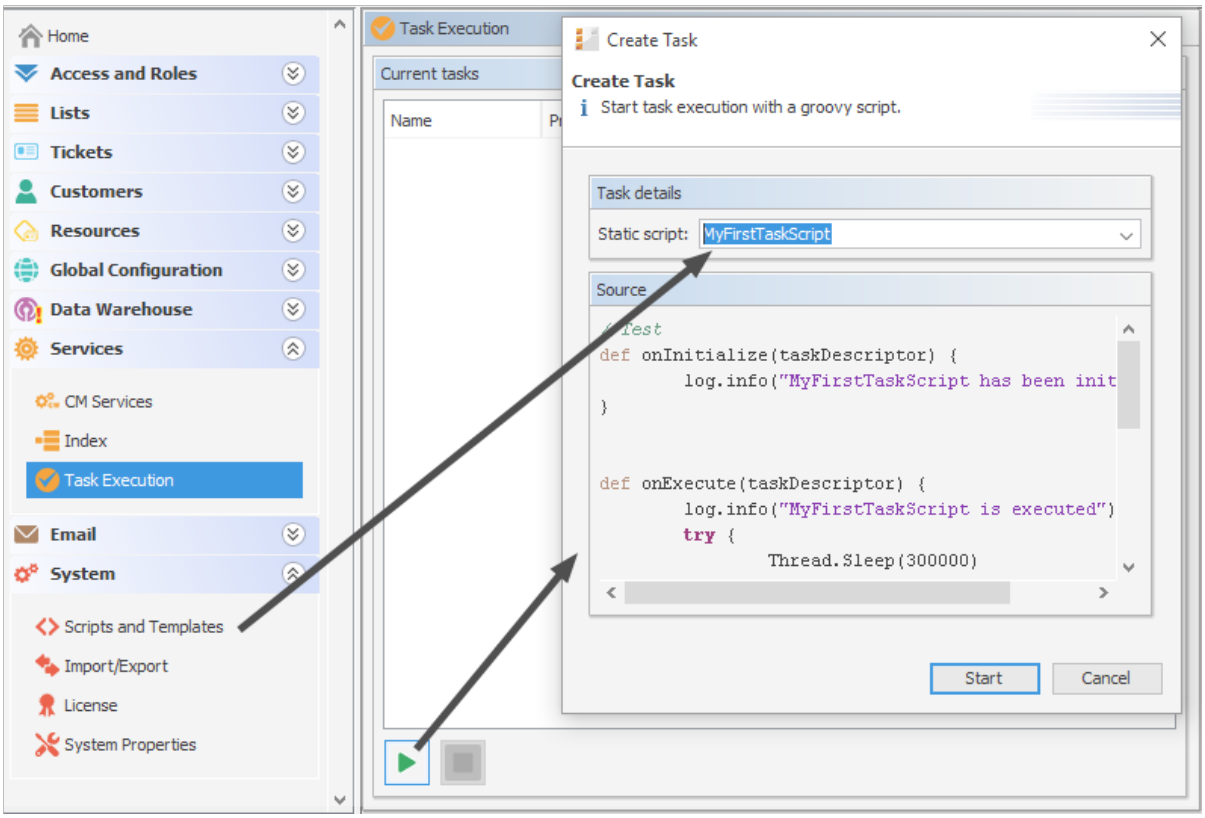


Figure 335: *ConSol CM Admin Tool - Services, Task Execution: Creating a task*

To start a script, click on the *Start* button and select the name of the Admin Tool script from the drop-down menu *Static script*. All Admin Tool scripts of type *Task* will be listed here. Click on *Start* to execute the script immediately. It is not possible to schedule a task using the Admin Tool GUI. If the start should be delayed, this has to be implemented within the script, see Defining the (First) Execution Date.

When a task is running, a progress bar is shown. You can stop (cancel) the task using the *Stop* button.

The update interval of the task list is defined using the CM system property cmas-app-admin-tool, task.panel.refresh.interval.seconds. The property is not present by default but has to be added manually.

The following figure shows a task being executed. You reach this screen by opening the navigation item *Task Execution* in the navigation group *Services*.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

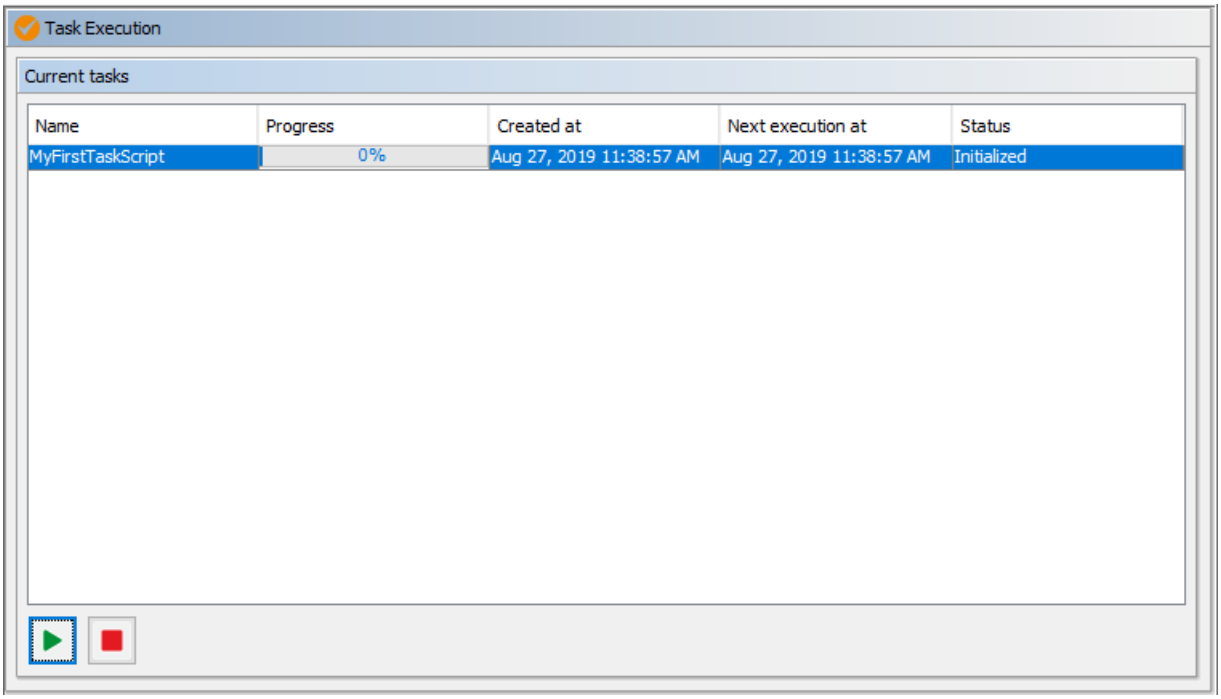


Figure 336: *ConSol CM Admin Tool - Services, Task Execution: Running task*

When a TEF script is executed manually using the Admin Tool, it will run on the machine where the Admin Tool is running. If you work in a clustered environment, where scripts are executed automatically, you can define the cluster node ID where TEF scripts should run using the CM system property cmas-core-server, task.execution.node.id.

F.5.3 Programming with Tasks

F.5.3.1 Introduction

In the current ConSol CM version, only one type of task is available, the **Groovy Task with a static script**. This refers to the Admin Tool script which defines the task, as described in the previous section.

The **Task Execution Service** (Groovy class `TaskExecutionService`, a singleton) runs in the background and scans the ConSol CM database for tasks (database table `cmas_task_descriptor`) with the status INITIALIZED. Like all ConSol CM services it is implicitly available as an object named `taskExecutionService` (see the following examples). When the start time of the task has been reached, the task is started.

Parameters for the new task have to be set either in the task descriptor (Groovy class `TaskDescriptor`) or in the task execution specification (Groovy class `ExecutionSpecification`). The task descriptor will also provide information about the running task, like the task's progress.

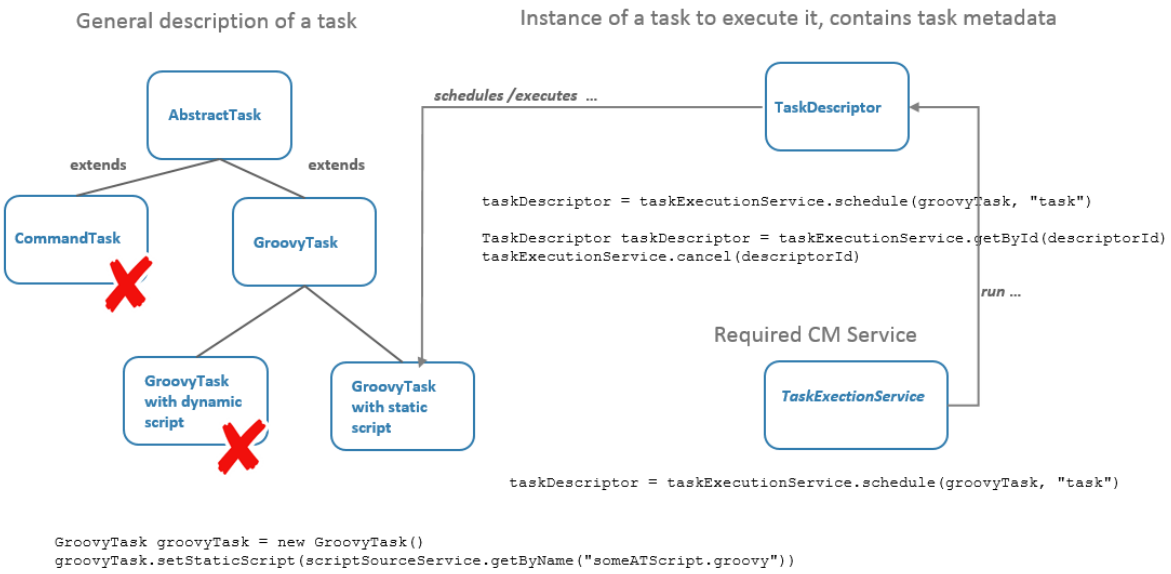


Figure 337: *Some TEF Groovy classes*

F.5.3.2 Coding Examples

Creating a Task

```
GroovyTask groovyTask = new GroovyTask();
groovyTask.setStaticScript(scriptSourceService.getByName("someATScript.groovy"));
taskDescriptor = taskExecutionService.schedule(groovyTask, "task");
```

Code example 51: *Creating a task descriptor*

Canceling (Killing) a Task

Part 1: Create the task descriptor and save its ID somewhere.

```
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 example 52: *Cancelling a task*

Part 2: Potentially used during the execution of the task.

```
taskExecutionService.cancel(myTaskDescriptorId)
```

Repeating a Task

If you set another execution date for a task after its job has completed, it will be rescheduled. This is accomplished from within the Admin Tool Task script, as demonstrated here.

```
def onInitialize(taskDescriptor) {}

def onExecute(taskDescriptor) {
   //some code to execute
   ........................
   //here, we return the special steering object with the new execution date of the
    task
   return new ExecutionSpecification().setRetryRequested(true).setExecutionDate(new
    Date(new Date().getTime() + 15000));
}

def onError(taskDescriptor) {}
def onCancel(taskDescriptor) {}
```

Code example 53: *Repeating a task*

Defining the (First) Execution Date

For a script which should not be started immediately, you can define a start time in the `onInitialize()` method.

```
def onInitialize(taskDescriptor) {
   return new ExecutionSpecification().setExecutionDate(new Date(new Date().getTime
    () + 15000));
}
```

Code example 54: *Scheduling a task*

When the task is started, the set execution date is displayed in the *Next execution at* column of the task overview in the Admin Tool.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Repeating a Task after an Error Occurs

```
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 example 55: *Repeating a task after an error occurred*

Working with the ContextReference

Using the `ContextReference`, it is possible to determine which context a script has been called in. For example, a TEF script might be required to behave differently depending on whether it is called from a workflow script in a certain activity or in the process from another activity. In such cases, the `ContextReference` will help you to deduce which workflow script the TEF script was called from. You just set a different `ContextReference`, which is a simple string as identifier, in each workflow script. Then you can retrieve the `ContextReference` within the task. Thus the task "knows" where it was called from.

Use the methods

- `void GroovyTask.setContextReference(String ContRef)`
- `String TaskDescriptor.getContextReference()`

See also example #2 below.

F.5.3.3 Examples for the Use of Task Scripts

Example 1: Using a Simple Task Script

In this example, a task script will be executed from a workflow activity. No delay is set, i.e., the task is scheduled to be executed immediately when the engineer executes the workflow activity using the Web Client. The script might then run in the background and the engineer will only see the results (like new ticket entries or new customer data) when the script is finished. No action is required on the engineer's part in between the script's start and completion.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

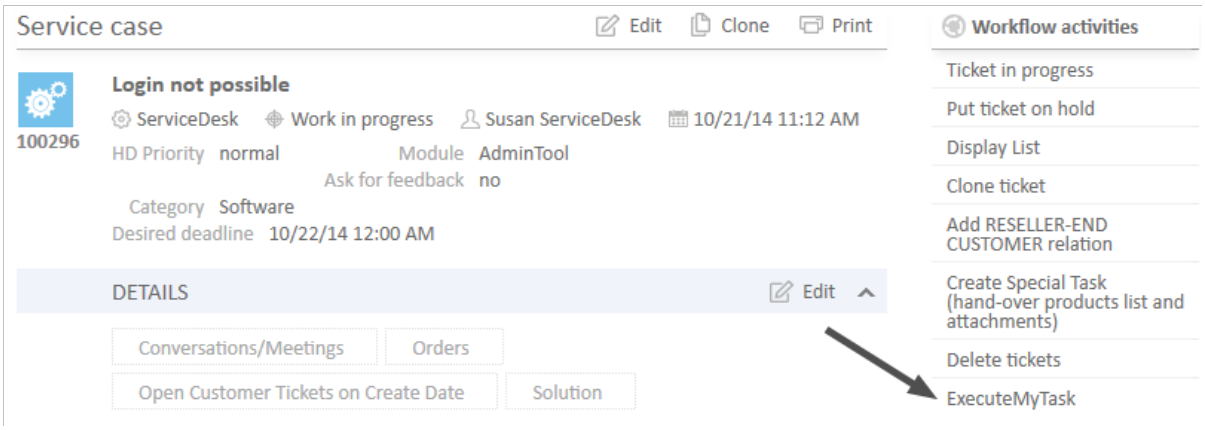


Figure 338: *ConSol CM Web Client - Workflow activity for task execution*

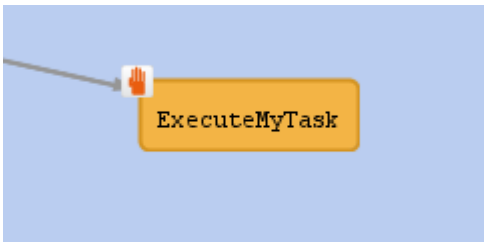


Figure 339: *ConSol CM Process Designer - Workflow activity for task execution*

```
def myNewTask = new GroovyTask()
myNewTask.setStaticScript(scriptSourceService.getByName("MyFirstTaskScript"))
def myTaskDescriptor = taskExecutionService.schedule(myNewTask, "myTaskGroup")
```

Code example 56: *Workflow activity script for task execution*

```
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}
```

Example 2: Working with the ContextReference, Simple Example with Log Output

The Task script (Admin Tool script of type *Task*) will display the ContextReference from which it has been called.

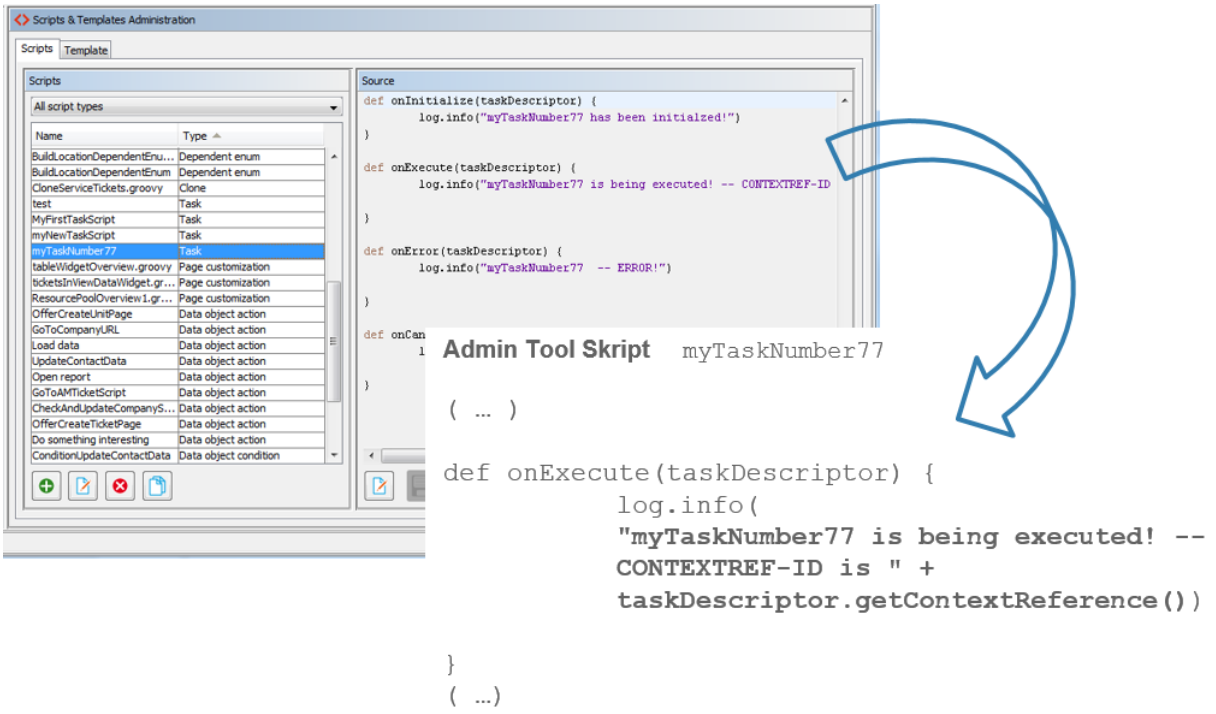


Figure 340: *ConSol CM Admin Tool - System, Scripts and Templates: Admin Tool script for a TEF task*

In different workflow activities, different ContextReferences are set as unique identifiers. When the TEF script is called, it will always display (in the log output) the ContextReference of the calling workflow activity.

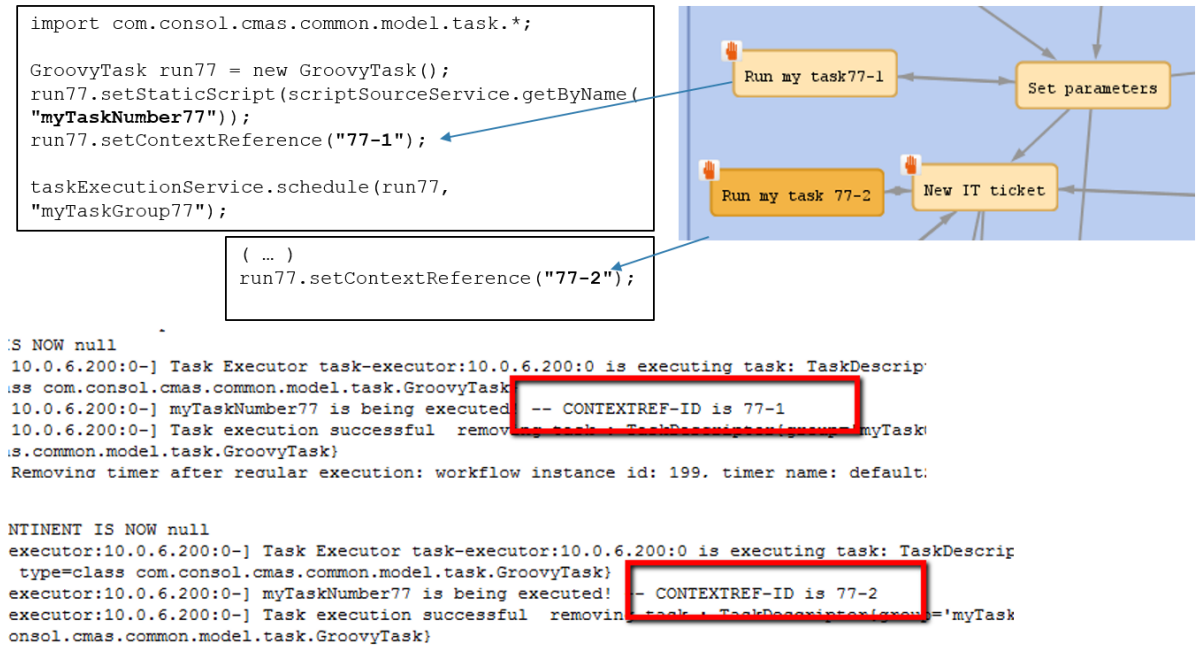


Figure 341: *Workflow activities and log output when one of the activities has called the TEF script*

F.5.4 System Properties Relevant for the TEF

Module	Property	Default value	Description
cmas-app-admin-tool	start.groovy.task.enabled	false	Enables the "start task" button in the Admin Tool
cmas-core-server	transaction.timeout.minutes	600	Sets the transaction timeout for the task execution service, i.e., one run of a task must finish before this timeout is reached
cmas-core-server	number.of.tasks	1	Thread pool size, i.e., number of tasks executed in parallel
cmas-core-server	task.execution.interval.seconds	5	Time to wait between execution of two tasks, in seconds

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.6 The ConSol CM Action Framework

The ConSol CM Action Framework comprises several types of actions which are NOT workflow actions. Workflow actions are executed for tickets, either manually or automatically, whereas the Action Framework provides actions for other types of objects. Additional components like Control Forms extend the spectrum of the framework.

Please see the following overview for a first impression of all components of the Action Framework and read the respective sections for detailed information.

F.6.1 Actions

- **Search actions**
  Always manual actions. Available for
  - tickets
  - customer (= units)
  - resources

  Explained in section Action Framework - Search Actions
- **Customer actions**
  Can be manual or automatic actions. Available for
  - companies
  - contacts

  Explained in section Action Framework - Customer Actions
- **Resource actions**
  Can be manual or automatic actions. Available for
  - resources

  Explained in section CM/Resource Pool - Resource Actions

F.6.2 Additional Components

- **Control Forms**
  Forms, similar to Activity Control Forms, which allow gathering data during the execution of an action of the Action Framework. They are explained in section Control Forms.
- **Scripts for the Action Framework**
  Scripts used to implement several functionalities of the Action Framework. The scripts are explained in section Scripts for the Action Framework.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.6.3 Scripts for the Action Framework

F.6.3.1 Introduction to Actions

The ConSol CM Action Framework provides the ability to perform actions which are not related to workflow activities, i.e., they can be executed in another context.

There are three types of actions, which are explained in the respective manual sections in detail.

- **Customer actions**
  Actions for contacts or companies, see section Action Framework - Customer Actions.
- **Resource actions**
  Actions for resources, see section CM/Resource Pool - Resource Actions.
- **Search actions**
  Actions for the result set of a search, see section Action Framework - Search Actions.

This section includes details about programming action scripts.

Action Types

An action can be of one of the following types:

- **Automatic actions**
  Automatic actions are performed by the system whenever the situation for which they are defined occurs in an object. The following automatic actions are available:
  - **Create**: Action performed when an object is created.
  - **Update**: Action performed when an object is modified.
  - **Delete**: Action performed when an object is deleted.
  - **Relation**: Action performed when a relation to the object is created or deleted.
- **Manual actions**
  Manual actions are performed by a user who clicks the respective action in the Web Client. There are two types of manual actions:
  - **Manual**: Action performed for a single object.
  - **Search**: Action performed for the results of a Detailed Search.

Script Types

The actions are based on Admin Tool scripts whose types correspond to the purpose of the scripts.

> For each **action**, a condition script can be defined. It has to be associated with the action in the action definition in the Admin Tool. This **condition script** will be executed before the action script.
>
> Thus, the action script will only be executed in the following situation:
>
> - No condition script is defined.
> - The condition script returns “true”.

The following script types are available for the Action Framework:

- For **customer actions**:
  - Customer condition
  - Customer action
- For **resource actions**:
  - Resource condition
  - Resource action
- For **search actions**:
  - For actions based on ticket result lists:
    - Search condition for tickets
    - Search action for tickets
  - For actions based on customer result lists, i.e., contact or company result lists:
    - Search condition for customers
    - Search action for customers
  - For actions based on resource result lists:
    - Search condition for resources
    - Search action for resources
- For **Control Forms** (see section Control Forms):
  - Control Form Condition
  - Control Form Prefill

F.6.3.2 Programming with the Action Framework

Available and Deprecated Objects

In all available script types (see section Script Types), the behavior of the system after the script can be defined by using the methods of the class `OperationResponseBuilder`. An object of this class, named `client`, is implicitly available in all action scripts, in automatic as well as in manual scripts. This object is also available in workflow scripts.

The `client` object includes methods for operations like the following:

- open a certain URL
- open the page to create a ticket, customer or resource

Besides those actions, the `client` object also includes methods to display messages in the Web Client, for example:

- display the info that the action has been executed successfully
- display a warning message if the action has failed, e.g., because certain criteria are not met by the entered data

Furthermore, the `client` object offers a `rollback()` method.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Please note:

- The classes and respective objects of `ActionScriptResult`, `ActionScriptResultFactory`, `PostActionScriptResult`, `PostActionType` and `postActivityExecutionHandler` are deprecated. Use the new `OperationResponseBuilder` implementation described above.
- The method `workflowApi.addValidationError()`, which can be used to display messages in the Web Client, is still valid.

Overview of the Methods

The following methods are available in the `client` object (class `OperationResponseBuilder`). For more details, please refer to the ConSol CM API documentation. A mapping to the old/deprecated objects and methods is provided - this might make it easier for you to exchange the code in your scripts.

New class/method	Deprecated class/method	Note
client.failure()	PostActionType.FAILURE	Sets the OperationType = FAILURE.
client.goToCreateResource (resource)	actionScriptResultFactory.getPostAction (PostActionType.CREATE_RESOURCE, resource)	Opens the create resource page prefilled with data from `resource` object.
client.goToCreateTicket(ticket)	actionScriptResultFactory.getPostAction (PostActionType.CREATE_TICKET, ticket)	Opens the create ticket page prefilled with data from `ticket` object.
client.goToCreateTicket(ticket).withCustomer(unit)	actionScriptResultFactory.getPostAction (PostActionType.CREATE_TICKET, ticket, unit)	Opens the create ticket page prefilled with data from `ticket` and `unit` object.
client.goToCreateUnit(unit)	actionScriptResultFactory.getPostAction (PostActionType.CREATE_UNIT, unit)	Opens the create customer page prefilled with data from `unit` object.
client.goToPage("url")	actionScriptResultFactory.getPostAction (PostActionType.GOTO_PAGE, "url");	Opens the given URL.
client.goToResource(resource)	actionScriptResultFactory.getPostAction (PostActionType.GOTO_RESOURCE, resource);	Opens the resource page of the given `resource` object.
client.goToTicket(ticket)	actionScriptResultFactory.getPostAction (PostActionType.GOTO_TICKET, ticket)	Opens the ticket page of the given `ticket` object.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

New class/method	Deprecated class/method	Note
client.goToUnit(unit)	actionScriptResultFactory.getPostAction (PostActionType.GOTO_UNIT, unit)	Opens the customer page of the given `unit` object.
client.goToResource (resource).openActionForm ("actionformname")	-	Opens the given action form of the resource.
client.goToUnit(unit).openAc-tionForm("actionformname")	-	Opens the given action form of the cus-tomer.
client.goToTicket(ticket) .openActivityForm("acfpath")	actionScriptResultFactory.getPostAction (PostActionType.GOTO_TICKET, ticket, acfExecutionContext)	Opens the ACF of the current ticket.
client.rollback()	-	Rolls back the action. Optionally, an error message can be provided.
client.showDebugMessage ("messageOrLabel")	actionScriptResultFactory.getPostAction (PostActionType.FAILURE, "mes-sageOrLabel")	Shows a red message, the message text can be entered as a string or using a label.
client.showErrorMessage ("messageOrLabel")	actionScriptResultFactory.getPostAction (PostActionType.FAILURE, "mes-sageOrLabel")	Shows a red message, the message text can be entered as a string or using a label.
client.showInfoMessage("mes-sageOrLabel")	actionScriptResultFactory.getPostAction (PostActionType.SUCCESS, "mes-sageOrLabel")	Shows a green mes-sage, the message text can be entered as a string or using a label.
client.showMessage("mes-sageOrLabel")	actionScriptResultFactory.getPostAction (PostActionType.SUCCESS, "mes-sageOrLabel")	Shows a green mes-sage, the message text can be entered as a string or using a label.
client.showWarningMessage ("messageOrLabel")	actionScriptResultFactory.getPostAction (PostActionType.SUCCESS, "mes-sageOrLabel")	Shows a green mes-sage, the message text can be entered as a string or using a label.
client.success()	PostActionType.SUCCESS	Sets the Oper-ationType = SUCCESS.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

By default, the methods to show messages do not refresh the page. Therefore, any updates to the object performed within the action scripts are not yet visible when the message is displayed. You can append `withRefreshContent()` to the method to refresh the page before showing the message.

Examples

Open the Page to Create a Ticket

Use one of the following methods:

- client.goToCreateTicket(ticket)
- client.goToCreateTicket(ticket).withCustomer(unit)

The following example opens the page to create a ticket. The queue “Helpdesk” is preselected and some fields are prefilled.

```
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")
client.goToCreateTicket(ticket)
//to additionally set the main contact use withCustomer()
//client.goToCreateTicket(ticket).withCustomer(unit)
```

Code example 57: *Open the page to create a case*

Open a Ticket Page in View Mode

Use the following method:

- client.goToTicket(ticket)

The following example opens the ticket “newtic”. Instead of passing a ticket object to the method, you can also provide the ticket name or ID.

```
client.goToTicket(newtic)
```

Code example 58: *Open the ticket page in view mode*

Open a Ticket Page with ACF

Use the following method:

- client.goToTicket(ticket).openActivityForm("activity path")

The following example opens the ticket with the name “100270” and directly opens the activity form of the *Dismiss ticket* activity.

```
client.goToTicket("100270").openActivityForm("defaultScope/Service_Desk/Ticket_
 dismissed/Dismiss_ticket_")
```

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Code example 59: *Open a ticket page with an ACF*

Open the Page to Create a Customer

Use the following method:

- client.goToCreateUnit(unit)

The following example opens the page to create a new contact for the company. Some data fields are already prefilled.

```
// used for companies in end customers group to create new contacts easily
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", "CM")
myunit.set("customer.name", "Consultant")

client.goToCreateUnit(myunit)
```

Code example 60: *Company action script ("unit" is the current company) to open the page for creating a new contact*

Open a URL

Use the following method:

- client.goToPage(url)

The following example opens the URL which is saved in a company field.

```
// opens company's web site
def url = unit.get("companyData.url")

if (!url) {
   client.showErrorMessage("error.script.no.url")
} else {
   client.goToPage(url)
}
```

Code example 61: *Open a URL*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Display Result Message in the Web Client: SUCCESS, Green Messages

There are several methods to define messages with a green background:

- `client.success()`
  Displays the message “Action succeeded” with green background. Sets operation type to OperationType.SUCCESS.
- `client.showInfoMessage(string or label)`
  Displays the string or the message coded by the label with a green background.
- `client.showMessage(string or label)`
  Displays the string or the message coded by the label with a green background.
- `client.showMessage(String pMessageKey, OperationMessage.Severity.INFO)`
  Displays the string or the message coded by the label with a green background. Another severity level would display the message background in another color. In the following example, a label is used.

By default, the methods to show messages do not refresh the page. Therefore, any updates to the object performed within the action scripts are not yet visible when the message is displayed. You can append `withRefreshContent()` to the method to refresh the page before showing the message:

```
client.showMessage("controlForm.info.executionOK").withRefreshContent()
```

Display Result Message in Web Client: FAILURE, ERROR, Red Messages

There are several methods to define messages with a red background:

- `client.failure()`
  Displays the message “Action failed” with red background. Sets operation type to OperationType.FAILURE.
- `client.showErrorMessage(string or label)`
  Displays the string or the message coded by the label with a red background.
- `client.showDebugMessage(string or label)`
  Displays the string or the message coded by the label with a red background.
- `client.showMessage(String pMessageKey, OperationMessage.Severity.WARN)`
  Displays the string or the message coded by the label with a red background.

Working with Relation Actions

Relation actions are available for all three main object types and are defined in the Admin Tool. Please see the detailed explanations in the respective sections of this manual.

- **customers**
  explained in section Action Framework - Customer Actions
- **tickets**
- **resources**
  explained in section CM/Resource Pool - Resource Actions

A relation action is executed when the relation is

- created
- deleted

It is not executed if the comment of the relation has been changed.



Please note that a relation action is always executed when the object type for which the relation is defined is involved. As a consequence, two actions will be executed in case the source as well as the target object belong to a type with a relation action!

In scripts which define the relation actions, the following objects are available:

- Relation between to customers:
  - `unit` object
  - `relation` object (UnitRelation)
- Relation between a customer and a ticket:
  - `unit` object
  - `ticket` object
  - `role` object (ContactTicketRole)
- Relation between a customer and a resource:
  - `resource` object
  - `unit` object
  - `relation` object (ResourceUnitRelation)
- Relation between a resource and a ticket:
  - `resource` object
  - `ticket` object
  - `relation` object (ResourceTicketRelation)
- Relation between two resources:
  - `resource` object
  - `relation` object (ResourceResourceRelation)

It is possible to distinguish between operations where a relation has been added and operations where a relation has been removed. Use the context parameter `actionType` for this purpose. It can be used in the condition script, e.g., to control if the action script should be executed or not, or in the action script itself. The `actionType` can be either of two values:

- ADD
- REMOVE

Example for a Relation Action

In the following example, the person who is the contact person for an SLA should receive an email whenever a new relation for this SLA to a customer has been established or has been deleted.

Perform the following steps to create the relation action:

1. Define a relation, in this example the *SLA_to_Company_Relation*, with source = resource type *SLAs* and target = *company*, several customer groups.
2. Write the resource action script, in this example: `SLA_CompanyRelationAction.groovy`.
3. Define a resource action (in this example: *SLA_CompanyRelationAction*) of type “Relation” which uses the script as action script.

4. Assign the resource action to a resource type, in the example to the type *SLAs*.
5. Test the desired use case in the Web Client.

Example relation action script `SLA_CompanyRelationAction.groovy`:

```
import com.consol.cmas.common.model.mail.MailSendHolder

def myUnit = relation.targetUnit
def myUnitDef = relation.targetUnit.definition.name
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)
def contPersMail = resource.get("SLA_Fields_basic.responsible_person_email")
if (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'
}
```

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Code example 62: *Relation action script to send an email when a resource-company relation has been created or deleted*

Working With the Data of Control Forms

Control Form-Specific Objects

Two specific objects are available in scripts which deal with Control Forms (see section Control Forms):

- Action scripts (of the action which is executed when the control form has been filled in)
- Control Form condition scripts
- Control Form prefill scripts

The two objects are:

- **formFields** (Map<FieldKey, AbstractField>)
  The map entries represent the data fields and the respective values of the form. In Control Form condition scripts, the `formFields` are only available, after the Control Form has been filled for the first time.
- **controlForm** (class `TicketActionControlForm`, `UnitActionControlForm` or `ResourceActionControlForm`, depending on the context)
  Represents the action form. It may be null if no action form is used. In Control Form condition scripts, the `controlForm` object is only available, after the Control Form has been filled for the first time.

The class `ControlFormService` includes several methods to retrieve Control Forms.

Auto-binding

The auto-binding setting defines whether changes to the values in the form also affect the original object.

Auto-binding disabled

Add the following line to the script to disable auto-binding:

```
def isAutoBindingEnabled() {
   return false;
}
```

The values of the form are available only within the form. They can be retrieved using the `formFields` object. The original object, i.e., the ticket, customer or resource for which the form is opened, is **not** modified upon saving the form.

Auto-binding enabled

Add the following line to the script to enable auto-binding:

```
def isAutoBindingEnabled() {
   return true;
}
```

The values of the form are used to manipulate the values of the original object, because there is a binding between the data field in the form and the data field in the object. The form values are also available in the `formFields` object, but the original object, i.e., the ticket, customer or resource for which the form is opened, is also modified upon saving the form.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.6.4 Control Forms

F.6.4.1 Introduction

What Are Control Forms?

Control Forms offer the possibility to collect data, before a certain action is executed. Just like ACFs (Activity Control Forms, see section Tab Activity Form Data) for workflow activities, Control Forms collect data for actions of the ConSol CM Action Framework, i.e., for the following action types:

- Ticket search actions
- Customer actions
    - Manual customer actions
    - Customer search actions
- Resource actions
    - Manual resource actions
    - Resource search actions

The Control Form is opened and has to be filled in, before the action, i.e., the action script, is executed. Please read section Scripts for the Action Framework for a detailed introduction to scripts for actions of the Action Framework.

Like ACFs, Control Forms can be fine-tuned using scripts. Each Control Form can have the following scripts. They are both optional - a Control Form can also work perfectly without them.

- Form condition script
  Only when the condition is met (“true”), the Control Form is displayed. See section Form Condition Script, Optional.
- Form prefill script
  Can fill in values in the Control Form, i.e., when the form is displayed, the respective fields are filled out. See section Form Prefill Script, Optional.

Use cases like the following can be covered using Control Forms:

- **Company action**
  Enter new contact data into the form - create a new contact for the company with one click.
- **Ticket search action**
  Enter the new contract expiry date for each asset in a list of IT assets, found in a search operation, with one click.
- **Company search action**
  Enter all dates concerning an upcoming conference in each company of a list of companies you want to invite.
- **Contact search action**
  Enter only the name of the email template you want to use and send an email to a list of recipients:
    - inform job candidates about the start date of the assessment center
    - inform all employees who have a certain type of software license about the necessity to update the license

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

When a Control Form has been displayed and the respective data have been filled in or modified, the action script which implements the ticket/customer/resource action is executed. Specific methods are available which deal with the Control Form data. This is explained in detail in section Working With the Data of Control Forms.

See the following example to get an impression how Control Forms work.

First Impression of Control Forms - a Control Form of a Resource Action

The resource page of a HP printer is open and the engineer executes the resource activity *Create maintenance ticket for HP printer*. A form is opened where the deadline and priority for the new ticket have to be entered. Only when the engineer has clicked *Save and continue*, the form data is saved and the resource action is executed. In the current example, it would not have been possible to enter nothing, since the form contains mandatory fields (marked by a red asterisk).

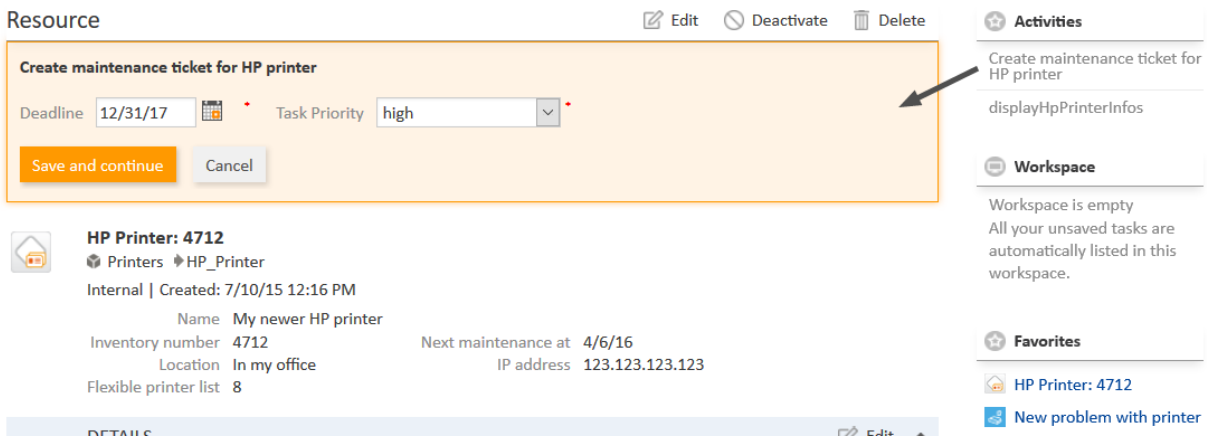


Figure 342: *ConSol CM Web Client - Control Form of a resource activity*

F.6.4.2 Defining Control Forms Using the Admin Tool

All components of Control Forms are defined in the Admin Tool.

A Control Form comprises at least one and can comprise three components:

- The Control Form Itself, Mandatory
- Form Condition Script, Optional
- Form Prefill Script, Optional

The Control Form Itself, Mandatory

It depends on the type of action to which the form should be assigned where in the Admin Tool you have to define the Control Form:

- For ticket search action:
  Navigation group *Tickets*, navigation item *Search Actions*, Tab *Control Forms*
- For customer/unit action:
  Navigation group *Customers*, navigation item *Actions*, Tab *Control Forms*
- For resource action:
  Navigation group *Resources*, navigation item *Actions*, Tab *Control Forms*

In the following example , we will explain a customer/unit action. The principle applies to all Control Forms.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Perform the following steps:

- Define the Data Fields for the New Control Form
- Assign the Control Form to the Action Where It Should Be Opened
- Test the New Control Form

Define the Data Fields for the New Control Form

In order to create a new Control form for a customer action, open the navigation group *Customers*, navigation item *Actions*, Tab *Control Forms*. Click *Add* and select the data fields which should be part of the new Control Form in the *Available* section and move them to the *Assigned* section.

Please note that, unlike in an ACF for tickets where only ticket fields are available, in a Control Form, all types of data fields are available:

- ticket fields
- customer fields:
  - company fields
  - contact fields
- resource fields

This implies that a Control Form can comprise data other than the data which belongs to the object of the action! For example: in a contact action, not only data of this contact, like email or phone, can be requested, but also every other data field like data of the ticket or of the company. Two things are of major importance here:

1. The action script decides what happens to the data of the form! So the developer of this script is responsible for the data processing!
2. The behavior of the action script concerning “auto-binding” is crucial! See section Auto-binding about this!

In the drop-down menus which work as filters above the list of data fields, you can select which type of fields you want to have displayed:

- ticket
- unit
- resource

In the second drop-down menu/filter, you can select the data field group of which the fields should be displayed in the *Available* list. If you have selected a certain field type in the first drop-down menu/-filter, only the matching field groups are displayed, e.g., only fields of the customer field group *Reseller-CompanyData* (in our example).

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

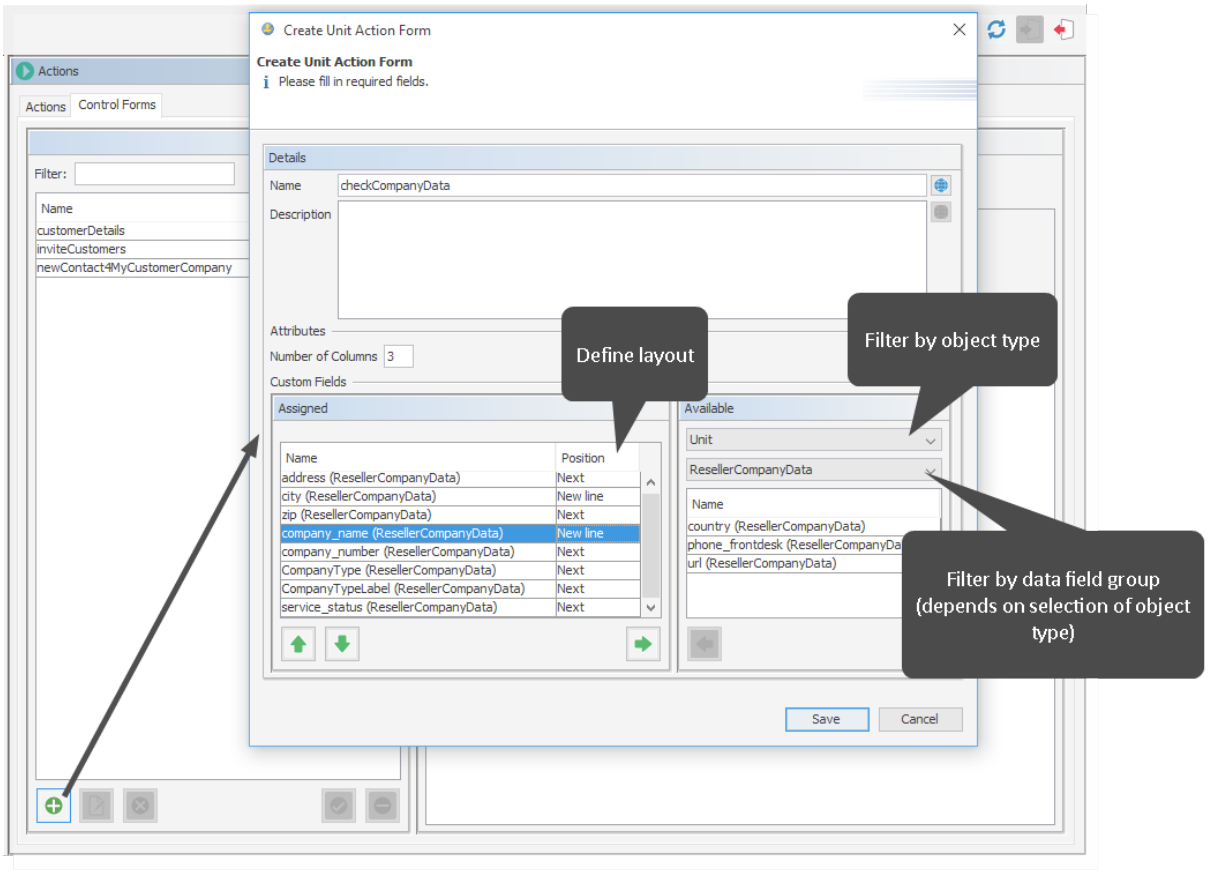


Figure 343: *ConSol CM Admin Tool - Customers, Actions: Creating a new Control Form for a customer action*

In the example, some company fields of the customer field group *ResellerCompanyData* have been selected.

Assign the Control Form to the Action Where It Should Be Opened

When the new Control Form has been defined, you have to assign the form to an action of the Action Framework. In our example, to a company action.

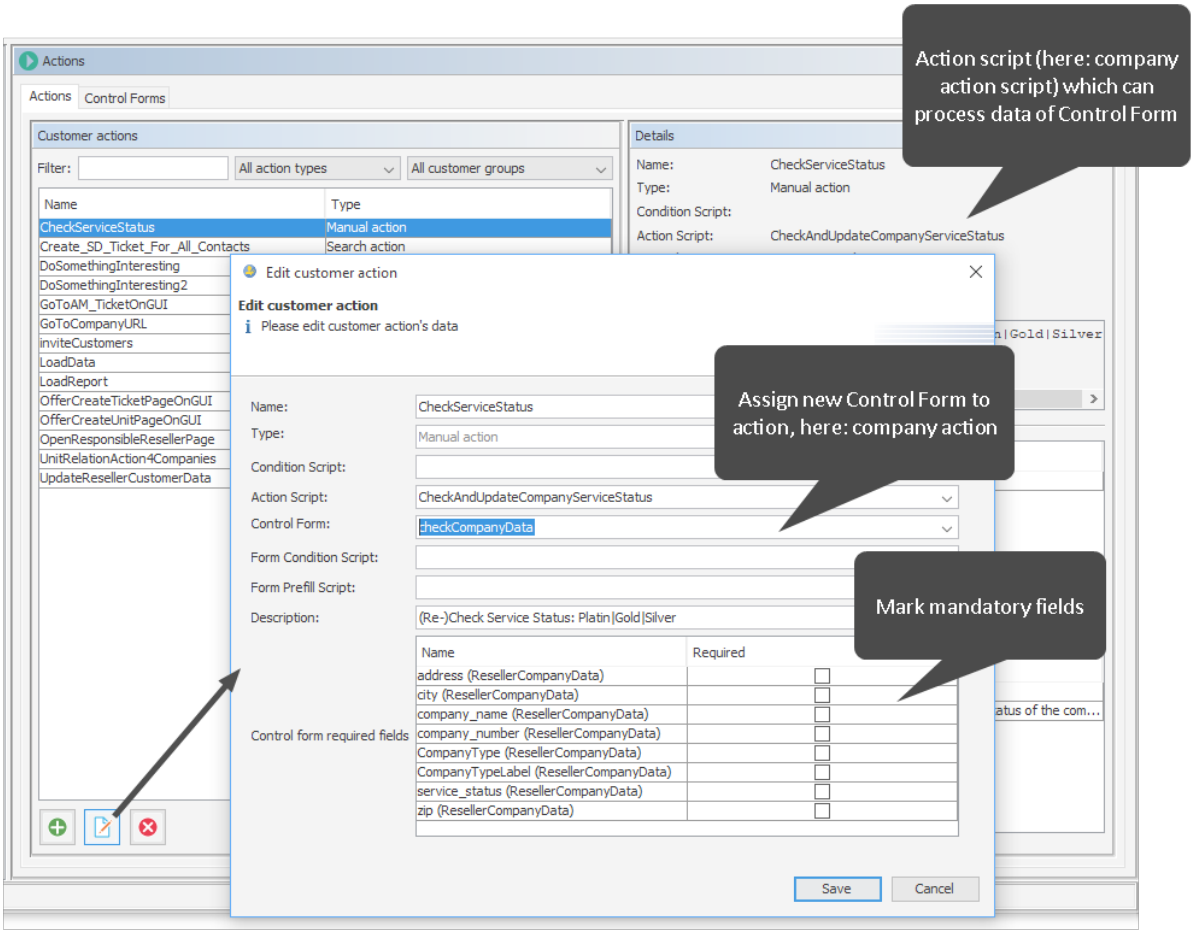


Figure 344: *ConSol CM Admin Tool - Customers, Actions: Assigning a new Control Form to a company action*

Test the New Control Form

Open a company page in the customer group where the manual action has been assigned and execute the company action. The Control Form is opened and can be modified.

In the current example, the fields are already filled in, because the company fields have been filled already and the data is simply displayed. It could be modified within the form.

> Please note the different behavior of the system concerning the saving of the form data depending on the “auto-binding” settings! See section Auto-binding about this!

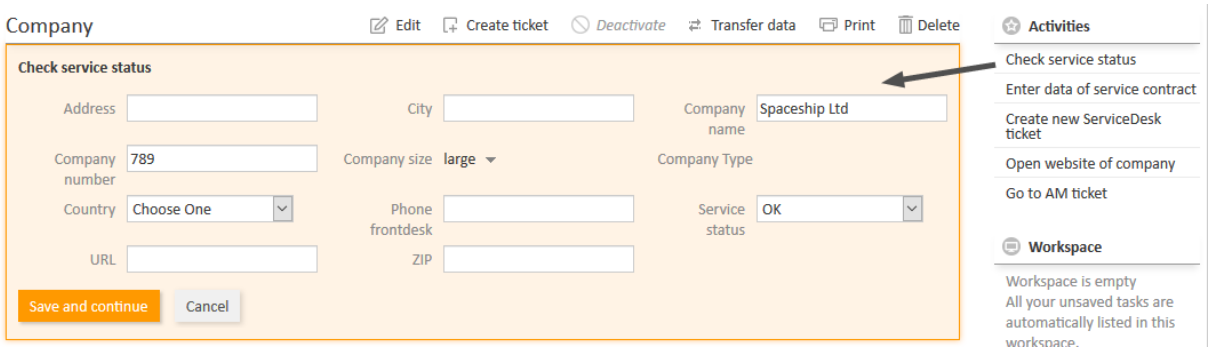


Figure 345: *ConSol CM Web Client - Control Form of a company action*

Form Condition Script, Optional

In the Web Client, a Control Form is only displayed if

- no condition script is present

OR

- a condition script is present and has returned “true”

The form condition script is stored in the Admin Tool, in the scripts section and has to be of type *Control Form Condition*. It has to return either “true” or “false”. In order to implement a form condition script, perform the following steps:

1. Write the Control Form condition script.
2. Assign the script to the action as *Form Condition Script*.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

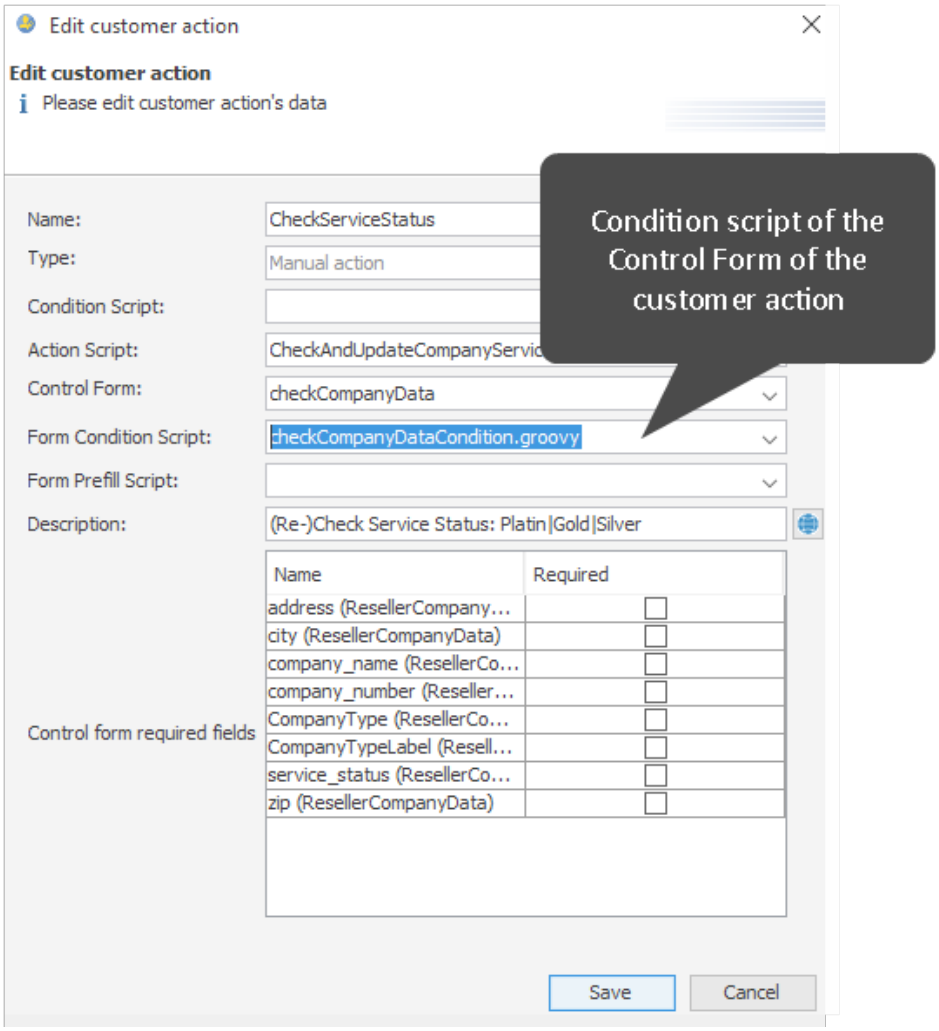


Figure 346: *ConSol CM Admin Tool - Customers, Actions: Assigning a condition script to a Control Form of a company action*

If the script returns “true”, the Control Form is displayed, if it returns “false”, the Control Form is not displayed, but the action is executed.

The following example shows a Control Form condition script which checks if there are empty fields in the resource. Only then the form is displayed in order to gather the missing data. If all fields are already filled, no form is displayed.

```
if(resource.get("product.quantity") == null || resource.get("product.price") ==
 null) {
   return true
}
return false
```

Code example 63: *Control Form condition script which checks if fields are filled*

Form Prefill Script, Optional

If a form should be opened with some fields already filled in with specific values, you have to assign a form prefill script to the respective customer/ticket/resource action. In the script the fields can be filled.

For example, the following Control Form prefill script could be used to make sure the value for service status is set to a current value by an engineer.

```
company.set("ResellerCompanyData.service_status","unknown")
```

In the Web Client, the respective field is pre-filled with the given value:

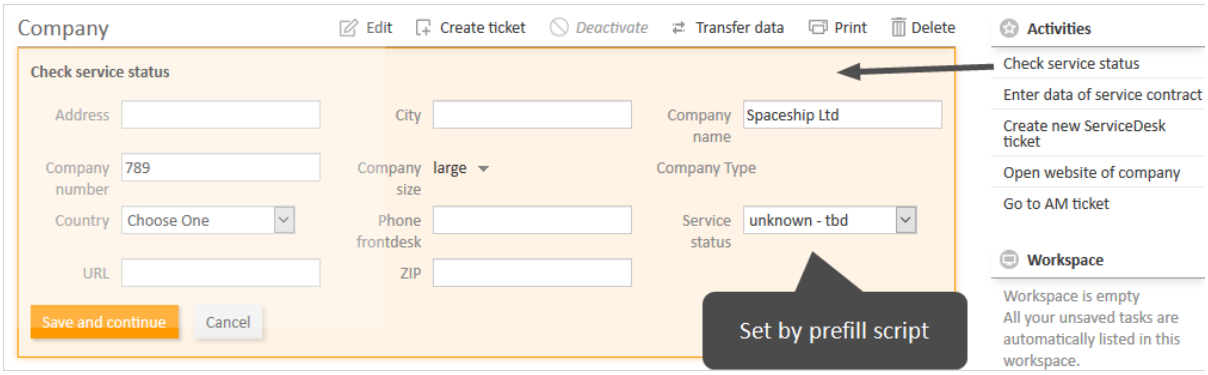


Figure 347: *ConSol CM Web Client - Customer action and Control Form where one field has been pre-filled by a script*

> Please note the different behavior of the system concerning the saving of the form data depending on the “auto-binding” settings! See section Auto-binding about this!

When the form is opened:

- It depends on the auto-binding definition (see info box!) if the fields which belong to the context object are filled or not.
- Values of the form prefill script overwrite the original values of the object.

When the form is saved:

The values which are set within a form prefill script are only used to fill out the form. The data is not persisted. If the data is really saved in the objects, depends on the action script, namely on

- the definition of the auto-binding (see info box!)
- the usage of the content of the `formFields` (see section Working With the Data of Control Forms about `formFields`)

F.7 Email Configuration

ConSol CM includes a sophisticated email interface. CM fetches emails from one or more mail servers and can send emails. Incoming as well as outgoing emails are included in the respective tickets, hence you always have access to all information regarding a case.

Basic CM email functionalities are explained in the following sections:

- Email
- Email Backups

In case your company wants to use email encryption, you can employ server and/or client certificates, see section:

- Email Encryption

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.7.1 Email

F.7.1.1 Introduction to Emails in ConSol CM

Sending and receiving emails is one of the core functionalities of ***ConSol CM***. The application interacts with one or more mail servers to fetch emails and send emails.

The following sections provide information about the following email-related topics:

- Email Functionalities in ConSol CM (sending and receiving emails, email duplication)
- Email Configuration Details
- Email Configuration Using the Admin Tool

You can find more information about email management during system operation in the *ConSol CM Operations Manual*, section *Email*.

F.7.1.2 Email Functionalities in ConSol CM

This section contains a short introduction on emailing with ConSol CM. Sending and receiving emails is a core functionality of the application.

Sending Emails from ConSol CM

Manual Emails

Emails can be sent manually by an engineer or automatically by the system. Manual emails are sent using the Ticket Email Editor. In most systems, by default, the ticket's main customer is the receiver of the email, but the engineer can select or type any other email address. The system-wide default value can be changed by an administrator using Page Customization, using the attribute mailToSelection in the Page Customization section. Furthermore, the engineer can use email templates and/or quote ticket text. Please see the *ConSol CM User Manual* for a detailed introduction about working with the Ticket Email Editor.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

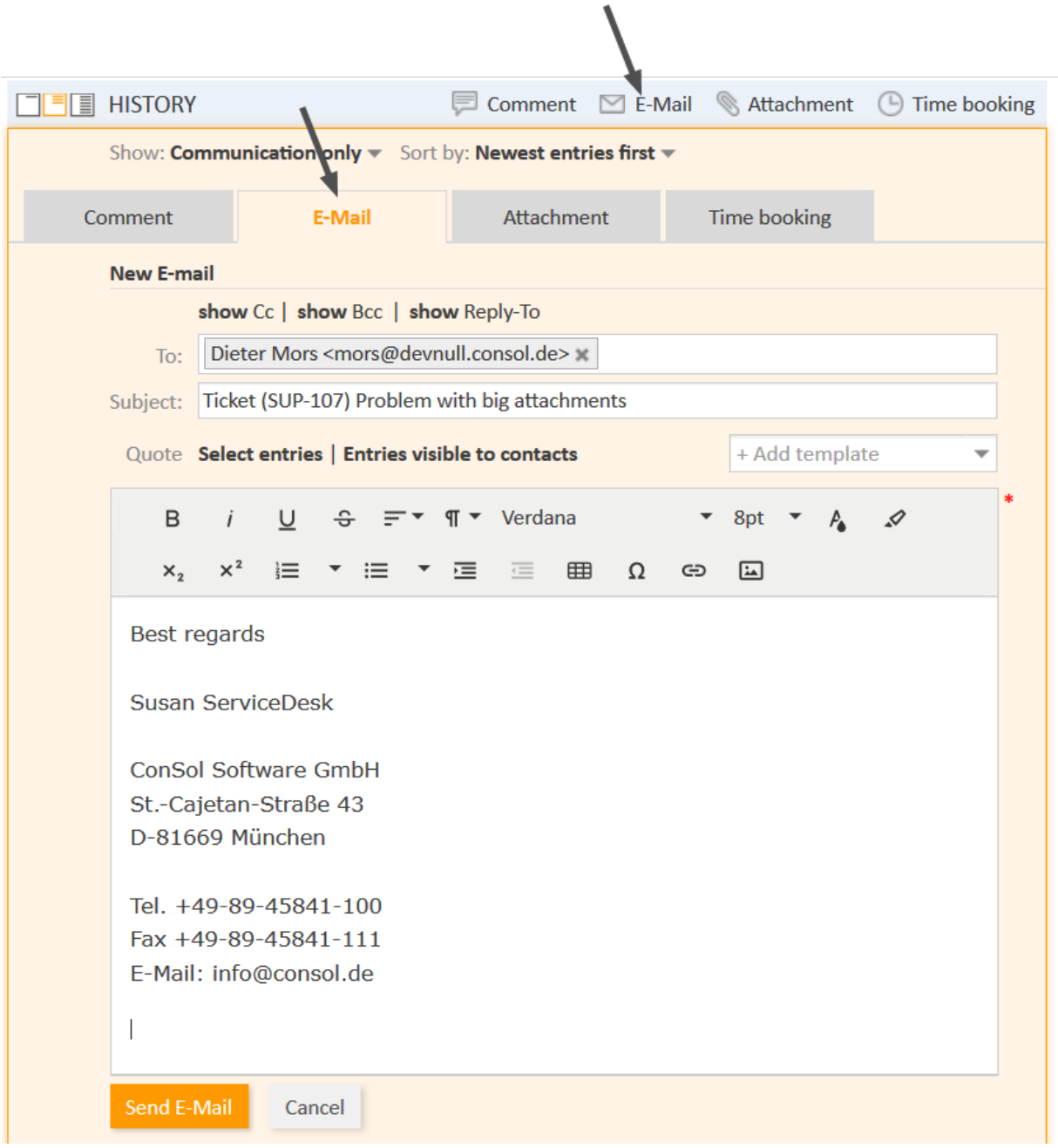


Figure 348: *ConSol CM Web Client - Ticket Email Editor*

Automatic Emails

Automatic emails may be sent by ConSol CM in situations like the following:

- Initiated by the workflow engine, e.g.,
  - when the engineer to whom the ticket is assigned should be reminded to attend to the ticket.
  - when customers should receive an automatic confirmation that a ticket has been opened for them.

    - when customers should receive an automatic confirmation that a ticket of theirs has been closed.
    - when a supervisor or approver should receive a message that a new case has to be approved.

  In any workflow activity, an email can be sent to any valid email address. Please see the *ConSol CM Process Designer Manual* for a detailed explanation of the methods to use.
- Initiated by the system in case of an error or for a success message, e.g.,
    - system error
    - email error
    - DWH synchronization (error or success)
      Usually, those emails are sent to the ConSol CM administrator. However, for most special error cases a special receiver email address can be configured using system properties. Please see section System Properties for details.
- Initiated by the ConSol CM system to remind engineers
    - When an engineer receives a ticket or a ticket is retrieved from the engineer, an email can be sent to this engineer. This can be configured on a per-queue basis, as described in section Queues.

Receiving Emails with ConSol CM

The ConSol CM system can fetch emails from one or more mailboxes (= email accounts) on one or more mail servers. The mailboxes are configured in the Admin Tool (Email Functionalities in ConSol CM). Please keep in mind that ConSol CM works with mailboxes here. Each of the mailboxes can be reached by at least one email address. In certain cases, one mailbox might be used for more than one email address. This can be significant when writing Scripts of Type Email.

As far as the mail server is concerned, ConSol CM is just a regular email client fetching emails using a standard mail protocol: IMAP(S) or POP3(S). Depending on the mail server configuration and on the ConSol CM system property cmas-nimh, mailbox.default.task.delete.read.messages, the emails are deleted from the mailbox on the mail server after ConSol CM has picked them up. The default setting is **mails are not deleted after pick-up**.



If you do not want ConSol CM to delete emails from the mail server, please make sure to monitor the mailbox(es) to avoid a data overflow and server or performance problems.

All incoming emails are first stored in an incoming email pool in ConSol CM and are then processed in a chain of email scripts. Please see section Scripts of Type Email for a detailed explanation of those scripts. When an email cannot be processed, the administrator will receive a notification email. The unprocessed email is listed under Email Backups.

There are different possibilities concerning the default system behavior for an incoming email:

- The subject of the email does not contain any ticket number with a valid syntax (i.e., it does not contain the pattern which is defined as regular expression (RegEx) for the ticket subject):
  A new ticket is created.
- The subject of the email does contain a ticket number with a valid syntax (RegEx) and the ticket is still open:
  The email is attached to the existing ticket.

- The subject of the email does contain a ticket number with a valid syntax (RegEx), but the ticket is closed:
  A new ticket is created and a reference to the old ticket is established.

By modifying the email scripts (see section Scripts of Type Email), the default system behavior can be changed. However, this can corrupt core functionalities of the system and should not be done or only done by very experienced ConSol consultants!

Email Duplication in the ConSol CM Web Client

Please see explanations on the *Page Customization* page at showCloneOption and appendOrReplaceOnClone.

F.7.1.3 Email Configuration Using the Admin Tool

This section explains the navigation item *Email* in navigation group *Email* in the Admin Tool.

IMPORTANT INFORMATION

In ConSol CM versions prior to 6.11, there were two modes to receive incoming emails:

- **Mule/ESB** - This mode is deprecated in CM versions 6.11 and up. Therefore, it is no longer treated in ConSol CM manuals. If you run an older CM system with Mule/ESB, please refer to older manuals. All ConSol CM 6.11 systems must run in NIMH mode. Please refer to older Administrator manuals to learn how to switch from Mule/ESB to NIMH.
- **NIMH** - *New Incoming Mail Handler*, available since version 6.9.4. This is the only available incoming email mode in ConSol CM version 6.11 and up.

The active email mode is set in the system property cmas-core-server, nimh.enabled:

- **true** - NIMH mode
- **false** - Mule/ESB mode

The sending of emails, i.e., the SMTP server configuration, is not influenced by the incoming email mode.

General Email Configuration (Navigation Item *Email*)

In this navigation item you can set the parameters for the email connection.

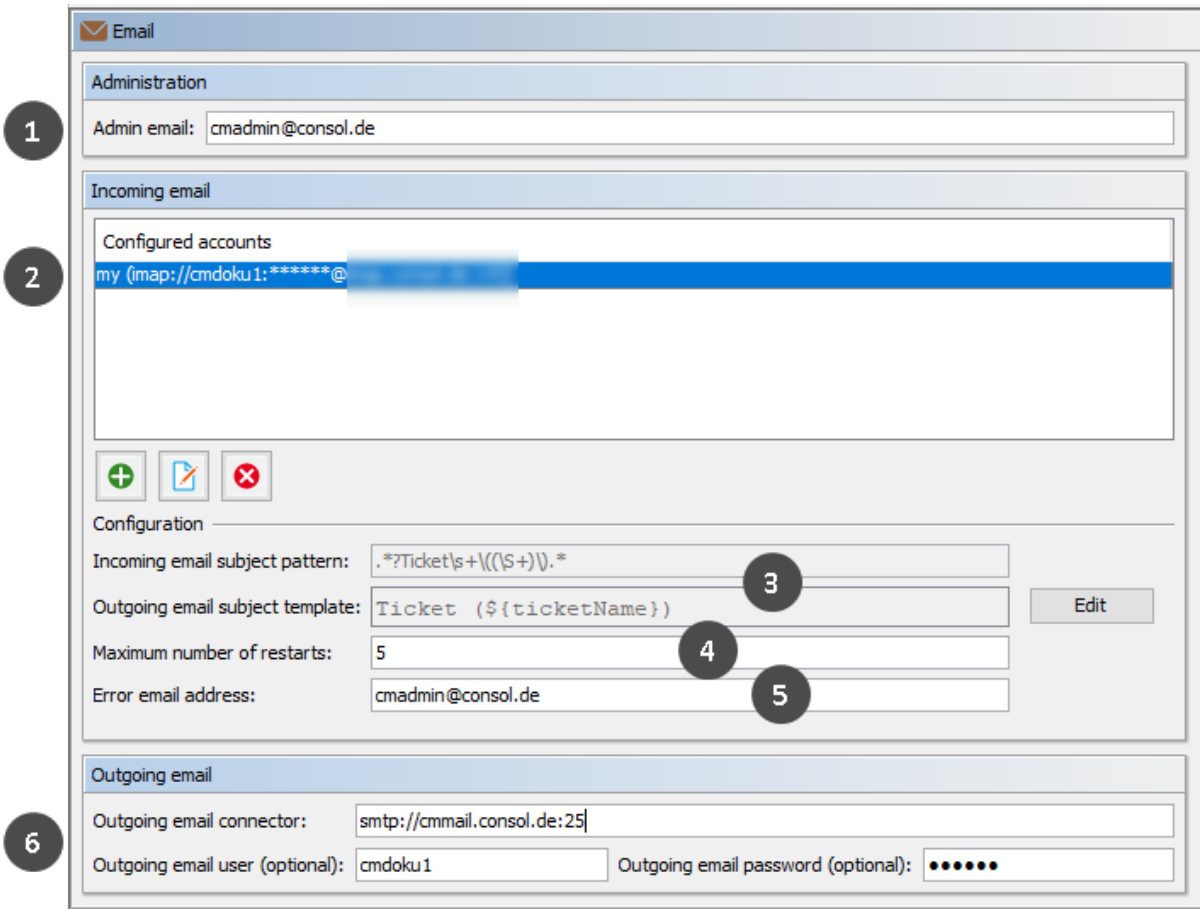


Figure 349: *ConSol CM Admin Tool - Email*

The *Email* screen shows the following information:

- General admin email address for error messages (1)
- Email accounts (not email addresses!) on one or more email servers where ConSol CM will fetch emails (2)
- Configuration of RegEx pattern for the recognition of existing tickets (incoming emails) (3)
- Number of restarts/retries after an error with email delivery (4)
- Email address for email-related error messages (5)
- Address of outgoing email server (SMTP) with user name and password (6)

Administration

Admin email:

Enter the email address which should receive general messages or warnings from the system. Using the GUI, you can enter one admin email address. Multiple addresses separated by commas are possible, but they have to be entered directly in the navigation item System Properties. Use the system property cmas-core-security, admin.email. The total number of characters (for both GUI and property) must not exceed 72. If there are many recipients, we recommend using a mailing list on the mail server system.

Incoming Email

The configuration of incoming email is divided into two areas:

- **Configured accounts:**
  Here you can use a pop-up window to add or edit accounts from which emails are retrieved. The connection to the mailbox is checked during set-up, so it is not possible to configure an account that cannot be used when the system is in operation (provided the mail server has not changed etc.). The value(s) are saved in the system properties in the module `cmas-nimh`. Please see the email properties section in System Properties for detailed information. Required values are:
  - **Identifier**
    The name of the mailbox. If a name is provided for the mailbox, the mailbox details are displayed in parenthesis after the mailbox name in the list of email accounts. The identifier is optional.
  - **Protocol**
    The protocol used to retrieve emails from the server. Supported protocols are IMAP4, IMAP4s, POP3, and POP3s. Please keep in mind that ConSol CM behaves like a regular email client. When the secure protocol version is used, the corresponding certificate is required! This has to be stored in the security store of the application server.
  - **Server name**
    The DNS-resolvable name or IP address of the mail server.
  - **Port**
    The port on the mail server where the mail daemon/service is listening.
  - **User name**
    The user name of the email account.
  - **Password**
    The password of the email account.

    > ⚠ Please keep in mind that one email account can have more than one email address. So here, you are dealing with the account name, i.e., with the mailbox. When you edit the Admin Tool script(s) that process the incoming emails, it might be required to use the email address. The email address is also required when you configure the Reply-To address, the From address, and queue-specific email addresses! So be sure to use the correct parameter: mailbox or email address!

- **Configuration:**
  - **Incoming email subject pattern:**
    Describes the elements that the subject of an incoming email has to contain in order to assign this email to a certain ticket. The pattern is defined in form of a regular expression (RegEx).
    Example: *.*?Ticket\s+\((\S+)\).** would match every subject line that contains *Ticket (<Ticket number>)*.
  - **Outgoing email subject template:**
    Describes the pattern which is used to create the ticket ID in the subject of an outgoing email. The template should be matchable by the incoming email subject pattern. Via the *Edit* button on the right you can modify the incoming email subject pattern and

outgoing email subject template and verify if they match.
Example: *Ticket (${ticketName})* would match the example RegEx above.

> ⚠ You can check if the pattern for the incoming email subject pattern and for the outgoing email subject template match by using the *Edit* button and the editor that is opened. Please make sure that the email subject has been set correctly at **all** locations, e.g., also in all workflow scripts and Admin Tool scripts!

- **Maximum number of restarts:**
  Shows the maximum number of restarts after an error when ConSol CM fetches emails. Valid for all mail pollers.
- **Error email address:**
  Email address to which messages and warnings of the mail sub-system are sent. This is usually the same as the general administrator address.

For the configuration of incoming email you might also want to check the email-related system properties, see System Properties. Particularly, the polling interval (the time interval for fetching emails from the mail server, system property mailbox.default.task.interval.seconds (module `cmas-nimh`) or the respective property of another mailbox (e.g. `mailbox.1.task.interval.seconds`) might be of interest.

Outgoing Email

The connection data for outgoing emails are set in the *Outgoing email* section. Please fill in the fields as follows:

- **Outgoing email connector**

  The protocol can be “smtp” or “smtps”. In addition you need to provide the IP/URL and port of the email server. The user name and password can be specified in two ways:

  - Enter the protocol, URL and port in this field using the following syntax and specify the user name and password in the respective fields below.

    ```
    <protocol>://<IP address or URL of mail server>:<port>
    ```

  - Enter everything in this field using the following syntax.

    ```
    <protocol>://<user>:<password>@<IP address or URL of the mail
     server>:<port>
    ```

- **Outgoing email user (optional)**

  Enter the user name of the email account.

- **Outgoing email password (optional)**

  Enter the password of the email account.

Please note: If user name and/or password which have to be provided contain the character “@”, the “@” has to be coded as %40.

To enable SMTPS for outgoing emails, set the system property cmas-core-server, mail.smtp.tls.enabled to “true”.

F.7.1.4 Email Configuration Details

This section describes emailing in a single-server environment. Please contact the ConSol CM support or your ConSol CM consultant for information about emailing in a cluster.

IMPORTANT INFORMATION

In ConSol CM versions prior to 6.11, there were two modes to receive incoming emails:

- **Mule/ESB** - This mode is deprecated in CM versions 6.11 and up. Therefore, it is no longer treated in ConSol CM manuals. If you run an older CM system with Mule/ESB, please refer to older manuals. All ConSol CM 6.11 systems must run in NIMH mode. Please refer to older Administrator manuals to learn how to switch from Mule/ESB to NIMH.
- **NIMH** - *New Incoming Mail Handler*, available since version 6.9.4. This is the only available incoming email mode in ConSol CM version 6.11 and up.

The active email mode is set in the system property cmas-core-server, nimh.enabled:

- **true** - NIMH mode
- **false** - Mule/ESB mode

The sending of emails, i.e., the SMTP server configuration, is not influenced by the incoming email mode.

Email Configuration Using NIMH

NIMH, the ***New Incoming Mail Handler***, is a proprietary module of ConSol CM. The following picture provides an overview of all components.

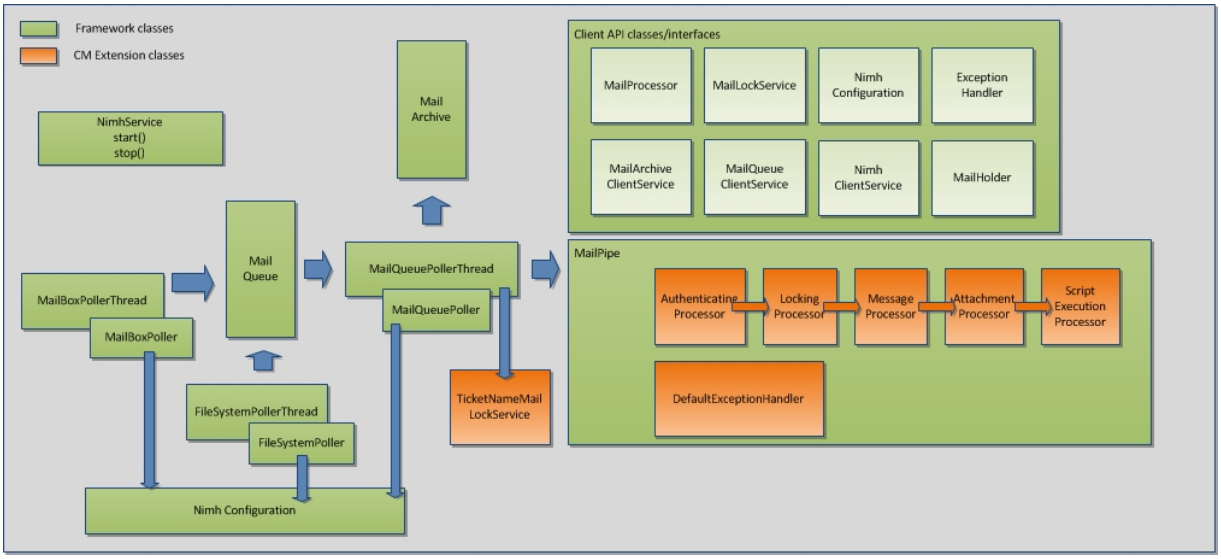


Figure 350: *NIMH components*

When NIMH is enabled, the following apply:

- NIMH runs as a (single) ConSol CM service, see section CM Services.
- Emails from mailboxes on a mail server are retrieved by ConSol CM using the MailBoxPoller.

- Emails can also be fetched from a data directory using the FileSystemPoller.
- All ConSol CM emails are stored in the ConSol CM database (nothing is stored in the file system).
- The MailQueuePoller retrieves the emails from the database and forwards them to the core ConSol CM system where the emails run through the email script pipeline.
- Emails which could not be processed are stored in a separate database table and can be re-sent to the ConSol CM system. See section Management of Email Backups with NIMH.
- NIMH-specific system properties are used:
    - General NIMH properties
    - Default mailbox properties which are used when a property is not set as a mailbox-specific property
    - Mailbox-specific properties for each mailbox which has to be retrieved
    - FileSystemPoller properties
    - MailQueuePoller properties

The following figure provides an example of NIMH properties. Please also refer to the detailed explanation of NIMH system properties in List of System Properties by Area).

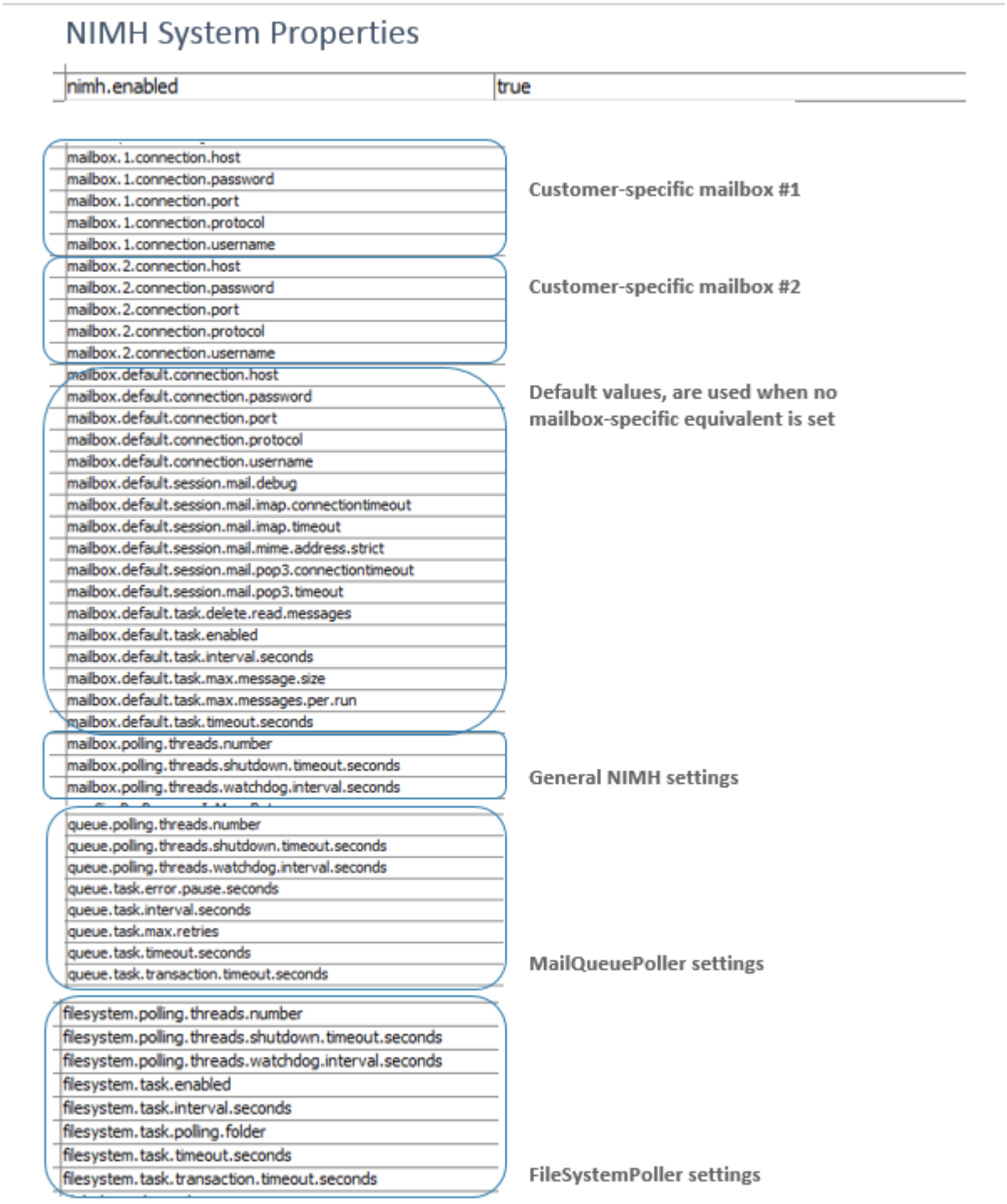


Figure 351: *System properties for NIMH, 1*

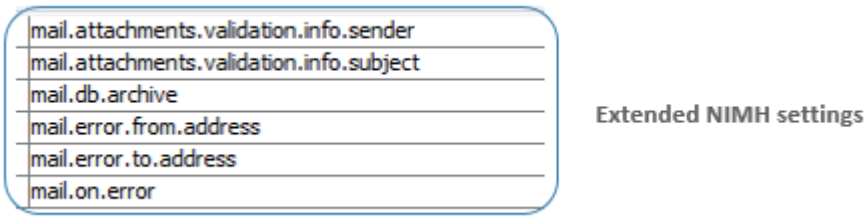


Figure 352: *System properties for NIMH, 2*

F.7.2 Email Backups

F.7.2.1 Introduction

Incoming emails which could not be processed are stored in the CM database.

You as an administrator can then try to re-send the emails to the system manually. The emails stored here can also be deleted, e.g., spam emails.

IMPORTANT INFORMATION

In ConSol CM versions prior to 6.11, there were two modes to receive incoming emails:

- **Mule/ESB** - This mode is deprecated in CM versions 6.11 and up. Therefore, it is no longer treated in ConSol CM manuals. If you run an older CM system with Mule/ESB, please refer to older manuals. All ConSol CM 6.11 systems must run in NIMH mode. Please refer to older Administrator manuals to learn how to switch from Mule/ESB to NIMH.
- **NIMH** - *New Incoming Mail Handler*, available since version 6.9.4. This is the only available incoming email mode in ConSol CM version 6.11 and up.

The active email mode is set in the system property cmas-core-server, nimh.enabled:

- **true** - NIMH mode
- **false** - Mule/ESB mode

The sending of emails, i.e., the SMTP server configuration, is not influenced by the incoming email mode.

If an email which could not be processed cannot be saved in the ConSol CM database either, e.g., because of its size, it is saved in the file system instead. The folder where these emails are stored is located in the ConSol CM data directory inside the `mail` folder. This is configured in the system property cmas-nimh, mailbox.polling.unstorable.backup.folder.

Emails which are saved to the file system are not shown in the navigation item *Email Backups* of the Admin Tool. Instead, a notification is sent to the administrator.

F.7.2.2 Email Backups in the Admin Tool

The emails which could not be processed are listed in the navigation item *Email Backups* in the navigation group *Email*.

Usually, when an email cannot be processed, this is caused by one of the following reasons:

- One of the email scripts (see section Scripts of Type Email) has a bug and therefore the email which has been fetched from the mail server cannot be processed further in CM.
- The attachment of the email is too large. For NIMH, the size limit is defined by the system property cmas-nimh, mailbox.default.task.max.message.size or by a mailbox-specific value, e.g., `cmas-nimh`, `mailbox.1.task.max.message.size.`

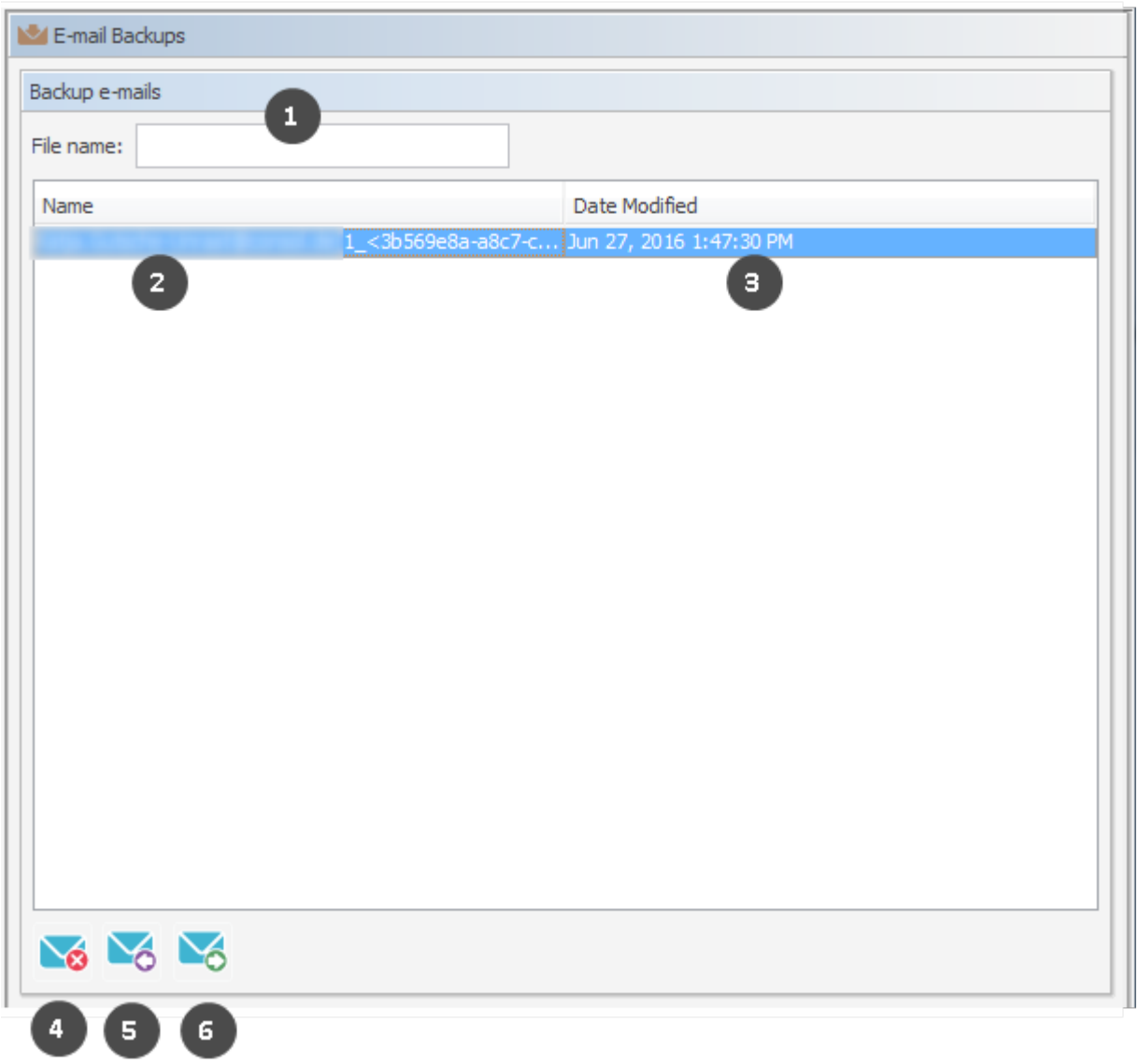


Figure 353: *ConSol CM Admin Tool - Email, Email Backups*

The list panel for unparsable emails contains the following elements:

- **File name (1)**
  This field provides a filter. When you enter the name or part of the name of email files, only the matching file names will be displayed in the list.
- **Name (2)**
  The name of the email file (usually with an .eml extension).
- **Date modified (3)**
  The last date when the file was modified. Usually the date when the email was stored on the ConSol CM server.

In an error-free ConSol CM system, the list panel for unparsable emails should be empty. If there are emails listed, an error has occurred in the processing of those incoming emails. Please see section Scripts of Type Email for a detailed explanation of the processing pipeline.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

You can perform the following actions with email backups:

- **Delete (4)**
  To delete an email from the list, select the list entry and click *Delete*. Please keep in mind that the information will be lost! It will not be saved or transferred to ConSol CM in any way!
- **Re-Send (Import) (5)**
  You can also re-send (*import*) the email to the processing pipeline (e.g., when a script was not working correctly and has been fixed) by selecting it in the list and by clicking *Resend*.
- **Export to file system (6)**
  You can export an email as .eml file to the file system of the machine where the Admin Tool is running (e.g., for further analysis of an error-prone email).

F.7.2.3 Management of Email Backups with NIMH

Incoming emails which could not be processed are stored as .eml files in the ConSol CM database table `cmas_nimh_archived_mail`. The emails which are saved in this table and have the value 0 in the `processed_success` column are displayed in the email backups.

The system behavior concerning email backups with NIMH is controlled by the system property cmas-nimh-extension, mail.db.archive.

If the property is set to “true”, all incoming emails are archived in the database. If the property is set to “false”, the emails which can be processed are deleted after processing. The emails which cannot be processed will be stored in the database as long as they have neither been processed nor deleted (which can be achieved by reprocessing or deleting such emails, see section above). An email database entry will be deleted automatically after being successfully processed.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.7.3 Email Encryption

F.7.3.1 Introduction

Due to security policies, it might be required to encrypt email traffic (including the emails which are sent and received by the ConSol CM installation) using standard S/MIME encryption. If the topic is new to you, you might want to read some articles about it, e.g. the Public-key cryptography article in Wikipedia.

In order to enable the use of encrypted emails with ConSol CM, you first have to enable the email encryption in the system:

1. **Mandatory:**
   Set the system property cmas-core-server, mail.encryption to “true”. It is set to “false” as default value. This is the basic configuration to enable email encryption for the entire system.
2. **Optional:**
   Set the page customization attribute mailEncryptionAvailable to “true”. This activates an option in the Web Client to choose whether an email should be encrypted.

General Explanation about Email Encryption in ConSol CM

There are two types of certificates:

- **Server certificates, mainly used for incoming emails**
  The public key (in the certificate) and the private key of the receiving email address can be manually imported into the CM system from a PKCS12 file (.p12 or .pfx). If the certificate is password-protected, the administrator has to provide this password when the certificate is imported.
- **Client certificates, used for outgoing emails**
  For outgoing emails, the certificates (public keys) of the receiving email addresses are required. These certificates can be imported into the CM system in two ways:
  - **Manually**
    By selecting (uploading) the respective X.509 file (.cer or .crt)
  - **Automatically**
    From an LDAP repository where it is stored in the correct format (i.e. in the same format which is required for the file import). This automatic retrieval of the requested certificate can be performed on-the-fly when the email is sent.

> ⚠ The certificates discussed here are used for email encryption only and **not** for the access of ConSol CM (as email client) to the email server! That has to be managed using certificates which are stored in the security store of the application server.

The following figure shows the basic principle of email encryption for incoming and outgoing emails in ConSol CM.

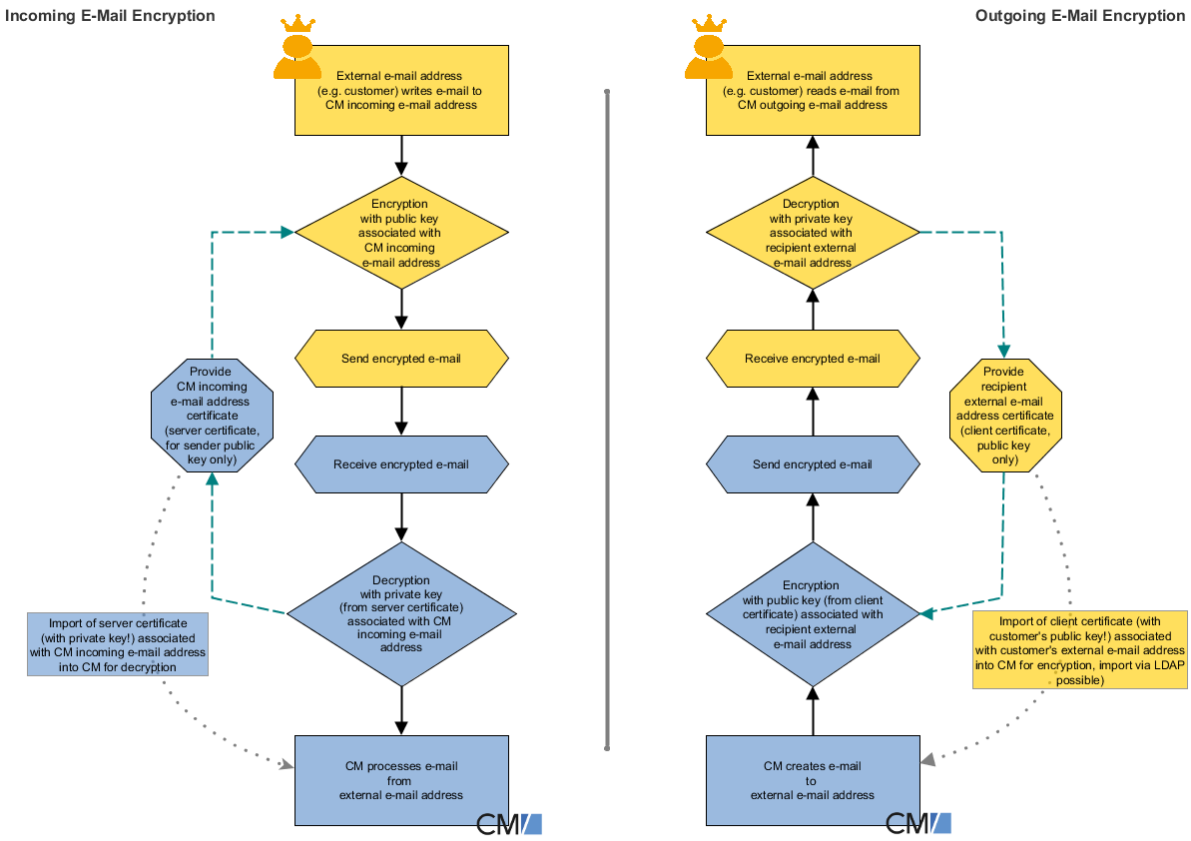


Figure 354: *Basic principle of ConSol CM email encryption*

Requirements

- The client certificate must contain the email address of the customer (receiver of the email) in the attribute *SubjectDN* (*E=* or *EMAILADDRESS=*) or the *X509v3 Subject Alternative Name* element from the *Extensions* section of the certificate.
- *Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files* has to be installed on the server and on the machine where the Admin Tool is started. This is required to enable the Admin Tool to import certificates.
- *X.509 Base64*-encoded certificates are supported.

Certificate Import from LDAP (Available for Client Certificates)

If LDAP is configured, ConSol CM will look up the client certificate for the requested contact in the LDAP repository. This is done as follows:

1. Someone tries to send an encrypted email.
2. The cryptography service looks for a client certificate (with the public key) of the recipient.
3. If a client certificate is found, the email is encrypted using the client's public key and sent.
4. If a client certificate is not found in the Admin Tool or it has expired, it is looked up in the LDAP repository.

5. If it is found, it is imported to ConSol CM and the email is encrypted and sent.
6. If it is not found, the email is sent unencrypted.

The following configuration properties have to be set to enable certificate lookup via LDAP:

- ldap.certificate.basedn
- ldap.certificate.searchattr
- ldap.certificate.content.attribute

Please see section *LDAP certificate parameters* in System Properties for details.

F.7.3.2 Certificate Management in the Admin Tool

In the Admin Tool, the navigation items *Server Certificates* and *Client Certificates* in the navigation group *Email* are used to configure the CM environment for email encryption.

Server Certificates

Server certificates are used to decipher **incoming** email messages. In some exceptional cases, they are also used to encrypt outgoing emails: if one of the recipients is the same as one of the incoming email accounts, the server certificate will be used to encrypt that message. Server certificates each contain the public and the private key for the given email address. If you define an incoming email account (see section above), you have to upload a server certificate for that email address (or for all email addresses covered by this mailbox) to be able to receive encrypted messages (because the server certificate contains the respective private key). If you have several incoming accounts, you either have to upload a server certificate for each of them or you can upload one certificate with all required email addresses.

When you open the navigation item *Server certificates*, a list of all existing server certificates is displayed. To add a new server certificate, click the *Add* button and use the file browser to find the required certificate. The certificate is validated before it is imported. If there are any incompatibilities, an error message is displayed and the certificate is not imported.

Supported formats for server certificates are:

- PKCS #12 archive file containing certificate (public) and private key (password protected). Supported filename extensions for PKCS #12 files are:
  - .p12
  - .pfx

Client Certificates

A client certificate contains only the public key of an external recipient (e.g., a customer). It allows encrypting messages which are sent to that user, i.e. client certificates are used for **outgoing** emails.

When you open the navigation item *Client certificates*, a list of all existing client certificates is displayed. To add a new client certificate, click the *Add* button and use the file browser to find the required certificate. The certificate is validated before it is imported. If there are any incompatibilities, an error message is displayed and the certificate is not imported.

Supported formats for client certificates are:

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- X509 standard format.
  Supported file name extensions for X.509 certificates are:
  - .cer
  - .crt
  - .der
  - .pem

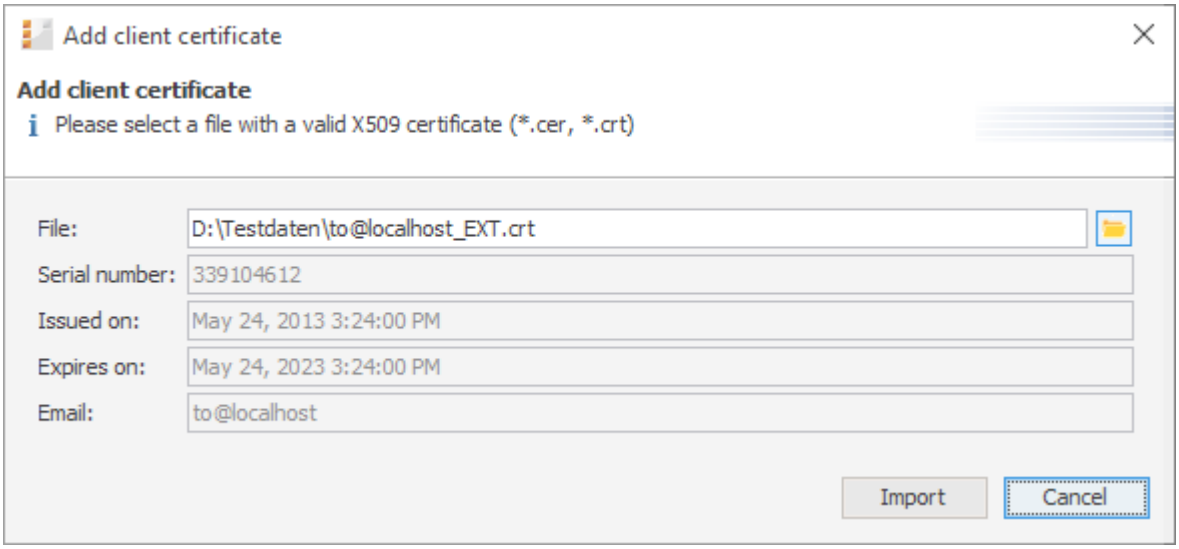


Figure 355: *ConSol CM Admin Tool - Email, Client Certificates: Pop-up window for adding a client certificate*

Here are some example use cases:

1. An engineer works in the ConSol CM Web Client and writes an encrypted email using the Ticket Email Editor. When the *Send* button is clicked, the ConSol CM system looks up the receiver address in the list of email addresses under *Client certificates* and uses the public key of the recipient to encrypt the outgoing email. If ConSol CM cannot find a matching certificate (the email address is not in the certificate list), the certificate for the email address is loaded from LDAP. If no compatible certificate is found there either, the email is sent unencrypted.
   If one of the recipients is one of the incoming email accounts, the server certificate (public key) will also be used to encrypt that message.
2. ConSol CM receives an email and checks the To address. If it is found in the list under *Server certificates*, ConSol CM uses the private key given in this certificate to decrypt the message and to either create a new ticket or append the message to an existing ticket.

Sending Encrypted Emails

Choosing if Emails Should Be Sent Encrypted from the Web Client

If the page customization property mailEncryptionAvailable has been set to “true”, a checkbox *Send encrypted* is available in the Ticket Email Editor in the Web Client. Thus, the user can choose whether the email should be sent encrypted.

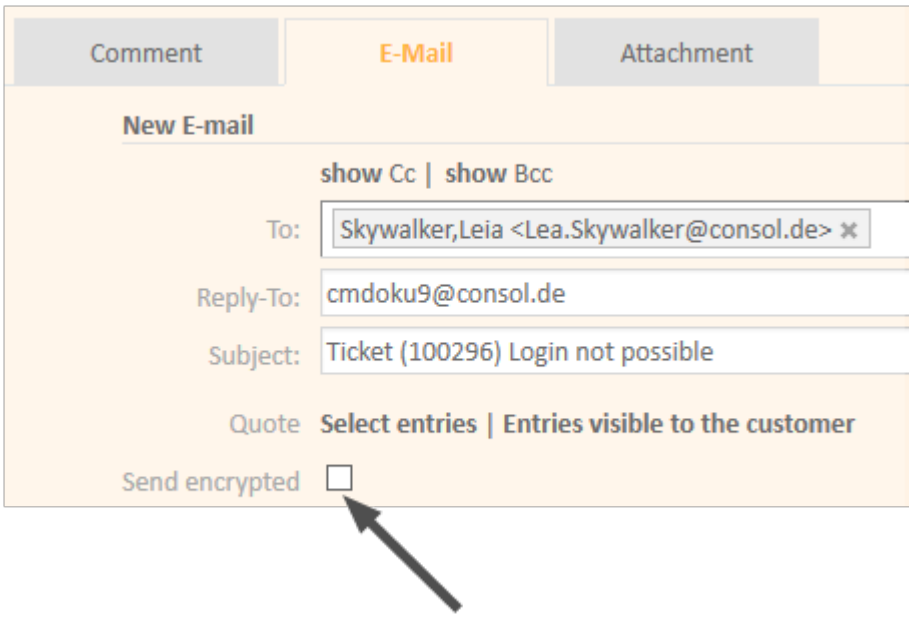


Figure 356: *ConSol CM Web Client - Send encrypted email*

Sending an Encrypted Email from the Workflow

An encrypted email can be sent by using the method `enableEncryption()`. Please see the *ConSol CM Process Designer Manual* for a detailed explanation.

Sending Encrypted Emails by Default

If the system property cmas-core-server, mail.encryptionis set to “true”, all outgoing emails from the workflow and Web Client are encrypted by default.

If users would like to send selected emails unencrypted, they can uncheck the checkbox *Send encrypted* in the Web Client. For emails sent by the workflow the method `disableEncryption()` can be used for this purpose.

F.8 Scripts and Admin Tool Templates

In this chapter, you will learn how to work with scripts and templates that are stored in and managed with the Admin Tool.

Scripts are used in various contexts in ConSol CM, particularly in the Process Designer within workflows. Please see the *ConSol CM Process Designer Manual* for a detailed explanation of this topic. However, various scripts are also stored in the Admin Tool, in the *Scripts* section. This is explained in section Admin Tool Scripts.

Templates are also stored in several locations:

- Accessible via Web Client:
  - In the Text Template Manager (email templates)
  - In the Document Template Manager (Microsoft Word or OpenOffice templates in CM/Doc)
- Accessible via Admin Tool:
  - In the *Scripts and Templates* section
    - Some special email templates
    - Templates for customer data
    - Templates for resource data
    - Password reset template
    - CM/Phone templates
    - <more templates, depending on customization>

For an explanation of using email templates using the Text Template Manager, and for configuring CM/Doc, please refer to section Working with Templates. For an explanation of templates in the Admin Tool, please read section Admin Tool Templates.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.8.1 Admin Tool Scripts

F.8.1.1 Introduction to Scripts in the Admin Tool

Scripts are stored in the *Scripts and Templates* section of the Admin Tool. They are written in Groovy and should only be edited by experienced ConSol CM consultants and administrators.

To work with scripts, open the navigation item *Scripts and Templates* in the navigation group *System* in the Admin Tool. The tab *Scripts* will be shown.

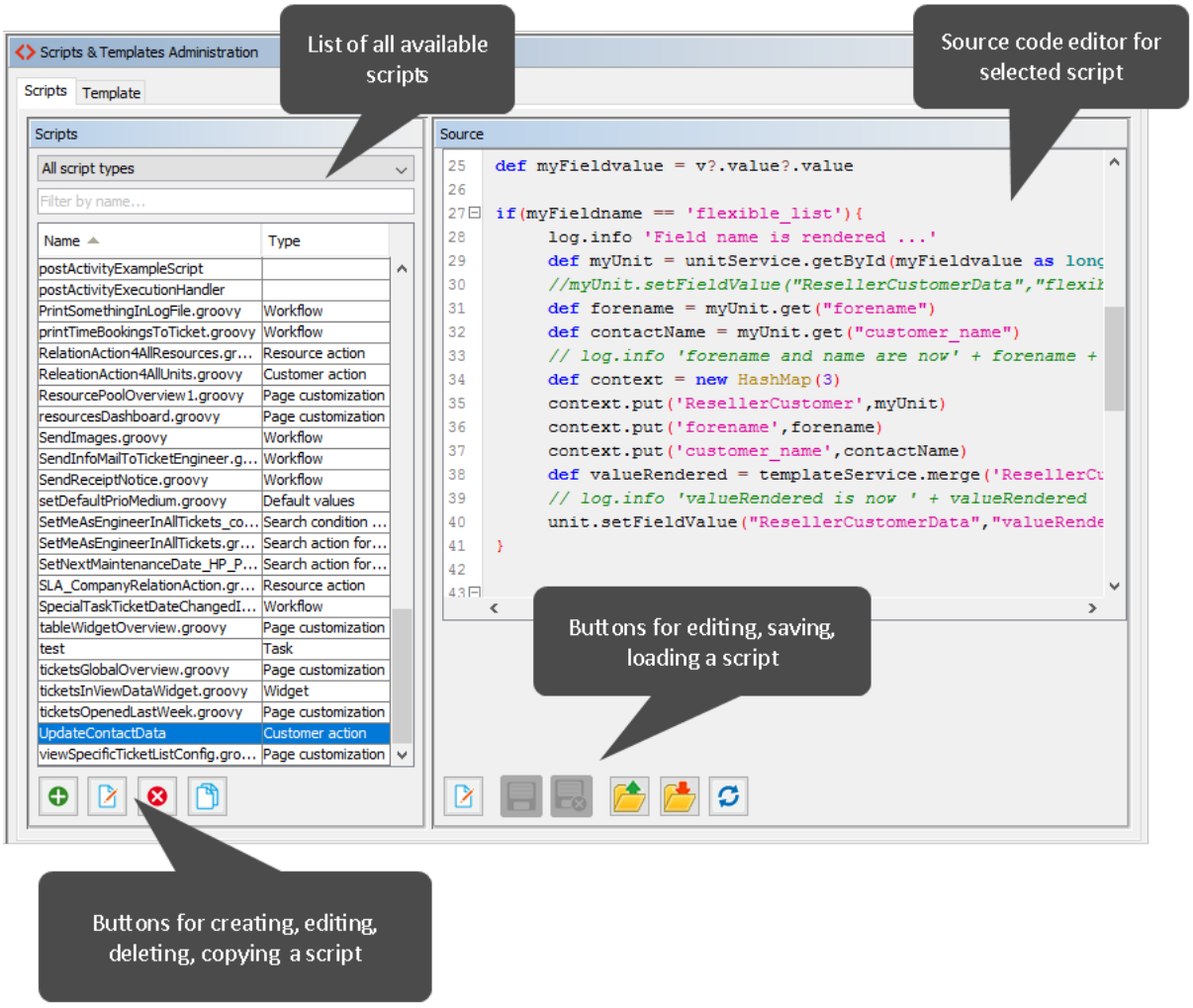


Figure 357: *ConSol CM Admin Tool - System, Scripts and Templates*

On the left you see the list of all scripts. The list can be filtered using

- the drop-down menu where the script type can be selected.
- the input field for a text filter for the script name.

Two parameters have to be set for each script:

- **Name**
  This is the name by which the script will be referenced, e.g., from the workflow or from other objects like queues.

- **Type**
  The script type. One of the following possible script types has to be selected:
  - **Calendar integration**
    Scripts of this type provide the connection information for the integration of Microsoft Exchange calendars. See section Microsoft Exchange Calendar Integration.
  - **Clone**
    Script which is executed when the *Clone* option is selected for a ticket. Has to be assigned to a queue. See section Scripts of the Type Clone for details.
  - **Control form condition**
    Script which is executed to determine if an action form for a customer action, resource action, or search action should be displayed. Please see Control Forms for further information.
  - **Control form prefill**
    Script which is executed when an action form is opened. It prefills data fields in the action form. Please see Control Forms for further information.
  - **Customer action**
    Script which is executed when a customer action has occurred, see section Action Framework - Customer Actions for details.
  - **Customer condition**
    Script which is executed to evaluate whether a customer action should be made available to the Web Client, see section Action Framework - Customer Actions for details.
  - **Custom form condition**
    Script which is executed when a client tries to access a web form. It determines if the web form can be used by the client. See section CM/Forms for details.
  - **Default values**
    Scripts of this type are used to define default values, i.e., values that are (pre)set in data fields when a new ticket is to be created. Please see section Scripts of Type Default Values for details.
  - **Dependent enum**
    Scripts of this type are used to define *dependent enums*, structures which provide hierarchical lists. Please see section Scripts of Type Dependent Enum for details.
  - **Email**
    Scripts of this type are used to manage incoming and outgoing emails. Please see section Scripts of Type Email for details.
  - **Field visualization**
    Scripts of this type are used to configure the visualization of data fields. Please see section Scripts of Type Field Visualization for details.
  - **Integration**
    Scripts of this type are used to implement services for the CM *Webhooks* interface. See section Webhooks for details.
  - **News Publisher**
    Scripts of this type are used to implement news for the Web Client and/or CM/Track. See section Scripts of Type News Publisher for details.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Page customization**
  Scripts of this type are referenced by page customization settings. Please see section Dashboards for details. The script type *Widget* can be used interchangeably with this type. Both types work exactly the same way. However, by using scripts of type *Page customization* for scripts which are referenced by page customization scopes or types and using script type *Widget* for scripts which are used to implement dashboard widgets, you can distinguish the scripts easily in the list of scripts (by sorting by type).
- **PostActivityExecutionScript** (no specific type indicated)
  An (optional) script which is executed after every manual workflow activity. See section PostActivityExecutionScript for details.
- **Relation graph**
  Scripts of this type are used to implement the graph display of relations in either standard sections or the expert section in ticket, customer (contact/company) and resource pages. For a detailed explanation, please refer to section Configuration of the Graphical Representation of Relations.
- **Resource action**
  Script which is executed when a resource action has taken place, see section CM/Resource Pool - Resource Actions for details.
- **Resource condition**
  Script which is executed to evaluate whether a resource action should be made available to the Web Client. See section CM/Resource Pool - Resource Actions for details.
- **Search action for customers**
  This script type is part of the Action Framework, namely the search actions, and is described in section Action Framework - Search Actions.
- **Search action for resources**
  This script type is part of the Action Framework, namely the search actions, and is described in section Action Framework - Search Actions.
- **Search action for tickets**
  This script type is part of the Action Framework, namely the search actions, and is described in section Action Framework - Search Actions.
- **Search condition for customers**
  This script type is part of the Action Framework, namely the search actions, and is described in section Action Framework - Search Actions.
- **Search condition for resources**
  This script type is part of the Action Framework, namely the search actions, and is described in section Action Framework - Search Actions.
- **Search condition for tickets**
  This script type is part of the Action Framework, namely the search actions, and is described in section Action Framework - Search Actions.
- **System documentation**
  This script type is used to create a file with the system documentation, please see section Scripts of Type System Documentation.
- **Task**
  Scripts of this type are used by the TEF (Task Execution Framework), please see section The Task Execution Framework (TEF).

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Text Autocomplete**
  This script type is used to implement text autocomplete fields. Please see section Scripts of Type Text Autocomplete for details.
- **Widget**
  This script type is used for scripts which implement dashboards, see Scripts of the Type Widget. The script type *Widget* can be used interchangeably with the script type *Page customization*. Both types work exactly the same way. However, by using scripts of type *Page customization* for scripts which are referenced by page customization scopes or types and using script type *Widget* for scripts which are used to implement dashboard widgets, you can distinguish the scripts easily in the list of scripts (by sorting by type).
- **Widget visualization**
  This script type is used for scripts which implement dashboards using generic widgets (see section HTML widgets. Please see section Scripts of Type Widget Visualization for details.
- **Workflow**
  Scripts of this type are referenced from the workflow. Please see section Scripts of Type Workflow for details.

The buttons below the list offer the standard Admin Tool functionalities:

- Add a script
- Edit a script
- Delete a script
- Copy a script

On the right you see the *Source Code Editor*. The script which is selected in the list on the left is displayed. Here you can write/edit the script source code when using edit mode.

F.8.1.2 The Source Code Editor

The Source Code Editor provides an editing panel with

- syntax highlighting
- code completion (use CTRL+SPACE, the classes in the locally executed tool, i.e. in the CLASSPATH, will be offered, not all classes and methods in ConSol CM)
- code validation (see display in the *Compilation result* box, since the validation is performed on the server side, all classes and methods of ConSol CM will be checked).
  - The interval between two code checks can be configured using the CM system property cmas-app-admin-tool, script.validation.interval.seconds.
  - The code validation can be disabled per session using the checkbox *Disable validation* below the editor pane.
  - By default, a simple version of the code validation is used to avoid false positives. You can activate a strict validation, which also checks if the invoked methods exist for the objects, by adding the following line to the script:

```
//enable-strict-mode
```

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Please note that the strict mode might lead to false positives, for example when calling other scripts and using classes and methods defined in other scripts. Use the following syntax to add type hints to avoid such false positives:

```
//type-hint[someProperty:someClass]
```

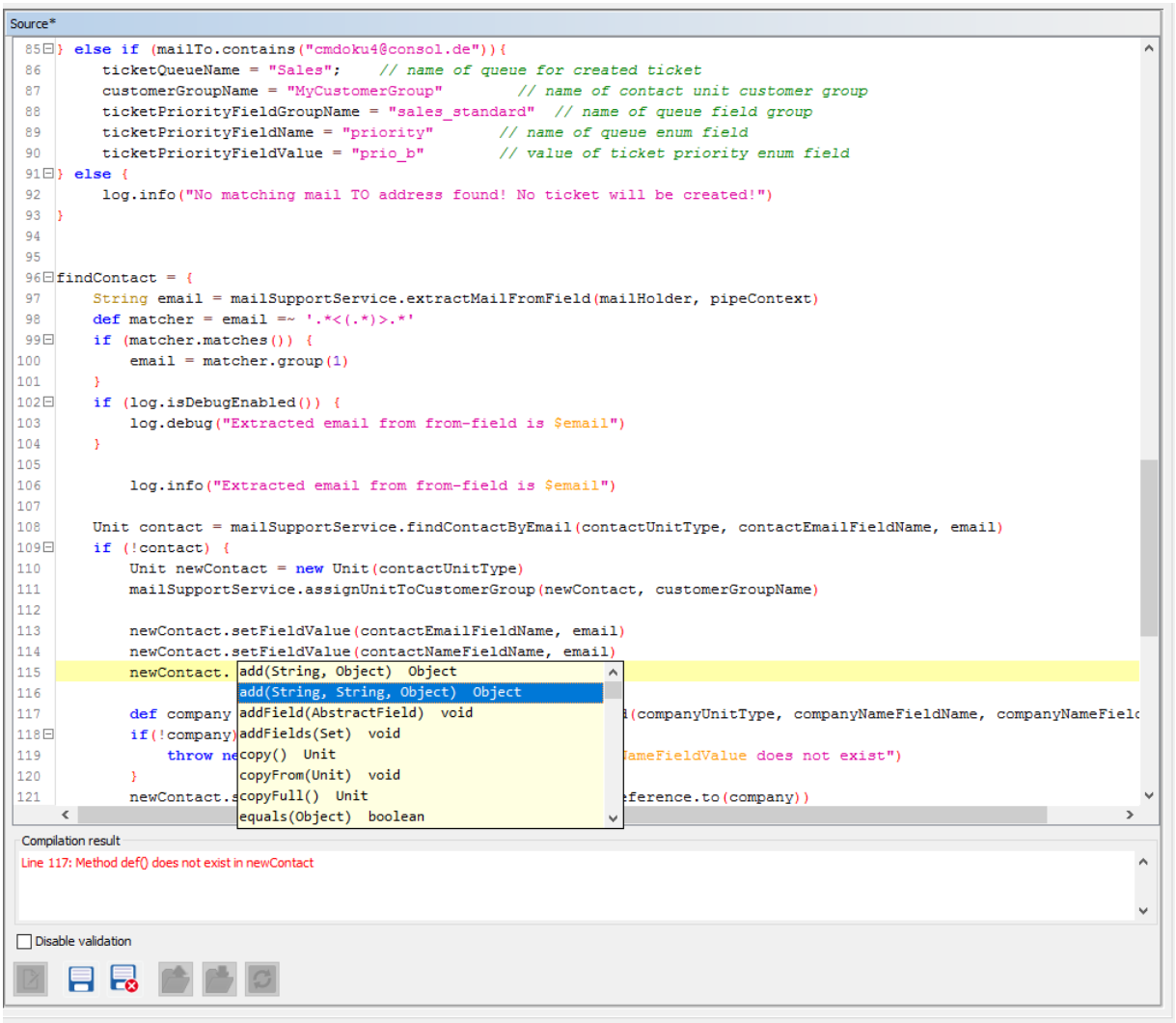


Figure 358: *ConSol CM Admin Tool - System, Scripts and Templates: Source Code Editor*

The lower section of the Source Code Editor has the following buttons:

- **Edit**
  Click this button to switch to edit mode in the Source Code Editor. When you open the navigation item *Scripts and Templates* in the Admin Tool, all scripts are in read-only mode to prevent an administrator from accidentally changing something.
- **Close edit mode and save changes**
  Save the script and quit edit mode, i.e., switch to read-only mode again.
- **Close edit mode without saving changes**
  Switch to read-only mode, without saving any changes you might have made to the source code.

- **Open script from file**
  This option opens a file browser. Two cases are possible:
  - The name of the selected file and the name of the script are identical.
    In this case, the code of the script (in the Source Code Editor) will be completely replaced.
  - The name of the selected file and the name of the script differ.
    In this case, a new script will be created with the name of the selected file. The new script will be displayed in edit mode in the Source Code Editor.
- **Save script to file**
  Here you can save the text of the script as a plain text file in the file system of the machine the Admin Tool is running on.
- **Update script from file**
  This option opens a file browser. The code in the Source Code Editor will be replaced by the text in the file, no matter if the names of the script and the file name are identical or not.

F.8.1.3 Script Types

In the following section, the available script types are explained. Some examples are provided to give you an idea of potential uses for scripts.

The following script types are explained here in this section. Please refer to the links provided in the list above for a complete overview of all script types with links to their respective documentation.

- Scripts of Type Text Autocomplete
- Scripts of the Type Clone
- Scripts of Type Email
- Scripts of Type Default Values
- Scripts of Type Dependent Enum
- Scripts of Type Field Visualization
- Scripts of Type News Publisher
- Scripts of Type System Documentation
- Scripts of Type Widget Visualization
- Scripts of Type Workflow
- PostActivityExecutionScript

F.8.1.4 Scripts of Type *Text Autocomplete*

Scripts of the type *Text autocomplete* can be used to implement text autocomplete fields. These are text fields which display a list of suggestions according to the entered text.

The following figure shows an example of a text autocomplete field which contains the results of an engineer search.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

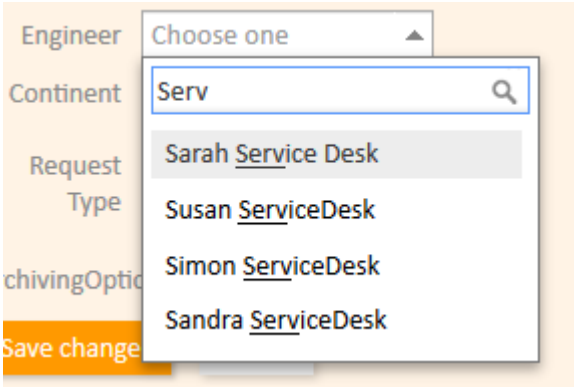


Figure 359: *ConSol CM Web Client - Text autocomplete field*

Text autocomplete fields offer a number of advantages:

- reference tickets, customers, resources or engineers saved in ConSol CM
- search within the list values
- add custom logic to determine the selectable values

Examples for the use of text autocomplete fields:

- Create a field which contains the responsible engineer of a customer.
- Provide text autocomplete fields as a helper for creating relations within an ACF.
- Save the name of the engineer who should receive notifications about a ticket.

Steps to use a text autocomplete script:

1. Write a script of the type *Text autocomplete* in the *Scripts* section of the Admin Tool. See Creating the Text Autocomplete Script.
2. Create the data field for which the script should be used. See Creating the Data Field.

Creating the Text Autocomplete Script

Text autocomplete fields store two values:

- **Internal value**
  The internal value depends on the field type. For fields of the type *Ticket*, *Unit*, *Resource*, or *Engineer*, it is the reference to the respective object. For fields of the type *String*, it is the string itself.
- **Display value**
  The display value is the value which is shown to the users in the Web Client or CM/Track. There are two options for the display value:
  - **Dynamic**
    The display value is retrieved from the referenced object itself. If there are changes to the referenced object, the display value is updated accordingly.
  - **Static**
    The display value is a string which is defined by the script. This can be a value from the referenced object or any other text. The display value does not change when there are changes to the referenced object.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Dynamic display values use templates to render the list values in the drop-down list with the suggestions, and the data field after a value has been selected. The following templates are used:

- **Engineers**: *engineer description template name*
- **Customers**: *Customer search result template*. If it is not present, the *Default* template is used.
- **Resources**: *Search template*. If it is not present, the *Default* template is used.
- **Tickets**: Fixed internal template containing the ticket name and subject.

Structure of the Script

The script has to implement the method `onSearchInput`. This method has two signatures, one for regular data fields and one for data fields which belong to lists. When you create a new script of type *Text autocomplete*, the method template is inserted automatically, so you can easily locate the correct method.

The script has to return an object of the class `ScriptAutocompleteResult` which contains the search results. The objects added as search results need to match the object type defined for the data field (see Creating the Data Field), i.e., you can add either tickets, or customers, or resources, or engineers, or strings.

The default message displayed if the search does not return any results is *No matches found*. You can use the method `noResults('String')` to customize this message, e.g.:

```
return ScriptAutocompleteResult.noResults('No matching engineers found')
```

Methods to Add Values to the List

The type of display value (static or dynamic) depends on the method used to add values to the list.

First, you create an object of the class `ScriptAutocompleteResult`.

```
ScriptAutocompleteResult resultset = new ScriptAutocompleteResult();
```

Then you use one of the following methods to add the values to the list.

- **Static display values**

  Add new items to the list by passing the object and a string with the display value as parameters:

  ```
  resultset.add(ticket, ticket.getSubject());
  ```

  ```
  resultset.add(ticket, "my display value");
  ```

- **Dynamic display values**

  Add new items to the list by passing the object as a parameter:

  ```
  resultset.add(ticket);
  ```

Please see the ConSol CM API Documentation of the class `ScriptAutocompleteResult` for details about the method signatures.

Uses Cases for Returning Strings

There are two additional use cases for `ScriptAutocompleteResult` objects which contain strings:

- Search in external sources, e.g., an external database
- Display a message that the user has to enter a certain number of characters to start the search

Working with the Context

You can use the `pContext` variable to retrieve the object which the data field belongs to. The content of `pContext` depends on the type of object.

- **Ticket**:

  `pContext.ticket` contains the current `ticket` object
- **Customer**:

  `pContext.unit` contains the current `unit` object
- **Resource**:

  `pContext.resource` contains the current `resource` object

This allows you to influence the search results based on the actual ticket, customer or resource which the text autocomplete field belongs to.

Creating the Data Field

Text autocomplete fields can be used in ticket, customer, and resource fields. Please proceed as follows to create a new text autocomplete field:

1. Create a field of the data type *autocomplete*.
2. Select the desired object from the *Type* drop-down list. The available options are *Ticket*, *Unit*, *Resource*, *Engineer*, or *String*.
3. Select the script in the *Script* drop-down list. Only scripts of the type *Text autocomplete* are available. See Creating the Text Autocomplete Script.

Coding Example

The following example script implements a text autocomplete field containing resource search results. The variable `pSearchStr` is used to check how many characters have been entered. In this example, the drop-down list shows a message that at least 3 characters are required if the user does not enter enough characters to start the search. If no results have been found, a customized message is displayed. Otherwise, the drop-down list contains the search results. The script works with dynamic display values.

```
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. More about this later in the
 specification (Context section)
*/

ScriptAutocompleteResult onSearchInput(String pSearchStr, FieldKey pKey, Context
 pContext) {
   ScriptAutocompleteResult resultset = new ScriptAutocompleteResult();
   // Add matching resource when entering more than 2 characters
   if (pSearchStr.length() < 3) {
      return new ScriptAutocompleteResult("NOTE: Please enter at least 3
       characters!");
   } else {
      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 example 64: *Script of a text autocomplete field for a resource field*

The list which is implemented by the script shown above is displayed in the Web Client as follows:

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

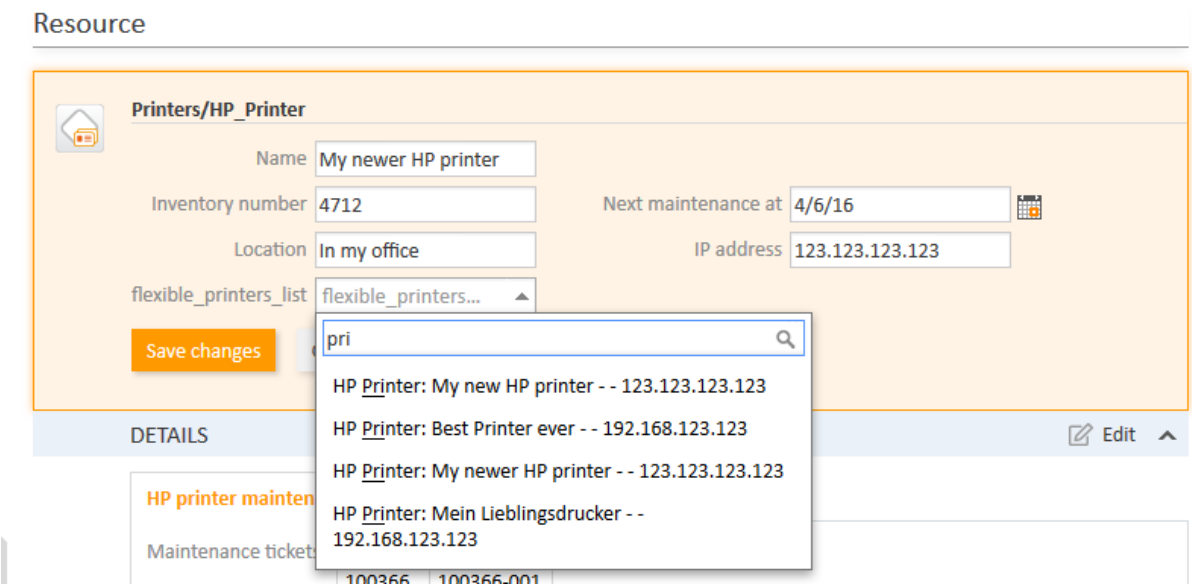


Figure 360: *ConSol CM Web Client - Text autocomplete field of type resource*

Retrieving and Setting Values of Text Autocomplete Fields

The value of a text autocomplete field belongs to the class `AutocompleteValue`. It can be retrieved using the methods `getInternalValue()` and `getDisplayValue()` of this class.

The following example shows how to write the value of the text autocomplete field *engineers_autocomplete* to the log file. The field contains a reference to an engineer and belongs to a ticket.

```
// display engineer of ticket field (not engineer field in header!)
def ticket = workflowApi.ticket

AutocompleteValue myAutocompleteValue = ticket.get("serviceDesk_fields.engineers_
 autocomplete")
def intValue = myAutocompleteValue.getInternalValue()
log.info 'The internal value is ' + intValue
def dispValue = myAutocompleteValue.getDisplayValue()
log.info 'The display value is ' + dispValue
```

Code example 65: *Script which writes values of a text autocomplete field to the server.log file*

To set the values of a text autocomplete field, you need to create an object of the class `AutocompleteValue` with the internal value and the display value. Use the method `setDynamicDisplayValue(true)` to create a field with a dynamic display value.

The following example shows how to reference the engineer “Susan ServiceDesk” with the ID “26” in the ticket autocomplete field *engineers_autocomplete* using a dynamic display value.

```
def ticket = workflowApi.ticket

AutocompleteValue myAutocompleteValue = new AutocompleteValue("26","Susan
 ServiceDesk");
myAutocompleteValue.setDynamicDisplayValue(true);
ticket.set("serviceDesk_fields.engineers_autocomplete", myAutocompleteValue);
```

Permissions in Text Autocomplete Fields

Permissions are checked during the execution of the text autocomplete script when a field is edited in the Web Client and CM/Track, i.e., the user can only see and select entries referring to objects which he is allowed to view according to his assigned roles. When viewing text autocomplete fields, permissions are not checked. Therefore, the user might see labels referring to objects for which he has no permissions in text autocomplete fields and history entries.

> ⚠ **Note regarding permissions in CM/Track:**
>
> Permissions related to text autocomplete fields work the same way as for the Web Client. CM/Track-specific restrictions, namely the limitation that a user only sees his own tickets (and tickets of other contacts of his company, if configured) and contacts of his company, are not part of the default configuration. This means that, by default, the user has permissions to see and select entries referring to tickets belonging to queues for which the Track user profile has at least read permissions and customers belonging to customer groups for which the Track user profile has at least read permissions. In short, the customer would see data belonging to other customers.

Adapting the Permissions for CM/Track

The class `customerSecurityCriteriaBuilder` provides methods to adapt the behavior of text autocomplete fields to the standard permissions in CM/Track, so that the users can only see results which refer to their tickets or to contacts which belong to their company.

The following example shows how to limit the search results of a text autocomplete field used in CM/Track to tickets belonging to the current user.

```
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 example 66: *Script of a text autocomplete field which can be displayed in CM/Track*

General Information About Text Autocomplete Fields

This new way of configuring text autocomplete fields replaces the previous method, which used the annotation *text-type* = "autocomplete". Existing fields which use the old method will remain functional. These fields are not modified during an update. The previously used method `onEditDisplayEntered` still works for existing fields. Nevertheless, it is not needed anymore and will be removed in a future version of ConSol CM.

The following limitations exist when using the new way of configuring text autocomplete fields:

- No transfer to the DWH. Text autocomplete fields are not transferred to the DWH and cannot be used in reports.
- No support for ETL. Import and export of text autocomplete fields using ETL is not supported.
- Text autocomplete fields cannot be indexed. Therefore they are not available for search.
- If an object which is referenced in an text autocomplete field is deleted, the field might point to a non-existing object. This has to be considered in case the field value is later used in scripts.
- Text autocomplete fields cannot be used in text templates and document templates.

> Please refer to the *ConSol CM Administrator Manual* version 6.11.1.0 for information about the previous way of configuring text autocomplete fields.

F.8.1.5 Scripts of the Type *Clone*

In the Web Client, a ticket can be duplicated using the *Clone* option in the ticket menu.

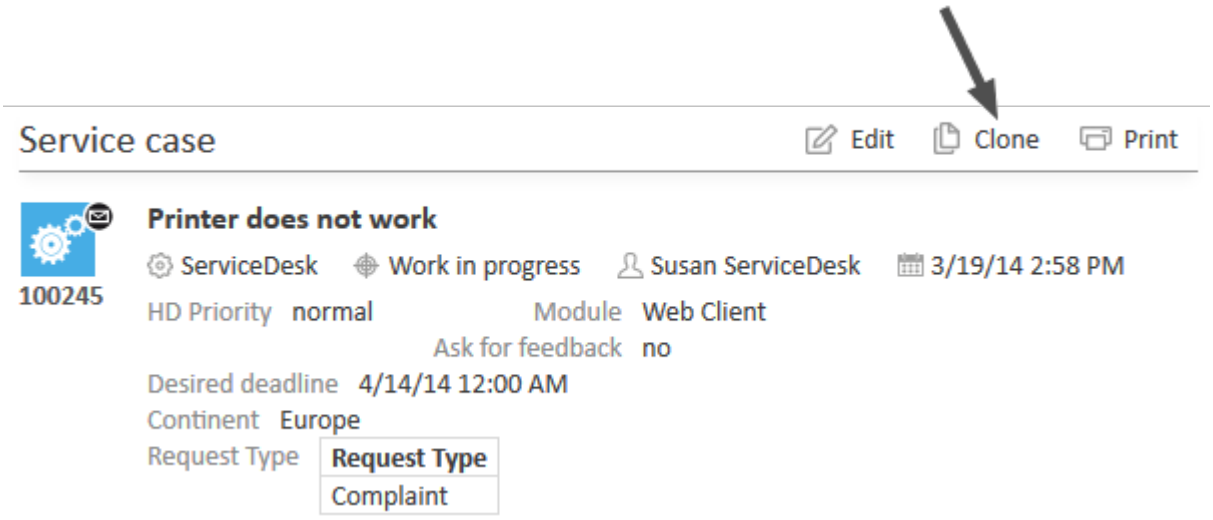


Figure 361: *ConSol CM Web Client - Clone option in ticket menu*

When a ticket is cloned, the following data is transferred from the original ticket:

- ticket subject
- queue
- assigned engineer (if set)
- values of all ticket fields (header and Details section)
- main customer
- additional customers

The following data is **not** transferred from the original ticket:

- ticket history, including comments, emails and attachments
- additional engineers
- related tickets or resources

A clone script allows to overwrite this default behavior. You can use it to preset values in the newly created ticket, similarly to using a default values script. The engineer can change the values if required.

Steps to use a clone script:

1. Write a script of the type *Clone* in the *Scripts* section of the Admin Tool.
2. Assign the script to the desired queue (see Queues). Each queue can only have one clone script.

Coding Example

In the following example, the clone script is used to reset the data field *Desired deadline* to avoid having incorrect deadlines in cloned *ServiceDesk* tickets.

```
ticket.set("serviceDesk_fields.desiredDeadline", null)
```

Code example 67: *Clone script to reset ticket field for desired deadline*

If the script is assigned to the queue *ServiceDesk*, the field for the desired deadline is empty in the cloned ticket instead of containing the deadline from the original ticket.

Please keep in mind that in a clone script, you do not work in the workflow context. That means the `workflowApi` object (implementation of `WorkflowContextService`) is not available.

F.8.1.6 Scripts of Type *Default Values*

A default values script allows to preset values on the ticket creation screen in the Web Client and CM/Track. This saves time during the ticket creation as the users do not have to enter all the values. In addition, it is a way to indicate the most commonly used values in order to improve consistency. If required, the users can change the preset values.

The following figure shows the ticket creation screen with one preset field (*HD priority*). The other fields are empty.

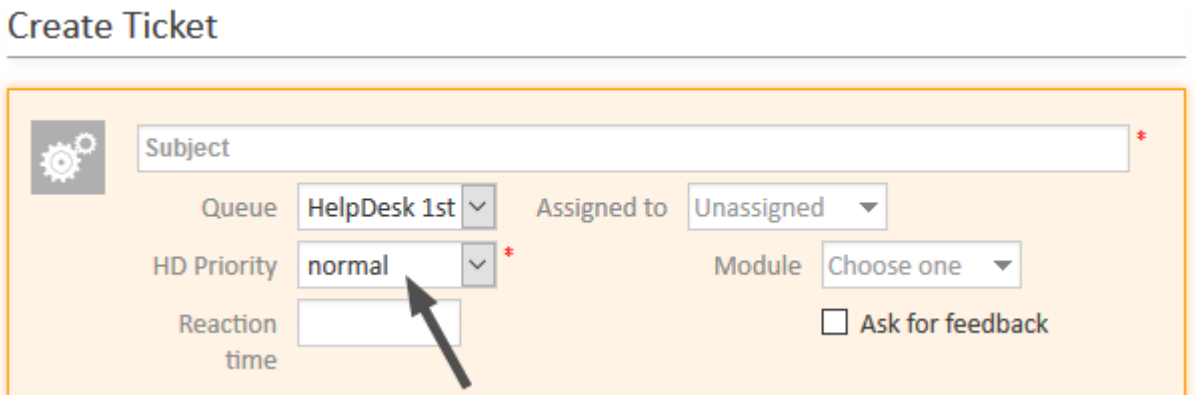


Figure 362: *ConSol CM Web Client - New ticket with default value*

Steps to use a default values script:

1. Write a script of the type *Default values* in the *Scripts* section of the Admin Tool.
2. Assign the script to the desired queue (see Queues). Each queue can only have one default values script.

Coding Example

The following example sets the default value for the priority field according to the client which is used to create the ticket: If the ticket is created in CM/Track, the default priority is “high” and if the ticket is created in the Web Client, the default priority is “normal”.

```
if (pClient == "track") {
   ticket.set("helpdesk_standard.priority", "high")
} else {
   ticket.set("helpdesk_standard.priority","normal")
}
```

Code example 68: *Default values script with different values according to the client*

The variable `pClient` can have two values:

- **web**: ticket created in the Web Client
- **track**: ticket created in CM/Track

> Please keep in mind that in a default values script, you do not work in the workflow context. That means the `workflowApi` object (implementation of `WorkflowContextService`) is not available.

Overwrite Mode for Default Values Scripts

By default, default values scripts do not overwrite existing values. This means that there might be circumstances where not all values which are defined in a default values script are applied. If you change the queue of a ticket when some values are already set (by a default values script or manually), these values will not be overwritten by the default values script of the new queue.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

You can change this behavior by enabling the overwrite mode. Add the following code at the beginning of the default values script to overwrite existing values:

```
import com.consol.cmas.common.model.ticket.TicketPrototypeContext
TicketPrototypeContext.enableOverwriteMode()
```

F.8.1.7 Scripts of Type *Dependent Enum*

Dependent enums are hierarchical data structures. In contrast to MLAs (see MLAs), the values of each level are calculated based on the value selected on the parent level. Therefore, instead of displaying all the values which belong to a given enum, it is possible to display only a subset of the available enum values. In addition, it is possible to have empty levels within a dependent enum. Dependent enums can be used in ticket, customer and resource data models.

In the Web Client, the selected value on each level is displayed.

Steps to use a dependent enum script:

1. Create the required enums in the *Enums* section of the Admin Tool.
2. Write a script of the type *Dependent Enum* in the *Scripts* section of the Admin Tool.
3. Assign the script to the desired data field group. A data field group can have several dependent enum scripts.

Example

The following example shows a dependent enum used to manage countries. Three enums are used in the example: *Continent*, *Subcontinent* and *Country*. On the first level, the user selects the continent. Depending on the selection, the user can either select a subcontinent on the second level, or directly select a country on the third level.

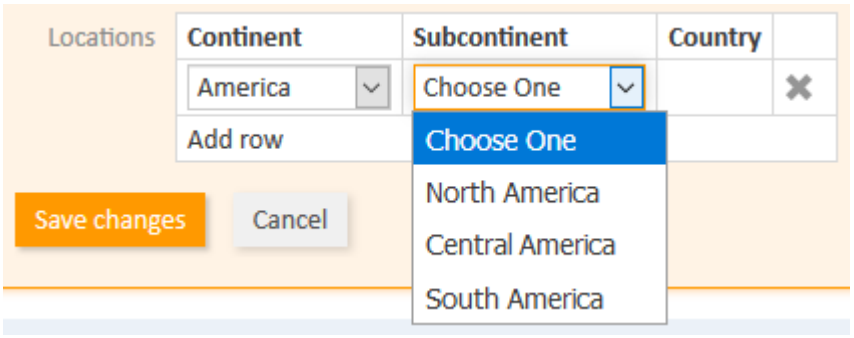


Figure 363: *ConSol CM Web Client - Sub-list of continent America*

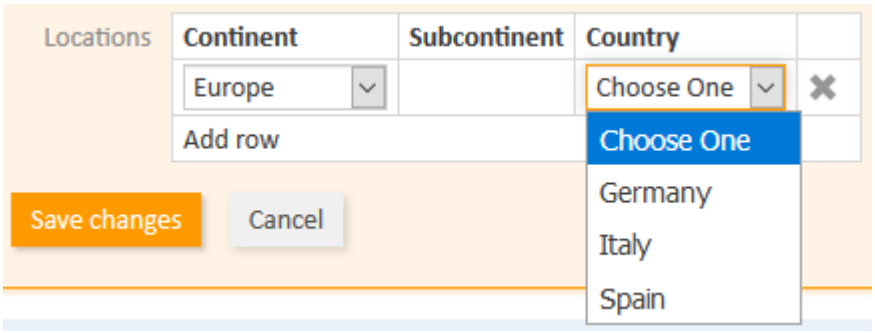


Figure 364: *ConSol CM Web Client - Sub-list of continent Europe*

F.8.1.8 Scripts of Type *Email*

Scripts of this type are used to configure the email functionalities of ConSol CM. Some of the scripts are part of the default system configuration and have to be modified according to the customer-specific system configuration. You can also add your own scripts.

Email Scripts for Incoming Emails

Incoming emails are processed using NIMH, see Email Configuration Using the Admin Tool. When ConSol CM receives an email, it is processed by several scripts.

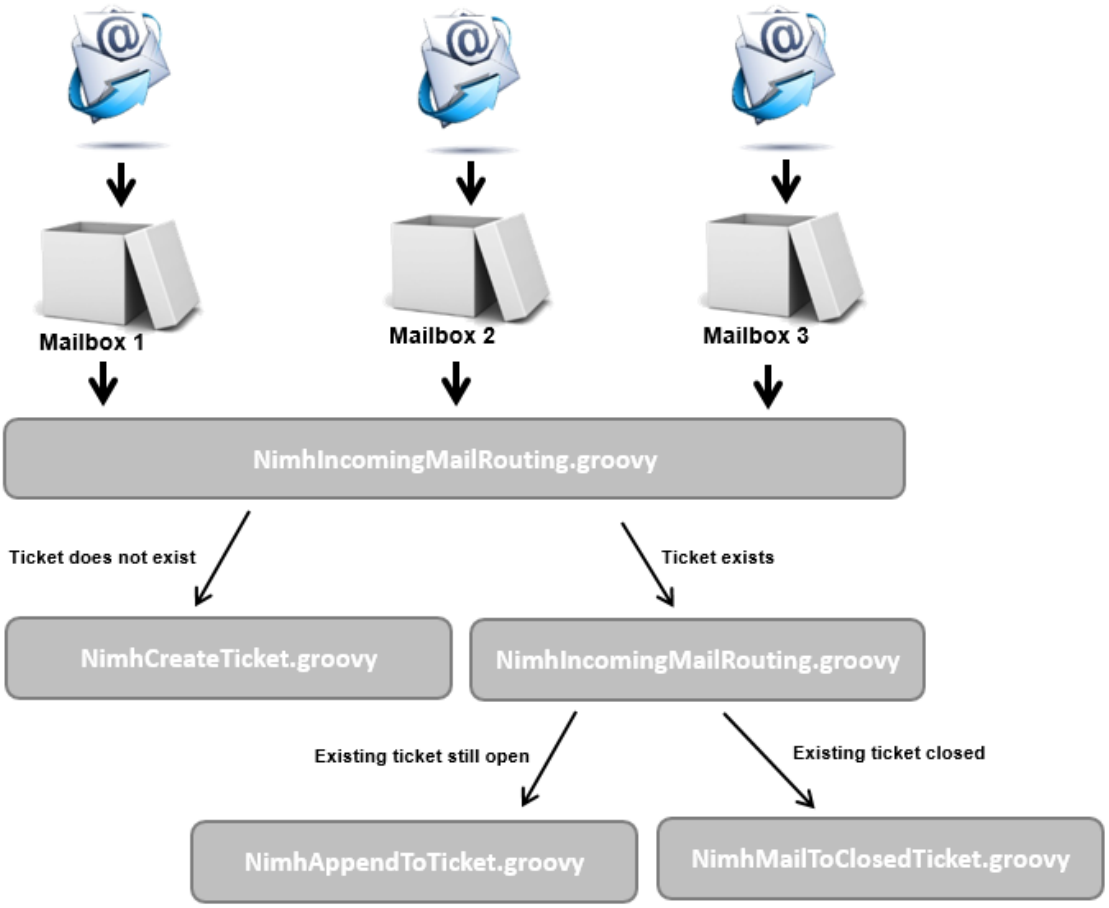


Figure 365: *Email scripts (NIMH) in ConSol CM*

- **NimhIncomingMailRouting.groovy**
  This is the first script that is executed when one of the configured mailboxes receives an email (see Email Configuration Using the Admin Tool). It determines how to handle this email. There are three cases:
  - The email does not belong to any ticket.

    -> The script `NimhCreateTicket.groovy` is executed and a new ticket is created.
  - The email belongs to an open ticket.

    -> The script `NimhAppendToTicket.groovy` is executed and the email is added to the history of the ticket.

- The email belongs to a closed ticket.

  -> The script `NimhMailToClosedTicket.groovy` is executed and a new ticket is created and linked to the closed ticket.

  > If the ticket which receives the email should be reopened instead of creating a new ticket, you need to introduce a reopen activity in the workflow and adapt `NimhIncomingMailRouting.groovy` to reopen the ticket and call `NimhAppendToTicket.groovy`. Please refer to the *ConSol CM Process Designer Manual* for further details.

Usually, it is not required to adapt this script.

- **NimhCreateTicket.groovy**
  This script is executed if the email subject does not match the regular expression for appending the email to an existing ticket. It creates a new ticket for the incoming email. The following aspects need to be defined in this script:
  - Queue in which the ticket should be created
  - Values of ticket fields which should be set automatically during ticket creation
  - Fields which are used for the name and email address of the contacts in the desired customer data model
  - Field which is used for the name of the default company
  - Customer group

  By default, the script searches for a contact whose email address matches the `From` address of the incoming email. If no such contact is found, a new contact is created in the default company.

  You can further adapt the script to fine-tune ticket creation, e.g., create tickets in different queues depending on the `To` address of the email.

  > Usually, this script has to be adapted considerably. Please ask a ConSol CM consultant for support with this task.

- **NimhAppendToTicket.groovy**
  This script is executed if the email belongs to an open ticket. It appends the email to the ticket history. The assignment of the email to the ticket is performed by comparing the email subject with the regular expression set in *Incoming email subject pattern* in the *Emails* section of the Admin Tool (see Email Configuration Using the Admin Tool).

  Usually, it is not required to adapt this script.

- **NimhMailToClosedTicket.groovy**
  This script is executed if the email belongs to a closed ticket. It creates a new ticket for the contact and a reference to the closed ticket.

  Usually, it is not required to adapt this script.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Email Scripts for Outgoing Emails

The processing of outgoing emails can be fine-tuned by queue (see Queues). Each queue can have an email script which is executed for every email written from a ticket in this queue. This applies to emails which are sent automatically by the workflow and to emails which are sent manually by an engineer.

A common use case for outgoing email scripts is to set queue-specific `Reply-to` or `From` addresses.

The default configuration includes an example script for outgoing emails:

- **ChangeOutgoingMail.groovy**
  Template for outgoing emails scripts

Within the outgoing email script, the Java object `mailEntry` is implicitly available as the object `mail`. You have to set all required attributes for the outgoing email using the `mail.setAttribute()` or `mail.setAttributes()` methods.

```
def queueReplyAddress = "serviceteam@mycompany.com"
// you might also use system properties for the queue-specific email addresses and
 fetch an
// address using the configurationService!

mail.setAttribute('Reply-to', queueReplyAddress)
```

Code example 69: *Email script*

Common email attributes are:

- Bcc
- From
- Reply-to
- To
- Cc
- Subject

Reply-to addresses in ConSol CM

There are four potential Reply-to addresses which you deal with:

- **System property `mail.reply.to`**
  If this system property is set, its value is displayed in the email editor in the Web Client. The email address can be overwritten by the outgoing email script. If the page customization attribute *showReplyTo* for the type *mailTemplate* is set to “false”, the Reply-To address is not displayed in the email editor, but used anyway.
- **Queue-specific outgoing email script**
  If a Reply-to address is set in the outgoing email script, this address is used. It overwrites the system property `mail.reply.to`, although the address set in this property is displayed in the email editor.



- **System property `mail.from`**
  Only relevant if the Reply-to address is neither set in the system property `mail.reply.to` nor in the outgoing email script. In this case, most email clients use the From address as Reply-to address.
- **Email address of the current user**
  Only relevant if the Reply-to address is neither set in the system properties `mail.reply.to` or `mail.from` nor in the outgoing email script. In this case, the personal email address of the currently logged in user is used as Reply-to address for emails from the Web Client.

The Reply-to address which was really used is displayed in the ticket history of the Web Client.

Best practice:

Always set a Reply-to address to prevent personal email addresses from being used, as this would lead to customer emails being sent to the users' personal email accounts instead of ConSol CM. There are two places to set the Reply-to address:

- **System property `mail.reply.to`**
  This is the simplest way to set the Reply-to address. This address is also displayed in the email editor.
- **Queue-specific outgoing email script**
  If different Reply-to addresses are required depending on the queue, we recommend to write one outgoing mail script where queue names are mapped to specific Reply-to addresses. This can then be extended for Bcc, Cc or other addresses.

You can combine the `mail.reply.to` system property and queue-specific Reply-to addresses. In this way the email address set in the outgoing email script is used for queues with such a script, and the email address from the system property is used for the rest of the queues.

Usage in workflow scripts

When sending emails in workflow scripts, you need to set the Reply-to address within the script.

- Retrieve the value of the system property `mail.reply.to` and set it as a Reply-to address in the outgoing email:

```
configurationService.getValue("cmweb-server-adapter","mail.reply.to")
```

- Use the `Mail` object if the outgoing email script should be used. This overwrites the system property `mail.reply.to`.

```
mail.useDefaultScript()
```

See the *ConSol CM Process Designer Manual* for a detailed explanation.

F.8.1.9 Scripts of Type *Field Visualization*

Scripts of the type *Field visualization* enable to customize the display of field content in the Web Client and CM/Track. In addition, they allow to display other content on ticket, customer and resource pages in the Web Client and on ticket pages in CM/Track.

Examples for the use of field visualization scripts:

- Changing the color, font size or background color of data fields to highlight important information
- Organizing data fields in several groups with headlines
- Displaying maps
- Displaying tables with additional features
- Displaying graphs

Field visualization scripts can use resources, e.g., images, stylesheets, JavaScript libraries, or other files, which are stored in the file system. Alternatively, the resources can also be retrieved using URLs.

Steps to use a field visualization script:

1. Save the required resources in a folder called `resources` within the ConSol CM data directory.
2. Write a script of the type *Field visualization* in the *Scripts* section of the Admin Tool. See Writing the Field Visualization Script.
3. Configure the data field:
    a. Create a data field of the desired type.
    b. Assign the annotation `common`, `visualization` to the data field and indicate the name of the field visualization script as value of the annotation.
    c. Optional: Assign the annotation `common`, `visualize-when-empty` to the data field if the field visualization script should be executed if the field has no content. Otherwise, the visualization content is only rendered if the field has a value.
    d. Optional: If the field visualization script displays content with `position: absolute`, you need to assign the annotation `visualize-allow-absolute-position` = `true` to the data field. The visualization content can then be displayed anywhere within the current object (i.e. case, contact or resource). The field label is hidden automatically.

Writing the Field Visualization Script

Field visualization scripts have to implement two methods:

- **render()**
  Includes the logic for rendering the field value. Returns the content to be displayed in HTML syntax.
- **resources()**
  Allows to include additional resources. The resources can be stored in the file system. A new folder `resources` can be created in the ConSol CM data directory for this purpose. Alternatively, resources can also be retrieved using URLs.

The methods provide several parameters:

- **pContext**: object from which the script is called
- **pFieldKey**: key of the field which calls the script

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **pFieldValue**: value of the field
- **pClient**: type of client, can be “web”, “track”, or “rest”

These parameters allow you, for example, to adapt the behavior to the field value or to configure a different behavior for the Web Client and CM/Track.

Coding Example: Display a Chart

You can use a field visualization script to display graphics like statistics data or small reports. The ***Highcharts*** library is used for rendering the data.

In the following example, a customer field, positioned in the *Details* section, displays statistics. The example uses fixed values in the script, in real life you work with values which are retrieved from real data.

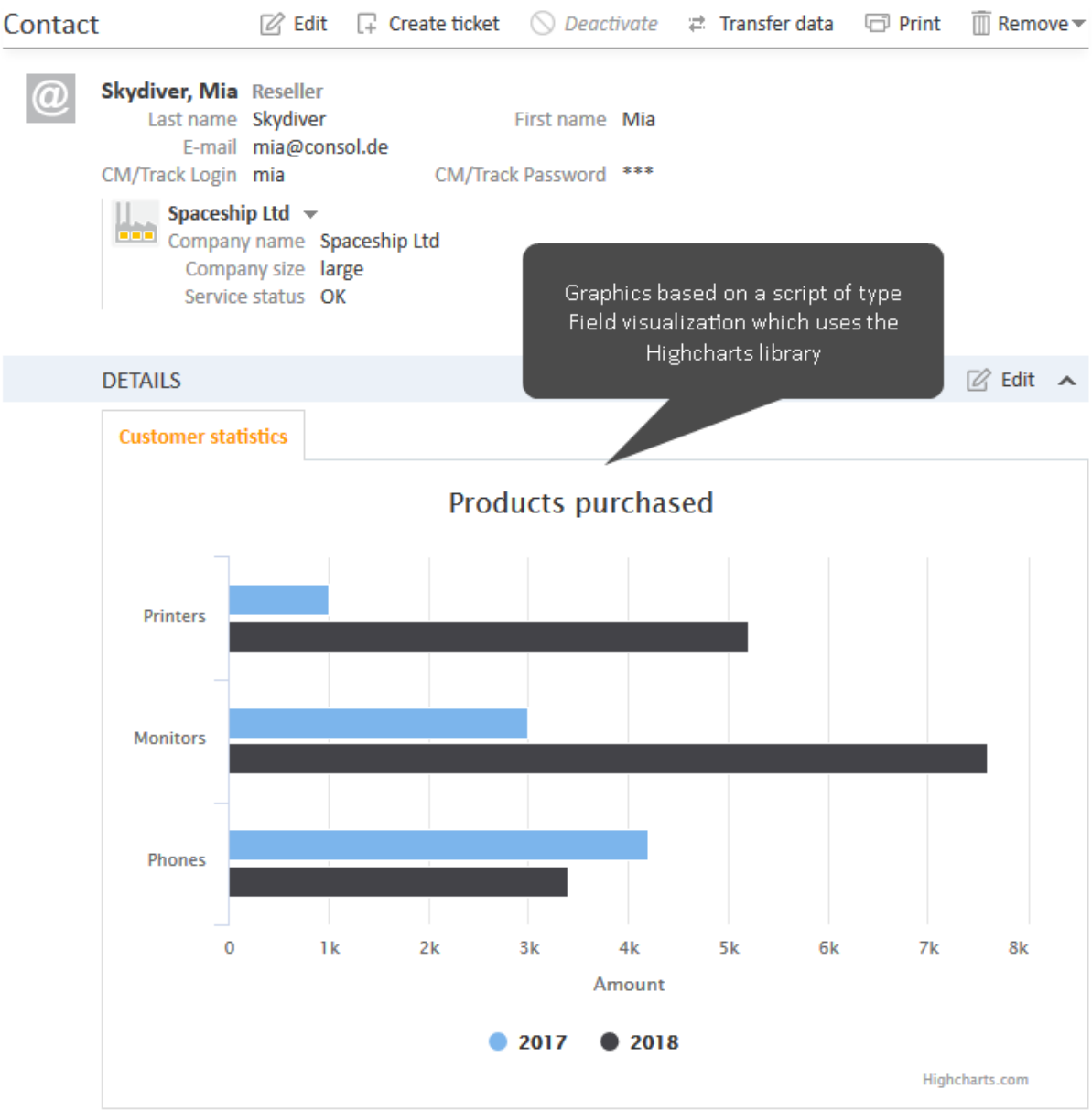


Figure 366: *ConSol CM Web Client - Field visualization rendering a graph on a contact page*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

The following example shows the relevant excerpts of an example script which uses static values.

```
import com.consol.cmas.common.model.customfield.meta.FieldKey;
import
 com.consol.cmas.common.model.customfield.visualization.FieldVisualizationContext;

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
}

def resources(FieldKey pFieldKey, Object pFieldValue, String pClient,
 FieldVisualizationContext pContext) {
  List<String> resources = [
     "https://code.highcharts.com/highcharts.js"
  ] as String[];
  return resources;
}
```

Code example 70: *Script to display a graph with static data based on the Highcharts library*

F.8.1.10 Scripts of Type *News Publisher*

Scripts of the type *News publisher* are used to determine which news are displayed on the Web Client dashboard and/or on the welcome page of CM/Track. The news which are returned by the script can be retrieved from several sources, for example, ticket comments or RSS feeds, or created directly

within the script.

The news script returns a list of news items to be displayed in the Web Client or CM/Track. There are two ways of creating this list:

- Create a list of objects of the class `NewsItem`.
- Use the convenience method `getNewsFromTicketEntries` of the class `NewsService` to retrieve news directly from ticket comments.

If you use `getNewsFromTicketEntries`, you need to define the following in the news script:

- Queues which contain the news tickets
- Scopes which the tickets have to be in
- Classes of text which the comments have to be marked with

From each ticket, the latest comment marked with one of the text classes is displayed in the news. Only history entries of the type **comment** can be used in news scripts. If any other entry, for example an email, is marked with the configured text class, the entry is ignored.

The default values for news items from ticket comments are:

- **subject**: the name and subject of the ticket which the comment belongs to
- **author**: the engineer or customer who added the comment
- **content**: the latest comment of the ticket which has the specified text class, including inline images and formatting
- **creation date**: the date when the comment was added
- **update date**: the date when the comment was updated
- **color**: the background color of the text class

Steps to use a news script:

1. Write a script of the type *News publisher* in the *Scripts* section of the Admin Tool.
2. Assign the script in the news configuration:
    a. **Web Client**: Assign the script to the news widget in the page customization (see News widgets).
    b. **CM/Track**: Reference the script in the client configuration of the welcome page (see Displaying News on the Welcome Page)
3. Adapt the default values for displaying news in the news configuration (optional).

Coding Example 1: Pseudocode to Retrieve News

The following pseudocode example shows how to build a list of `NewsItems` from an RSS channel.

```
def newsList = []
def url = "myurl"
Locale.setDefault(Locale.US)
try{
   def rss = new XmlSlurper().parseText(url)
   rss.channel.item.each {
      def date = new Date().parse("EEE, dd MMM yy HH:mm:ss",
       "${it.pubDate}".substring(0, 23))
      newsList.add(new NewsItem("${it.title}" as String, null, "${it.description}",
       date, null, null))
   }
}catch(Exception e){
   newsList.add(new NewsItem("No response from server" as String, ""))
}
return newsList
```

Code example 71: *Pseudocode to create news items from an RSS channel*

Coding Example 2: Retrieving News from Ticket Comments

The following example shows a script of the type *News publisher* used to retrieve news from ticket comments:

```
Set queues = ["ServiceDesk"]
Set scopes = ["defaultScope/Service_Desk/Work_in_progress"]
Set textClasses = ["Solution"]
return newsService.getNewsFromTicketEntries(queues, scopes, textClasses)
```

Code example 72: *Script to retrieve news from ticket entries*

The script retrieves the entries as objects of the class `TicketEntryNewsItem`. It is possible to extract data from the retrieved entries and modify them as desired.

F.8.1.11 Scripts of Type *System Documentation*

Introduction

You can export parts of the current configuration of ConSol CM into a file (see System Documentation). The script of the type *System documentation* determines the scope of this documentation, i.e., which information is exported. In addition, the script allows to apply some formatting and to add descriptions to the items.
The following items of the system configuration can be included in the system documentation:

- incoming and outgoing email connection
- customer data models: customer objects and customer fields
- customer groups
- workflows: scopes, activities, triggers, image of the workflow (only available after deploying the workflow)
- queues: workflow, assigned ticket field groups with ticket fields, customer groups, projects and classes of text
- resources: resource categories and resource types with resource fields

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- roles: queue, customer group, resource type and global permissions, views, engineer functions
- views: queues, scopes, view criteria



The ConSol CM distribution contains the two example scripts `Documentation_Generator_script_EN` and `Documentation_Generator_script_DE` for generating system documentation in HTML format in English and German.



You might need to adapt the system documentation script if you want to create documentation from a scene export.

Creating the Script

The methods for retrieving system documentation data belong to the package `com.consol.cmas.common.model.documentation`. Please refer to the *ConSol CM API Documentation* for detailed information about the available methods.

You need to consider the following aspects when writing the system documentation script:

1. **Document creation:**
   Write the document, the example scripts use the Java class `OutputStream` and the Groovy class `MarkupBuilder`.
2. **Content:**
   Determine which content should be included in the system documentation. The API allows to retrieve information about the objects listed in the introduction of this feature. The available information depends on the type of object.
3. **Languages:**
   If your system has more than one language, decide whether all languages are needed and whether they should be included in the same document.
4. **Enums:**
   The API retrieves technical enums in some places, e.g., for the types of data fields, workflow elements or permissions. If you want to provide a more user-friendly description for these enums, you can create a map containing a user-friendly label for each technical enum value.
5. **Workflow images:**
   If you want to include workflow images in the system documentation, you need to deploy the workflows first. The image is generated during workflow deployment and is therefore not directly available after updating the ConSol CM system.
6. **Layout:**
   Use HTML tags to properly structure the content of the system documentation. Typically, you would use standard HTML syntax to create headings, paragraphs, lists, and tables. A simple approach to layout the document is to define the display of the used HTML tags by creating simple CSS styles and adding them to the `<head>` tag of the output.

F.8.1.12 Scripts of the Type *Widget*

Widget scripts are used to implement widgets which are displayed on dashboards (see Dashboards). They implement logic to retrieve and prepare the data which is shown in the widget. A widget scripts needs to return the data in the format required by the type of widget.

Widget scripts are used for the following widget types:

- Chart Widgets
- KPI Widgets
- Table Widgets

The following general principles apply to widget scripts:

- If you set attribute values both in the widget script and in the page customization, the values set in the script overwrite the values entered in the page customization.
- The results of the methods which retrieve data varies depending on the context. By default, the script is executed with user permissions. If you need to use data which is not available in the user context, you need to execute the script with administrator permissions (checkbox *run with admin privileges* in the page customization).
  Example: You want to show the number of escalated cases of all the assignees. Therefore, you need the script to retrieve all cases not only the cases which the current user is allowed to see.
- The widget script is executed every time the page with the respective widget is opened. Complex scripts can decrease the system performance. If complex scripts are used, it is recommended to use tabs on the dashboard and place such widgets in a tab which is not opened directly.

The following sections describe the return values of the different widget types.

Chart Widgets

Chart widgets are based on the Highcharts library. The script has to return a HashMap containing the attributes which should be set. See Chart widgets for a detailed description of the available attributes.

The following example shows the script which is used for the default chart widget showing the number of cases in the current view.

```
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 example 73: *Script for the default chart widget ticketsInViewDataWidget.groovy*

The following example shows the script of a funnel widget. Some static data is used for demo purposes.

```
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 example 74: *Script for a sales funnel chart with demo data*

KPI Widgets

KPI widgets are based on the `jquery-kpiwidget` library. The script has to return a HashMap containing the attributes which define the KPI representation. See KPI widgets for a detailed description of the available attributes.

```
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 example 75: *Script for a KPI widget which counts the cases which have been opened during the last week*

Table Widgets

Table widgets are based on the Datatables library. The script has to return a HashMap containing the attributes which should be set. See Table widgets for a detailed description of the available attributes.

```
// 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'],
```

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

```
   [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 example 76: *Script for a table widget*

F.8.1.13 Scripts of Type *Widget Visualization*

Scripts of the type *Widget visualization* enable the use of generic widgets on dashboards.

Example use cases for generic widgets are:

- display data from external applications, e.g., news, weather
- display data from ConSol CM in a customized way

Widget visualization scripts can use resources, e.g., images, stylesheets, JavaScript libraries, or other files, which are stored in the file system. Alternatively, the resources can also be retrieved using URLs.

Steps to use a widget visualization script:

1. Write a script of the type *Widget visualization* in the *Scripts* section of the Admin Tool.
2. Reference the script in the attribute *visualizationScript* of the widget's scope in the page customization, see HTML widgets.

Writing the Widget Visualization Script

Widget visualization scripts have to implement two methods:

- **render()**
  Includes the logic for rendering the field value. Returns the widget content in HTML syntax.
- **resources()**
  Allows to include additional resources. The resources, e.g., images, stylesheets, or JavaScript files, can be stored in the file system. A new folder `resources` can be created in the ConSol CM data directory for this purpose. Alternatively, resources can also be retrieved using URLs.

The methods provide the following parameter:

- **pParams**: The string entered in the *parameters* attribute of the page customization (see HTML widgets).

Example: Display Weather Information

The following example script creates a weather widget using https://weatherwidget.io/.

```
def render(String pParams) {
return """
<div>
<a class="weatherwidget-io" href="https://forecast7.com/en/40d71n74d01/new-york/"
 data-label_1="NEW YORK" data-label_2="WEATHER" data-theme="original" >NEW YORK
 WEATHER</a>
<script>
!function(d,s,id){var js,fjs=d.getElementsByTagName(s)[0];if(!d.getElementById(id))
 {js=d.createElement
 (s);js.id=id;js.src='https://weatherwidget.io/js/widget.min.js';fjs.parentNode.ins
 ertBefore(js,fjs);}}(document,'script','weatherwidget-io-js');
</script><div>
</div>
""" as String

}
def resources(String pParams) {
List<String> resources = [
] as String[];
return resources;
}
```

Code example 77: *Script for a HTML widget displaying weather data*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

In the Web Client, the widget is displayed as follows:

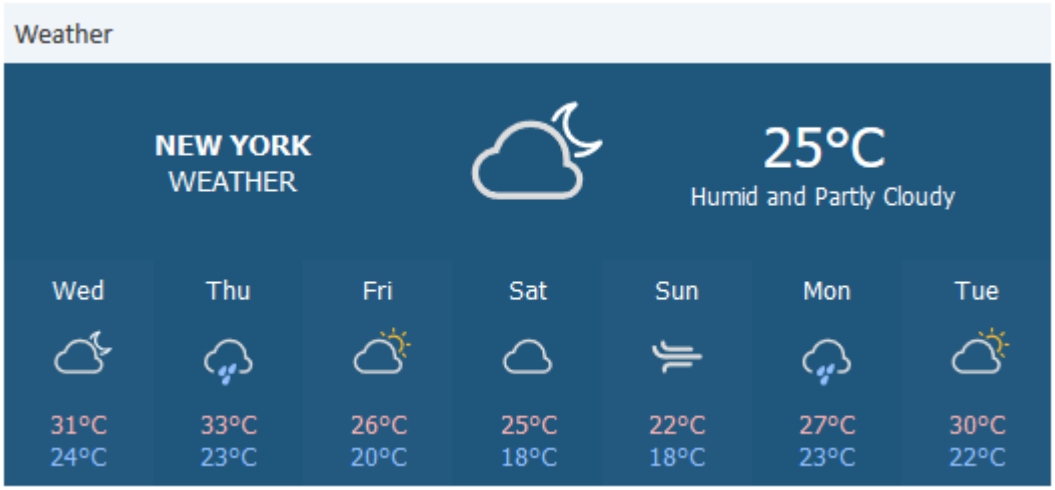


Figure 367: *ConSol CM Web Client - Generic widget on the Web Client Dashboard*

F.8.1.14 Scripts of Type *Workflow*

When using scripts in a workflow, you can either save the scripts within the workflow (in the Process Designer) or reference a script stored in the Admin Tool from within the workflow. Saving workflow scripts in the Admin Tool is useful if the same script is used in several places, e.g., in different activities or workflows. In addition, scripts can be tested more easily during workflow development because changes to scripts in the Admin Tool take effect immediately after saving, so you do not have to deploy the workflow.

Steps to use a workflow script:

1. Write a script of the type *Workflow* in the *Scripts* section of the Admin Tool.
2. Reference the script from the desired workflow item, e.g., a workflow activity, using the Process Designer

   ```
   scriptExecutionService.execute("myscript.groovy")
   ```

   Optionally, you can pass parameters to the script, e.g., the ticket:

   ```
   def params = [ "myticket": workflowApi.ticket ]
   scriptExecutionService.execute("myscript.groovy", params)
   ```

You can find detailed information about workflow programming in the *ConSol CM Process Designer Manual*.

F.8.1.15 PostActivityExecutionScript

The PostActivityExecutionScript mechanism is deprecated and was removed in ConSol CM version 6.12.0. In previous ConSol CM versions, this script was executed after each manual workflow activity. It could be used to define that specific pages (tickets, customers, or resources) in the Web Client should be opened after executing certain workflow activities.



If your ConSol CM system contains a PostActivityExecutionScript after updating to version 6.12.0, the required functionalities need to be re-implemented using the methods of the Action Framework. Please see Overview of the Methods for detailed information.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.8.2 Admin Tool Templates

F.8.2.1 Introduction to Templates in the Admin Tool

In ConSol CM, several types of templates are used:

- **Email templates** are stored in one of the following places:
  - the *Text Template Manager*, see section Text Templates.
  - the *Templates* section of the Admin Tool. Details on that topic are provided in this chapter.
- **Document templates** are stored in
  - the Document Template Manager (part of CM/Doc), see section CM/Doc
- **Data representation templates** are stored in
  - the navigation item *Scripts and Templates* of the Admin Tool. They are used for:
    - the definition of customer templates, see section Templates for Customer Data
    - the definition of resource templates, see section CM/Resource Pool - Templates for Resource Data
- **General templates** are stored in
  - the navigation item *Scripts and Templates* of the Admin Tool. They are explained in this chapter.

In this chapter, the general templates in the navigation item *Scripts and Templates* of the Admin Tool are explained.

Admin Tool templates are written using ***FreeMarker*** notation (see FreeMarker web site) and should only be edited by experienced ConSol CM consultants and administrators. A ConSol CM standard installation already contains system templates and some example templates which might help you, as an administrator, to define new templates for your special use cases.

F.8.2.2 The Admin Tool Template Editor

To work with templates, open the navigation item *Scripts and Templates* in the navigation group *System* and switch to the *Template* tab.

In the templates list, all templates are listed with:

- **Name**
  Mandatory. A template is referenced by its name when it is referenced by other objects.
- **Group**
  Optional. Groups help you temporarily sort the templates in the templates list. This setting does not have any technical implications and will be lost once you reload the data in the Admin Tool.

To open a template in the editor panel, mark it in the list and open it by clicking the *Edit* button. Each template must have a name, whereas the group name is optional.

If your system works with various languages, you can define each template for each language. Use the drop-down menu *Language* above the editor panel. The Web Client will display the template for the configured locale of the web browser. If there is no template for this language, the default language will be used. Every template has to be defined for the default language.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

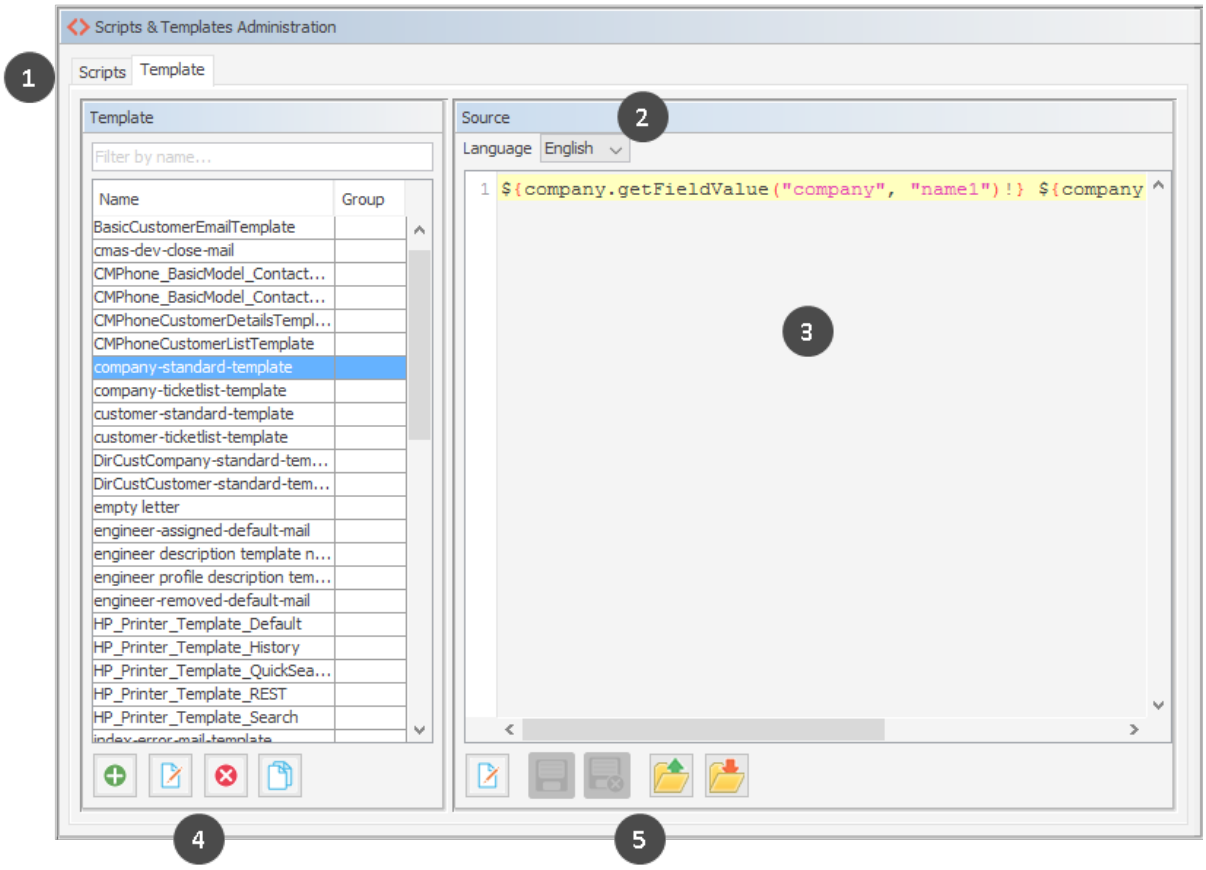


Figure 368: *ConSol CM Admin Tool - System, Scripts and Templates: Template editor*

The *Templates* screen shows the following items:

- Tabs to switch between scripts and templates (1)
- Language selector (2)
- Editor panel (3). Displays the template which is selected in the list.
- General buttons to add, delete or copy a template and to edit the template name and group (4)
- Buttons referring to the open template (5)
  - Edit the selected template
  - Save the template
  - Cancel (quit without changing the changes)
  - Upload a text file (as template) from the file system
  - Save the template as a file in the file system

F.8.2.3 Working with Admin Tool Templates

The Admin Tool templates represent a template pool. Each template can be referenced from different modules of the system and is always referenced by its name. In the following paragraphs, all modules where templates can be used are explained. Within a template, the data fields are referenced by group name and field name.

The following example shows how the field *company_name* within the customer field group *Reseller-CompanyData* is referenced.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

```
${ResellerCompany.getFieldValue("ResellerCompanyData","company_name")!}
```

For a detailed explanation of working with ticket fields, please see section Ticket Fields (Setting Up the Ticket Data Model). Customer fields are explained in section Setting Up the Customer Data Model. Resource fields are explained in section CM/Resource Pool - Setting Up the Basic Resource Model.

⚠ Do not use line breaks in template statements!

System Templates

A default ConSol CM installation comes with several system templates. They are used in standard situations like error messages sent to an administrator. Please see the following list for an overview of the system templates:

- **attachment-type-error-mail-template**
  An email with this template is sent to the email administrator (email address given in system property cmas-nimh-extension, mail.on.error) when the attachment type of an incoming or outgoing email is not supported and thus the email cannot be processed.
- **engineer description template name**
  Template used to render the engineer label, e.g., ticket owner. The template name “engineer description template name” is the default value. You can define another name for the template which should be used to render engineer names by using the CM system property cmas-core-server, engineer.description.template.name.

  If you need the engineer name as determined by the template in scripts, you can use the method `getDisplayName()` from the class `EngineerService`. The following example shows how to print the engineer name in German to the log files.

  ```
  log.info engineerService.getDisplayName(engineer, new Locale("de"));
  ```

- **engineer profile description template name**
  Template used to render the label on a header of the page, next to the logout button.
- **mail-receive-too-large**
  Template for the notification email which is sent when an incoming email could not be processed due to its size. The sender of the email receives a notification which shows the size of the email which he tried to send and the maximum allowed size.

  ⓘ Add the parameter `From:` with the desired email address to the template to use another address than the value of the system property cmweb-server-adapter, mail.-from as a from address.

- **mail-send-failure-template**
  Template for the notification email which is sent when a manual email could not be sent due to problems with the email server.
- **password-reset-template**
  Template for the body of the email which is sent when a user requests a password reset (on login page).

- **representation_info_email_html**
  All emails sent by CM to the represented engineer are also sent to the representing engineer (see *Global Permissions: Representation Permissions* in section Roles). The template is used to configure the text which is added to the forwarded email.
- **representation_info_email_plain_text**
  Same as above, as plain text.
- **representation-create-email**
  The template for the email which is sent when a new representation configuration has been set up, e.g., when an engineer has selected a colleague of his as representing engineer, because he will be on vacation for some time. The email is sent to both, the new representing engineer and the engineer who is represented. All engineer-specific properties are available as variables in the template, for the representing engineer, e.g.:
  - Last name: ${representing.lastname}
  - First name: ${representing.firstname}
  - Login: ${representing.name}
  - Email: ${representing.email}
  - Fax: ${representing.fax}
  - Mobile: ${representing.mobile}
  - Phone: ${representing.phone}

  The variables for the engineer who is represented are built according to the same pattern, e.g., Last name: ${represented.name}.
- **representation-delete-email**
  The template for the email which is sent when a representation configuration has been stopped, e.g., when an engineer has deleted the representation configuration for a colleague of his as representing engineer, because he has returned from vacation. The email is sent to both, the old representing engineer and the engineer who was represented. For variables which are available, please see the previous paragraph.

Templates to Define the Display of Customer and Resource Data in the Web Client

The representation of **customer data** and **resource data**, the so called **display name**, used in the different sections of the Web Client can be defined using templates. The definition has to be made for each customer object and resource type. The template can use the available customer or resource fields for the respective object.

The following example shows the use of a template to format the contact name in the ticket list, so that the first name and last name of the contact are displayed.

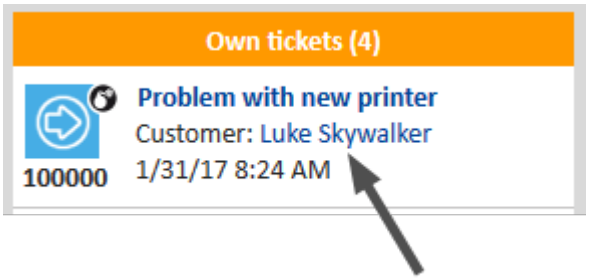


Figure 369: *ConSol CM Web Client - Example of a customer template (ticket list)*

The following template is used:

```
${ResellerCustomer.getFieldValue("ResellerCustomerData","forename")!}
 ${ResellerCustomer.getFieldValue("ResellerCustomerData","customer_name")!}
```

Code example 78: *Customer template*

Please see Templates for Customer Data and CM/Resource Pool - Templates for Resource Data for detailed information about the available template types and the assignment of the templates.

Localizing Enum Values in Templates

It is possible to display localized enum values in templates by using the `localize()` method. This method retrieves the localized enum value as defined in the Admin Tool to display it in the Web Client. Otherwise, the technical enum value would be displayed.

The following example shows a template for a resource. The resource field *SLA_country* is an enum containing a list of countries. The `localize()` method is used to display the country name in the respective browser locale.

```
${resource.getFieldValue("SLA_Fields_basic","SLA_Name")!} (${localize
 (resource.getFieldValue("SLA_Fields_basic","SLA_country"))!})
```

Abbreviating Values in Templates

It is possible to abbreviate the values retrieved by a template using the `abbreviate()` method. In this way, long values are shortened, so all fields are displayed even if the first field contains a long value.

The `abbreviate()` method needs two parameters:

- The value which should be abbreviated.
- The number of characters to be displayed. Please note that this is the number of all characters which are displayed, i.e., including the three dots. Thus, for example, to display five letters and three dots, use “8” as a parameter. The minimum value is “4”.

The following example shows the template for a company and how the abbreviated company name is displayed in the ticket list.

```
${abbreviate(ResellerCompany.getFieldValue("ResellerCompanyData","company_
 name"),8)!} - ${ResellerCompany.getFieldValue("ResellerCompanyData","company_
 number")!}
```

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH



Figure 370: *ConSol CM Web Client - Abbreviated value in ticket list*

Using Links

You can use the methods `linkInWebClientTo()` and `linkInTrackTo()` to add links to tickets, customers and resources to a template. The following methods are available:

- Link to a ticket in the Web Client : `${linkInWebClientTo(ticket)}`
- Link to a customer in the Web Client: `${linkInWebClientTo(unit)}`
- Link to a resource in the Web Client: `${linkInWebClientTo(resource)}`
- Link to a ticket in CM/Track: `${linkInTrackTo(ticket)}`

The following example shows how to use these methods:

```
[#assign ticketUrl = "${linkInWebClientTo(ticket)}"]
<p>This is a link to ticket: <a href="${ticketUrl?no_esc}">ticket</a></p>
```

Handling Missing Values in Templates

You can use if statements in combination with the `??` operator or the `?has_content` operator to check if an object exists or a field has a value for the given customer or resource.

```
<#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 example 79: *Example of a customer template with if statement (has to be written in one line!)*

Alternatively, you can specify a default value using the `!` operator to avoid exceptions due to missing values. The default value can be empty.

```
${ResellerCompany.getFieldValue("ResellerCompanyData","company_name")!} -
 ${ResellerCompany.getFieldValue("ResellerCompanyData","company_number")!}
```

Code example 80: *Example of a customer template with empty default value (has to be written in one line!)*

Setting Number Format Patterns in Templates

Freemarker allows to set a number format pattern.

```
<#setting number_format="0.#"/>${customerModelCompany.getFieldValue("groupName",
 "numberValueField")!}
```

Code example 81: *Setting a number format, so only one decimal place is displayed*

Ticket Assignment Templates

Engineers are notified in automatic emails when a ticket is assigned to them or they are unassigned from a ticket. The templates used for these emails are called ticket assignment templates.

These templates are defined per queue in the Queues. There are templates for the use cases **assign** and **remove**.

- The **assign** template (*Assign*) is used as text template for an automatic email which is sent by the system to the (new) engineer when a ticket is assigned to the engineer.
- The **remove** template (*Unassign*) is used as text template for an automatic email which is sent by the system to the (old) engineer when a ticket has been taken from the engineer.

You have to write and save the templates in the *Template* section first. Then they will be available in the drop-down menu in the *Ticket assignment templates* section of the queue details (see section Queues). You can define a name of your choice - we recommend using a name which describes the use of the template.

As default `From` address, the email address of the CM administrator (CM system property admin.email) is used. You can also set the from address explicitly within the template (see following example).

The following example shows a ticket assignment template as defined in the *Templates* section.

```
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 example 82: *Content of an assignment template*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Password Reset Templates

When users forgot their passwords, they can request a new password using ConSol CM standard functionality. The user might be an engineer who works with the Web Client or a customer who has a login to CM/Track. For both cases, the administrator has to configure ConSol CM in a way that an email can be sent to the user who needs a new password. This email is based on a template which is stored in the Admin Tool, in the *Template* section. The email contains a hyperlink to a page where the password reset can be performed.

Password Reset Template for Engineers in the Web Client

When an engineer has forgotten his Web Client password, he can request a new password by using the link *Forgot your password?* on the initial login page. An email with a link to an URL where the engineer can set the new password is sent to the engineer.

> ⚠ Please note that this can only work if a valid email account is available for this engineer and if the respective value has been entered as email address for the engineer in the engineer data!

The email which is sent to the engineer is based on the template `password-reset-template`. This template name is mandatory, required for the password reset to work. The template could look like the following example:

```
Subject: Password reset procedure
From: no-reply@consol.de

Dear ${engineer.firstname} ${engineer.lastname},
Please click the following link to reset your password:

http://localhost:8888/cm-client/passwordChange?resetCode=${resetCode}

This link expires at ${expirationDate?string("yyyy.MM.dd HH:mm:ss")}.
```

Code example 83: *Template to reset the password for the Web Client*

The `From` address of the email which is sent to the engineer is defined by the CM system property cmas-core-security, password.reset.mail.from.

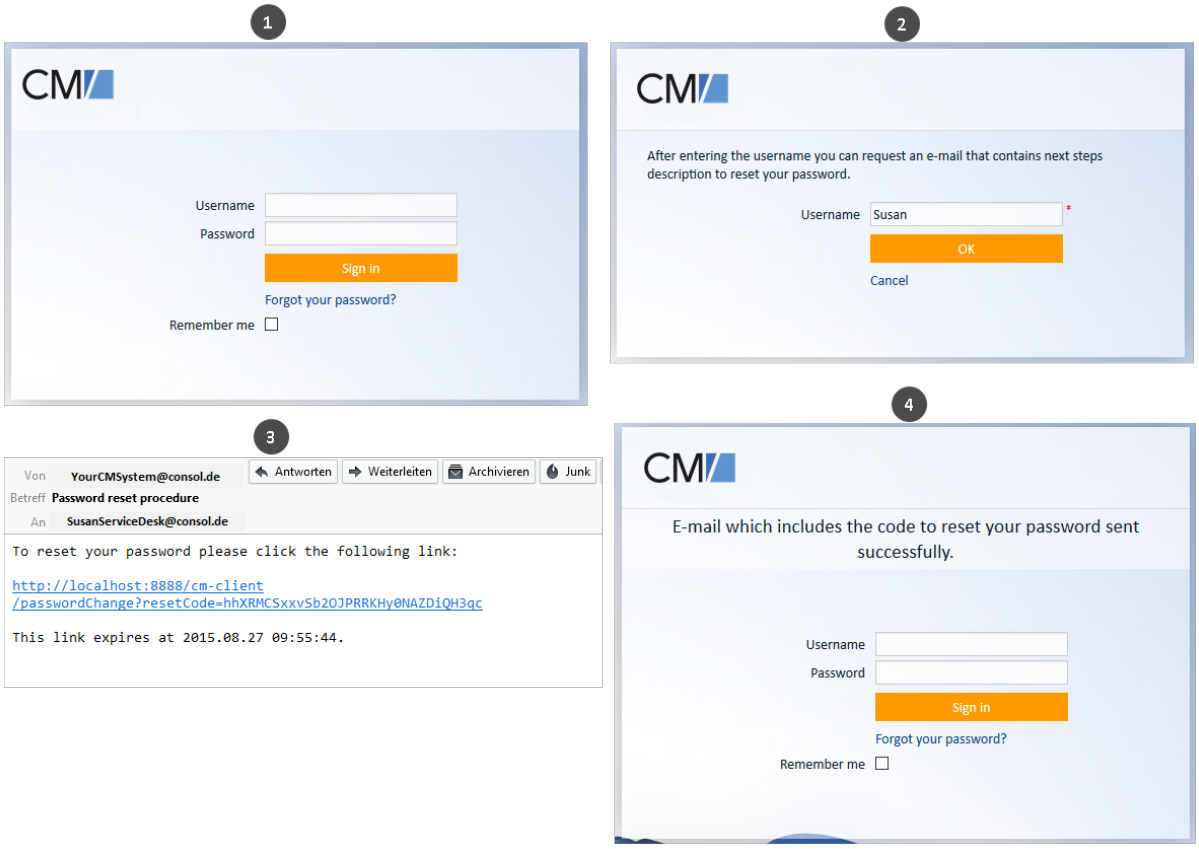


Figure 371: *ConSol CM Web Client - Password reset by an engineer*

> Please note that the password reset in the Web Client is only possible if the standard mode is used. It is not possible if LDAP or Kerberos authentication is in operation. See section Authentication Methods in ConSol CM for an explanation of all possible authentication modes.

Password Reset Template for Customers Using CM/Track

When a customer has forgotten his CM/Track password, he can request a new password by using the link *Forgot your password?* on the initial login page. An email with a link to an URL where the customer can set the new password is sent to the customer.

> Please note that this can only work if a valid email account is available for this customer and if the respective value has been entered as email address for the customer in the respective customer field.

The email which is sent to the customer is based on the template `track-password-reset-template`. This template name is mandatory, required for the password reset to work. The template has to be created/added manually, it will not be present in the system by default.

The template should be formatted like the following example (you can add any text you would like to send to your customers, but please note the `resetCode` variable and the URL!):

```
Subject: Your Password Reset Link
From: no-reply@consol.de

Dear ${unit.firstname} ${unit.lastname},
Please click the following link to reset your password:

${urlTrack}/#/password-reset/resetCode-${resetCode}

This link expires at ${expirationDate?string("yyyy.MM.dd HH:mm:ss")}.
```

Figure 372: *Template to reset the password for CM/Track*

The following variables are configured using system properties:

- **expirationDate**
  By default, the link expires 24 after the password reset request was made. To change this value, create the system property cmas-core-security, resetCode.expirationPeriod with the desired expiration date.
- **urlTrack**
  Value of the system property cmas-core-server, url.track.

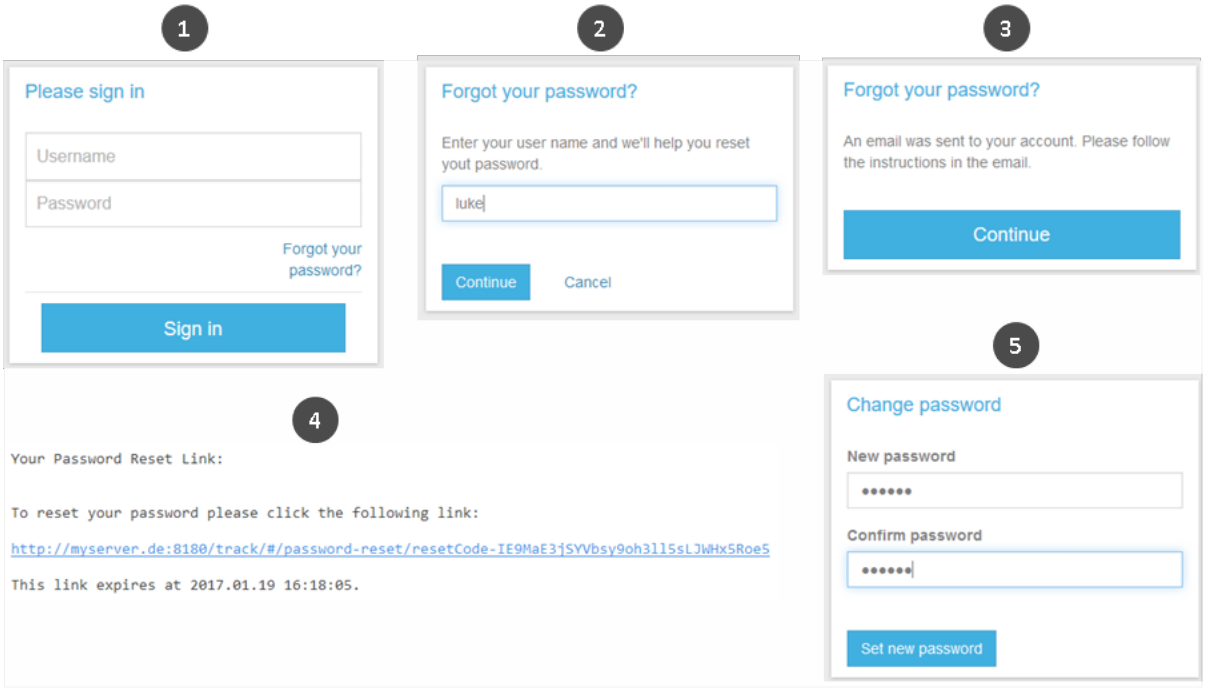


Figure 373: *CM/Track - Password reset by a customer*

Please note that the password reset in CM/Track is only possible if the DATABASE mode is used. It is not possible if LDAP authentication is in operation. See section Authentication Methods for Customers in CM/Track for an explanation of all possible authentication modes.

Custom-Defined Templates

You can define any template that is required and store it in the *Templates* tab. The API methods `renderTemplate()` and `mergeAndInlineImages()` can be used to reference a template when used in automatic emails which are sent in a workflow activity. However, most email templates should be managed using the *Text Template Manager* (see section Text Templates). There are only very few use cases which might require that email templates, or parts of email templates, have to be stored in the *Templates* tab of the Admin Tool.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.9 Deployment (Import/Export)

This chapter discusses the following:

F.9.1 Introduction 619

F.9.2 Scenarios 619

F.9.3 Deployment (Import/Export) Using the Admin Tool 620

F.9.1 Introduction

ConSol CM offers the option to export the system configuration with or without run-time data into a file and to import this file into another ConSol CM system. The transfer file is called a *scenario* (sometimes also called a *scene*). It can contain various data - details are explained in the following sections.

Usually a scenario with configuration data is used to transfer data from a test to a staging or production system.



Since ConSol CM works with transfer keys for all objects it is highly recommended to transfer test to staging or production environments by using a scenario, and **not** to implement the same functionality in both systems. In case a transfer (export/import) is performed later on, there will be duplicate objects! Please read the detailed explanations below.

F.9.2 Scenarios

A scenario is a file in a proprietary ConSol CM format (similar to `.zip`, `.jar`, and `.tar`) that contains the data of a ConSol CM installation. It can be exported from one ConSol CM system and imported into the same or another system. This can be very helpful, e.g., when a test scenario is built on a test system which can then be transferred to a production server.

When an export file is created (see detailed explanation in the sections below) the administrator can decide which data should be included.

A scenario will **always** contain:

- all customer-specific system properties,
  i.e., system properties where the module name starts with *custom-*, please see section CM system properties for details!
- all page customizations
- all REST client GUI configurations

A scenario **can** contain, depending on the selection of the administrator (see figure below):

- run-time data
- configuration data

A scenario will **never** contain:

- general (not customer-specific) system properties (e.g., mail server, LDAP directory, etc.)

F.9.3 Deployment (Import/Export) Using the Admin Tool

In the Admin Tool, the deployments are managed using the navigation item *Import/Export* in the navigation group *System*. In this navigation item you can import or export scenarios (i.e., the whole configuration or part of it) in an application-specific format. You usually do this to transfer data between different ConSol CM installations. A typical example is transfer of the configuration from a test system to a production system.

All export/import actions are logged in the file `transfer.log`. A detailed description of this file is provided in the section Logging and Log Files.

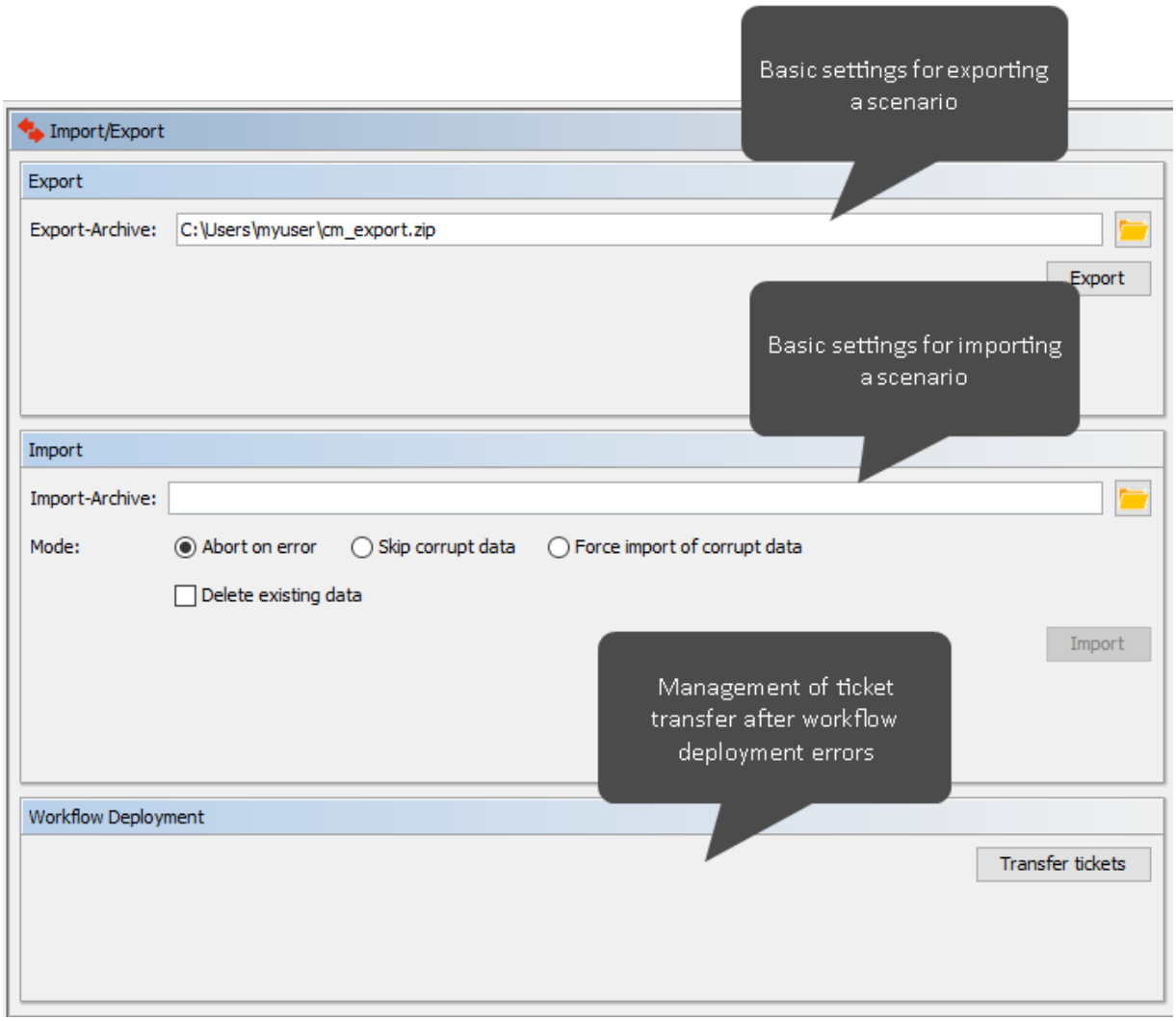


Figure 374: *ConSol CM Admin Tool - System, Import/Export*

> ⚠ The import of external data can modify, overwrite, or delete existing data irrecoverably. Although the user is prompted for confirmation at critical points during deployment, this cannot prevent erroneous handling. Use this function **only** if you are very sure what you are doing. In case of doubt please ask the ConSol CM support team or a ConSol CM consultant for assistance.

F.9.3.1 Export

- **Export-Archive:**
  Path to the export archive. When you open the navigation item *Import/Export*, the default path for the export (`<home directory of the admin user>/cm_export.zip`) will be set. Use this value or enter the path and name of the file you want to create. Alternatively, you can click the *Folder* button to open the file browser.

Click on *Export* afterwards to start the data export.

You will have to select the data that should be included in the export file (scenario). The following options are available:

- **Runtime data**
  This refers to data that is stored as operating data, e.g., tickets, resource data, and customer data.
  - **All**
    Ticket data, customer data, is exported completely **and** the complete configuration is exported. When you select the checkbox *All*, all other checkboxes are selected automatically. If CM/Resource Pool is active, the resource data is also exported.
  - **Single ticket**
    If the checkbox *Single ticket* is ticked, you have to insert the ticket name in the input field. (The ticket name is indicated below the ticket icon in the Web Client.)
    Tick the checkbox *Anonymized* to anonymize (hash) the exported data. In this way, no real customer, engineer, or resource information will be transferred to the importing system. The following data will be anonymized:
    - Content of ticket fields (some values as enum aren't anonymized)
    - Ticket attachments
    - Ticket history
    - Content of customer fields
    - Unit history

    > The *Single ticket* option is not supposed to be used for regular system operation, but it is a functionality which is offered to facilitate bug-tracking. The export produces a .zip/.jar file containing the ticket data in XML files. All data required for the ticket (i.e., all related objects) are exported with the ticket, so be careful when the ticket is imported into another system - data might be changed here!

  - **All without tickets**
    The complete installation beside the tickets is exported, i.e., the customer data **and** the complete configuration. When you select the checkbox *All without tickets*, all other checkboxes except for *All* are selected automatically. If CM/Resource Pool is active, the resource data is also exported.
  - **Only customer data**
    Only customer data (i.e., the customer data model and the actual customer data) is exported. Nothing else. (The checkbox *Customer model* is checked automatically.)

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Only resource data**
  Only resource data (i.e., the resource data model and the actual resource data) is exported. Nothing else. (The checkbox *Resource model* is checked automatically.)

- **Configuration data**
  This refers only to the configuration in the Admin Tool, no run-time data is exported. Depending on the selection you have made, other check boxes might be checked automatically in order to cover all dependencies within the exported data.
  - **All**
    The complete configuration is exported. When you select the checkbox *All*, all other checkboxes under *Configuration data* are selected automatically.
  - **Enum definitions**
    The definitions and localizations of sorted lists (enums) are exported. This represents the data which is defined in the navigation item *Enums* (navigation group *Lists*). For details about enums, please refer to section Managing Sorted Lists: Enums.
  - **MLA definitions**
    The definitions and localizations of MLAs (multi level attributes) are exported. This represents the data which is defined in the navigation item *MLAs* (navigation group *Lists*). For details about MLAs, please refer to section MLAs.
  - **Page Customization**
    Only the page customizations which have been defined specifically for the system (i.e. not the default values) are exported. These are the values which are stored in the database tables `cmas_web_customization` and `cmas_web_customization_values`. Default values are exported only if they were set to another value and have been changed back to the default value manually by clicking the *Update* button. For a details overview of this topic, please refer to section Page Customization.
  - **Engineers**
    Only the engineers with their data are exported. This also includes the roles the engineers have been assigned.
  - **Admin Tool templates**
    Only the Admin Tool templates (see section Admin Tool Templates for details) are exported.
  - **Scripts**
    Only the Admin Tool scripts are exported (see section Admin Tool Scripts for details).
  - **Text templates**
    Only the templates that are stored in the Text Template Manager (see section Text Templates for details) are exported.
  - **Document templates**
    Only the Microsoft Word or OpenOffice templates are exported, this is only relevant if CM/Doc is in operation (see section CM/Doc for details).
  - **Customer model**
    Only the customer fields that are used to define the customer model are exported. No run-time customer data is included.
  - **Resource model**
    Only the resource fields that are used to define the resource model are exported. No run-time resource data is included.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Classes of Text**
  - If set to “true”, all classes of text which are available in the CM system will be exported.
  - If set to “false”, only the classes of text which have been assigned to a queue which is included in the export will be exported. As a consequence classes of text without queue association will not be exported at all in this case.
- **Custom forms**
  Only web forms are exported (see CM/Forms for details).
- **Queue-related and other data**
  Only queue configuration and general configuration settings are exported (workflows, queues, ticket fields, enum values, MLAs, roles, views, system properties, ...), in short: everything which is not included above. Transfer the names of all queues you would like to include in the export file (scenario) into the list with *Exported Queues*.
- **System documentation**
  Only the system documentation is exported. The export file can be used to generate a system documentation in the navigation item *Documentation Generator*, see System Documentation.

- **Workflow Export**
  This panel is activated when you have selected *All* or *Queue-related and other data* in the *Configuration data* panel.
  The selection in the *Workflow Export* panel might influence the size of the exported scenario considerably! You might want to delete some workflows before you export a scenario which contains workflows.
  - **Only latest/installed versions**
    Only the installed versions of the workflows will be exported, i.e., the versions which are currently deployed and active in the system. All older versions will not be exported.
  - **All versions**
    All versions, of all workflows will be exported, currently deployed and older versions. Please be aware that checking this check box might cause the export to take very long and to a huge scenario, depending on the number of workflows which are stored in the system.

If you would like to export the complete configuration, select *All* in the *Configuration data* section. The export/import of subsets (e.g., templates only) is usually applied when selected data (e.g., from a test environment) has to be transferred to another (e.g., live) system.

F.9.3.2 Import

General

⚠ Please note that during the import, the DWH “LIVE” mode must be switched off.

The *general principles* of ConSol CM scenario import are:

- If the checkbox *Delete existing data* has **not** been **selected**, the scenarios are **merged**, as described here:
  - Data are only **added**, nothing is deleted.
  - If the imported scenario contains the **same field/parameter** as the original scenario, the value from the imported one **overwrites** that of the original scenario.
    Example: For the field *priority*, the imported scenario has the annotation `position` = “0;2”. The original scenario contains the value `position = 2;2` for the field *priority*. Thus, in the resulting scenario after the import, the value for `position` is “0;2”.
  - If the imported scenario contains **more parameters** than the original scenario, the parameters are **added** to the original one.
    Example: The imported scenario has the annotation `visibility` = “none” for the field *PersonID*. In the original scenario, the field *PersonID* is present, but does not have the annotation. Thus, in the resulting scenario after the import, the field *PersonID* will have the annotation `visibility` = “none” and will thus be invisible.
  - If the imported scenario contains **less data/fewer parameters** than the original one, the original data will be present in the resulting scenario. **Nothing is deleted**.
    Example: If the field *PersonID* in the imported scenario no longer contains the annotation `visibility` = “none”, but the original scenario does contain the annotation, it will remain. Thus, in the resulting scenario the field *PersonID* is still invisible.
  - For **scripts and templates**, the **latest version** (according to the time stamp) is used, no matter from which scenario.
  - **Objects** are identified by an **internal key** (*transfer key*).
    When an imported scenario contains an object with the same name but another transfer key, technically, these are two objects, and the new object will be added from the import to the original scenario (e.g., when a user *Mr. Miller* exists in both scenarios, there will be one user *Mr. Miller* and one user *Mr. Miller (1)* in the resulting scenario after the import.
    To make sure you can transfer another import scenario from the same source (test system), you can delete the original *Mr. Miller* user and transfer the tickets to *Mr. Miller (1)*, an operation that is supported by the ConSol CM Web Client. Then rename *Mr. Miller (1)* to *Mr. Miller*. Now, the *Mr. Miller* user has the transfer key that originated in the import scenario and during the next import there will be no problem.
    The general use case is: The transfer key is created by the ConSol CM system and allows the re-import and/or the update of the configuration data.
  - **CM System properties** which start with *custom-* are added (i.e. created) in an installation during import if they did not exist before. In case the custom-property is already present in the importing system, the value is updated, i.e. the imported value is set. This behavior is implemented in CM versions 6.11.0.5 and up. In versions 6.11.0.4 and below, the value was not overwritten.
- If the checkbox *Delete existing data* has been **checked**, the entire system is deleted, i.e., **all** existing data are **deleted**. All data means:
  - Configuration data
  - Run-time data

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

> ⚠ That means if *Delete existing data* has been selected, it is not possible to preserve anything from the original scenario. **Everything is deleted! Only** system properties are **not** deleted.

- Import of runtime data:
  - Export and import of runtime data is designed for the use with test systems or development systems. For data export and import on a large scale, ETL tools have to be used and **not** the export/import mechanism!
  - If an object already exists (identified by its transfer key) in the target database, the object will by no means be modified by the import, i.e. no fields of the object are added or deleted, and existing fields are not modified. In this way, existing objects (tickets, customers, resources) are protected from being modified accidentally with test data.

The following parameters have to be set for an import operation:

- **Import-Archive:**
  Enter path and file name of the archive from which the data shall be imported. Alternatively, you can click the *Folder* button to open the file browser.
- **Mode:**
  Here you can choose what the import should do if an error occurs:
  - **Abort on error**
    This mode is recommended for production systems.
  - **Skip corrupt data**
    This mode is recommended for imports into test systems. It might even be reasonably applied to production systems, because an unexpected error can lead to a corrupt system, but the import continues even if an error occurs. The problem can be probably handled quickly afterwards, whereas a new import might take longer to perform.
    Example: A referenced object is not found during the import of a view which references a queue which cannot be found.
  - **Force import of corrupt data**
    Choose this mode only if you want to clone a system with corrupt data, e.g., on a development server or if the support team is performing error analysis.

Click on *Import* afterwards to start the data import.

> ⓘ Runtime data are only imported into the target fields if these fields are empty. If a field contains an entry, this entry will not be overwritten during an import of a scene. Please use ETL (extract - transform - load) operations for export/import of runtime /operating data! This is not a use case for ConSol CM scene export / import.

Configuring System Behavior During Scene Import

The import of the configuration data in a scene is separated into several database transactions. This helps limit memory usage on the database server. However, in case of fatal errors during the import this could result in an inconsistent system state.

A ConSol CM system property can be used to configure the system in a way that all configuration items of a scene import are encapsulated in one large database transaction. Therefore, when enabling this option, a fatal error during import of a configuration item cannot leave the system in an incon-

sistent state. Rather the whole encapsulating transaction will be rolled back putting the system configuration-wise into the state before the import. The transaction encapsulation does not include localization strings or runtime data which are handled independently of this setting.

Use the ConSol CM system property cmas-core-server, config.import.global.transaction.enabled.

> Enabling this transaction encapsulation may result in longer import execution times and higher memory requirements on the database server!

F.9.3.3 Transformation of User Name and Password Fields During Import into CM 6.11

In ConSol CM version 6.11, the behavior of two customer fields (former data object group fields) has changed:

- The customer field which is annotated with `username`.
  The user name is now controlled to be unique.
- The customer field which is annotated with `password`.
  The passwords are only stored as encrypted strings.

Thus, when you import a scenario from a ConSol CM system with a version lower than 6.11 to a system running version 6.11 (or higher), two transformations will be performed automatically. The checkbox *Delete existing data* for the import operation does not have any influence on this behavior.

- **username:**
  The customer field which is annotated as `username`will be used. If several identical user names are found in the customer data, the later ones will be appended by a number like *Huber_1* for the second original username *Huber* and so on. The dialog after the import will inform about these user name changes.
  Please note that user names can be defined as case sensitive in CM versions 6.11 and up. Use the CM system property cmas-core-security, policy.track.username.case.sensitive.
- **password:**
  The customer field which is annotated as `password` will be used. The passwords will be stored as encrypted strings using the new mechanism.

F.9.3.4 Workflow Deployment (for Deployment Error Recovery Only)

Usually, all operations concerning workflow design and deployment are performed using the ***Process Designer***. However, if an error occurs during workflow deployment, you can transfer the tickets that could not be transferred into the new workflow using the following options.

First select the queue(s), then choose the transfer mode:

- **Remain at last activity**
  The ticket will try to stay at its position in the process:
  - If the activity and scope have not been changed, i.e., no change in position for the ticket.
  - If the activity is no longer present, i.e., the ticket goes as far back in the process as necessary to find the last consistent position in the process.
- **Restart process**
  The ticket goes back to the *START* node of the process/workflow.

Please also read the detailed explanation of the workflow deployment process in the *ConSol CM Process Designer Manual*.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.10 License Management

F.10.1 General Information about Licenses in ConSol CM

A ConSol CM license file is a text file which contains entries for several modules. For each module, the number of valid licenses is indicated. For example, the following excerpt of a license file shows the ConSol CM Web Client, CONCURRENT_USERS section. Ten licenses have been purchased.

```
[CONCURRENT_USERS]
contractParty = Demo-Licence ConSol
products = WEB_CLIENT,REST
version = 6.10
expirationDate = 31.12.2016
licenses = 10
signature = XXX
```

ConSol CM works with **concurrent users** (sometimes also called floating licenses), i.e., the number of users who are logged in simultaneously is registered, no user names are checked. That means the number of engineers who are managed in the Admin Tool (see section Engineers) does not have to be identical to the number of Web Client licenses.

A license is consumed when the user logs in. The license is handed back to the server when the user session is terminated, i.e., when the user logs out or when the user session is terminated automatically by the server because the session timeout has been reached (see system property cmas-core-server, server.session.timeout).

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.10.2 Sections of a License File

A ConSol CM license file can contain the following sections. All licenses are **concurrent** licenses, see explanation above.

- [ADMINTOOL_USERS]
  The number of users who can log in to the CM Admin Tool.
- [CONCURRENT_USERS]
  The number of CM engineers who can log in to the Web Client
- [PROCESS_DESIGNER]
  The number of users who can log in to the CM Process Designer
- [TRACK]
  The number of customers who can log in to the portal CM/Track
- [TRACK_USERS]
  The number of user profiles for the portal CM/Track. This is the number of engineers who are marked as *Track* in the Engineers.
- [REST_USERS]
  The number of users who can access the REST API. The number of TRACK_USERS is not included in this number (CM/Track also uses the REST API.)
  CM/Phone will also consume REST licenses, one license per client (PC/laptop) where CM/Phone is installed and active.



Since several (or even a great number of) customers can use the same user profile in CM/Track, the license numbers of [TRACK] and [TRACK_USERS] might differ considerably. For example, there might be two user profiles and 1000 customers should be allowed to log in to CM/Track at the same time. This would mean 2 [TRACK_USERS] licenses and 1000 [TRACK] licenses. Please read section CM/Track: System Access for CM/Track Users (Customers) for a detailed explanation.

F.10.3 Managing the ConSol CM License Using the Admin Tool

You have to import a valid license for your ConSol CM system in the navigation group *System*, navigation item *License*. You will receive a license for a test and/or a productive system when you have signed the respective software contracts with ConSol. If the license has expired or will expire soon, you can import a new license file. Of course the Admin Tool will always start, even if the license has expired. In the Web Client, in CM/Track, and in the Process Designer, the login is not possible when the license has expired.

Please ask your consultant for details. The license is a plain text file. The license can be modified during ConSol CM operation, no system downtime is required.

> ⚠ There is no *Back* button to undo changes with one click when you enter or delete text in the *License* field. If you accidentally change parts of the license, close the Admin Tool **without** clicking *Save*. This will discard all changes you made to the license text. When you restart the Admin Tool afterwards, the license will be in the same condition as it was before you made the changes.

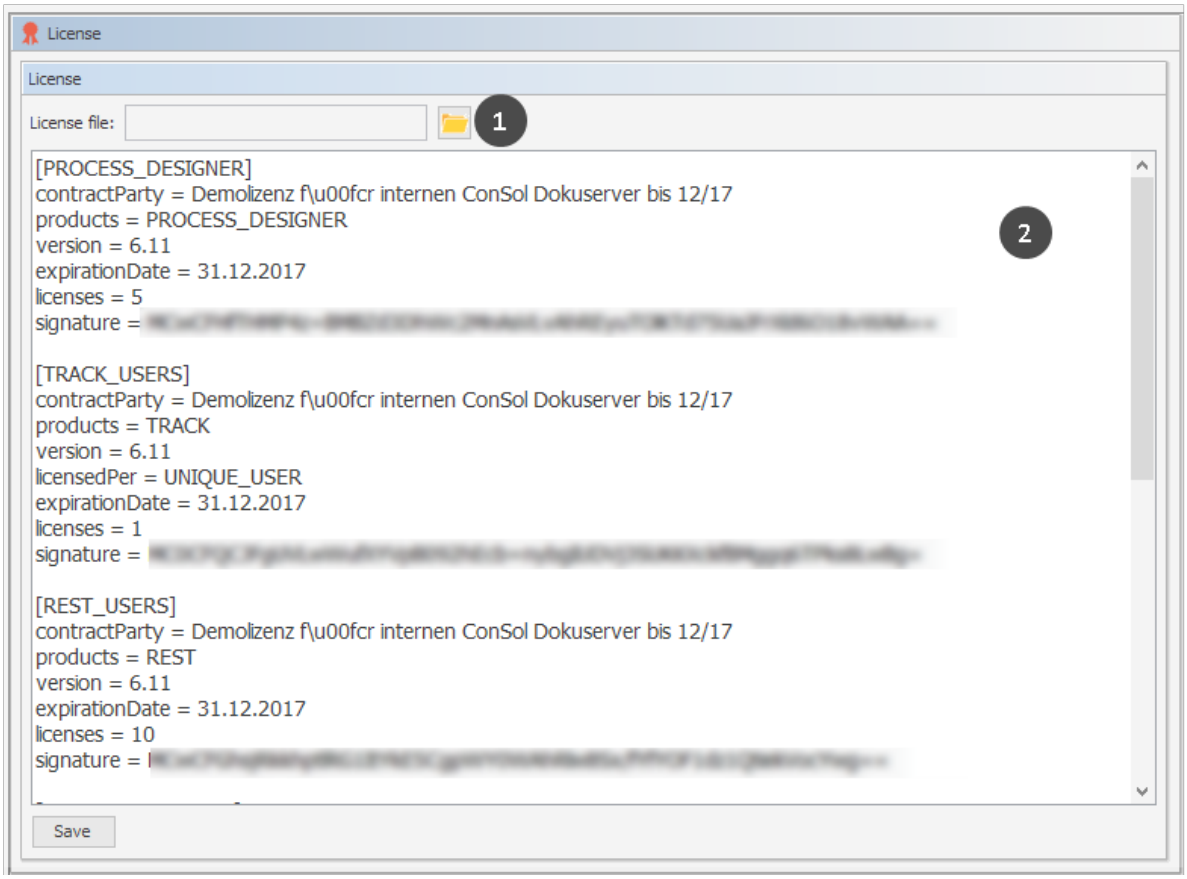


Figure 375: *ConSol CM Admin Tool - System, License*

Choose one of these two ways to import your ConSol CM license file:

- Load the license using the file browser next to the field *License file*. Click on *Save* (1).
- Insert the entire text of the license file via copy and paste. In case an old license is present, just replace the entire text. Click on *Save* (2).

You should receive a message that the license has been imported into the system successfully. It goes into effect immediately, without further action.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.10.4 Expert Information about Accessing Content of CM Licenses

The content of CM licenses can also be queried using the MBean `licenseDeployer`. This MBean offers three methods:

- getRemainingDays
- deployLicence
- getLicenseInfo

If you are interested in using this feature and would like to have support implementing a system which uses the MBean access, e.g., to set up monitoring for your CM system, please contact your CM consultant.

The following figure shows an example using the JConsole.

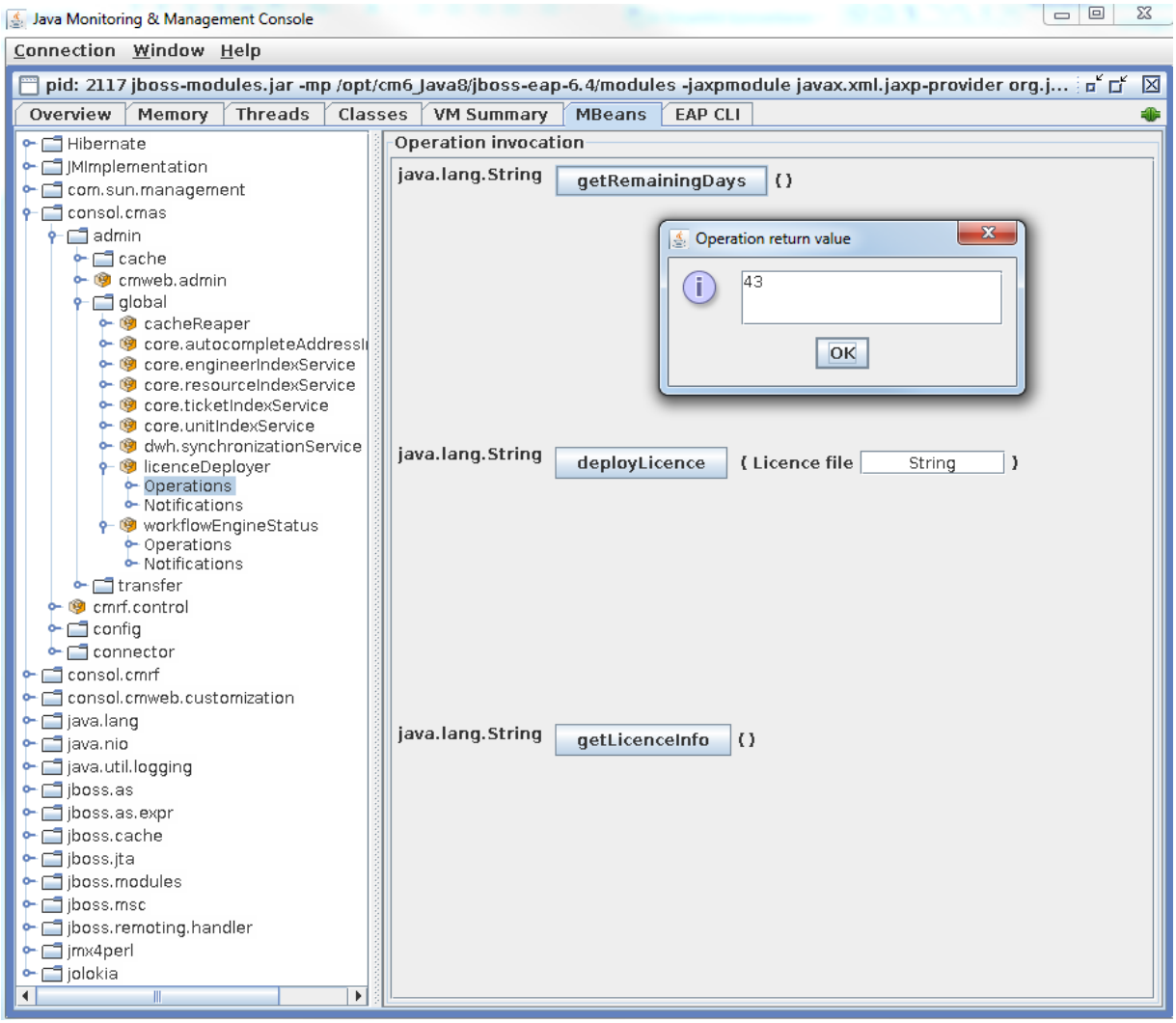


Figure 376: *Retrieving the remaining days of a CM license using the JConsole*

> In case you have purchased a CM license with a limited period of validity, we recommend to set up a monitoring for the license which sends a notification a certain time before the license expires.

F.11 System Properties

This chapter discusses the following:

F.11.1 Introduction 632
F.11.2 System Property Overview 633
F.11.3 Setting System Properties 635
F.11.4 Programming with System Properties 636

F.11.1 Introduction

In the navigation item *System Properties* in the navigation group *System*, you can add, modify, or delete system settings, so called *system properties*, for the ConSol CM application. System properties are used to store, for example, values for the number of seconds for a session timeout, for admin email addresses, or for the configuration of the search page size.



DO NOT WORK ON THIS NAVIGATION ITEM UNLESS YOU KNOW EXACTLY WHAT YOU ARE DOING !!!

On this page, you have access to basic system settings called *system properties*.

Do not modify, edit, or delete any values of system properties unless you know exactly what the impact will be.

F.11.2 System Property Overview

A system property has the following parameters:

- **Module**
  Mandatory. This indicates in which ConSol CM module the system property will be used.
- **Property** (name)
  Mandatory. The name of the system property. The system property is referenced by this name throughout the system.
- **Type**
  Mandatory. Data type of the system property, i.e., of the value.
  - **String**
    A regular string field
  - **Password**
    A field which will contain a password and will therefore not be displayed in plain text.
  - **Email**
    A field which contains an email address, i.e., it has to be formatted according to the standard email format (<address>@<domain>).
  - **Boolean**
    A boolean (true/false) field.
  - **Integer**
    A whole number field (no fractional part). Used, for example, for time intervals or number of restarts.
- **Value field**
  The value of the system property. Must be set according to the given data type.
- **Description**
  Optional. A text description for the system property.
  In case you add company-specific system properties which contain integers for time intervals, we strongly recommend that you use the description to explain the unit of measurement, since it makes a big difference whether an escalation goes off within five minutes, five hours, or five days!
- **Restart required**
  Boolean field (checkbox) to indicate whether a change of the system property will only become active after a system restart.
- **Optional**
  Boolean field (checkbox) to indicate if the system property has to be present or is optional. The following behavior will be configured:
  - optional = “true”:
    - The value of the system property can be NULL.
    - The system property can be deleted, e.g., using the Admin Tool.
  - optional = “false”:
    - The value of the system property cannot be NULL.
    - The system property cannot be deleted, i.e., there must be a value in the *Value* field.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **REST accessible**
  Boolean. Starting with CM version 6.11, the value of a system property can be retrieved via REST API if this checkbox is set to “true”. For detailed information about the REST API and the required commands, please refer to the *ConSol CM REST API Documentation*.

  > ⚠ Please be very careful with the decision to set this value to “true”! Using REST, a regular system account is sufficient for retrieving data!

The system properties are managed in a table where three of the system property parameters are displayed as columns:

- **Module**
  Module name
- **Property**
  The name of the system property.
- **Value**
  The value of the system property.

You can sort the table according to a column by clicking on the column header. Another click reverses the order.

The following example shows the pop-up menu which is used to set a system property, here `cmas-core-server, server.session.timeout`. You reach this screen by opening the navigation group *System*, the navigation item *System Properties* and by selecting the property in edit mode.

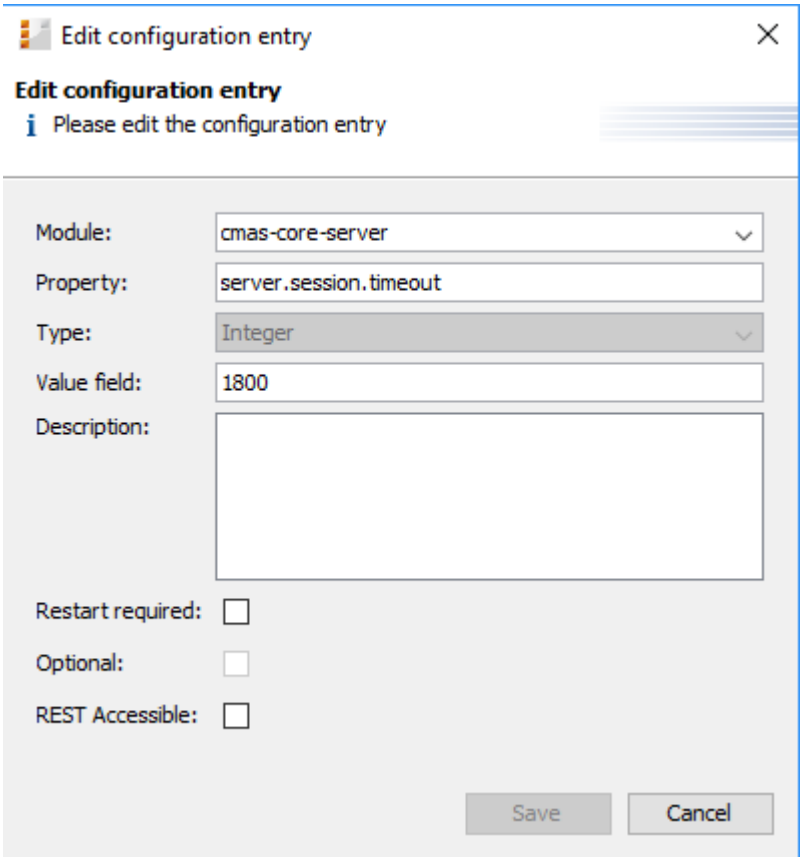


Figure 377: *ConSol CM Admin Tool - System, System Properties: Editing one system property*

F.11.3 Setting System Properties

The values for system properties are set in different ways. System properties can be ...

- filled automatically with default values by the ConSol CM system, e.g., `cmweb-server-adapter, ticketListRefreshIntervalInSeconds` is set to “180” seconds. You can modify the values if necessary.
- prepared by the system, i.e., the system properties are in the list but they are not filled with the required values. For example, LDAP-related system properties will only be filled (manually by an administrator) when you configure the LDAP authentication.
- filled by Admin Tool configurations which you perform by using the graphical user interface of the Admin Tool. For example, when you enter mailbox names in the *Email* configuration, the values will be entered into the respective NIMH system properties, e.g., `cmas-nimh, mailbox.0.connection.host`. In those cases, you should always use the graphical configuration and not edit the system property directly!
- not set at all in a default installation. Then they have to be added manually, e.g., the activation of the TEF (Task Execution Framework) by adding the system property `cmas-app-admin-tool, start.groovy.task.enabled` and setting its value to “true”.
- added to a system manually as customer-specific system properties. If the module name starts with *custom-*, those system properties are exported in a scenario (see section Deployment (Import/Export)). For example, you can define and add your own system properties to manage escalation times. In this way, you can store the values here in the system properties administration and an administrator can change the values without requiring any programming knowledge. In the script code where the system properties are referenced, only the system property name is used. This is explained in the following section.

F.11.4 Programming with System Properties

To use a system property in a Groovy script, i.e., to retrieve the system property's value, use the following class and method:

- `configurationService.getValue(String pModule, String pProperty)`.

For example, to retrieve a specific escalation time, use:

```
def mytime = configurationService.getValue("custom-
 servicedesk","escalation.time.medium2")
```

This will retrieve the value “10”.

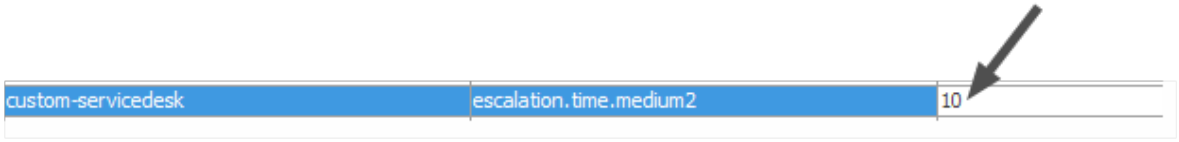


Figure 378: *ConSol CM Admin Tool - System, System Properties: Customer-specific system property*

F.12 System Documentation

The navigation item *Documentation Generator* in the navigation group *System* allows to export parts of the current configuration of ConSol CM into a file. The scope of the documentation, i.e., which information is included, and the layout of the document are determined by a script of the type *System documentation*.

The following items of the system configuration can be included in the system documentation:

- incoming and outgoing email connection
- customer data models: customer objects and customer fields
- customer groups
- workflows: scopes, activities, triggers, image of the workflow (only available after deploying the workflow)
- queues: workflow, assigned ticket field groups with ticket fields, customer groups, projects and classes of text
- resources: resource categories and resource types with resource fields
- roles: queue, customer group, resource type and global permissions, views, engineer functions
- views: queues, scopes, view criteria

Please proceed as follows to create a system documentation:

1. In the navigation item *Scripts and Templates*:
   - Create a script of the type *System documentation* (see Scripts of Type System Documentation).

     OR
   - Adapt the default documentation script included in the ConSol CM distribution to your needs (see Scripts of Type System Documentation).
2. In the navigation item *Documentation Generator*:
   a. Click the file explorer icon next to the *Output file* field and select the path where you want to save the documentation. Enter a file name with the desired file extension.
   b. Select the script of the type *System documentation* in the *Script* drop-down list.
   c. Optional. Select the checkbox *Input export file* and provide the path to the scene export for which the system documentation should be generated. The scene needs to contain the `systemDocumentation.xml` file which is created when enabling the checkbox *System documentation* in the scene export configuration (see Deployment (Import/Export)).

   > ⚠ You might need to adapt the system documentation script if you want to create documentation from a scene export.

3. The document containing the system documentation is created and saved at the provided path. If a scene export file is selected, the system documentation is generated for this scene. Otherwise, it is generated for the current scene in the Admin Tool.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH



The ConSol CM distribution contains the two example scripts `Documentation_Generator_script_EN` and `Documentation_Generator_script_DE` for generating system documentation in HTML format in English and German.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.13 Working with Templates

Text templates are pre-defined texts which an engineer can open and either use as-is or modify. Text templates may be used for emails or ticket comments anywhere text, headers, and footers can be specified. Another example is documents which have to be edited using Microsoft Word or OpenOffice.

In both cases, the templates offer not only text, but certain data fields can also be pre-filled with data from the ticket, e.g., customer name or ticket subject.

ConSol CM includes two modules which provide text templates:

- The Text Template Manager for editing and managing email and ticket comment templates (see section Text Templates)

and

- CM/Doc with the Document Template Manager for editing and managing Microsoft Word and OpenOffice templates (see section CM/Doc).

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.13.1 Text Templates

F.13.1.1 Introduction to Text Templates in ConSol CM

Text templates are boilerplates consisting of text and variables, which allow to automatically insert data of the current user or ticket. They are used in emails and comments. Text templates help to keep emails consistent regarding wording and layout, and increase the productivity of the users who do not have to type as much text.

Concepts, Terms and Definitions

Concept	Other terms	Definition
text template		Template which is used when writing emails or comments.
document template	Office template	Microsoft Office or OpenOffice file used as a template for CM/Doc.
Freemarker		Template engine used for creating templates in ConSol CM.

Purpose and Usage

Text templates are used for facilitating the work of the Web Client users by providing them with drafts for emails and comments. In addition, the text of automatic emails or automatically added comments can be stored in text templates.

Text templates can include variables which are filled with data from the ticket where the template is used. Variables can retrieve ticket data, data of the ticket's customer or engineer, and data of the current engineer.

When using a text template, the template content is added to the comment or email editor. The user can edit it as desired to create the final comment or email. It is possible to use several templates for a comment or email.

Text templates are saved in the *Text templates* section of the Web Client. Templates which are only used in the workflow and should not be available for selection in the Web Client, can also be saved in the Admin Tool Templates section of the Admin Tool.

F.13.1.2 Available Settings for Templates

Text templates have the following general settings:

- **Name:**
  Mandatory. The name of the text template. It must be unique within the language. On the overview screen, templates which have attachments are indicated by a small paperclip icon next to the template name.
- **Group:**
  Mandatory. The group which the template belongs to. This setting is useful to organize your templates and locate them more easily, both in the template administration and in the email or comment editor.
- **Subgroup:**
  Optional. The subgroup which the template belongs to. This setting is useful if your groups contain many templates.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Language:**
  Mandatory. The language of the template. This is useful if you communicate in several languages, so that you need to have the same template in each language. You can then use the language setting to retrieve the correct template in scripts.
- **Usage:**
  Indicates how many times the template was used. Only shown on the overview screen.
- **Release:**
  Mandatory. The version number of the template. Only shown in the template details.
- **Active:**
  Mandatory. Indicates if the template is enabled or disabled. The checkbox is available in the template details. On the overview screen, the disabled templates are displayed in gray italics.
- **Is workflow:**
  Only available for templates of the type *Include*. Indicates whether the include can be used in letters which are used in the Web Client or letters which are used in the workflow. If the setting is “No”, the letter cannot be used in workflows. Otherwise, it can be used both in the Web Client and in workflows.



If a template uses data fields which are no longer available, the line of the respective template is displayed in red on the overview screen. The broken references are displayed in red and you need to remove them to be able to save the template.

Types of Templates

There are four types of text templates. A text template is always of one of the types.

- **Letter:**
  The text template which can be used for comments or emails.
- **Include:**
  Includes can be used within letters. They allow reusing text for several templates. A common use case for includes is the email signature, which is required for all email templates. When creating an include, you need to decide whether is should be available only for letters which are used in the Web Client, or for both letters which are used in the Web Client and in the workflow.
- **Text block:**
  Text blocks can be used within letters. They are useful for manual emails and comments because the user can choose which text blocks to include in the final comment or email. See Using Text Blocks.
- **Script:**
  Scripts can be used within letters, includes or text blocks. They are written in Freemarker and allow adding logic to templates. If you want to use HTML tags in the script, you need to select the checkbox *Allow HTML*. Scripts can only be edited by users with administrator permissions. See Using Scripts in Text Templates.

Template Usage

There are four places where text templates of the type *Letter* can be used. You can select one or several of the places:

- **Workflow:**
  The template can be used for emails and comments created automatically in the workflow.

  > ⚠ Templates which can be used in the workflow must not contain text blocks or additional parameters, as these elements require manual interaction by the user.

- **Email:**
  The template can be used for manual emails sent using the email editor in the Web Client.
- **Ticket create:**
  The template can be used for comments written in the comment editor of the Web Client when creating a new ticket.
- **Ticket edit:**
  The template can be used for comments written in the comment editor of the Web Client when adding or editing a comment in an existing ticket.

For text templates of the type *Include*, there is one setting:

- **Is workflow:**
  Indicates whether the include can be used in letters which are used in the Web Client or letters which are used in the workflow. If this checkbox is not selected, the letter cannot be used in workflows. Otherwise, it can be used both in the Web Client and in workflows.

F.13.1.3 Basic Tasks

Finding a Text Template

The list of text templates is displayed directly after accessing the *Text templates* section of the Web Client. You can filter the list to locate a template more easily:

- **Text search**: Perform a free text search in the template name and content.

  Enter a text in the *Enter template name or content* field.
- **Template group**: Display only templates which belong to a certain group.

  Select the template group or subgroup in the *All groups* drop-down list.
- **Queue**: Display only templates which are available in a certain queue.

  Select the queue in the *All queues* drop-down list.
- **Template usage:** Display only templates which are available in a certain context.

  Select the context in the *Available for* drop-down list.
- **Template type:** Display only templates of a certain type.

  Select the type in the *Template type* list.

You can select the checkbox *Include inactive templates* if you want to display disabled templates.

Managing Text Templates

You need to have template administration permissions to manage templates, see permission *Write template* in the Tab Global Permissions section.

Best practice

Set up a specific role which includes only the *Write template* permission and assign it to the desired users.

Most actions to manage templates can be accessed from two places:

- **Context menu**

  Click the name of the template in the *Template name* column and select an entry in the context menu.

- **Preview**

  Click any column except for the *Template name* column and click one of the icons displayed on the left.

The following actions are available for templates:

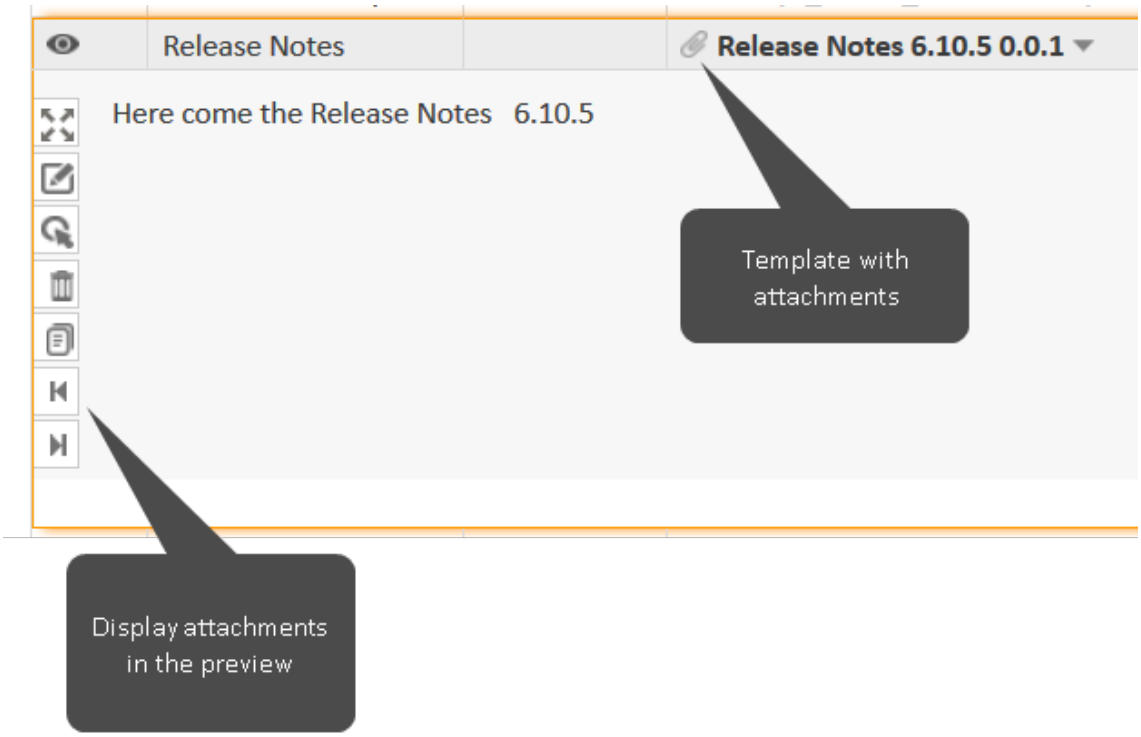
- **Create a new template**

  Click the *Create new template* button above the list of templates. Fill out the basic template data (see Available Settings for Templates) and write the template content (see Writing Text Templates).

  If the new template is similar to an existing template, you can prefill the *Create new template* screen with the data of an existing template by selecting the *Clone* option.

- **Preview the attachments of a template**

  You can click the arrow icon in the preview of a template to display the attachments in the preview window.



CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Edit a template**

  Click the *Edit* icon of the template preview or select the *Edit* option from the template's context menu. You can update the basic template data (see Available Settings for Templates) or change the template content (see Writing Text Templates).

- **Disable a template**

  Click the *Disable* icon of the template preview or select the *Disable* option from the template's context menu. Disabled templates cannot be used anymore. They are displayed in gray italics on the overview screen.

- **Enable a template**

  Click the *Enable* icon of the template preview or select the *Enable* option from the template's context menu. Enabled templates can be used in the Web Client and workflow.

- **Delete a template**

  Click the *Delete* icon of the template preview or select the *Delete* option from the template's context menu to delete templates which are not needed anymore. Templates which are in use, either by the workflow or because they are referenced by other templates, cannot be deleted.

- **Set the default templates for comments and emails**

  Only available for templates of the type *Letter*. You can set a default template for both emails and comments. This template is inserted automatically in the editor when a user adds a comment or email. To set a template as default template, click the option *Use as e-mail standard* or *Use as comment standard* in the template's context menu. This option is only available if no other template is in use as standard template for emails / comments. If another template is already selected for this purpose, you have to remove this setting first by clicking *Unset comment standard* or *Unset email standard* in the context menu of the affected template. You can locate the default templates by looking for the text *E-mail standard* or *Comment standard* after the template name.

  > Usually, a signature is used as a standard template for emails. A standard template must not contain text blocks or parameters.

Writing Text Templates

You can perform the following steps to write a text template:

1. Create a template of the type *Letter*. You can enter text and format it using the rich text editor. In addition you can insert variables from the library of markers. See Using Variables in Text Templates. You can add attachments to the template. If the template should only be available in specific queues or if specific data fields are filled out in the ticket, you can restrict the template accordingly. See Binding a Text Template.

   > Alternatively, you can create a new template from an outgoing email in the ticket history. Click the entry *Use as template* in the context menu of the email to open the template creation screen.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

2. Optional. Add attachments to the letter. When selecting the template in the comment or email editor, the attachments are preselected. They are attached to the ticket upon saving the comment or sending the email. When an email template with attachments is used by the workflow, the attachments are ignored.
3. Optional. Create a template of the type *Text block*. You can enter text and format it using the rich text editor. In addition you can insert variables from the library of markers. See Using Variables in Text Templates.
4. Optional. Create a template of the type *Include*. You can enter text and format it using the rich text editor. In addition you can insert variables from the library of markers. See Using Variables in Text Templates. You can use all kinds of variables except for text blocks and other includes. A common use case for includes is the email signature, which is added to all letters used for emails.
5. Optional, administrator permissions required. Create a template of the type *Script*. You can enter text and use Freemarker code to add logic. If you select the *Allow HTML* checkbox, you can use HTML tags to format the script content.
6. Optional. Reference text blocks, includes and scripts in the letter. See Using Variables in Text Templates.

> It is not possible to use executable JavaScript code in text templates. Any <script> tags are escaped automatically by the system.

Using Variables in Text Templates

The library of markers allows to insert variables into text templates. There are three kinds of variables:

- **References to other text templates (includes, text blocks, scripts)**

  The referenced templates are included automatically. The only case where there is user interaction are text blocks, which the user can select or deselect.

  > Text blocks are not available in letters which can be used in the workflow.

- **References to ticket and customer fields**

  The referenced fields are automatically replaced by their values in the current ticket / ticket customer. If the field is empty, it is omitted in the template. No user interaction is required.

  > If you use customer fields, the template is only available in tickets whose main customer belongs to a customer group whose data model includes the respective fields.

- **Parameters to be filled out by the user**

  There are two kinds of parameters: string parameters and enum parameters. The user needs to fill out the variable when using the text template.

  > Parameters are not available in letters which can be used in the workflow.

The following table lists the available variables with a short description. Different colors are used in the template administration to make it easier to distinguish between the different kinds of variables.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Variable	Description	Color
Customer data models	Allows to insert customer fields. Select the customer group and customer object (company or contact) to see the available data fields.	Blue
Customer Group	Allows to insert the name of the customer group.	Blue
Queue	Allows to insert the name of the queue.	Blue
Custom fields	Allows to insert ticket fields. Select the field group to see the available data fields.	Blue
Ticket	Allows to insert basic ticket data (ID, name, subject, creation date, escalation date). You can click the *Engineer* option to insert data of the assigned engineer, e.g. the name or phone number (see Engineers for details about the available fields).	Blue
Engineer	Allows to insert data of the current engineer, e.g. the name or phone number (see Engineers for details about the available fields).	Blue
Includes	Allows to insert templates of the type *Include*.	Orange
Text blocks	Allows to insert templates of the type *Text block*.	Green
Workflow includes	Allows to insert templates of the type *Include* which have the option *Is workflow* checked, so that they can be used in templates which are used in the workflow.	Orange
Scripts	Allows to insert templates of the type *Script*.	Lila
String parameter	Displays a text field above the comment / email editor where the user can enter a text which is inserted in the configured place within the template.	Red
Enum parameter	Displays a selector above the comment / email editor where the user can select one of the enum values to insert it in the configured place within the template.	Red

Please proceed as follows to insert references to text templates and fields:

1. Place the cursor at the desired position within the text.
2. Navigate to the desired item in the *Library of markers* section and click to insert the item.

Please proceed as follows to insert parameters:

1. Place the cursor at the desired position within the text.
2. Insert the parameter in the *Library of markers* section:

   String parameter:

   a. Click *Add string parameter*.

   b. Enter the name of the parameter in the field below *New parameter*.

   c. Click *Add parameter*.

   Enum parameter:

a. Click *Add enum parameter*.

b. Enter the name of the parameter in the field below *New parameter*.

c. Select an enum. The checkbox *Template enabled* needs to be selected for the enum in the Managing Sorted Lists: Enums section.

d. Click *Add parameter*.

See Using Parameters for information about the user perspective.

Binding a Text Template

The binding determines in which tickets a letter template is available. The availability of a text template depends on two factors:

- **Settings in the *Binding* section of the template**

  You can limit the availability of templates depending on the queue and ticket fields. The following settings are available:

  - **Queue**: Select the queue which the ticket has to be in.
  - **Custom field**: Select the value which needs to be set for a ticket field (only fields of the type ENUM).

  You can add several criteria by using the + icon. The following rules apply when combining several criteria:

  - **Queue**: Several queues are combined with OR, i.e. the ticket needs to be in one of the queues.
  - **Ticket field**: Several ticket fields are combined with AND. Several values of the same ticket field are combined with OR. This means that the ticket needs to have one of the specified values for each of the ticket fields.
  - **Queue and ticket field**: Queue and ticket fields are combined with AND, i.e. the ticket needs to be in the selected queue and have the specified value in the given ticket field.

  There are two kinds of binding: soft and hard. By default, the hard binding is used for all criteria, meaning that the template is only available if the criterion is met. You can select the checkbox *soft* next to a criterion to allow the Web Client users to “overwrite” the binding. In this case, the users can display the template by unchecking the option *Show templates suitable for this context* in the template selector of the email / comment editor, when the criterion is not met.

- **Customer group of the ticket’s main customer**

  If variables from customer data models are used (see Using Variables in Text Templates), the template is only available if the ticket’s customer belongs to a customer group with the respective data model.



If you do not need all the ticket fields of the type ENUM for text templates, you can exclude specific ticket fields using the page customization of markersLibrary (Type). This improves the load time of the template details page because the excluded field do not need to be loaded in the *Binding* section.

Using Text Templates

Text templates can be used in the comment and email editor of the Web Client and in the workflow. Text templates which are used in the workflow work in the same way as regular templates.

The following section describes how users work with text templates in the Web Client.

The user can select the desired templates in the *Add template* selector of the comment and email editor. He can edit the template text before saving the comment or sending the email. If a default template is used, the user can remove or edit the content of this template. The template is inserted as text, the text which belongs to variables has the same color as in the template administration. The purpose of the colors is to highlight the parts which are dynamically retrieved, the sent email or saved comment does **not** have these colors.

The following figure shows a template with two text blocks in the comment editor. Both text blocks are selected, so that they are included in the comment.

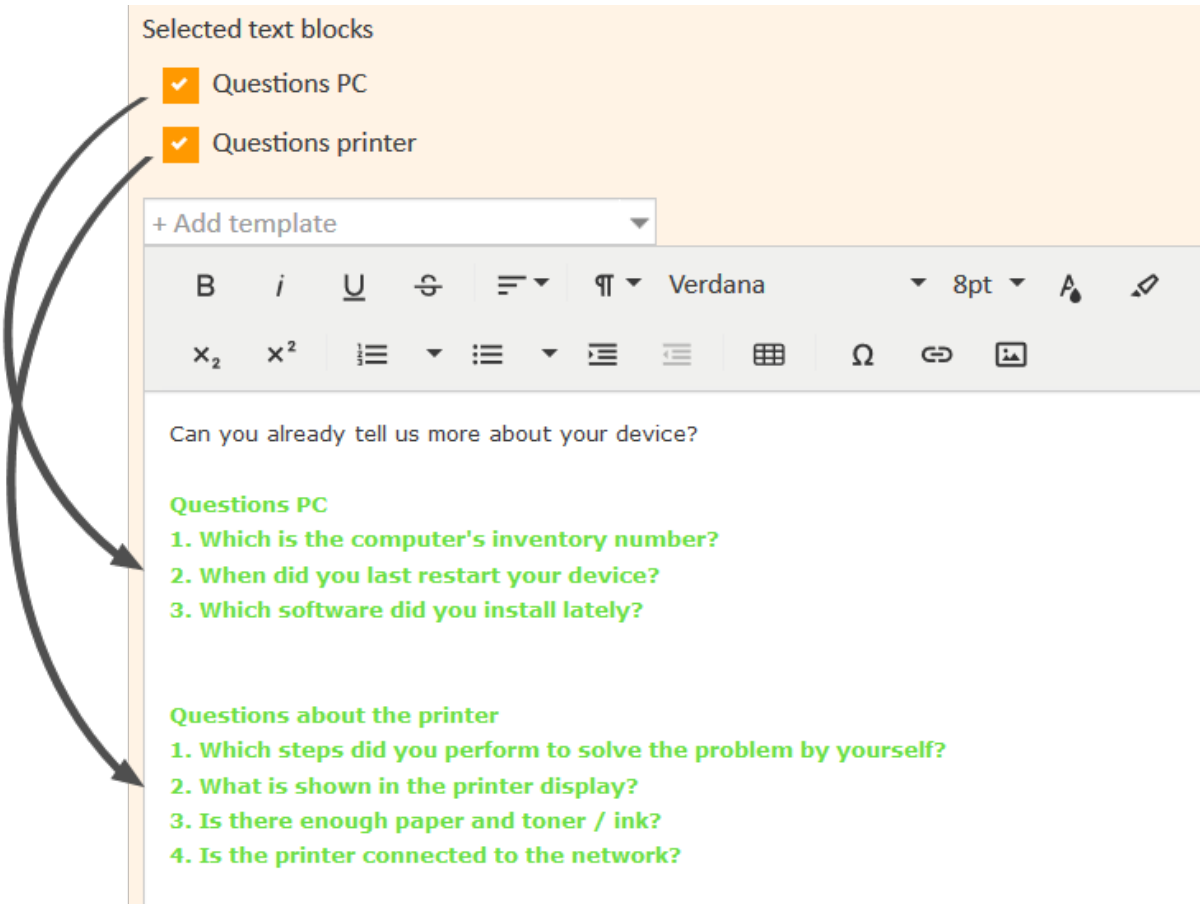


Figure 379: *Template with text blocks (user perspective)*

The following figure shows a template with one string parameter and one enum parameter. The user can provide the desired values above the editor to insert them.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

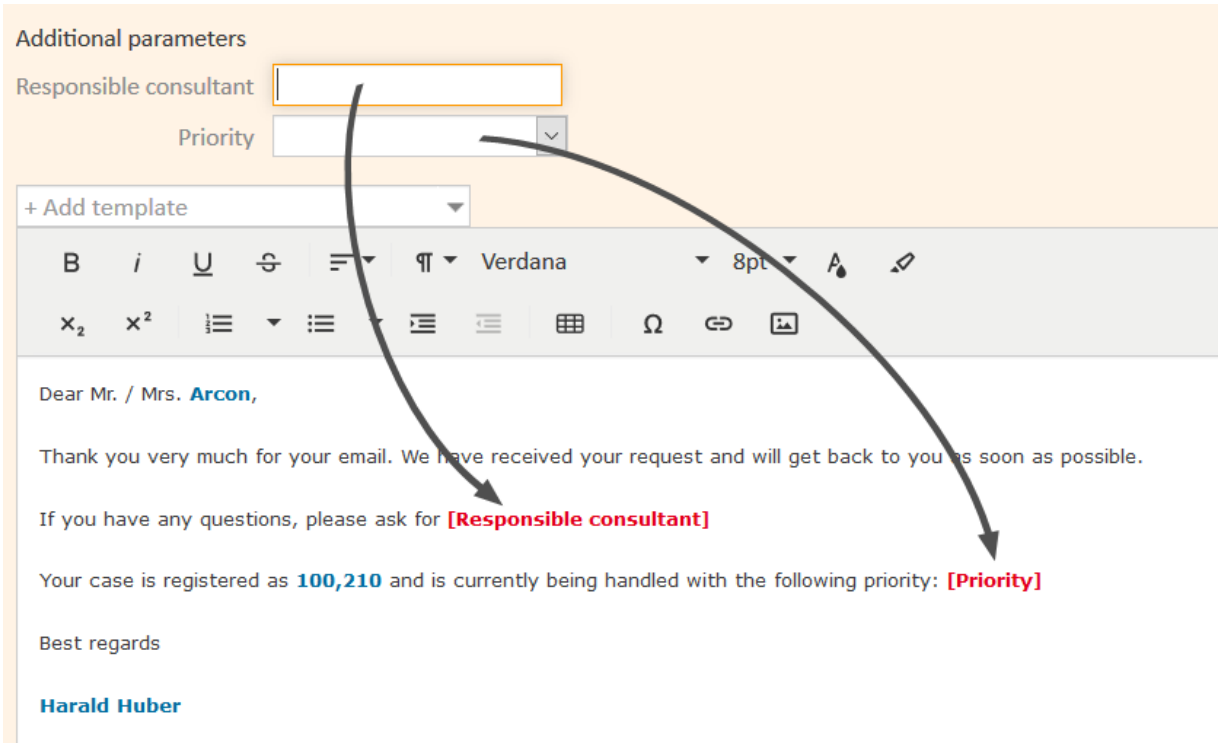


Figure 380: *Template with parameters (user perspective)*

You can configure several editor settings regarding templates in the page customization of mailTemplate (Type).

F.13.1.4 Advanced Tasks

Using Scripts in Text Templates

Scripts allow to add logic to the templates. They can only be edited by users who have administrator permissions, see Tab Global Permissions. Using scripts you can implement for example customized salutations depending on contact fields.

Script templates are written in Freemarker (see https://freemarker.apache.org/). If you select the *Allow HTML* checkbox, you can use HTML tags to format the script content.

You have to use square brackets [] instead of the angle brackets used in standard Freemarker notation.

⚠ Scripts with the option *Allow HTML* can include links. The links have to start with one of the allowed protocols (http, https, ftp, ftps, file, mailto), otherwise they are removed.

Therefore, you cannot directly use a script function within a link to dynamically retrieve a complete link from a ticket or customer field. The following code using a function within the link tags in a script will **not** work:

```
<a href="${ticket.get('fieldgroup.field')}">My Link</a>
```

Instead, the field value must not include the protocol. You can use a variable which stores the return value of the function:

```
[#assign link=ticket.get('fieldgroup.field')!]
<a href="https://${link}">My link</a>
```

Coding Example: Queue-specific Signatures

The following example shows a script template used to include a specific signature depending on the current queue of the ticket. This way, you can use different signatures for different departments of the company.

```
[#if ticket.queue.name??]
   [#if ticket.queue.name=="ServiceDesk"]
      [#include "Signature ServiceDesk"]
   [#elseif ticket.queue.name=="HelpDesk_1st_Level"]
      [#include "Signature Helpdesk"]
   [#else]
      [#include "Default signature"]
   [/#if]
   [#else]
      [#include "Default signature"]
[/#if]
```

Coding Example: Using List and Struct Fields in Templates

Data fields of the type LIST or STRUCT cannot be selected directly using the library of markers. Instead, you need to create a script and include it in the letter template. The *Allow HTML* checkbox needs to be selected for the script template. The following example shows how to write a script which displays the content of the struct `order_list` in the ticket field group `helpdesk_standard`. The table header is displayed in bold font and contains the field names.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

```
[#if helpdesk_standard.order_list?has_content]
   <table>
      [#list 0..helpdesk_standard.order_list?size-1 as i]
      <tr>
         [#list helpdesk_standard.order_list[i] as value]
            [#if i==0]
               <td style="border:1px solid #d9d9d9; padding: 0 2px;font-
                weight:bold;">${value}</td>
            [#else]
               <td style="border:1px solid #d9d9d9; padding: 0 2px;">${value}</td>
            [/#if]
         [/#list]
      </tr>
      [/#list]
   </table>
[/#if]
```

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.13.2 CM/Doc

F.13.2.1 Introduction to CM/Doc

Even in companies where most processes are managed by IT applications, a great number of documents still have to be printed out or are required in `.doc` or `.pdf` format. These can be, for example:

- invoices
- contracts
- documents concerning the acceptance of IT systems
- orders

ConSol CM offers the add-on CM/Doc to print documents directly from the business management process. CM/Doc supports Microsoft Word documents and OpenOffice documents.

Templates guarantee that ...

- all documents of one type are identical (text and layout).
- all documents match the company's corporate design.
- the engineers do not have type the same text over and over again.

Data from the ticket can be integrated into the template automatically, these can be:

- ticket data (e.g., amount in an invoice, service level in a contract)
- customer data (name and address of the main customer (and of the company, if the main customer is a contact belonging to a company), additional customers are **not** relevant in CM/Doc!)
- engineer data (name, phone number, email address of the engineer who works on the case)

When CM/Doc is active in ConSol CM, the tab *Document* is displayed in the ticket history section, where the engineer can select the required document template. The document is opened in Microsoft Word or OpenOffice automatically with all required data fields already filled in. The engineer can then work on the document and save it. It is automatically attached (as a regular attachment) to the ticket and can be opened by users who have **read** access to the ticket and who have installed the required software (Microsoft Word, OpenOffice) on their PCs.

With special adaptations implemented by the ConSol CM consulting team, the ConSol CM system can be extended in a way that `.docx` documents can also be converted to `.pdf` files in order to make sure that no further changes can be made to the document.

F.13.2.2 Requirements for Using CM/Doc

On the client PC or laptop, the following requirements have to be met to use CM/Doc:

- Microsoft Windows operating system
- Locally installed CM/Doc application
- Locally installed Microsoft Word / OpenOffice

> Detailed information about the supported Microsoft Word and OpenOffice versions can be found in the current *System Requirements*.

The engineers can download the CM/Doc application for Windows from the *Downloads* section of their user profiles.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

If you use CM/Doc with Microsoft Internet Explorer or Microsoft Edge, some additional installation steps are required.

For both browsers: Go to *Internet Options -> Security* in Microsoft Internet Explorer. Select *Local Intranet*. Click *Sites*. Click *Advanced*. Enter the URL of the ConSol CM server and add it to the websites belonging to the local intranet zone.

If you use Microsoft Edge, you need to execute the following command as an administrator:

```
CheckNetIsolation LoopbackExempt -a -n="Microsoft.MicrosoftEdge_
 8wekyb3d8bbwe"
```

If HTTPS is used, the engineers need to trust the URL configured for CM/Doc (see cmweb-server-adapter, cmoffice.websocket.port) in their browsers. The prompt to trust CM/Doc will be displayed automatically in the default browser. If the engineers use the Web Client in another browser, they need to manually add the certificate as trusted in this browser. The corresponding URL can be copied from the window which is opened when clicking *Preferences -> Web Browser* in the context menu of the CM/Doc application in the system tray.

For Microsoft Internet Explorer, the certificate generated by CM/Doc (file `public.crt` in `C:\Users\<your user name>\AppData\Local\cmdoc\security`) also needs to be imported manually to the *Trusted Root Certificate Authorities* certificate store. Go to *Internet options -> Content -> Certificates -> Trusted Root Certificate Authorities* to import the file.

F.13.2.3 Configuring the ConSol CM System for CM/Doc

CM/Doc is enabled using the system property cmweb-server-adapter, cmoffice.enabled. Go to the navigation item *System Properties* in the navigation group *System* and click the respective entry. Set the value of the system property to “true”.

If CM/Doc is enabled for the ConSol CM system, the engineers can enable or disable it on an individual basis in their user profiles (section *General settings*, option *Document generation via CM/Doc*).

Enabling CM/Doc with OpenOffice

If CM/Doc is used with OpenOffice, the engineers have to set the path to the OpenOffice installation within the CM/Doc application. They have to perform the following steps:

1. Right-click the icon of the CM/Doc application in the system tray.
2. Go to *Preferences -> Open Office*.
3. Select the OpenOffice executable in the file browser.

Configuring the Saving Strategy for CM/Doc

There are two saving strategies for CM/Doc:

- A new document is attached to the ticket with each save operation in Microsoft Word or OpenOffice. This option creates a history of all changes to the document but might lead to a great number of attachments, which might increase the ticket size considerably.

- A new document is attached to the ticket after the first save operation in Microsoft Word or OpenOffice. Further save operations overwrite this attachment as long as the document is not closed in Microsoft Word or OpenOffice. This option limits the number of attachments and hence the ticket size, so that it does not matter how often the engineers use the *Save* button in Microsoft Word or OpenOffice.

The saving strategy is configured using the CM system property cmweb-server-adapter, cmoffice.strict.versioning.enabled:

- **true**(default)
  A new version of the Word/ OpenOffice document is saved and attached to the ticket with each SAVE operation from within the document.
- **false**
  A SAVE operation from within the Word / OpenOffice document overwrites the existing attachment with the same name as long as Word / OpenOffice has not been restarted in the meantime.

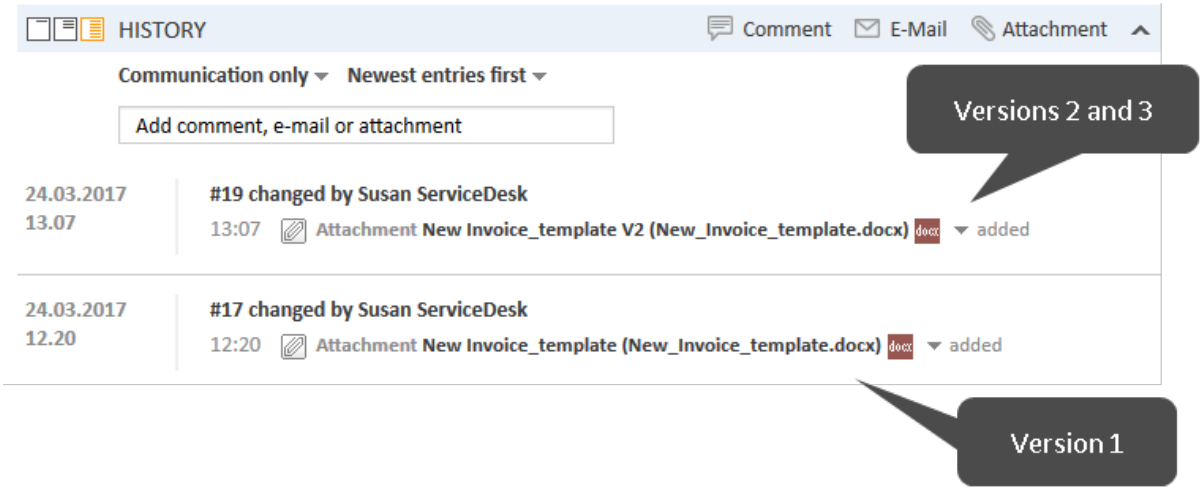


Figure 381: *ConSol CM Web Client, documents attached to a ticket, cmoffice.strict.versioning.enabled=false*

Creating a Role with Permissions for the Document Template Manager

The menu item *Document templates* which is used to create templates for CM/Doc in the Web Client is only available to engineers who have the permission *Write template* from the *Global permissions* tab of the role administration. So, as a second step, you have to create one or more roles with the respective permissions. For a detailed explanation about setting role permissions, please refer to section Roles.

> We recommend to create a role (e.g., *TemplateManager*) that has only the permission Write template, with no queue permissions or other permissions. Every user who should have access to the Document Template Manager can be given this role. In this way, there is no merge between regular user permissions and document template permissions and you can grant and retrieve the document template permission in a very flexible way.
>
> However, the permission *Write template* also grants access to the Text Template Manager (for text templates, see section Text Templates).

F.13.2.4 Creating Document Templates

Once CM/Doc is enabled, you can create the document templates. This is done using Microsoft Word or OpenOffice. Please create `.doc`, `.docx` or `.odt` files as templates. Do **not** use the template format of the word processor (`.dot` or `.ott` files)!

Adding a Document Template

Log in to the Web Client and click on *Document templates* in the main menu to open the Document Template Manager.

As the next step, you have to fill in the requested data fields as *merge* fields in the document template, i.e., you create a CM/Doc template from your regular Microsoft Word or OpenOffice document. This is done using the ConSol CM Web Client.

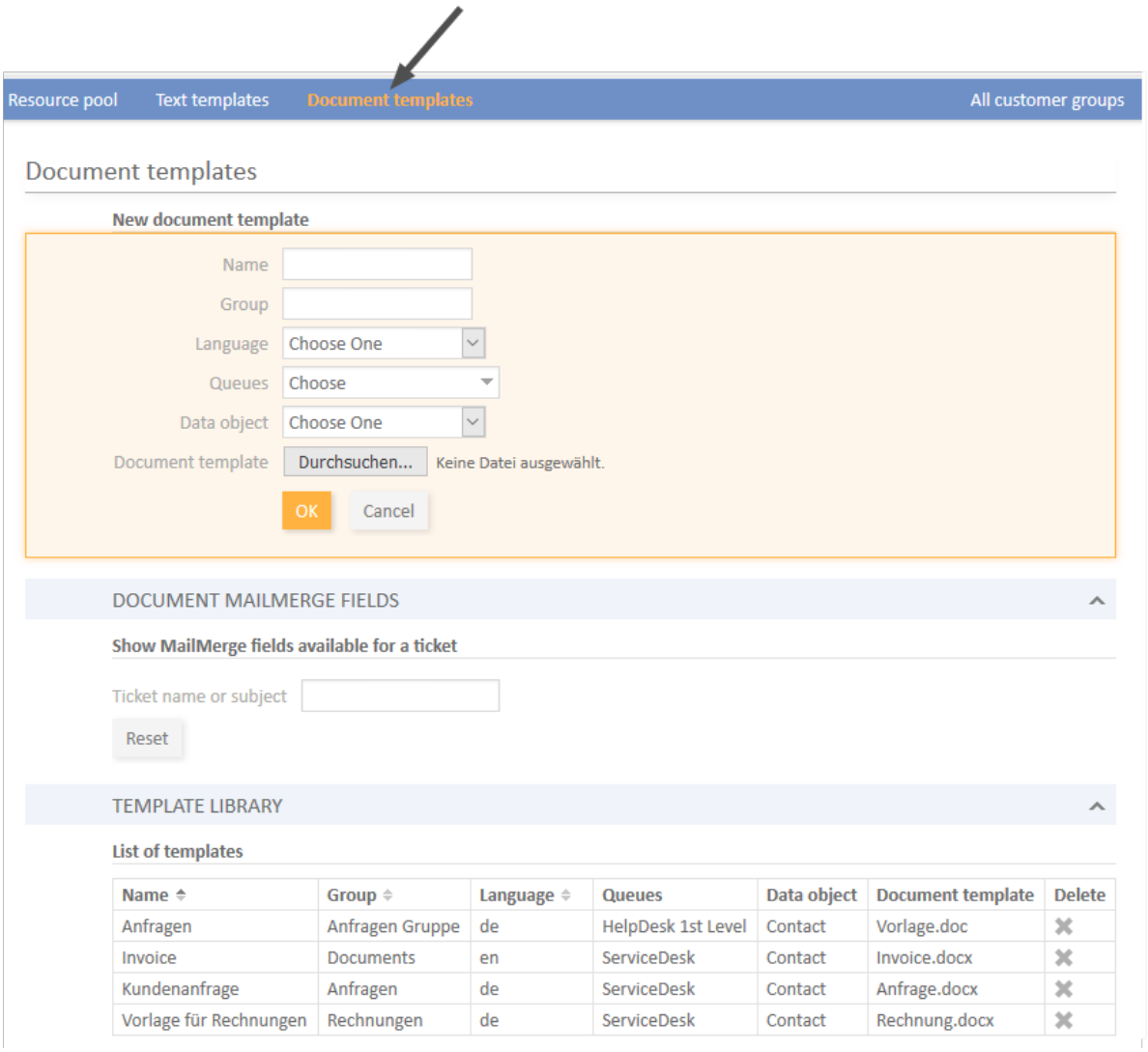


Figure 382: *ConSol CM Web Client - Document Template Manager*

The Document Template Manager is opened.

Enter the following data for each new template, then click on *OK*:

- **Name**
  The name of the (new) template. This will be displayed for engineers in the Web Client.

- **Group**
  The name of the template group. This does not have any technical implications but serves as an easy-to-use parameter to sort the templates in the template list in the Document Template Manager.
- **Language**
  Select the required language. All languages which have been activated in the Admin Tool are listed.
- **Queues**
  Select the queues for which the template should be available.
- **Data object**
  For the selection of the data object which should be used as reference object for customer data within the template. All data objects are listed, please see section Selection of the Data Object and Its Consequences for a detailed explanation about which data object you have to select.
- **Document template**
  Use the file browser to select the `.doc` or `.docs` file that should serve as a template in the file system.

Selection of the Data Object and Its Consequences

In the form where you have to fill in all fields for a document template, you also have to select a data object, i.e., a customer object (contact or company object) from your customer data model. If you are not familiar with the ConSol CM customer data model concept, please read the Customer Data Model first.

In the drop-down menu *Data object*, the data objects are listed in the following format:

- [localized name of the data object (localized name of the customer data model)]

For a data object to be listed in the drop-down menu, the following conditions have to be met:

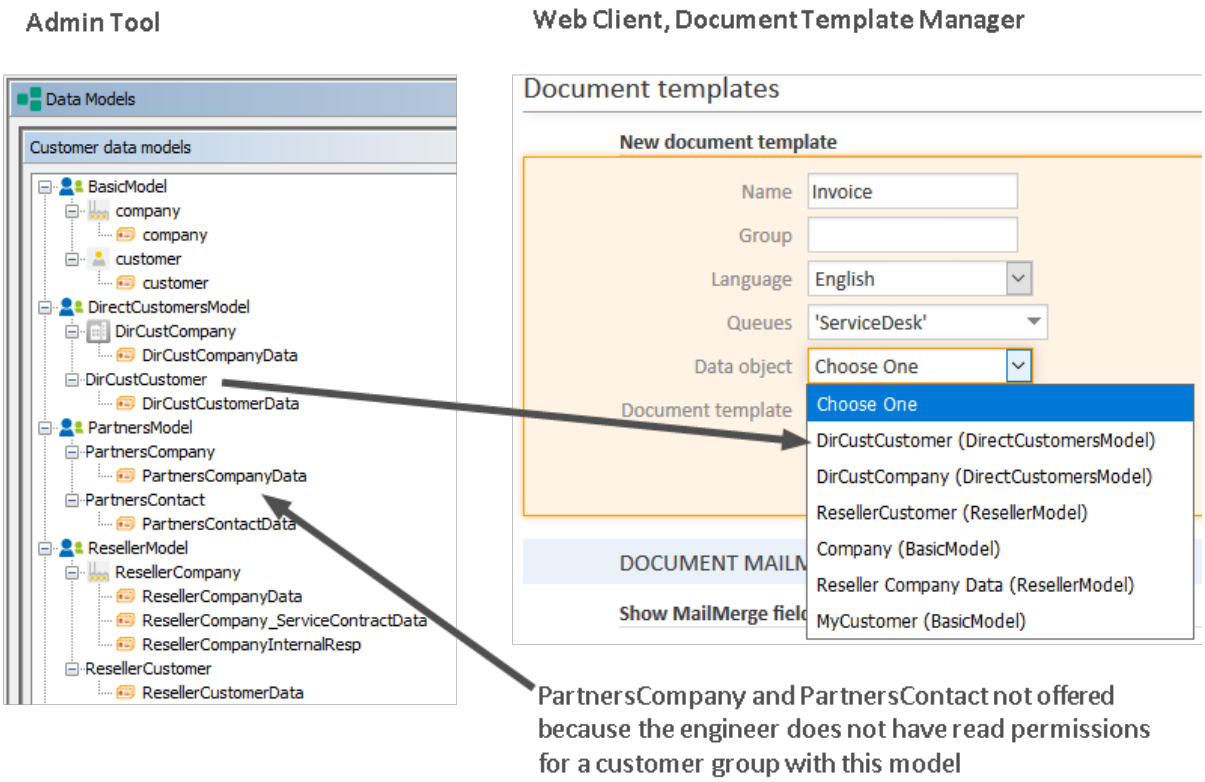
- The customer data model with the respective data object has to be created in the Admin Tool.
- The customer data model has to be assigned to at least one customer group.
- The engineer who works on the document templates has to have at least **read** access to at least one customer group with the respective customer data model.

You have to select the data object which represents the main customer of the tickets. The document template will only be available in tickets which have main customers from a customer group with this customer data model. This means if you want to use the same document template text for two customer groups with two different customer data models, you have to create two document templates, one for each customer data model. You can, of course, use the same document template for two or more customer groups which share the same customer data model.

If you try to assign a data object to a document template which is not compatible with the selected queue (because the selected data model is not assigned to the selected queue), you will get an error message.

Please be aware that it makes only sense to select an object at company level if the parameter *Company as customer* has been set for this customer data model.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH



Please be aware that if you leave the *Data object* field empty, the document template will theoretically be available for all customer groups, but there might be runtime errors if the required fields are not found when the template is loaded. ConSol CM will display a message that field content is missing and the fields will be filled with the string 'n/a'. However, we recommend that you always select a data object to keep the system in good working order.

Availability of Document Templates in Tickets

If the document template is available in a certain ticket (see section Using Document Templates in the Web Client below) depends on two parameters:

- **Queue**
  The template is only listed for tickets in the selected queue(s).
- **Customer data model**
  The template is only listed in tickets which have, as main customer, a customer object which is based on the selected data object.

The following figure shows a document template which is defined for the queue *ServiceDesk* and for the data object *ResellerCustomer*.

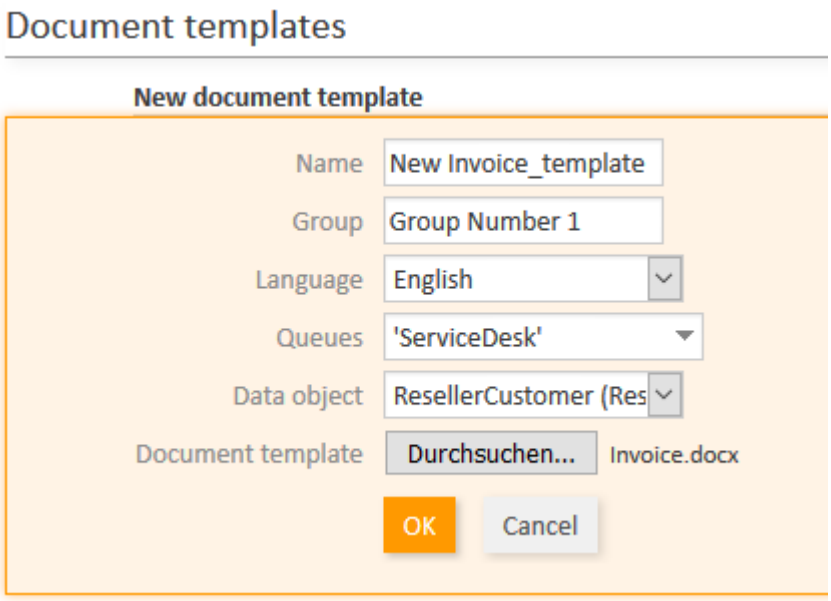


Figure 383: *ConSol CM Web Client - Definition of a document template for one queue and for a certain data object*

The availability of the document template depends on the queue and customer group of the current ticket:

- In the Web Client, the template **can** be used in a ticket which is in the queue *ServiceDesk* **and** which has a main customer of the *ResellerCustomer* data object type.
- In the Web Client, the template **cannot** be used in a ticket which is in the queue *ServiceDesk* and which has a main customer of a data object which is **not** of the *ResellerCustomer* data object type.
- In the Web Client, the template cannot be used in all tickets which are **not** in the queue *ServiceDesk*.

Creating and Editing the Document

When you have filled in all fields and clicked *OK*, the new template appears in the *Template library*, *List of templates*. After entering the new template data, you can perform the following step directly or you can click on the name of the template file (in the *Document template* column) to download and edit the template.

In the next step, *MailMerge* fields, which represent the fields of ticket and customer data, can be added to the template. Select a ticket which has all the required fields by using the *Document MailMerge fields* section. Enter the ticket name or subject in the field under *Show MailMerge fields available for a ticket* and select the correct one from the search result list.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

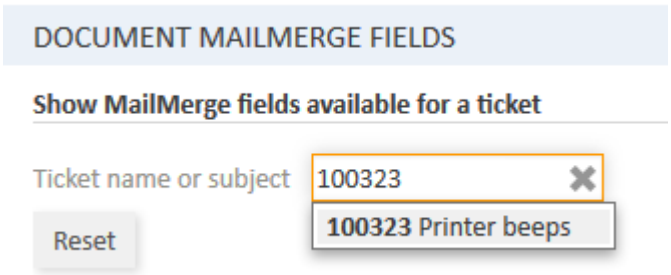


Figure 384: *ConSol CM Web Client - Selection of a ticket to see all required data fields*

All available *MailMerge* fields are displayed.

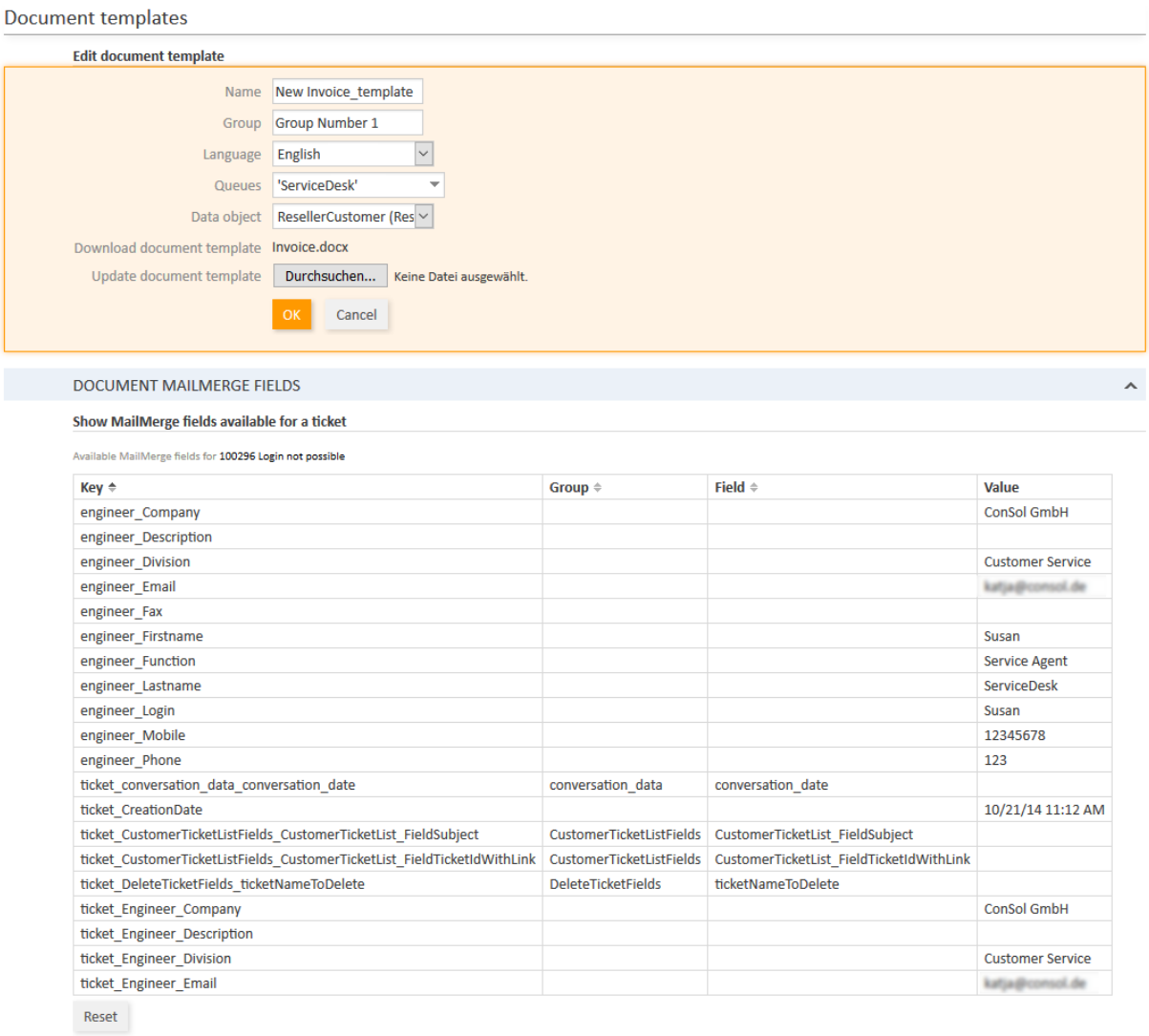


Figure 385: *ConSol CM Web Client - Show available MailMerge fields*

The list contains the following columns:

- **Key**
  The name of the field (this is the one you will have to use for the MergeField later on)

- **Group**
  This is set for customer data and for ticket data. For customer data, the customer field group is shown (here, only the customer field groups from the selected data object are offered). For ticket data, the ticket field group is shown.
- **Field**
  This is set for customer data and for ticket data. For customer data, the customer field name is shown (here, only the customer fields from the selected data object are offered). For ticket data, the ticket field name is shown.
- **Value**
  The value of this field in the selected ticket. You do not have to use this in the document template.

You can use all fields which are available in the ticket, i.e.:

- **Company fields**
  Customer fields from the selected data object if this is a company, or from the company of the selected data object if the selected object is a contact. The keys start with *ticket_queue_fields_* and have a customer field group name as *Group*.
- **Contact fields**
  Customer fields from the selected data object if it is a contact, or from the contact of the selected data object if the selected data object is a company. The keys start with *ticket_queue_fields_* and have a customer field group name as *Group*.
- **Ticket data fields**
  Ticket fields from the ticket. The respective keys start with *ticket_* and have a ticket field group name as *Group*.
- **Engineer data**
  - Data concerning the current engineer of the ticket. The respective keys start with *ticket_Engineer_* and do not have *Group* and *Field* names.
  - Data from the engineer who is currently logged in. This must **not** be confused with the engineer of the ticket! The respective fields start with *engineer_* and do not have *Group* and *Field* names.

Download and open the document template from one of the following locations. In both cases, the technical file name of the template is shown:

- from the *Edit document template* section (at *Download document template*)
- from the list in the *Template Library* (*Document template* column)

Add the required fields to the document. See Editing the Document in Microsoft Word and Editing the Document in OpenOffice for detailed steps. The following figure shows an example of a document template with some MailMerge fields.



Figure 386: *Example for a document template*

Save the document in the local file system and then upload it using the field *Update document template*. This will store the template in the Document Template Manager.

Editing the Document in Microsoft Word

Please proceed as follows to add a field to the Microsoft Word document:

1. Go to the menu entry *Insert -> Quick parts -> Field*.
2. Select *MergeField*.

3. Copy and paste the key of the MailMerge field from the Web Client into the field *Field name* below *Field properties* (see following figure).
4. Click the *OK* button to insert the field into the document at the current cursor position.

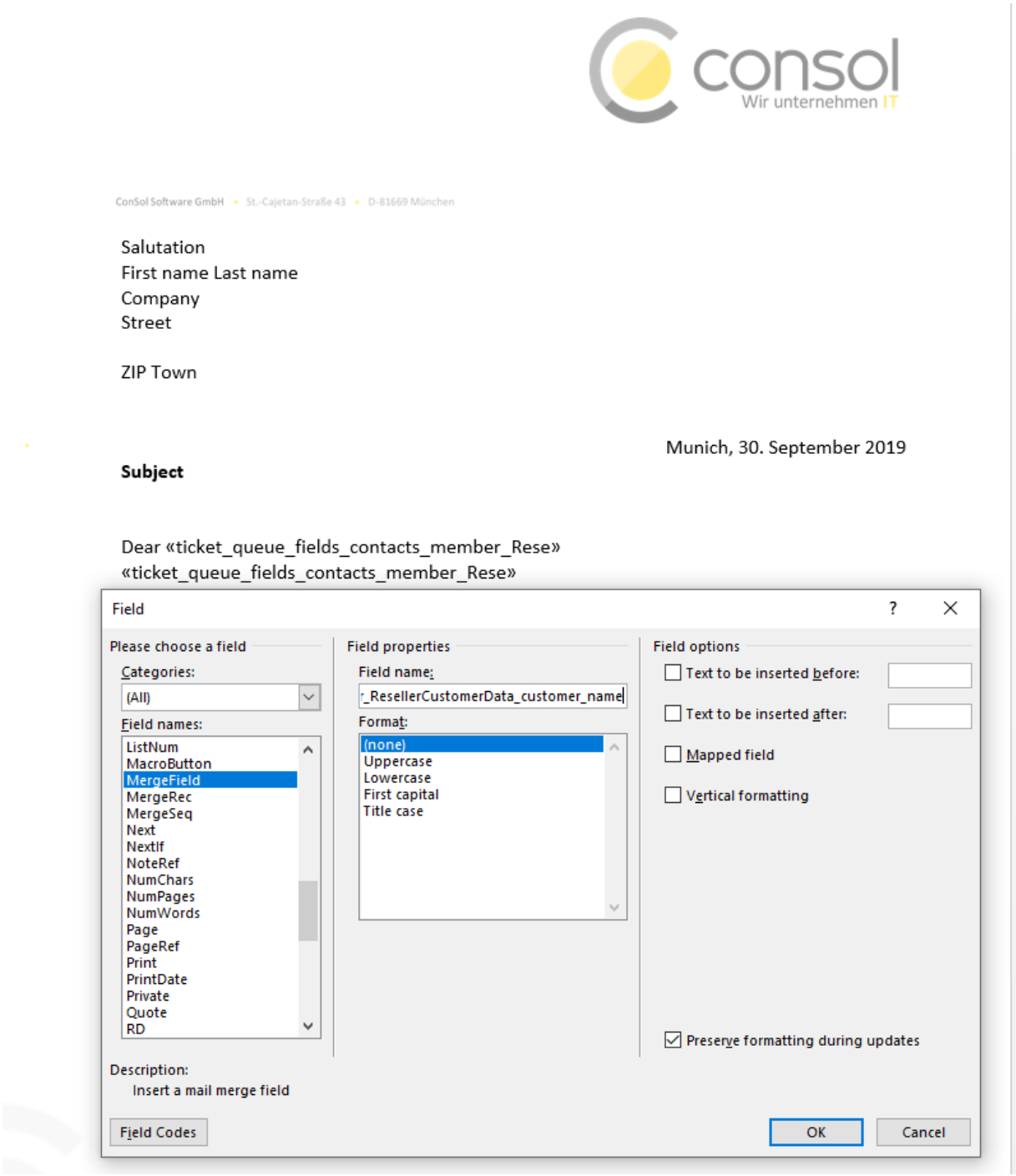


Figure 387: *Microsoft Word - Insert MergeField into document*

Editing the Document in OpenOffice

Please proceed as follows to add a field to the OpenOffice document:

1. Go to the menu entry *Insert -> Fields -> Other*.
2. In the *Fields* dialog, select the tab *Variables*.
3. Select *User Field* as a *Type* and *Text* as a *Format*.
4. Enter the field name (from the *Key* column in the *Document MailMerge fields* section) in the *Name* field below the *Type*.
5. Add a useful *Value* for recognizing the field in the document.
6. Click the green checkmark to the right of the field *Value*, so that the field is shown under *Selection* in the middle, and make sure it is selected there.
7. Click the *Insert* button to insert the field into the document at the current cursor position.

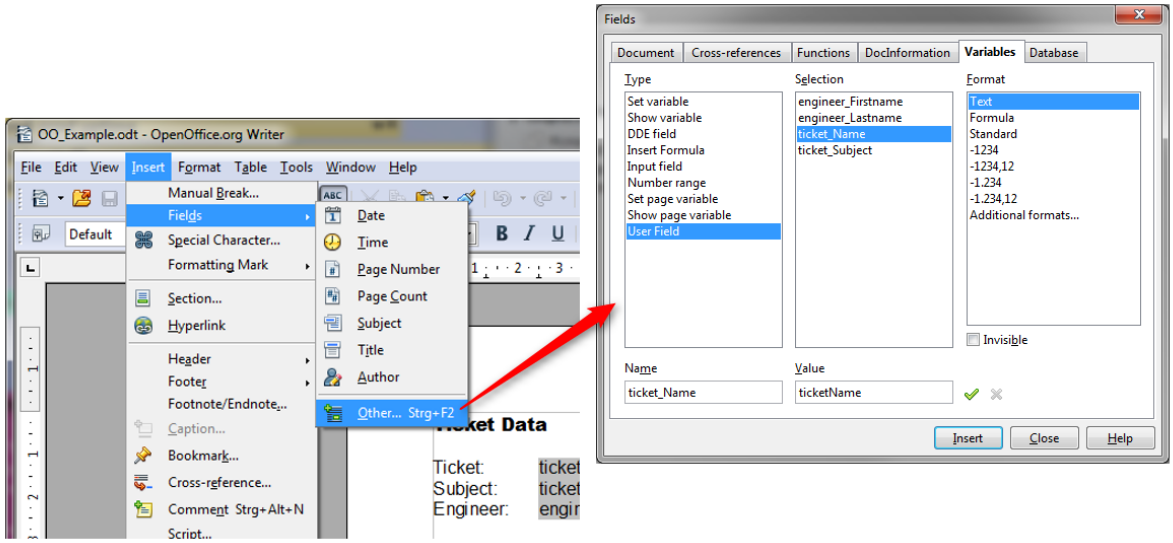


Figure 388: *Creating a document template using OpenOffice*

Details about the Keys for MailMerge Fields

For customer data, there are always two keys for the same value, i.e., two keys which in the end retrieve the value from the same customer field. You can always use either one - there is no difference as far as the behavior in the templates is concerned.

The two keys are:

1. A generic field, e.g., ticket_queue_fields_contacts_member_companyReferenceField_**company_name**
2. A field which comes from the specific customer data model according to the following syntax: xxx_[data object]_[customer field group]_[customer field], e.g., ticket_queue_fields_contacts_member_companyReferenceField_ResellerCompany_ResellerCompanyData_**company_name**

F.13.2.5 Using Document Templates in the Web Client

Creating a New Attachment Using a Document Template

When document templates are available for the ticket, an engineer can use them by clicking *Document* in the history section of a ticket and selecting the requested template. If CM/Doc is not running

yet, it is started. Afterwards, Microsoft Word or OpenOffice is started and a document based on the selected template is created. The document is opened, with all values/parameters filled in at the correct positions. This might look like the example in the following figures.

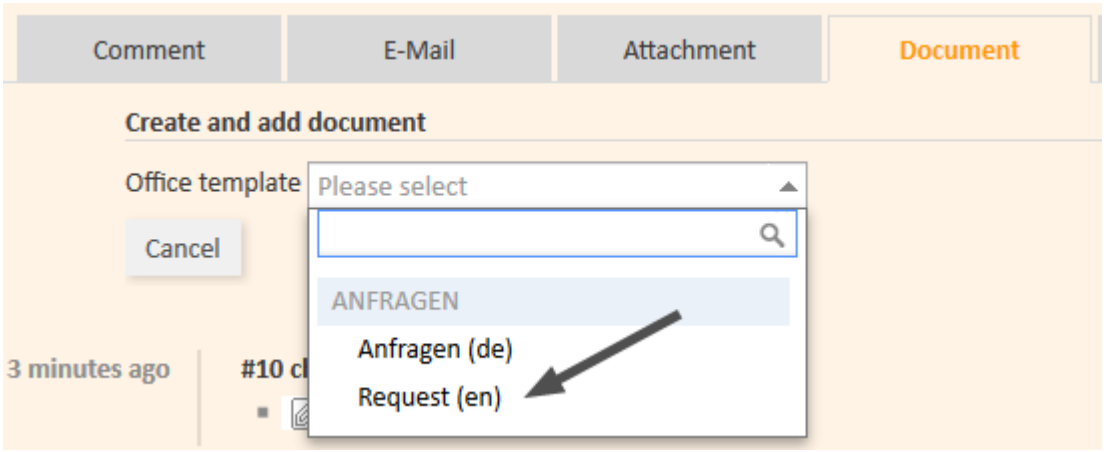


Figure 389: *ConSol CM Web Client - Document templates in Document tab*



Figure 390: *Microsoft Word example document*

The engineer can then edit the document, if required, and save it. This will automatically attach the new version of the document to the ticket.

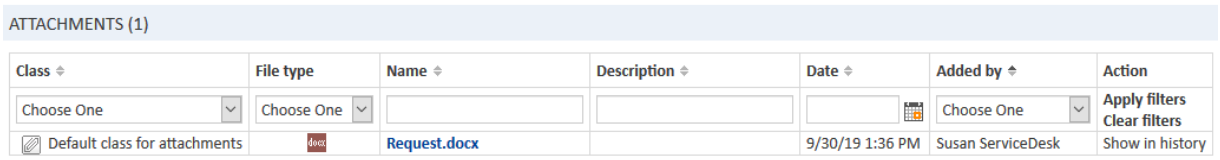


Figure 391: *ConSol CM Web Client - Newly edited Word document as attachment at the ticket*

From here on, working with the Word document is exactly the same as working with regular Word (.doc, .docx) attachments.

Working with Existing Document Attachments

An engineer can also open a document which has been attached to the ticket. As an engineer, click on the attachment name. This will open the file in Microsoft Word or OpenOffice. Edit the document and save it. A new version of the document will be attached to the ticket automatically.

F.13.2.6 Using Document Templates in Scripts

You can use document templates in automatic actions to attach a document, which is based on a certain template, to a ticket, customer or resource. It is also possible to generate a PDF document from a template or to create a document from scratch.

This is done using methods of the class `ContentFileTemplateService`:

- `generateWordDocumentByTemplate` to generate a DOCX document from a CM/Doc template
- `generateWordDocumentFromXHTML` to generate a DOCX document from XHTML input
- `generatePDFDocumentFromXHTML` to generate a PDF document from XHTML input
- `generatePDFDocumentFromWord` to generate a PDF document from a DOCX document

> If you attach a DOXC document generated from a document template to a ticket, the merge fields are filled with the data from the respective ticket and ticket customer.

The following example shows a workflow script which attaches four documents to the current ticket. The file extension is added automatically, so you only need to provide a file name. If you create a document from a template, the name of the template is used as a file name.

```
Ticket ticket = workflowApi.ticket;
String html =
 "<html><header></header><body><h1>H1Text</h1><p>Text</p></body></html>";
ContentFile docxContentFile;
ContentFile pdfContentFile;

// generating docx from office template (template with testDocx name must be
 prepared before)
String templateName = "testDocxFromTemplate";
docxContentFile = contentFileTemplateService.generateWordDocumentByTemplate
 (templateName, ticket);
addContentFileAsAttachmentEntry(ticket, docxContentFile);
```

```
// generating pdf from docx
pdfContentFile = contentFileTemplateService.generatePDFDocumentFromWord
 (docxContentFile);
addContentFileAsAttachmentEntry(ticket, pdfContentFile);

// generating docx from XHTML
String fileName = "testDocxFromXhtml";
docxContentFile = contentFileTemplateService.generateWordDocumentFromXHTML
 (fileName, html);
addContentFileAsAttachmentEntry(ticket, docxContentFile);

// generating pdf from XHTML
fileName = "testPdfFromXhtml";
pdfContentFile = contentFileTemplateService.generatePDFDocumentFromXHTML(fileName,
 html);
addContentFileAsAttachmentEntry(ticket, pdfContentFile);

def addContentFileAsAttachmentEntry(Ticket pTicket, ContentFile pContentFile) {
AttachmentEntry attachmentEntry = new AttachmentEntry();
attachmentEntry.setFile(pContentFile);
attachmentEntry.setDescription(pContentFile.getName());
attachmentEntry.setMimeType(pContentFile.getMimeType());
ticketContentService.createContentEntry(pTicket, attachmentEntry);
}
```

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.14 Time Booking Using ConSol CM

This chapter discusses the following:

F.14.1 General Introduction to Time Booking Using ConSol CM 668
F.14.2 Manual Time Bookings 668
F.14.3 Automatic Time Bookings 674
F.14.4 DWH Reports 677
F.14.5 Page Customization for Time Booking 677
F.14.6 Using Time Booking Data in Scripts 677

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.14.1 General Introduction to Time Booking Using ConSol CM

In ConSol CM, working hours can be booked to tickets. There are two booking modes:

- Manual bookings, see section Manual Time Bookings
- Automatic bookings, see section Automatic Time Bookings

F.14.2 Manual Time Bookings

In ConSol CM, an engineer can book working hours to a ticket. Those working hours can then be reported.

With manual time booking, working hours are always booked on projects which have to be assigned to one or more queues. For example, if your company plans to perform a migration from Windows 8 to Windows 10 clients and all the working hours should be registered for this migration project, the ConSol CM administrator has to create a migration project and assign it to all queues where tasks for this project might be performed. Then engineers can book their times on the project and can see their own reports for the project. Additionally, a report over all time bookings, of all engineers, may be implemented using the DWH (Data Warehouse, see section Data Warehouse (DWH) Management).

F.14.2.1 Configuration of Manual Time Booking Using the Admin Tool

In order to enable engineers to book working hours on projects the ConSol CM administrator has to perform two steps using the Admin Tool:

1. Create the projects on the navigation item *Projects* (navigation group *Global Configuration*), see section Projects.
2. Assign one or more projects to the desired queues within the Queues.

In the following example, three projects are created. Engineers in the *HelpDesk_1st_Level* queue should be able to book working hours on two of them. Thus, the two projects have to be assigned to the *HelpDesk_1st_Level* queue.

You reach the following screen by selecting the navigation group *Global Configuration*, navigation item *Projects*.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

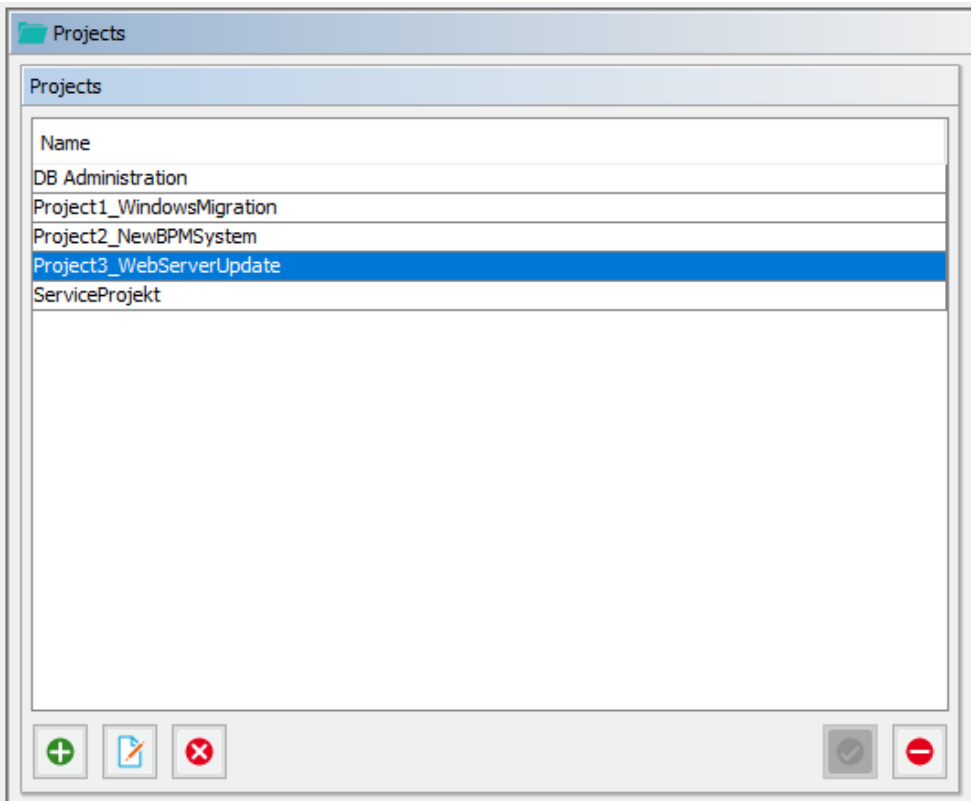


Figure 392: *ConSol CM Admin Tool - Global Configuration, Projects: Management of projects*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

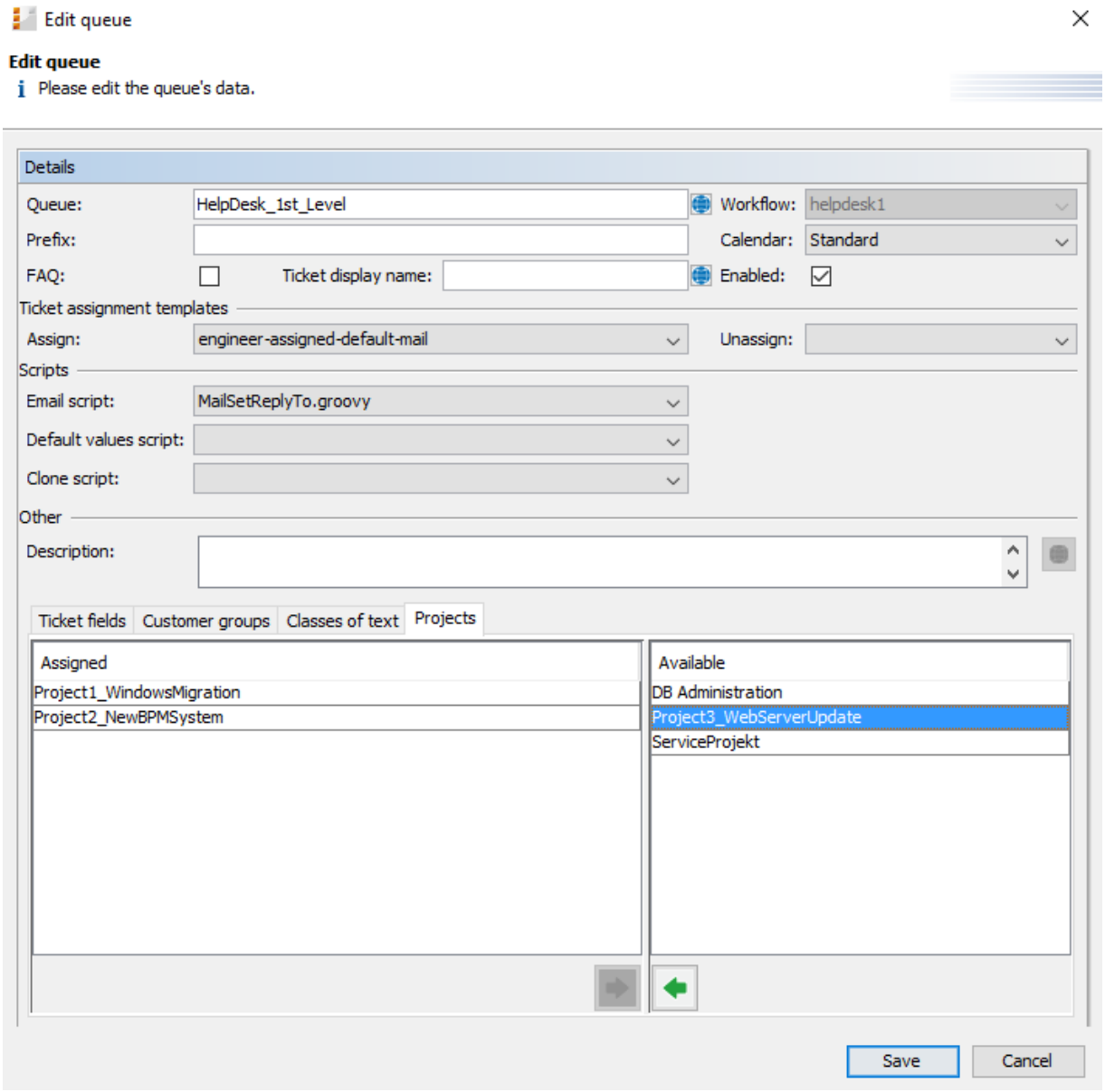


Figure 393: *ConSol CM Admin Tool - Global Configuration, Queues: Assigning projects to a queue*

F.14.2.2 Manual Time Booking from a User's Point of View (Web Client)

Please see the *ConSol CM User Manual* for a detailed explanation of the time booking feature. Here, only a brief overview is provided.

The user (engineer) can book working hours on a ticket using two different modes:

- Using the *Time booking* section in a **ticket** to book working hours directly on this ticket.
- Manual time bookings on the engineer profile page.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Using the *Time booking* Section in a Ticket to Book Working Hours Directly on This Ticket

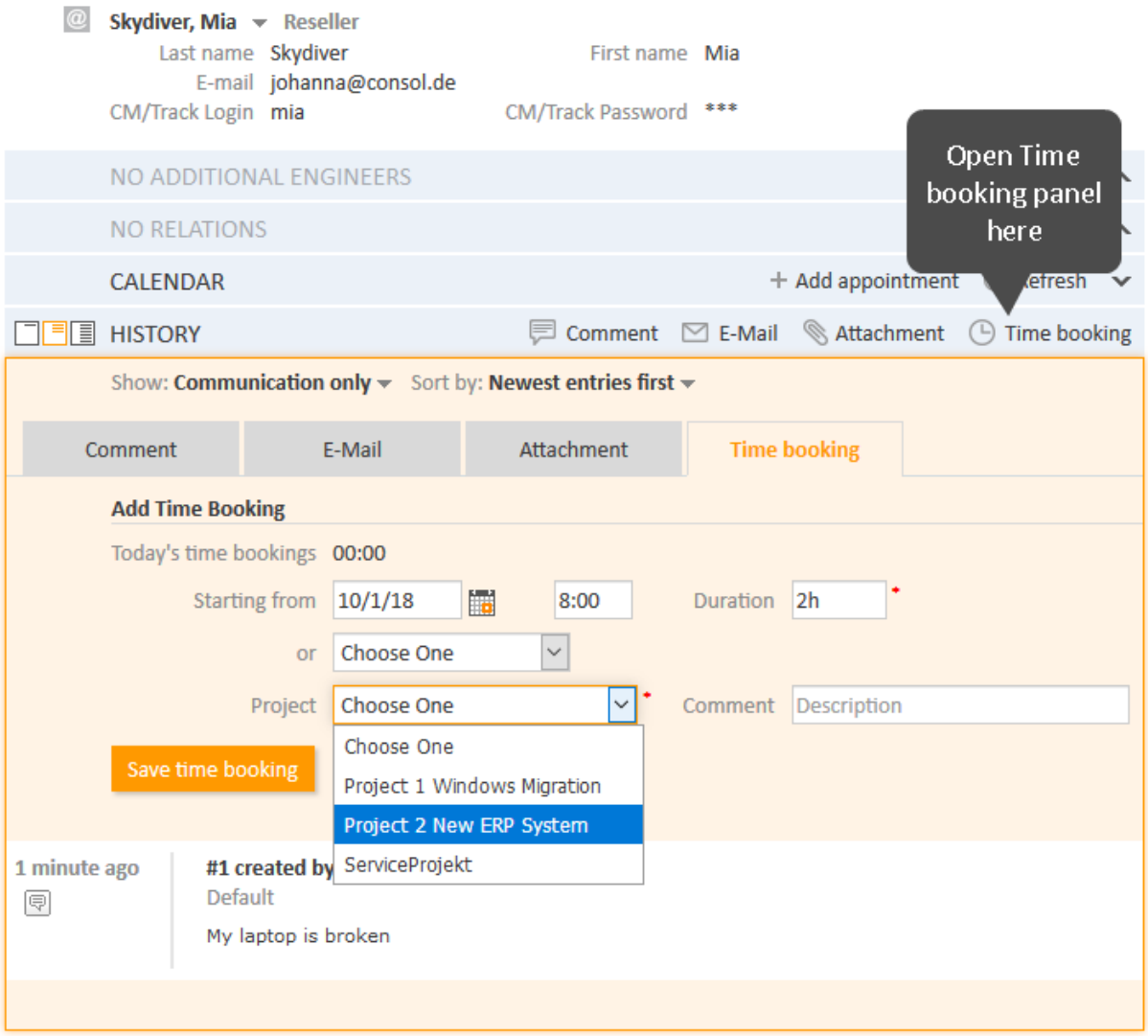


Figure 394: *ConSol CM Web Client - Time booking in a ticket*

Manual Time Bookings on the Engineer Profile Page

Using the *Time booking* section on the **engineer profile page** to book working hours on a ticket. Only tickets where the engineer has performed certain activities and tickets owned by the engineer can be selected. A project also has to be selected from the list.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

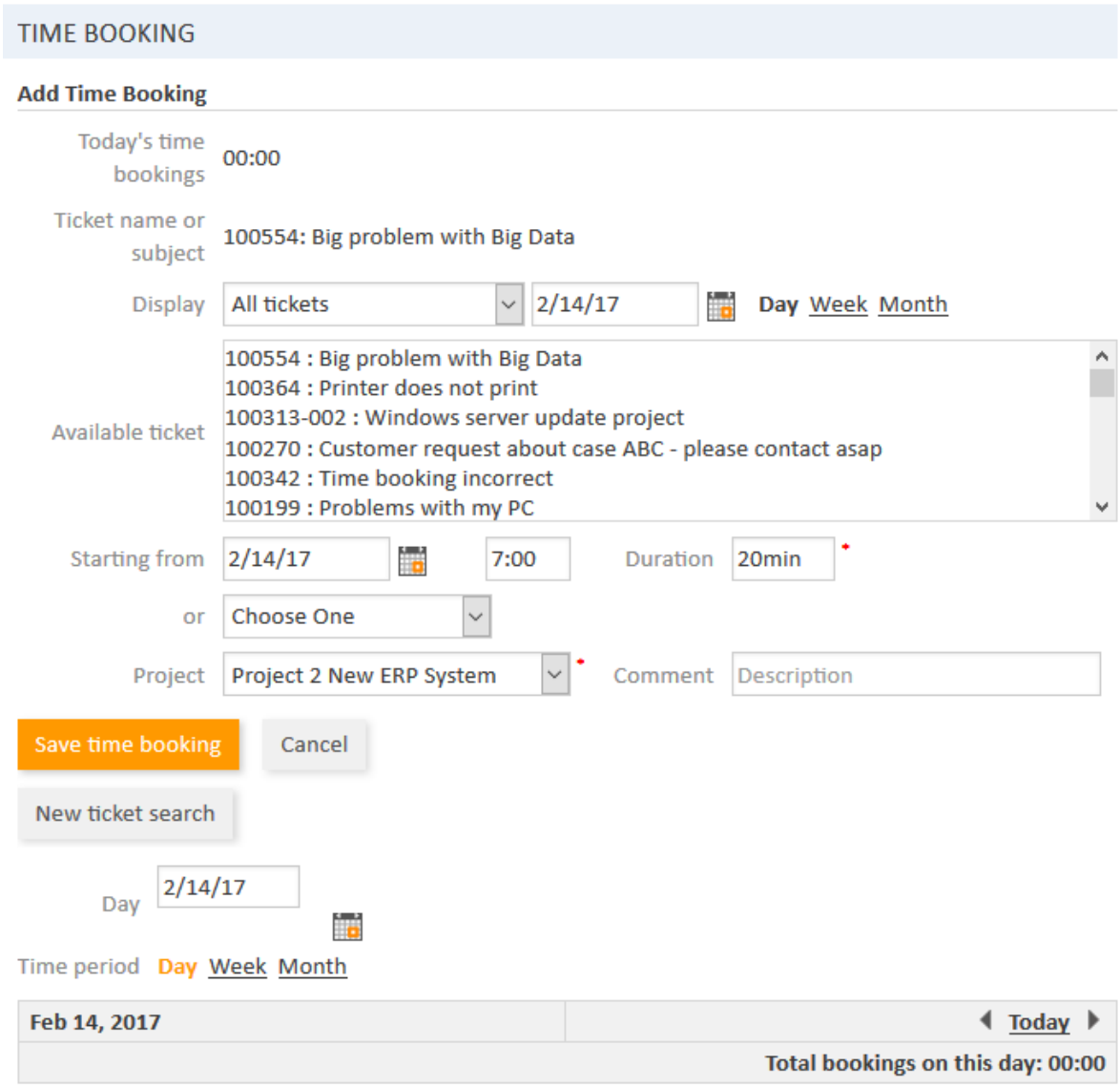


Figure 395: *ConSol CM Web Client - Time booking on the engineer profile*

Engineers can see a list of their time bookings on the engineer profile page. An example is shown in the following figure.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

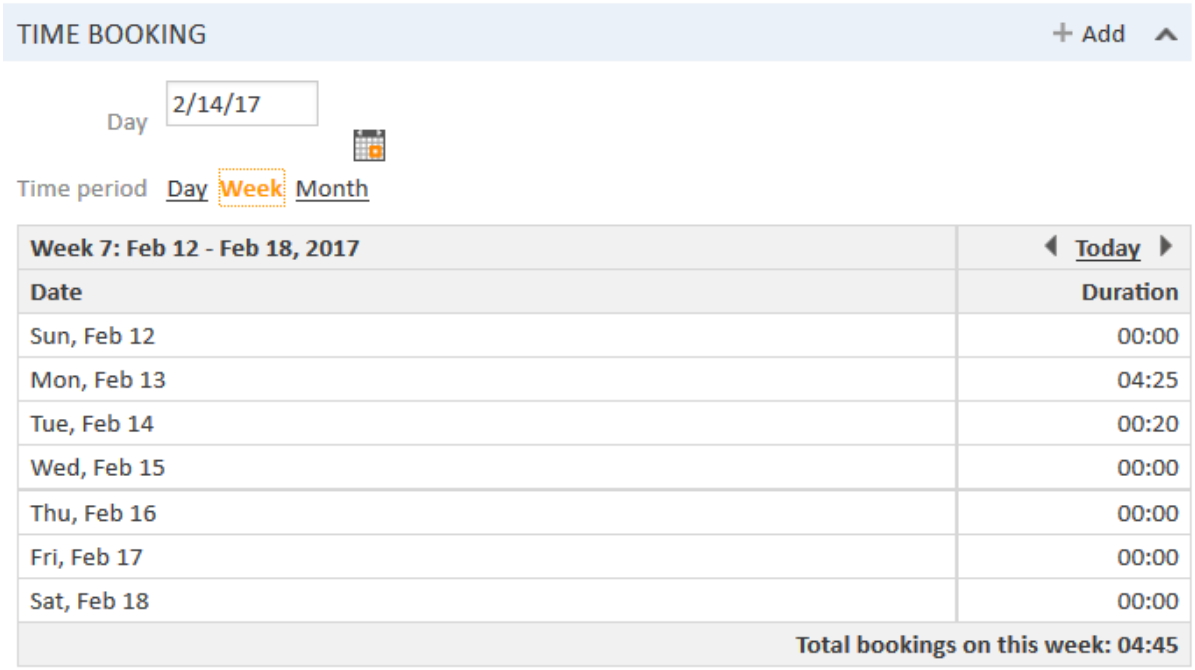


Figure 396: *ConSol CM Web Client - Time booking report on the engineer profile*

As an engineer, you can select whether you would like to see the bookings for the current day, week, or month. In the *Day* view, the projects are indicated, in the *Week* and *Month* view, only the sum of the booked times per day/week is indicated.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.14.3 Automatic Time Bookings

F.14.3.1 Introduction to Automatic Time Booking

ConSol CM can be configured in a way that working hours are tracked and booked on tickets automatically. These bookings always refer to tickets and cannot be linked to projects.

The following times are registered:

- **Duration of work with the Rich Text Editor**
  Started when the Rich Text Editor is opened and stopped when the *Add* button is clicked.
- **Duration of creating a ticket**
  Started when *Create ticket* is selected and stopped when the *Create* button is clicked.

Time booking is suspended when a ticket is transferred to the workspace and resumed when the ticket is brought back to the active work.

No times are booked on the ticket when ...

- an operation is canceled
- the engineer logs out manually
- the engineer session is ended automatically

Times are always booked with minute-precision and are always rounded up to the next full minute.

F.14.3.2 Configuration of Automatic Time Booking

In order to enable this functionality in your ConSol CM system, set the system property cmweb-server-adapter, automatic.booking.enabled to “true”.

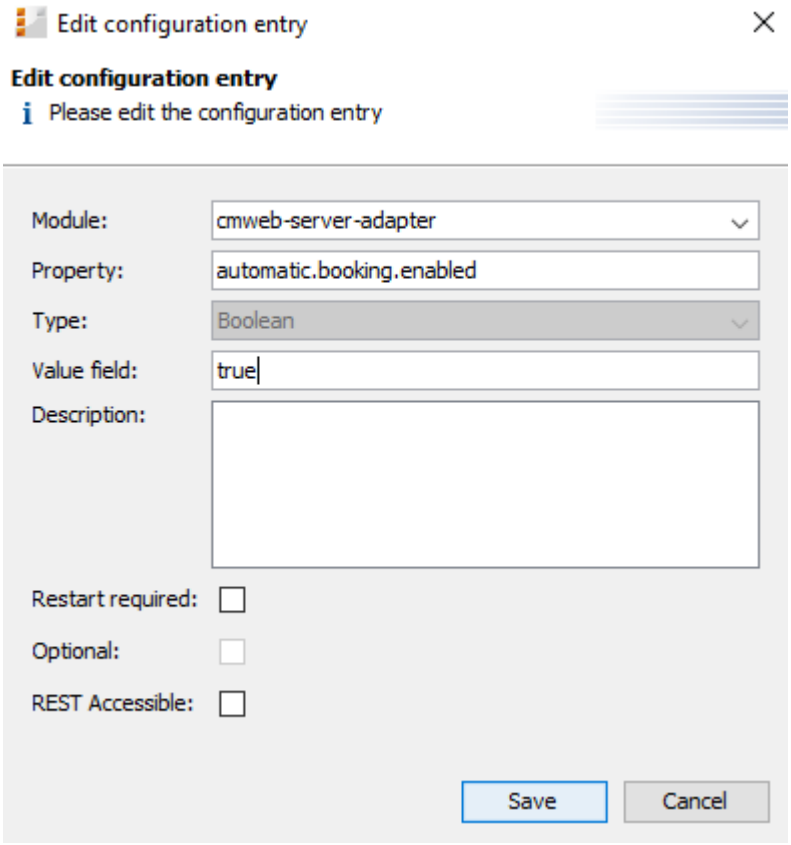


Figure 397: *ConSol CM Admin Tool - Set system property to switch on automatic time booking*

F.14.3.3 Automatic Time Bookings from a User's Point of View (Web Client)

The engineer does not have to do anything in particular to work with automatic time booking. When he enters a comment in a ticket or creates a ticket, the time is booked on this ticket automatically and can be seen in the **time booking report on the engineer profile page**. An example is shown in the following figure.

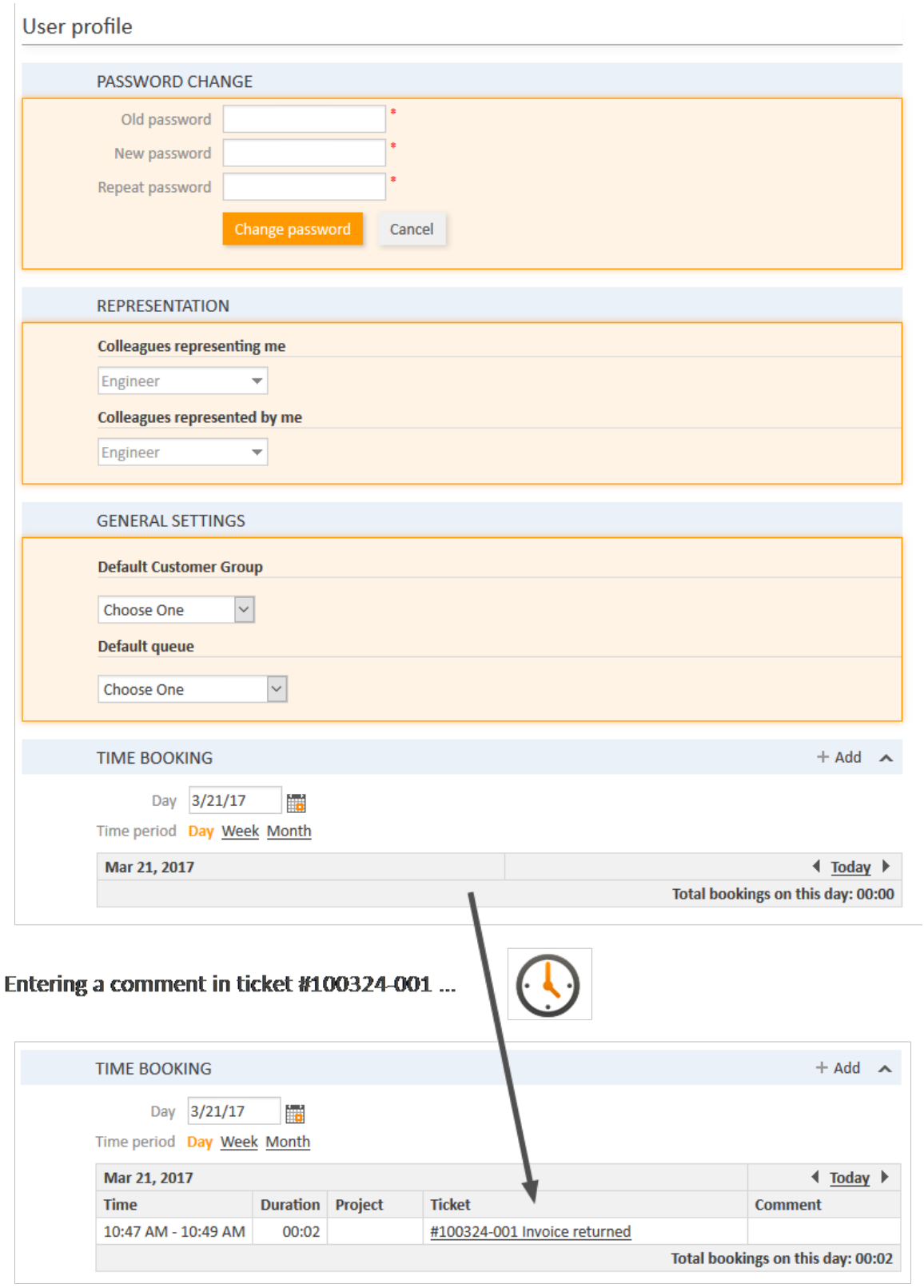


Figure 398: *ConSol CM Web Client - Automatically booked time in time booking report on engineer profile page*

F.14.4 DWH Reports

If your company would like to be able to report at a more detailed level, the DWH provides a good basis. Reports can be developed that use the DWH data and provide, e.g., the times booked on a certain project by all engineers.

F.14.5 Page Customization for Time Booking

If the time booking feature is not required, you can turn off the feature by using the *Page Customization*, see section Page Customization for details.

The following two parameters are relevant in this context:

- timeBookingSection
- timeBookingFeature

F.14.6 Using Time Booking Data in Scripts

Starting with ConSol CM version 6.11, the CM API contains a method to retrieve the time intervals booked on a ticket. Use the following method of the object `workflowApi`.

```
Set<TimeBooking> getBookings(Ticket pTicket)
```

You can use this method in Admin Tool scripts and in workflow scripts. Please note that in Admin Tool scripts, the workflow context might not be available!

The following example shows a script which can be called from a workflow activity. You might want to write the code directly into the script or place it in an Admin Tool script which is called from the workflow. For details about this principle, please refer to the *ConSol CM Process Designer Manual*.

F.14.6.1 Example of a Time Booking Script

The script:

```
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'

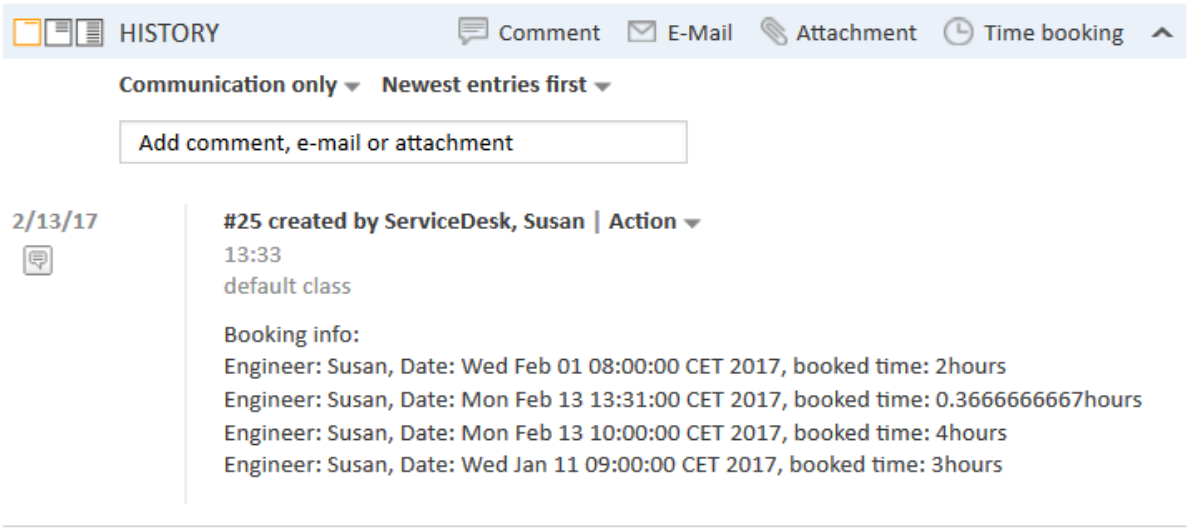
if (bookings.size() > 0){
   log.info 'Time bookings found in ticket ' + ticket.name
   bookings.each() { bk ->
      eng = bk.engineer.name
      per = bk.timePeriod
      per_date = per.bookingDate
      per_time = per.bookedTime /1000 / 60 / 60
      def text1 = 'Engineer: ' + eng + ', Date: ' + per_date + ', 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 example 84: *Admin Tool script, called from a workflow: all time intervals booked on the ticket are printed into the ticket history as comment*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

The output in the respective ticket:



CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.15 Authentication Methods in ConSol CM

For the authentication of users in ConSol CM, you can configure one of the following three methods:

- **Standard / Database**
  All user data is kept in the ConSol CM database. This authentication method is available for engineers (see Database Authentication for Engineers) and customers (see Database Authentication for Customers).
- **LDAP**
  The credentials are kept in an LDAP server. ConSol CM sends a request to this server to authenticate the users. This authentication method is available for engineers (see LDAP Authentication for Engineers in the Web Client) and customers (see LDAP Authentication for Customers in CM/Track).
- **Kerberos SSO**
  The credentials are taken from a valid session using Kerberos, see Single Sign-On in a Windows Domain Using Kerberos . This authentication method is only available for engineers in the Web Client.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.15.1 Authentication Methods for Engineers in the Web Client

There are three methods for authenticating engineers in the Web Client:

- **Database authentication**:
  The login and the password are stored in the CM database, see Database Authentication for Engineers.
- **LDAP authentication**:
  The login is stored in the CM database, in the engineer administration, the password is stored on an LDAP server, see LDAP Authentication for Engineers in the Web Client.
- **Kerberos authentication**:
  The login is stored in the CM database, in the engineer administration, the password is stored in Kerberos, see Single Sign-On in a Windows Domain Using Kerberos .

F.15.1.1 Database Authentication for Engineers

Setting the User Name and Password

The user name and password are provided in the engineer data (navigation group *Access and Roles*, navigation item *Engineers*, see Engineers). The following fields are relevant for database authentication:

- **Login:**
  Mandatory. This field contains the account name which has to be entered on the login page of the Web Client. Please use only international alphabetic and numeric characters, no blanks, punctuation marks, or special characters such as umlauts, hyphens, or the like.
- **Email:**
  Mandatory. The engineer's email address. Please use only international alphabetic and numeric characters, hyphens, underscores, periods, and the @ sign. The entry of multiple email addresses in one line is not allowed.
- **Password:**
  Mandatory. The engineer's password is mandatory. Please use only international alphabetic and numeric characters, and punctuation marks, do **not** use any special characters such as, e.g., umlauts. The password entered will be shown as a string of asterisks. Please see section Configuring the Password Policy for information about the optional password policy.

Configuring the Password Policy

This configuration is optional.

The following system properties can be used to implement a certain password policy. All following system properties are located in the module `cmas-core-security`. (For details about system properties, please refer to the section System Properties.)

- policy.password.pattern (String)
  RegEx pattern for the password, default value: “^.3,$” (at least 3 characters)
  Example: “^(?=.*[A-Z])(?=.*[0-9])(?=.*[a-z]).{5,}$” (at least 5 characters, at least one upper case letter, one lower case letter and one number)
- policy.password.age (Integer)
  Maximum validity period, in number of days, example “183” (6 months), default value: “5500” (= 15 years, i.e., no password change enforced).

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- policy.rotation.ratio (Integer)
  This defines the number of previous passwords which may not be identical, example and default value: “1”.
- policy.username.case.sensitive (Boolean)
  Defines whether the password is case-sensitive. Example and default value: “true”. Note that this setting is affected by the MySQL collation setting and needs the correct collation to work properly with MySQL.

Resetting an Engineer's Password

If an engineer has forgotten his password, he can request a new password by using the link *Forgot your password* on the initial login page. An email with a link to a URL where the engineer can set the new password is sent to the engineer.

Please note that this can only work if a valid email account is available for this engineer and if the respective value has been entered as email for the engineer in the engineer data!

The email which is sent to the engineer is based on the template `password-reset-template` which is stored in the *Templates* section of the Admin Tool. Please see section Admin Tool Templates for a detailed explanation of templates in general and section Password Reset Template for Engineers in the Web Client for details about the engineer password reset.

The password reset in the Web Client is only possible if the standard authentication mode is used. It is not possible if LDAP or Kerberos authentication is in operation. See section Authentication Methods in ConSol CM for an explanation of all possible authentication modes.

F.15.1.2 LDAP Authentication for Engineers in the Web Client

Introduction to ConSol CM LDAP Authentication

ConSol CM offers LDAP authentication for the Web Client as a standard feature, i.e., instead of managing the passwords for the ConSol CM engineers in the ConSol CM database, they can be retrieved from an LDAP server (like e.g., a ***Microsoft Active Directory*** server).

When engineers want to log in to the ConSol CM Web Client, they enter their user name and password and press *Enter*. Behind the scenes, the ConSol CM server sends a request with the engineer's user name and password and asks the LDAP server whether those credentials are correct.

If the credentials are correct, the approval is sent back to the ConSol CM server and the engineer is logged into the Web Client.

> Please keep in mind that the LDAP connection is only used to authenticate the user (confirm the identity). The authorization (i.e., the assignment of access permissions in the system) is done via the engineer and role administration in the Admin Tool. For every user who should work with the system as an engineer, an engineer account has to be created in the engineer administration!

Please see also the following picture for an explanation of the CM authentication process using LDAP.

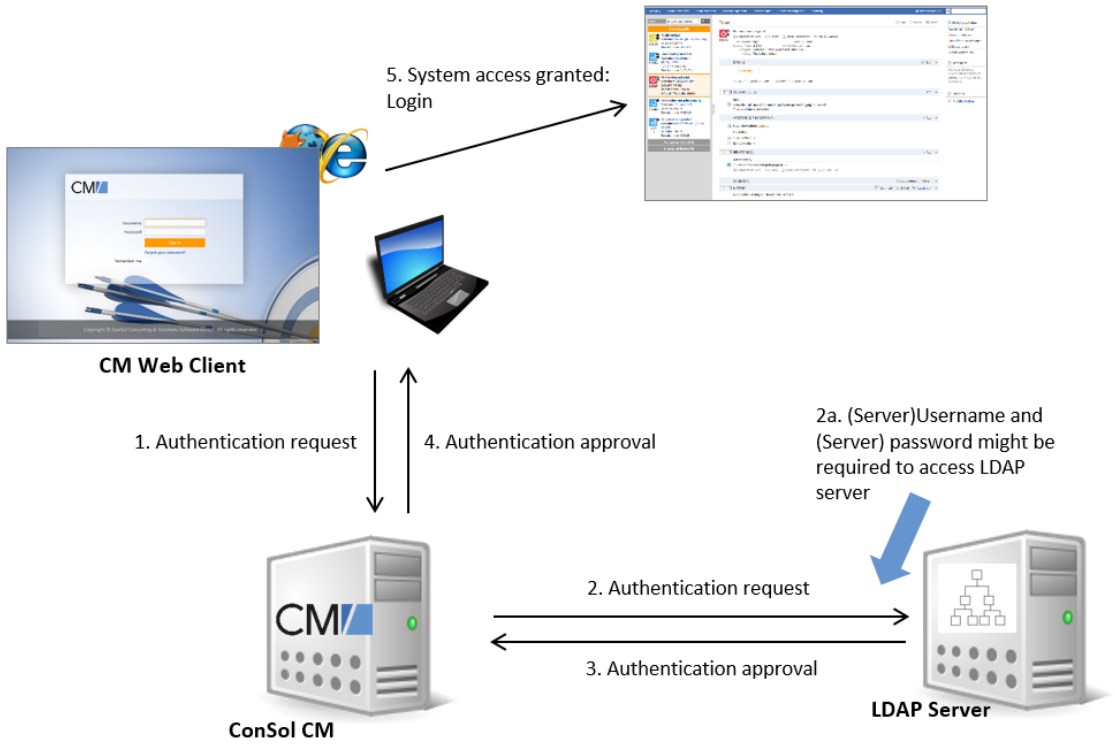


Figure 399: *ConSol CM - LDAP authentication process*

Configuring LDAP Authentication

There are two ways you can enable the ConSol CM system to use LDAP authentication for engineers in the Web Client:

- Select LDAP authentication during system setup and enter the requested parameters (system properties) after the setup.
- Set up the system with the regular authentication mechanism and switch to LDAP later on, i.e., enter all required system properties later on.

Configuring LDAP During Initial Setup

During system setup you can select *LDAP* as the authentication mode on the *Administrator* screen (third step). This will set the system property cmas-core-security, authentication.method (see below) to “LDAP”. No further parameters are entered. You have to set the LDAP parameters manually. Please see the next section for an explanation.

Switching the Authentication Mode to LDAP in a Running System

To switch the authentication method to LDAP, you have to set the required values in the system properties (navigation group *System*, navigation item *System Properties*, see System Properties):

- authentication.method
  LDAP
- ldap.authentication
  simple
- ldap.basedn
  The DN (distinguished name) of the LDAP (sub-)tree where the required attributes are located.

- ldap.initialcontextfactory
  The Java class name for the initial context factory of the LDAP implementation when using LDAP authentication. Should usually be `com.sun.jndi.ldap.LdapCtxFactory`.
- ldap.password
  Password for connecting to the LDAP server to look up users. Only needed if look-up cannot be done anonymously.
- ldap.userdn
  LDAP user for connecting to the LDAP server to look up users. Only needed if look-up cannot be done anonymously.

  > ⚠ A server user name/password pair might be required to access the LDAP server. If you are not sure, you might want to use an LDAP browser to confirm.

- ldap.providerurl
  The complete URL for the LDAP server:

  ```
  ldap://<HOSTNAME>:<LDAP PORT>
  ```

- ldap.searchattr
  Search attribute for looking up the LDAP entry connected to the CM login, i.e., the attribute which is used as user name for the authentication.

Using LDAPS (LDAP over SSL)

Introduction

Per default, when an LDAP client accesses an LDAP server, the information is transferred in clear text. In case you want the user name and password to be transferred to the LDAP server in encrypted form, you have to set up the LDAP authentication using LDAPS.

Preparations

You have to configure the CM server machine (Java) in a way that can use certificates. One way to do this for a Linux environment is described in the following section.

1. Retrieve the certificate:

   ```
   openssl s_client -connect dc2.mydomain.com:ldaps
   ```

2. The answer will contain a section which starts with `"---BEGIN CERTIFICATE "` and ends with `"END CERTIFICATE ---"`. Copy this section to a file, e.g., `/tmp/certificate2_dc2_mydomain_com.txt`
3. Import the certificate to the truststore of your machine, e.g., `/home/mydirectory/mytruststore`

   ```
   $JAVA_HOME/bin/keytool -import -alias <arbitrary> -trustcacerts -keystore
    /home/mydirectory/mytruststore -file/tmp/certificate2_dc2_mydomain_com.txt
   ```

You have to enter (set) a password.

4. Enter the truststore in the ConSol CM config file in JAVA_OPTS:

```
-Djavax.net.ssl.trustStore=/home/mydirectory/mytruststore -
 Djavax.net.ssl.trustStorePassword=<see above>
```

LDAPS Configuration in the ConSol CM Admin Tool (System Properties)

Configure the ConSol CM server as shown in the following example:

- 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

Depending on the LDAP server configuration, use one of the following values for the 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

Configuring Engineer Accounts for LDAP Authentication

Use the engineer administration (navigation group *Access and Roles*, navigation item *Engineers*) in the Admin Tool to configure the engineer accounts.

When LDAP is used as authentication method, it is not possible to set the ConSol CM password within the engineer administration. The pop-up window for engineer management provides the following fields which are relevant for LDAP authentication.

- **Login**
  If no LDAP ID has been provided, this is used as the user name during the LDAP authentication process which is looked up in the LDAP directory in the `ldap.searchattr` node.
- **LDAP ID**
  If you would like to employ special user names in ConSol CM which are not identical to the values used in the LDAP directory, fill in this field. During the LDAP authentication process, this LDAP ID is used as the user name which gets looked up in the LDAP directory in the `ldap.searchattr` node.

Please refer to section Engineers for details concerning the other (non LDAP-related) data fields.

F.15.1.3 Single Sign-On in a Windows Domain Using Kerberos

Short Introduction to Kerberos

Kerberos is a network protocol which is used to authenticate a user or a system to verify if a user or system has the identity it claims to have. The authentication mechanism is based on so called *tickets*. When a user or system has been authenticated by a Kerberos system, other systems, e.g., a ConSol CM server, can rely on this authentication and grant access for the user or system to their own resources. One of the advantages of using Kerberos is that no passwords are sent over the network.

Only information which has been encrypted using the password is sent. Once authenticated, a client has access to all resources within the Kerberos domain/realm without any further login (single sign-on, SSO).

The following components are important in a Kerberos infrastructure.

The **Key Distribution Center**, which contains two active components:

- **Authentication service**
  Performs the authentication (e.g., using Microsoft Active Directory) and creates a *ticket granting ticket (TGT)* if the user or system has been authenticated successfully.
- **Ticket granting service**
  Creates tickets which will grant access to other systems. The user or system who/which wants to receive a *service ticket* has to present a TGT in order to get this service ticket.

Short Introduction into ConSol CM with Kerberos in a Windows Domain

The single sign-on feature allows users to authenticate against ConSol CM automatically with their ***Windows*** credentials.

This authentication mechanism ...

- works completely transparent, no user interaction is required (i.e., filling in credentials on a login screen)
- does not interfere with existing authentication mechanisms. If the Kerberos authentication fails, the configured authentication mechanism (LDAP or database authentication, see cmas-core-security, authentication.method) is used.

The single sign-on feature is based on the ***Kerberos V5*** protocol, which is integrated in the ***Windows Active Directory***.

The ConSol CM server works as a **non-Windows Kerberos service** and can be installed on any operating system / application server. The domain controller is the Key Distribution Center and also runs the Active Directory which stores the user authentication information.

The following graphic provides a simplified overview of the steps which are required in order to provide a valid client/server session for a ConSol CM client and the respective ConSol CM server. The explanations in these sections will guide you through the entire configuration process for running ConSol CM with Kerberos authentication.

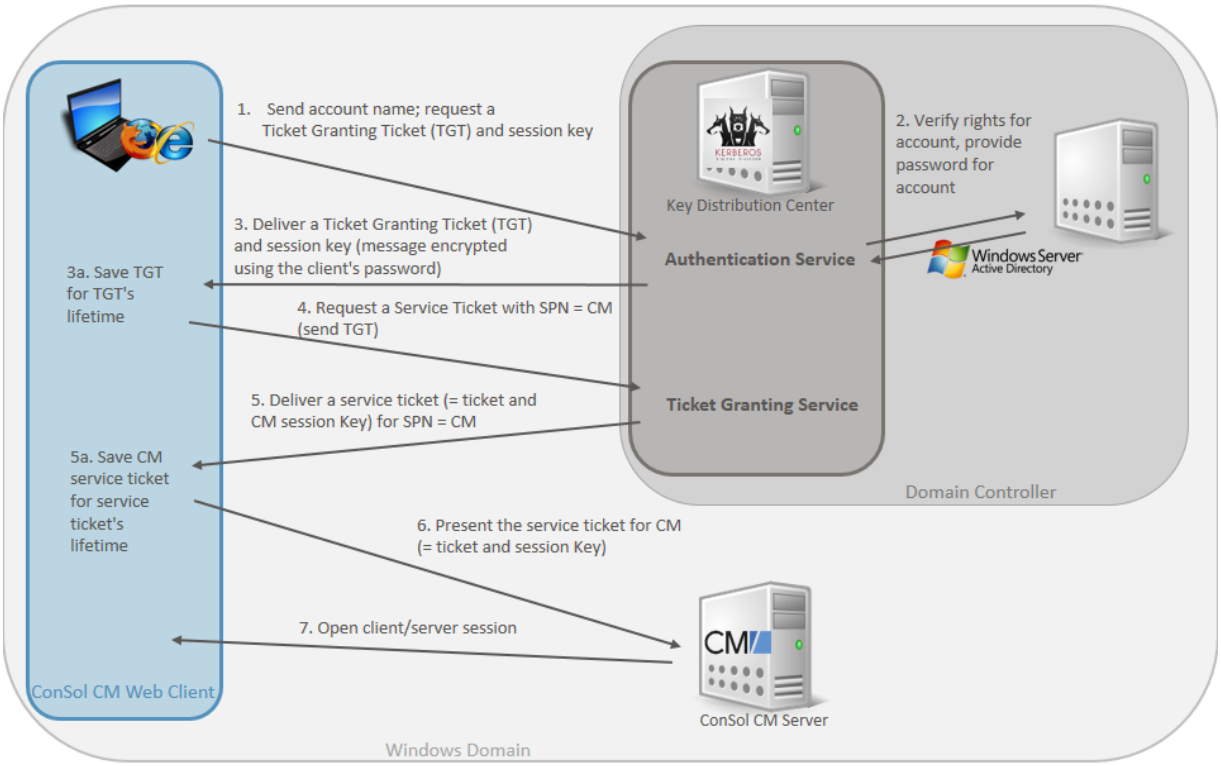


Figure 400: *Kerberos Authentication in a Windows domain for ConSol CM access*

Steps in the Authentication Process

Step 1: Send Account Name, Request a Ticket Granting Ticket (TGT) and Session Key

The user logs in to the computer with a user name and password. The computer sends a message to the authorization server requesting a ticket granting ticket (TGT).

If pre-authentication is enabled, a timestamp is encrypted using the user's password hash as an encryption key. The authentication server decrypts the timestamp with the user's password (stored in the Active Directory). If it reads a valid time, it knows that the request is not a renewal of a previous request.

In the example, the pre-authentication is disabled.

Step 2: Verify Rights for Account, Provide Password for Account

The authorization server verifies the user's access rights in the Active Directory and the Active Directory provides the user's password for step 3.

Step 3: Deliver a Ticket Granting Ticket (TGT) and Session Key (Message Encrypted Using the Client's Password)

The authentication service creates a ticket granting ticket (TGT) and a session key. These are sent in a message to the client and encrypted with a key derived from the user's password.

The computer prompts the user for a password and uses the password to decrypt the incoming message. If the decryption succeeds, the user is able to use the TGT to request a service ticket.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Step 3a: The TGT is saved for the user for the TGT's lifetime and can be used for all service ticket requests during this period of time.

Step 4: Request a Service Ticket for CM

When the user wants access to a service, i.e., when an engineer wants lo log in to ConSol CM, the workstation client application (Web Client) sends a request to the ticket granting service containing the SPN (Service Principal Name) of the requested service, the client name, the realm/domain name, and a timestamp (name and timestamp encrypted with the session key). The user proves his identity by sending an authenticator (TGT) received in step 3.

Step 5: Deliver a Service Ticket for CM (= Ticket and Session Key)

The ticket granting service decrypts the ticket and authenticator, verifies the request, and creates a service ticket for the requested service (SPN = ConSol CM). The service ticket contains the user name. It also contains the realm/domain name and service ticket lifespan. The service ticket is encrypted using the service's (CM's) secret. The ticket granting service sends the service ticket to the user.

Step 5a: The service ticket is saved for the user for the service ticket's lifetime and can be used for all requests to the SPN during this period of time.

Step 6: Present the Service Ticket for CM (= Ticket and Session Key)

The client application (Web Client) now sends a service request to the CM server containing the ticket received in step 5 and an authenticator. The service (CM server) authenticates the request by decrypting the session key. The CM server verifies that the service ticket and authenticator match, and then grants access to the service (ConSol CM application).

Step 7: Open Client/Server Session

The client/server session is in operation.

Encryption in the Process

Usually, Kerberos is run with symmetric encryption (i.e., one key is used to encrypt and decrypt the message). The encryption is based on the user's password. Optionally, public-key cryptography might be used in a Kerberos environment.

Microsoft Windows uses Kerberos as a default authentication method, using RC4 for encryption, together with HMAC (Hash-based Message Authentication Code).

Kerberos Terminology

Kerberos Principal

A *Kerberos principal* is a unique identity to which Kerberos can assign tickets. This might be a person's user account or a system user account. Kerberos principals are composed of a number of components. Each component is separated by a component separator, usually "/". The last component is the realm/domain, separated from the rest of the principal by the realm/domain separator, usually "@". If there is no realm/domain component in the Kerberos principal, then it will be assumed that the principal is in the default realm/domain for the current environment.

Usually, a Kerberos principal is composed of three parts: the primary, the instance, and the realm/-domain. The format of a typical Kerberos V5 principal is `primary/instance@REALM`.

- The primary is the first part of the principal, this is the user name (in case of a person's account) or the name/account of a system user or a host name.

- The instance is an optional string that qualifies the primary. The instance is separated from the primary by a slash (/). In the case of a user, the instance is usually `null`, but a user might also have an additional principal, with an instance called `admin`, which he uses to administrate a database. The principal `cmlxuser@CONSOL.DE` is completely different from the principal `cmlxuser/administrator@CONSOL.DE`, with a separate password and separate permissions. In the case of a host, the instance is the fully qualified hostname, e.g., `cmserver.consol.de`.
- The realm is the Kerberos realm, usually the domain name, in upper-case letters. For example, the machine `cmserver@consol.de` would be in the realm `CONSOL.DE`.

Keytab File

A `keytab` is a file containing pairs of Kerberos principals and the respective encrypted keys (which are derived from the Kerberos password). When a password is changed, you need to rebuild the keytab file. In the ConSol CM context this happens when the password of the system user (in our example *cmwinuser* for Windows and *cmlxuser* for Linux/Unix) has changed, **not** when a CM engineer has changed his password.

Setting Up Kerberos Single Sign-On for ConSol CM

Requirements

For Kerberos-based single sign-on you need:

- Domain controller for the Windows domain
- ConSol CM server
- Windows clients

> ⚠ Since in a Kerberos environment, timestamps are used to calculate and check the time period of validity of the tickets, it is absolutely indispensable that all components work with the same system time! A time server/service based on NTP can be used for example. (Some Windows domain controllers tolerate a maximum time difference of five minutes.)

The Windows domain used in the example is *SSO.CONSOL.DE* and the domain controller is called *winadsrv*.

> ⚠ **ConSol CM running on a Linux/Unix server**
>
> If ConSol CM is running on a Linux or Unix server, it needs to be reachable by the DNS name *cmserver.consol.de*. DNS is important because the communication does not work with the IP address alone.

Basic Principle and Required Steps

The user/account names and passwords of the CM engineers are managed in the Active Directory and the domain controller acts as Kerberos Key Distribution Center (KDC).

In the following chapters we distinguish between a CM server running on a Windows or running on a Linux/Unix system.

On Windows:

The computer running the CM server has to be a member of the Windows domain.

- Register the CM server as Windows domain member. Since it is not the system user (*cmwinuser)* who will work with CM but the CM engineers, the option *Trust this computer for delegation to any service* has to be set in the KDC. In this way, the system user can "impersonate" other users (the CM engineers).

  See Registering the ConSol CM Server and User.

- Build a keytab file for the system user *cmwinuser* for the CM machine. This registers CM as Kerberos service, so that Kerberos tickets can be processed on the ConSol CM server side.

  See Generating the keytab File.

On Linux/Unix:

- Register the CM server as Service Principal Name (SPN) on the Windows domain server.
- Create the domain account *cmlxuser* to which the SPN is assigned. Since it is not the domain user (*cmlxuser)* who will work with CM but the CM engineers, the option *Trust this computer for delegation to any service* has to be set in the KDC. In this way, the domain account can "impersonate" other users (the CM engineers).

  See Registering the ConSol CM Server and User.

- Build a keytab file for the SPN of the CM machine. This registers CM as Kerberos service, so that Kerberos tickets can be processed on the ConSol CM server side.

  See Generating the keytab File.

On Windows and Linux/Unix:

On the CM machine, it is required to configure the application server in a way that Kerberos can be used. CM needs to know where to find the keytab file to use Kerberos encryption and communication.

- Define the respective module in the application server.
- Store the keytab file on the CM server.
- Edit the `cm6-kerberos.properties` file.

See Configuring Kerberos.

For users who want to use SSO with Kerberos and CM, a CM engineer name must be assigned to a Kerberos principal name:

- Define Kerberos principal names for the CM engineers in the *Engineers* section of the Admin Tool.

See Configuring the CM Engineers for Kerberos.

Operations to Be Performed on the Domain Controller

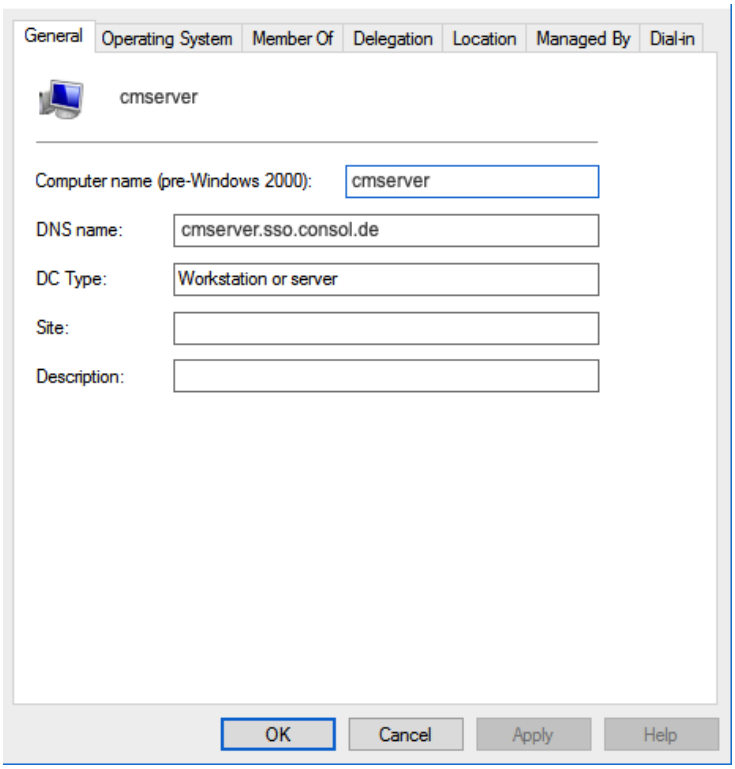
The first step is to configure the domain controller so that it knows the ConSol CM server, i.e., to integrate the CM server into the Windows domain.

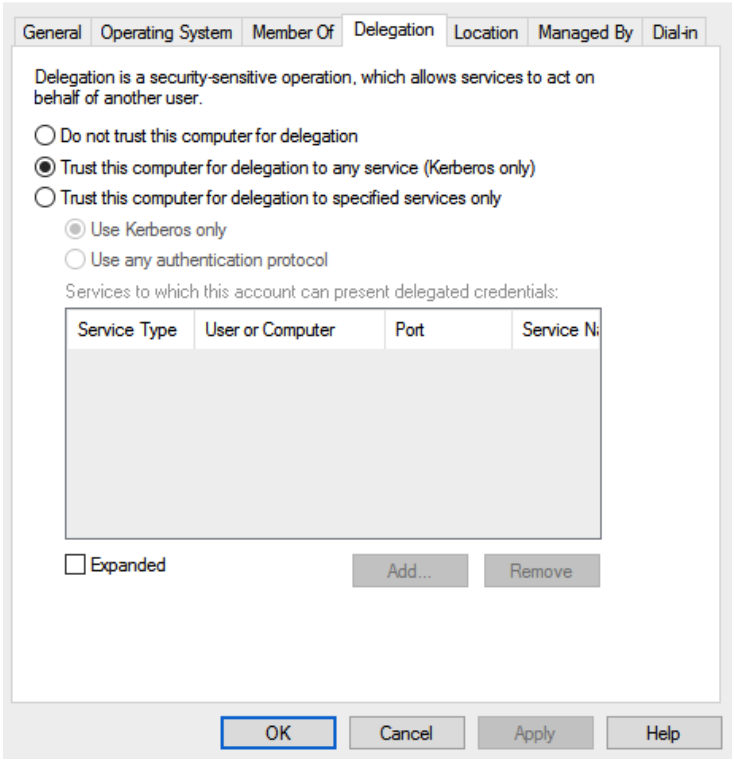
In our example the domain controller is called `winadsrv`, the domain is `SSO.CONSOL.DE`.

Registering the ConSol CM Server and User

On Windows:

The ConSol CM server machine needs to be registered in the Active Directory of the domain controller. This is done in the *Computers* administration of the Active Directory.





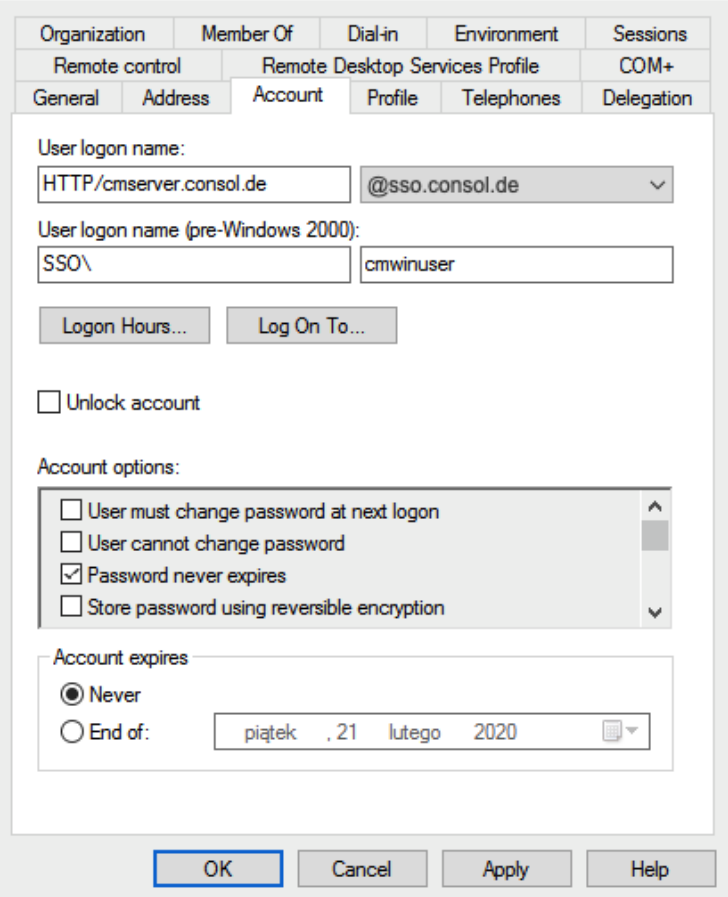
⚠ The radio button *Trust this computer for delegation to any service (Kerberos only)* must be enabled.

The user under which the ConSol CM server process will run is created and registered in the Active Directory (which acts as Key Distribution Center), in our example the user is called *cmwinuser*. This is done in the *Users* administration of the Active Directory.

The following account options must be enabled:

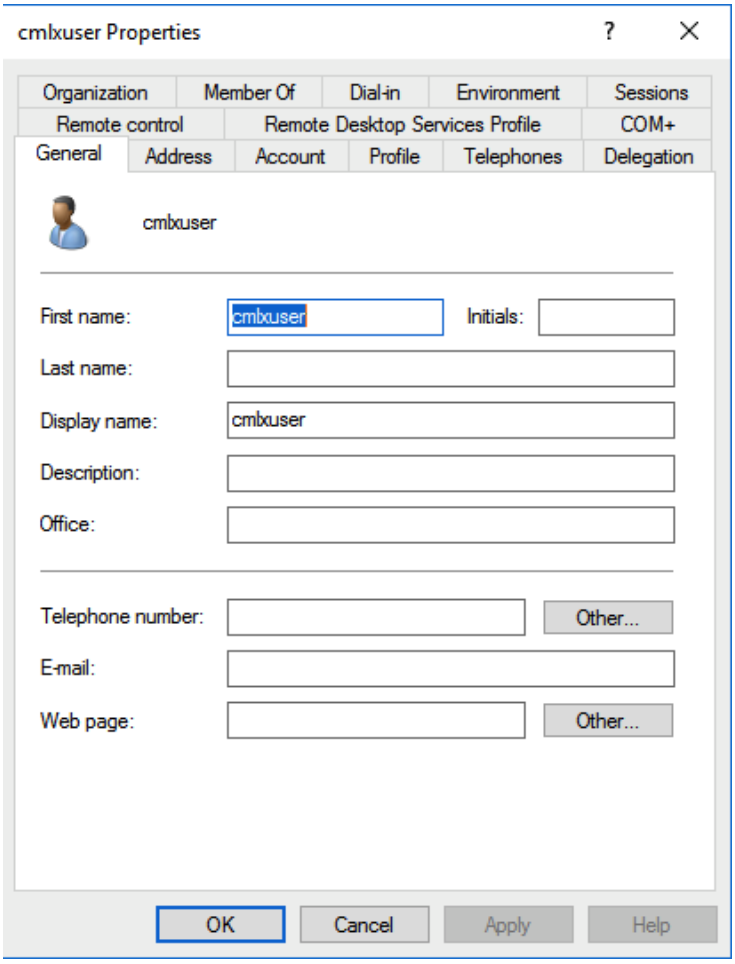
- Account is trusted for delegation
- No Kerberos pre-authentication needed

> Make sure that the CM application server process runs under the indicated user on the CM machine.

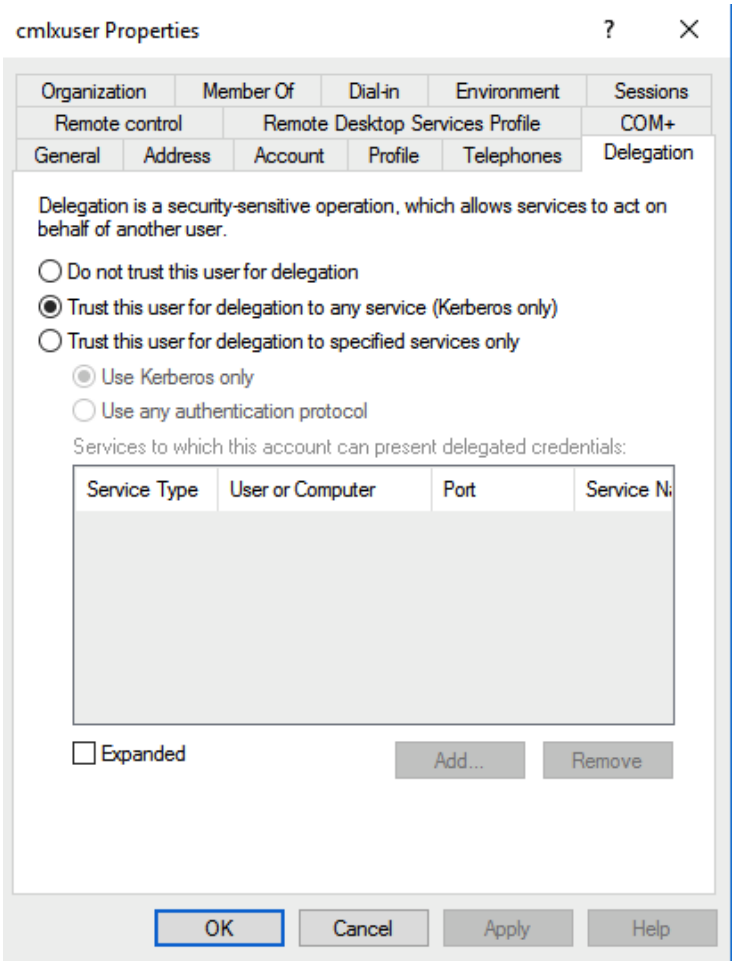


On Linux:

First it is necessary to create a Windows domain account. ConSol CM is assigned to this account in the next step. We use the name *cmlxuser* for this account.



⚠ The radio button *Trust this user for delegation to any service (Kerberos only)* must be enabled in the *Delegation* tab.



Use the following command to assign the ConSol CM server to the created account.

```
C:\Users\Administrator> setspn -S HTTP/cmserver.consol.de cmlxuser
Checking domain DC=sso,DC=consol,DC=de
Registering ServicePrincipalNames for CN=cmlxuser,CN=Users,DC=sso,DC=consol,DC=de
     HTTP/cmserver.consol.de
Updated object
```

You can show the assignment with the following command:

```
C:\Users\Administrator.SSO> setspn -l cmlxuser
Registered ServicePrincipalNames for CN=cmlxuser,CN=Users,DC=sso,DC=consol,DC=de:
HTTP/cmserver.consol.de
```

Generating the keytab File

On the domain controller, a Kerberos `keytab` file has to be generated. This file will be required later on the ConSol CM server machine. The `keytab` file contains the shared secret key of the service (SPN, Service Principal Name).

Use the following ktpass command:

```
C:\Users\Administrator> ktpass /princ HTTP/cmserver.consol.de@SSO.CONSOL.DE /pass
 consol123 /mapuser cmlxuser /pType KRB5_NT_PRINCIPAL /out c:\cmserver.http.keytab
 /crypto rc4-hmac-nt
```

⚠ If `ktpass` is not available, the ***Windows Server 2003 Support Tools*** must be installed, available here.

Explanation of the ktpass command

```
/out cmserver.http.keytab
```

=> Uses `cmserver.http.keytab` as output file

```
/ptype KRB5_NT_PRINCIPAL
```

=> Specifies the principal type.

- **KRB5_NT_PRINCIPAL** is the general principal type (recommended).
- **KRB5_NT_SRV_INST** is the user service instance.
- **KRB5_NT_SRV_HST** is the host service instance.

```
/princ HTTP/cmserver.consol.de@SSO.CONSOL.DE
```

=> Specifies the Kerberos principal name in the form http/computer.<domain>@<realm>.

```
/pass consol.123
```

=> Sets the password for the *princ* user indicated with `/princ`.

```
/mapuser cmlxuser
```

=> Maps the name of the Kerberos principal (`/princ` parameter) to the specified domain account.

```
/crypto rc4-hmac-nt
```

=> Specifies the keys that are generated in the keytab file (optional):

- **DES-CBC-CRC** is used for compatibility.
- **DES-CBC-MD5** adheres more closely to the MIT implementation and is used for compatibility.
- **RC4-HMAC-NT** employs 128-bit encryption.
- **AES256-SHA1** employs AES256-CTS-HMAC-SHA1-96 encryption.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **AES128-SHA1** employs AES128-CTS-HMAC-SHA1-96 encryption.
- **All** states that all supported cryptographic types can be used.

Operations to Be Performed on the ConSol CM Server

Run the application server under the system user which is registered in the Windows domain (in our example *cmwinuser* for Windows and *cmlxuser* for Linux/Unix).

Install ConSol CM as usual, then enable and configure Kerberos as described in the next steps.

Enabling Kerberos in ConSol CM

If you do an initial setup, you can choose whether Kerberos should be enabled. Please note that this is only a hint and additional configuration is needed (see next steps).

If your ConSol CM is already configured without Kerberos enabled, you can enable it in the Admin Tool by setting the system property cmas-core-security, kerberos.v5.enabled to “true”. A server restart is required to activate the new setting.

Configuring Kerberos

A ConSol CM server reads configuration parameters from the file `cm6-kerberos.properties` from the classpath. (Each cluster node may need separate configurations, so each node will read the `cm6-kerberos.properties` file from the classpath.)

Example configuration for JBoss 7:

1. Create a directory for the Kerberbos configuration at the following path:

```
mkdir -p ${jboss}/modules/system/layers/base/com/consol/cmas/main/
```

2. In this directory, create the file `module.xml` with the following content:

```
<module xmlns="urn:jboss:module:1.1" name="com.consol.cmas">
   <resources>
      <resource-root path="."/>
   </resources>
</module>
```

3. Save the `cm6-kerberos.properties` file in this directory.
4. Edit the configuration file, e.g., `${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 example 85: *JBoss 7*

In case you have a cluster of more than one ConSol CM servers in operation, each server has to have its own properties file.

The `cm6-kerberos.properties` file should contain:

- Reference to a Kerberos configuration file (e.g., `krb5.ini` or `krb5.conf`)
- One or more service principals, i.e., reference to the `keytab` file

```
# path to kerberos configuration
kerberos.config.location=C:\\conf\\krb5.ini

# one or more service principals (principal = path to keytab file)
HTTP/cmserver.consol.de@SSO.CONSOL.DE=C:\\conf\\cmserver.http.keytab
```

Code example 86: *cm6-kerberos.properties (Windows)*

```
# path to kerberos configuration
kerberos.config.location=/home/hotline/cmas/kerberos/krb5.ini

# one or more service principals (principal = path to keytab file)
HTTP/cmserver.consol.de@SSO.CONSOL.DE=/home/hotline/cmas/kerberos/cmserver.http.key
 tab
```

Code example 87: *cm6-kerberos.properties (Linux)*

```
[libdefaults]
   default_realm = SSO.CONSOL.DE
   default_tkt_enctypes = rc4-hmac
   default_tgs_enctypes = rc4-hmac

[realms]
   SSO.CONSOL.DE = {
      kdc = WINADSRV:88
      admin_server = WINADSRV:749
      default_domain = sso.consol.de
   }

[domain_realm]
      .SSO.CONSOL.DE = SSO.CONSOL.DE
      SSO.CONSOL.DE = SSO.CONSOL.DE
      .sso.consol.de = SSO.CONSOL.DE
      sso.consol.de = SSO.CONSOL.DE
```

Code example 88: *krb5.ini*

keytab File

Copy the `keytab` file generated on the Windows domain controller to the location you specified in `cm6-kerberos.properties`.

⚠ You have to restart the ConSol CM server process for this change to take effect!

Configuring the CM Engineers for Kerberos

If the system property cmas-core-security, kerberos.v5.enabled is set to “true”, the field *Kerberos Principle Name* is available in the engineer details in the *Engineers* section of the Admin Tool.

The *Kerberos Principal Name* is a user name along with the respective Kerberos realm/domain name, for example “SusanServiceDesk@consol.de”. This allows to authenticate users with the login “SusanServiceDesk” that belong to different Kerberos realms (“SusanServiceDesk@consol.de” and “SusanServiceDesk@consol.pl” are in fact different users). If the field *Kerberos Principal Name* is empty for an engineer, the user name (field *Login*) is used. If Kerberos authentication fails, the regular authentication mechanism (DATABASE or LDAP) is used. In that case, single sign-on is not possible and the engineer sees the login dialog.

Mapping Kerberos User Name to Engineer Name

Using Kerberos-based single sign-on, the Kerberos principal (i.e., the user’s Windows login) has to be mapped to a ConSol CM engineer name.

By default, this mapping is done using one of the following two ways:

- **Explicit mapping**
  Take the principal name and try to find a ConSol CM engineer who has this principal stored as *Kerberos Principal Name*. If such an engineer is found, this engineer is used.
- **Mapping via regular expression**
  The regular expression defined in the system property cmas-core-security, kerberos.v5.username.regex is taken and applied to the principal. The result is taken and a ConSol CM engineer with this login is searched:
  - The first matching regular expression group (in brackets) is used as engineer login name, e.g., the default property value *(.*)@.** converts *Huber@SSO.CONSOL.DE* to *Huber*.

If further customization is needed please refer to the *UsernameAdapter interface javadoc*.

Operations to Be Performed on the Client Machine

Micrsosoft Internet Explorer and Microsoft Edge

Internet Explorer and ***Edge*** need to be configured so that an automatic login is enabled. This is done in *Internet options -> Security* in the Windows Control Panel. By default, the security setting *medium-low* is used for the *Local Intranet* zone. This setting allows automatic login.

You can check it by clicking the *Custom level* button in the settings of the *Local Intranet* zone. The following settings for login behavior are available in the *User Authentication* section.



Figure 401: *Internet Explorer login configuration*

Each setting and the resulting behavior:

- **Anonymous logon**
  No single sign-on is possible, the user (CM engineer) will get the ConSol CM login dialog.
- **Automatic logon only in Intranet zone**
  Single sign-on is performed automatically but only if the site is part of the *Local intranet* zone.
- **Automatic logon with current user name and password**
  Single sign-on is performed automatically with the current user credentials.
- **Prompt for user name and password**
  OS displays a login dialog, user can enter OS login information, which is then used in Kerberos authentication.

Firefox

With the default settings, ***Firefox*** does not support Kerberos single sign-on. To enable single sign-on, you have to add the URI of the ConSol CM Web Client in the Firefox configuration.

To do this:

- Open `about:config.`
- Add the web server to the property `network.automatic-ntlm-auth.trusted-uris` (for example `http://mycm6server` if that is the URI).
  If this does not work, perform the following step:
- Add the web server to the property `network.negotiate-auth.trusted-uris` (for example `http://mycm6 server` if that is the URI).

You can set this property also in the file system. Open the file

`C:\Users\[USER]\AppData\Roaming\Mozilla\Firefox\Profiles\[XYZ].default\prefs.js`

and add/replace the following line:

```
user_pref("network.automatic-ntlm-auth.trusted-uris", "http://mycm6server");
```

or

```
user_pref("network.negotiate-auth.trusted-uris", "http://mycm6server");
```

⚠ You have to restart Firefox after this change.

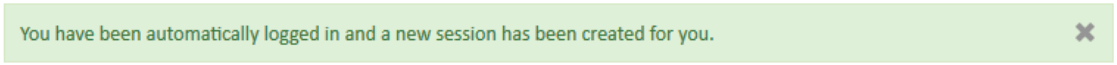
Using Kerberos Single Sign-On with ConSol CM

Single Sign-On from the User's Point of View

An engineer using single sign-on to log into ConSol CM will notice the following:

- No ConSol CM login screen is displayed.
- Instead, there may be (for a short time) an intermediate text screen (which is used to gather some client data via JavaScript) which immediately forwards the user to the main screen of the

ConSol CM Web Client, where a message is displayed:



It is still possible to log in as another ConSol CM user by clicking the logout button, which will lead you to the login page, or by explicitly using the `.../cm-client/login` URL.

Multi Domains Single Sign-On

For each domain which you will enable single sign-on for, create a new domain/user and Kerberos principal and put all of them into the `cm6-kerberos.properties` file:

```
# 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/cmserver.consol.de@SSO.CONSOL.DE=/home/hotline/cmas/kerberos/cmserver.http.key
 tab
HTTP/cmserver2.consol.com@SSO.CONSOL.COM=/home/hotline/cmas/kerberos/cmserver2.http
 .keytab
```

Starting and Stopping Kerberos Authentication

Kerberos authentication can be started/stopped in the Admin Tool navigation group *Services* -> navigation item *CM Services -> Kerberos v5 authentication provider*, see section CM Services.

F.15.2 Authentication Methods for Customers in CM/Track

F.15.2.1 Available Authentication Methods

There are three possible authentication modes:

- Against the ConSol CM database
  This is called *DATABASE mode*, see Database Authentication for Customers
- Against an LDAP server
  This is called *LDAP mode*, see LDAP Authentication for Customers in CM/Track
- Against an LDAP server and the ConSol CM database
  The order can be configured. This is called *Mixed mode*, see Mixed Authentication Mode

F.15.2.2 Defining the Authentication Method

The authentication mode is specified by the system property cmas-core-security, contact.authentication.method. A change of this property does not require a server restart, and is propagated to all cluster nodes.

The possible values (see also section System Properties) and their respective system behaviors are:

- **DATABASE**
  Attempt a database login if the customer has a database password. I.e., the login and password are stored in the ConSol CM database and are thus managed by the ConSol CM engineers, or indirectly by the customers themselves when they reset their password. The customer can reset his own password, see section Password Reset Template for Customers Using CM/Track.
- **LDAP**
  Try authentication using the available LDAP server(s), if an LDAP ID is provided. I.e., the password is stored in the LDAP directory and cannot be changed via ConSol CM, neither by the customer nor by an engineer.
- **LDAP,DATABASE**
  First attempt authentication using the available LDAP server(s), if an LDAP ID is provided. On failure, try a database login if the customer has a database password.
- **DATABASE,LDAP**
  First attempt a database login if the customer has a database password. On failure try authentication using the available LDAP server(s) if an LDAP ID is provided.

The values are case insensitive, and commas and whitespace are ignored.

F.15.2.3 Database Authentication for Customers

Database authentication is activated by setting the system property cmas-core-security, contact.authentication.methodto “DATABASE” (default value).

There are two steps which you need to perform to set up database authentication for customers using CM/Track:

- Create customer fields for the user name (login) and password in the Admin Tool (see Defining the Customer Fields for CM/Track Login and Password)
- Enter the user name and password for the actual customers in the Web Client (see Granting Access to CM/Track for Customers)

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

When database authentication is used, you can allow your customers to change their own passwords, see Configuring CM/Track for Password Reset by Customers.

Defining the Customer Fields for CM/Track Login and Password

The fields for login and password for a customer are regular customer fields at the contact level. Please see section Setting Up the Customer Data Model for an introduction to customer field management and GUI configuration for customer data.

Edit the customer data model in the Admin Tool (navigation group *Customers*, navigation item *Data Models*). If there are two levels you need to edit the fields of the contact level. Create the following two fields:

- Create a field for the login with the annotation `username` = “true”.

  > ⚠ Assigning the annotation `username` to a customer field is only possible, if there is no previous assignment of this annotation. Otherwise it will be prohibited. When assigning it, a warning dialog must be confirmed before it is executed, since it can be a longer running operation. Unassigning the annotation must be confirmed as well, because it cannot be undone: Unassignment deletes the user name values unrecoverably from internal storage.

- Create a field for the password with the annotation `password` = “true”. The annotation `text-type` = “password” guarantees that only stars/dots are displayed in the Web Client, not the clear text password.

  > ⚠ The annotation `password` requires confirmation when assigned.
  >
  > In case of an update from CM versions lower than 6.11 to 6.11 and up: When this annotation is set, the system reads the plain text passwords from the original field values, encrypts them and saves the encrypted values to the internal storage. Then the original field values are deleted and thus the plain text value cannot be recovered anymore.
  >
  > When trying to unassign the `password` annotation the operation must be confirmed as well, since the encrypted passwords are deleted from the internal storage. After the annotation unassignment the password information is completely lost and cannot be recovered at all.

When a scenario from a CM version lower than 6.11 is imported into a system with CM 6.11 (or higher), a transformation of user names and passwords is performed automatically. This is described in detail in section Transformation of User Name and Password Fields During Import into CM 6.11 .

Granting Access to CM/Track for Customers

The engineer working with the Web Client can then assign a user name, initial password, and a CM/Track user profile to every customer who should have access to the portal CM/Track. The user name has to be unique. This is checked by the system. You cannot enter a name a second time if this has already been assigned to another customer. The password is stored as encrypted string in the CM database. This means that an engineer can set a new password, e.g., when a customer calls and asks for this, but it is never possible to read the password from the system.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

> You, as an administrator, can define if the CM/Track user names should be case sensitive. Use the CM system property cmas-core-security, policy.track.username.case.sensitive. This is a boolean variable. When it is set to "true", the CM/Track user names are case sensitive. Please make sure that the database collation which is in use supports case sensitive strings!

The following example shows the customer data of an example contact in the ConSol CM Web Client. You reach this screen by opening a contact data set in edit mode.

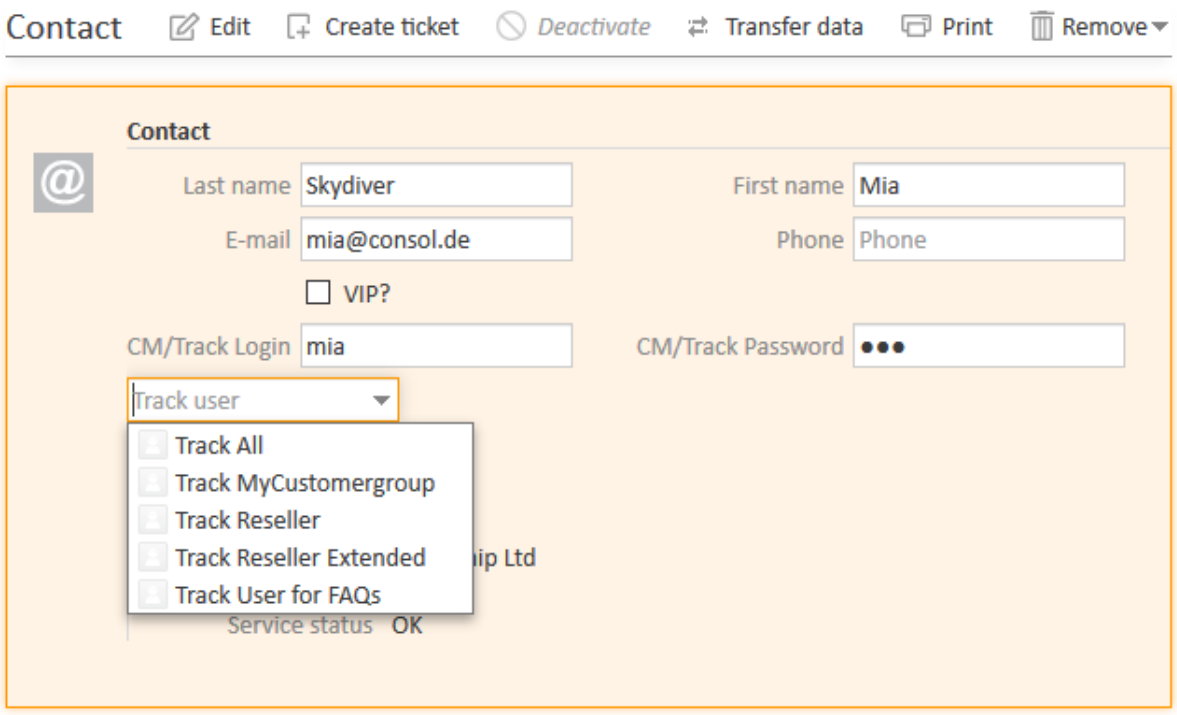


Figure 402: *ConSol CM Web Client - Contact page: CM/Track user data*

Configuring CM/Track for Password Reset by Customers

CM/Track can be configured to offer a hyperlink for customers where a customer can reset his password. This is based on the template `track-password-reset-template`. Please refer to section Password Reset Template for Customers Using CM/Track for a detailed explanation. The password reset in CM/Track is only possible when the DATABASE mode is used. It is not possible when LDAP authentication is in operation. See section Authentication Methods for Customers in CM/Track for the portal for an explanation of all possible authentication modes.

Please note that the `From`address of the email which is sent to a customer who has requested a new password can be set using the CM system property cmas-core-security, password.reset.mail.from.

F.15.2.4 LDAP Authentication for Customers in CM/Track

Introduction to ConSol CM LDAP Authentication

ConSol CM offers LDAP authentication for CM/Track as a standard feature, i.e., instead of managing the passwords for the ConSol CM customers in the ConSol CM database, they can be retrieved from an LDAP server (like e.g., a ***Microsoft Active Directory*** server).

When customers want to log in to CM/Track, they enter their user name and password and press *Enter*. Behind the scenes, the ConSol CM server sends a request with the customer's user name and password and asks the LDAP server whether those credentials are correct.

If the credentials are correct, the approval is sent back to the ConSol CM server and the customer is logged into CM/Track.

> Please keep in mind that the LDAP connection is only used to authenticate the customer (confirm the identity). The authorization (i.e., the assignment of access permissions in the system) is done via the assignment of a CM/Track user profile in the Web Client. The CM/Track user profiles are managed in the engineer and role administration in the Admin Tool.

Please see also the following picture for an explanation of the CM/Track authentication process using LDAP.

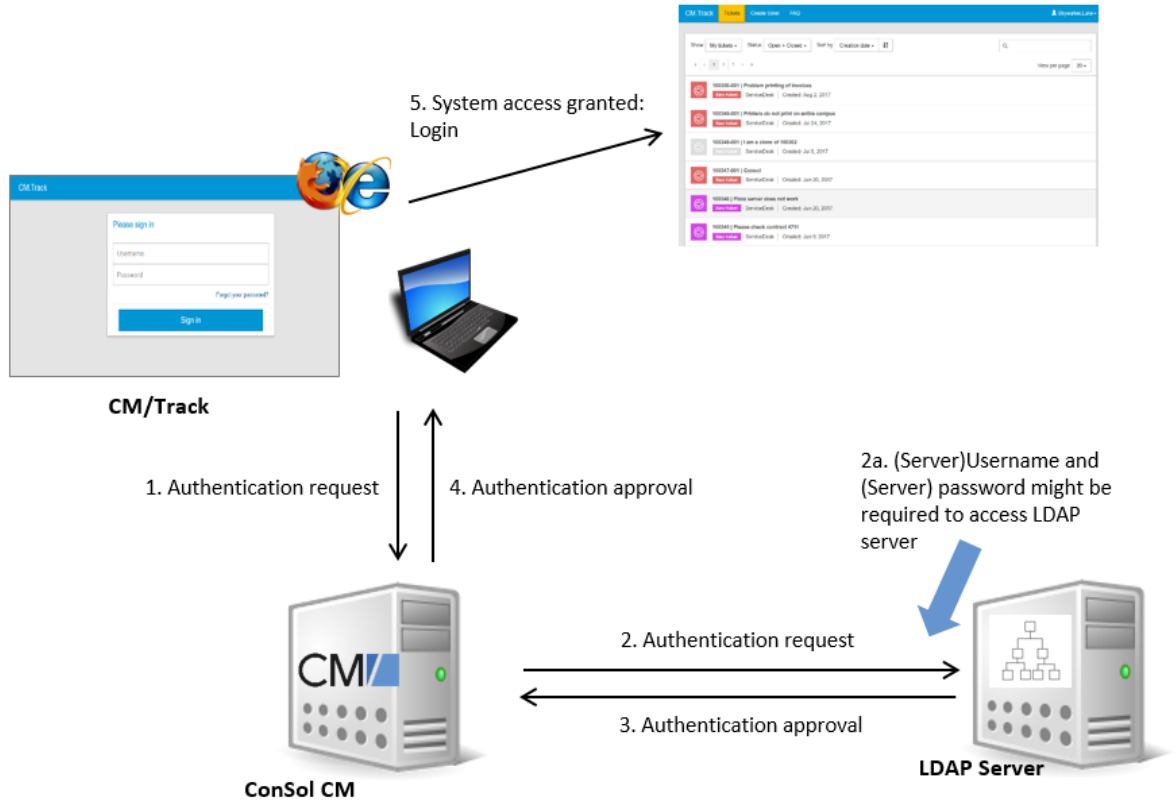


Figure 403: *ConSol CM - LDAP authentication process for CM/Track*

Configuring LDAP Authentication

LDAP authentication is activated by setting the system property cmas-core-security, contact.authentication.method to “LDAP”.

Then you have to set the required values in the system properties (navigation group *System*, navigation item *System Properties*, see System Properties).

The LDAP servers can be defined using the following system properties from the module `cmas-core-security`.

{name} is a string that you can choose to distinguish LDAP servers. It must always be set, even if only one LDAP server is configured. You should use a simple string for the {name}, not containing any keywords, like *internal* or *external*, and which does not contain special characters.

- cmas-core-security, contact.authentication.method
  LDAP
- ldap.contact.{name}.providerurl
  The property value is the address of the LDAP server in the form *ldap[s]://host:port*.
- ldap.contact.{name}.userdn
  The value is the user DN used to look up the contact DN by the LDAP ID. An anonymous account is used if the value is not set.
- ldap.contact.{name}.password
  The property contains the password to look up the contact DN by the LDAP ID. An anonymous account is used if the value is not set.
- ldap.contact.{name}.basedn
  This represents the base path to search for the contact DN by the LDAP ID, e.g., “ou=accounts,dc=mycompany,dc=de”.
- ldap.contact.{name}.searchattr
  The property value stands for the attribute to search for the contact DN by the LDAP ID, e.g., “uid”.

Initially, these system properties might not be present in your CM system. Just add them manually. Changes to any of the above system properties do not require a server restart and are propagated to all cluster nodes. The use of the placeholder {name} allows configurations to define several different LDAP servers.

- ldap.initialcontextfactory
  This is a predefined global property. If it is not set, “com.sun.jndi.ldap.LdapCtxFactory” is used as its value.

Authentication attempts against LDAP servers are made until first success, where the server order is determined by their {name} values (ascending alphabetical order of the values).

Mixed Authentication Mode

Set the system property cmas-core-security, contact.authentication.method depending on the desired order of authentication instances:

- **LDAP,DATABASE:**
  First attempt authentication using the available LDAP server(s), if an LDAP ID is provided. On failure, try a database login if the customer has a database password.
- **DATABASE,LDAP:**
  First attempt a database login if the customer has a database password. On failure try authentication using the available LDAP server(s) if an LDAP ID is provided.

The CM system will first contact the instance which is mentioned first, than the second one. For example, when the contact authentication method is set to “LDAP,DATABASE” and the customer (contact) uses the password which is only valid in the database, the login will succeed.

In `server.log` the following message will be displayed:

```
LDAP login failed: [LDAP: error code 49 - Invalid Credentials]; nested exception is
 javax.naming.AuthenticationException: [LDAP: error code 49 - Invalid Credentials]
```

Logging of LDAP Login Attempts in CM/Track

All LDAP errors encountered are logged without a stack trace using loggers with the following prefix:

- `com.consol.cmas.core.security.contact`

The stack trace of LDAP errors is not logged because failed login attempts on the first LDAP server would clutter logs if a following login on the second LDAP server succeeded.

Using LDAPS (LDAP over SSL)

Introduction

Per default, when an LDAP client accesses an LDAP server, the information is transferred in clear text. In case you want the user name and password to be transferred to the LDAP server in encrypted form, you have to set up the LDAP authentication using LDAPS.

Preparations

You have to configure the CM server machine (Java) in a way that can use certificates. One way to do this for a Linux environment is described in the following section.

1. Retrieve the certificate:

   ```
   openssl s_client -connect dc2.mydomain.com:ldaps
   ```

2. The answer will contain a section which starts with `"---BEGIN CERTIFICATE "` and ends with `"END CERTIFICATE ---"`. Copy this section to a file, e.g., `/tmp/certificate2_dc2_mydomain_com.txt`
3. Import the certificate to the truststore of your machine, e.g., `/home/mydirectory/mytruststore`

   ```
   $JAVA_HOME/bin/keytool -import -alias <arbitrary> -trustcacerts -keystore
    /home/mydirectory/mytruststore -file/tmp/certificate2_dc2_mydomain_com.txt
   ```

   You have to enter (set) a password.
4. Enter the truststore in the ConSol CM config file in JAVA_OPTS:

   ```
   -Djavax.net.ssl.trustStore=/home/mydirectory/mytruststore -
    Djavax.net.ssl.trustStorePassword=<see above>
   ```

LDAPS Configuration in the ConSol CM Admin Tool (System Properties)

Configure the ConSol CM server as shown in the following example:

- 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

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- 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

Depending on the LDAP server configuration, use one of the following values for the 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

Setting Up Customer Accounts for LDAP

There are two steps which you need to perform to set up LDAP authentication for customers using CM/Track:

- Set the required annotation for the customer field which should hold the LDAP ID in the Admin Tool.
- Enter the LDAP IDs for the actual customers in the Web Client.

When LDAP mode is used, the customer field which is used for the CM/Track user name (login) has to have two annotations:

- `username` = true
- `ldapid` = true

F.16 ConSol CM External Interfaces

ConSol CM is not a stand-alone system but can be integrated into your company's IT infrastructure. One possibility of integrating CM is using one of the external interfaces:

- The **REST API**
  - This is used for the implementation of CM/Track, see section CM/Track: The Customer Portal.
  - If you want to develop your own application which uses the REST API, please ask your ConSol CM consultant for a REST API documentation.
- The **Webhooks** interface
  - This offers the possibility to easily integrate CM with such applications as chat tools, shopping systems, or social media platforms.
  - See section Webhooks for details.
- **The ETL interface**
  - This provides the interface for ETL (extract- transform - load) tools. Those tools might be used to import large numbers of objects, e.g., customers, into the ConSol CM database. For a detailed explanation, please refer to the *ConSol CM ETL Manual*.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.16.1 Webhooks

F.16.1.1 Introduction to Webhooks in ConSol CM

Webhooks allow easy integration of ConSol CM with external applications, as shopping applications, social media platforms, chat tools, or customer-built applications. The applications can send data to ConSol CM using HTTP POST requests. ConSol CM processes the received data and returns a response to the requesting application.

Concepts, terms and definitions

Concept	Other terms	Definition
webhook		Mechanism to allow communication between ConSol CM and external applications
security provider		Mechanism to check if the data received by the webhook comes from an authorized application
content type		Format of the data sent using HTTP POST requests

Purpose and usage

Webhooks enable ConSol CM to communicate with external applications using HTTP POST requests. The ConSol CM webhook module provides services for these integrations. You can integrate several applications at the same time. In addition, webhooks are used for web forms (see CM/Forms).

The basic steps of the information transfer using webhooks are:

- The external application sends a HTTP request to the webhook service in ConSol CM.
- The request is processed by the webhook. The functionalities of the service are implemented in a script.
- The response is sent back to the requesting application which processes the response.

The following figure illustrates the interaction between the external application and ConSol CM:

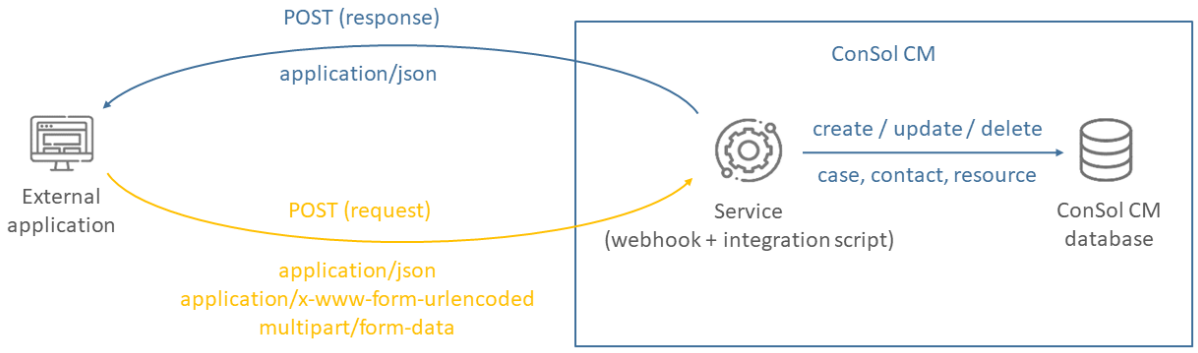


Figure 404: *Webhooks in ConSol CM*

Usage examples:

- Update data in ConSol CM based on master data managed in an ERP system.
- Integrate a social media platform to import customer data into ConSol CM.

- Use web forms for creating cases and adding content.

F.16.1.2 Available settings for webhooks

Supported content types

The HTTP POST requests which are sent by the requesting application can have the following content types:

- application/json
- application/x-www-form-urlencoded
- multipart/form-data

The response which is returned to the requesting application always has the content type `application/json`. It can contain custom-defined objects of the following types:

- httpStatusCode
- httpHeaders
- body

Security providers

The following settings exist for webhooks:

- **Enabled**: Select this checkbox to enable the webhook.
- **IP-based filtering**
  1. Select the checkbox *Enabled* to enable IP-based filtering as a security provider.
  2. Enter the IP addresses in CIDR notation in the field below, see Valid range of IP addresses.
- **Secret tokens**
  1. Select the checkbox *Enabled* to enable secret tokens as a security provider.
  2. Select the checkbox *HMAC* to use a secure hash for message authentication. Only supported for the content types `application/json` and `application/x-www-form-urlencoded`.
  3. Enter the name of the HTML header for the shared secret. The default value is “SECRET_TOKEN”.
  4. Enter the secret tokens in the field below. You can enter one token per line, see Secret tokens.

If both security providers are configured, they are checked in the following order:

- First the IP address is checked. If it is in the permitted range, the request can proceed.
- If the IP address is not in the required range, the secret token is checked subsequently. If it is correct, the request can proceed as well.
- If neither of the two criteria is met, the request is rejected with an “authorization failed” message.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

> ⚠ There is no other mandatory access validation beyond the security providers, specifically no login-based authentication, as for users, which imposes additional access restrictions. The interface has full system access, so securing it additionally with a proxy is mandatory for real-life usage.

> ⚠ It is highly recommended to use HTTPS as a communication protocol for transferring data using a ConSol CM webhook.

Valid range of IP addresses

The IP-based filter uses a list of IP address ranges defined in the CIDR notation (Classless Inter-Domain Routing). A valid entry in this notation is an IP address followed by a slash separating it from a number of bits defining the subnet mask. The example “10.20.30.40/24” stands for the IP address “10.20.30.40” and its subnet mask “255.255.255.0”. The address list for service configuration allows multiple IP ranges in this notation with each range entry on a separate line.

Secret tokens

The checkbox *HMAC* activates the validation of the request by a mandatory hash based on the token. The HMAC (Hash-based Message Authentication Code) value is derived from the unencrypted payload and the secret token. This code is transferred in the header of the request and decrypted by the server using the secret token. The only hashing algorithm supported by ConSol CM in this case is SHA-1. MD5 is not supported.

A corresponding hashing of the response is currently not supported.

The input field *Header* allows to define the HTTP header field name which shall be used to transfer the shared secret token. The value defaults to “SECRET_TOKEN”, but it is recommended to change this value. A non-standard name can mean a security improvement.

The script can access the used token in the headers map of the HTTP request, which contains a field named as defined in the webhook configuration under *Headers*.

> ⚠ The service behavior is completely based on the implementation of the corresponding script, i.e., the security based on the provided tokens has to be implemented and is not system-immanent!

F.16.1.3 Creating a webhook

The following steps are necessary to create a webhook:

1. Write a script of the type *Integration* to implement the logic of the webhook. This script has to process the data received from the external application and send a response to the application. See Writing the integration script.
2. Define a security provider which checks the requests. This can be a range of valid IP addresses or a secret token, see Configuring the webhook service and Security providers.

The service URL which receives the POST requests is:

```
https://<CM server address>/intg/<script name>/service
```

Writing the integration script

You need to create a script of the type *Integration* to process the data received from the external application and send a response to the application. The name of this script (without the extension `.groovy`) is the name of the service, which is part of the URL to access the interface.

Example:

- **Script name**: myWebHook.groovy
- **Service URL**: https://<CM server address>/intg/myWebHook/service

The script is executed in an administrative context for processing a request, so that it has full system access.

The integration script consists of two parts:

- Process the request, see Processing the received data.
- Generate the response, see Creating the response

Processing the received data

The first step for processing the received data is determining the content type of the payload. Either you already know which content type is used by the requesting application or you check the content type within the script:

- **application/json**
  The payload is a `String` object in JSON format.
- **application/x-www-form-urlencoded**
  The payload is a `MultiValueMap` object.
- **multipart/form-data**
  The payload is a `MultipartData` object.

Once the content type has been determined, you can process the data to create or update objects in ConSol CM.

Creating the response

Scripts of type *Integration* must return an object of the class `IntgServiceResponse`, which can include the following information:

- **httpStatusCode**
  Optional. Defines the HTTP status code. If not provided, the following supported default status codes are used:
  - **200 (OK)**
  - **401 (Unauthorized)**
    One or more security provider is enabled, but the data provided by the client did not match the expected data (e.g., wrong token or IP address)
  - **404 (Not Found)**
    A script for a webhook with the given name could not be found / webhook is disabled / no security provider is enabled for the webhook
  - **405 (Method Not Allowed)**
    Any HTTP method other than POST
  - **500 (Internal Server Error)**
    Any problems / exceptions on the server side (ConSol CM)

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **httpHeaders**
  Optional. Defines the response header. Only required if you want to define a specific httpHeader, i.e., one which is not part of the supported standard headers.
- **body**
  Optional. Defines the content of the response. If not provided, the response is empty. This is the object which can contain a JSON statement, see Coding Example.

Permissions

The webhook script is executed with administrator privileges. This means that all operations performed in the script are executed with these permissions. If this is not desired, the script can include the credentials of another user and some code to downgrade the session to the user's privileges. The method `executeWithUserPermissions` of the class `SecurityTemplate` is used for this purpose.

The following code example shows the usage of this method in a script. The session is downgraded to the permissions of the user identified by the login and password included in the JSON payload. Then, a customer search for the customer with the ID "12345" is performed with these permissions.

```
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 example 89: *Example of downgrading the session to a user's privileges*

Configuring the webhook service

The configuration page is available at the following URL:

```
https://<CM server address>/intg/<script name>/config
```

You need to log in with an administrator account. The configuration page allows to enable the webhook and define the security providers. Both steps are mandatory to use the webhook. See Security providers for details.

Testing the service

A simple way of testing your service is using a REST plugin in your browser:

- Set the required content type in the header, e.g. `application/json`.
- Enter the request content in the body using the syntax required by the content type, e.g. `{"ticket": "100200"}` to retrieve a case
- using JSON.

- Send a POST request to the service URL: `https://<CM server address>/intg/<script name>/service`

The response is returned as JSON. If the case is found, the response contains the case information defined in the script. Otherwise, the defined error message is returned.

Alternatively, you can also use CURL to test the service:

```
# Linux:
curl -X POST https://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 https://myserver.consol.de:8080/intg/myWebHook/service //
-H "content-type: application/json" //
-d "{\"ticket\": \"100200\"}"
```

The request payload in this example is the value of the `-d` option.

The following example shows the response generated by the script and returned to the requesting application:

```
{
   "status" : "OK" ,
   "ticket" : "100200" ,
   "subject" : "New Ticket for Resource 3" ,
   "engineer" : "Susan ServiceDesk"
}
```

F.16.1.4 Coding Example

The following example shows the implementation of a service named “myWebHook”. The script is called “myWebHook.groovy” and is stored in the script section of the Web Admin Suite / Admin Tool. It returns a JSON object with case number, subject, and assigned user as a response on a request with the case number.

```
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
def OK = false
if (message.ticket) {
   ticket = ticketService.getByName(message.ticket)
   if (ticket) {
      ticketname = message.ticket
      ticketsubject = ticket?.getSubject()
      if (ticket.engineer) {
         engineername = ticket.engineer.getFirstname() + " " +
          ticket.engineer.getLastname()
      }
      OK = true
   }
}
def response = new IntgServiceResponse()
if (OK) {
   response.httpStatusCode = 200
   response.body = JsonOutput.toJson([ticket: ticketname, subject: ticketsubject,
    engineer: engineername]) // optional
} else {
   response.httpStatusCode = 404
   response.httpHeaders = ['Content-Language':'en', 'Warning':'Required oject not
    found']
}
return response
```

Code example 90: *Example script which implements a webhook service*

F.17 System Architecture

ConSol CM is a Java EE application. It can be operated as

- **ConSol CM-only system**, see section Basic System Architecture

or as

- **ConSol CM system with reporting infrastructure**, see section System Architecture with Reporting Infrastructure

Furthermore, ConSol CM can be installed and operated in an application server cluster.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

F.17.1 Architecture of a CM System

F.17.1.1 Introduction to ConSol CM System Architecture

ConSol CM is a ***Java EE*** (Java Enterprise Edition) application that can be run in a standard application server on Unix/Linux or Windows systems. JBoss and Wildfly are supported.

In this chapter, a short overview of the ConSol CM system architecture will be provided.

> A detailed list of supported operation systems, application servers, database systems, and other systems, as well as storage and CPU requirements is given in the current *System Requirements*.

F.17.1.2 Basic System Architecture

ConSol CM is a Java EE application which is based on the classical three-tier architecture. The ConSol CM server is deployed in an application server and accesses a relational database. Two web interfaces are available as client interfaces: the standard interface is the ConSol CM Web Client, which is used by the engineers to work on the tickets. Another web client is the ConSol CM portal, CM/Track. This provides access to the system for customers who might want to know some basic facts about the status of their tickets. The two Java applications which are used to configure ConSol CM are the Admin Tool and the Process Designer. Both can be downloaded from the ConSol CM start page using Java Web Start (JWS). JWS is a component of every recent Java edition, so no extra installation is required on the PCs or Laptops you want to use to administer the system. On the contrary - you can do this from every regular web client with a supported web browser. Please make sure that the versions of all components which are used in your company meet the system requirements.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

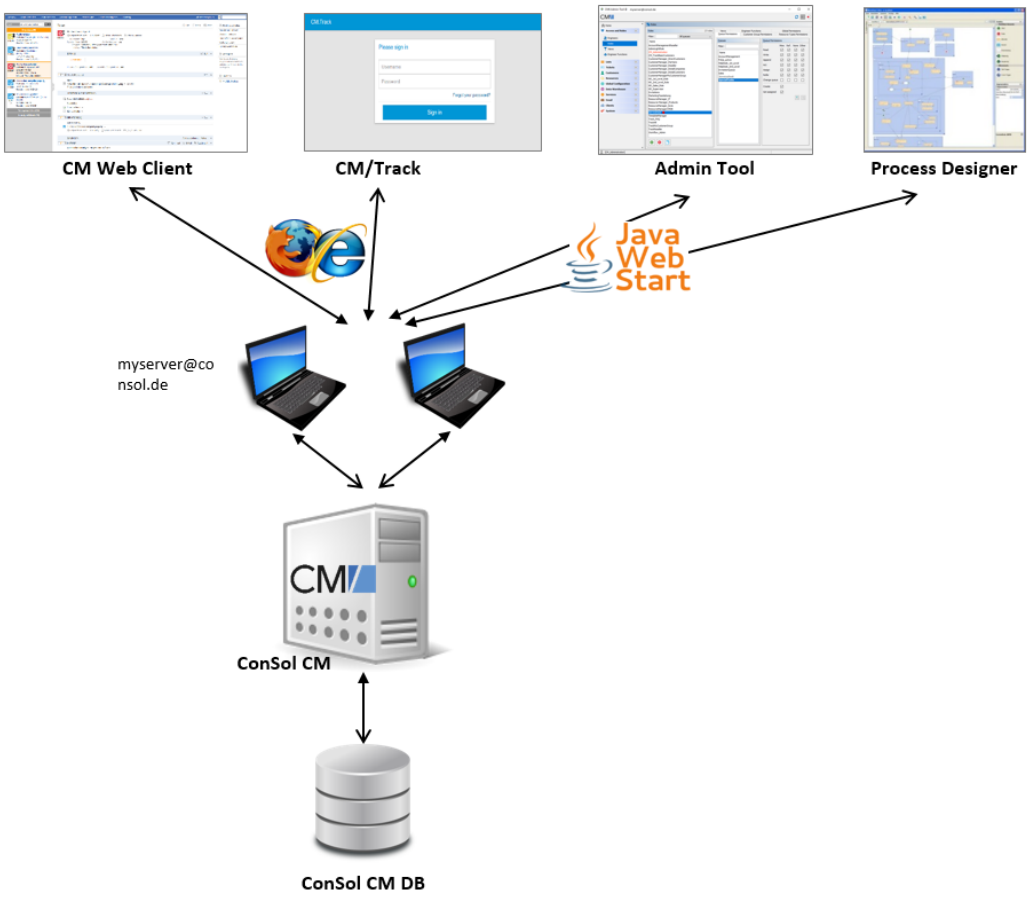


Figure 405: *ConSol CM - Basic system architecture*

CM Database

The ConSol CM database (CM DB) is a relational database which can be operated as Oracle, Microsoft SQL Server or MySQL system.

> A detailed list of supported operation systems, application servers, database systems, and other systems, as well as storage and CPU requirements is given in the current *System Requirements*.

- **Oracle**
  One database schema with one database user is used by ConSol CM.
- **Microsoft SQL**
  One database schema with one database user is used by ConSol CM.
- **MySQL**
  One database with one database user is used by ConSol CM.

F.17.1.3 System Architecture with Reporting Infrastructure

In order to allow Business Intelligence (BI) tools or other applications to build specific reports, OLAP cubes, and other analyses, ConSol CM provides a data warehouse (DWH) as one of its standard components. The DWH is a separate database (or database scheme, see below). The DWH is filled by a Java EE application called ConSol CM Reporting Framework (CMRF).

The ConSol CM standard function set comprises two components which enable reporting:

- **CMRF** (ConSol CM Reporting Framework)

  This is a Java EE application which synchronizes the ConSol CM database with the ConSol CM data warehouse (DWH). The CMRF can be deployed into the same application server as the core CM (overlay mode) or it can be run on a separate application server (standalone mode). The synchronization of CM data with the DWH is based on direct messaging. For a detailed explanation, please refer to the *ConSol CM Operations Manual*.section *Operating the Data Warehouse*.

- **DWH** (data warehouse)

  The ConSol CM DWH is a relational database which can be operated as Oracle, Microsoft SQL Server, or MySQL system. It stores the integrated/pre-processed data from the ConSol CM database.

> A detailed list of supported operation systems, application servers, database systems, and other systems, as well as storage and CPU requirements is given in the current *System Requirements*.

Separate application servers for ConSol CM and CMRF (standalone mode):

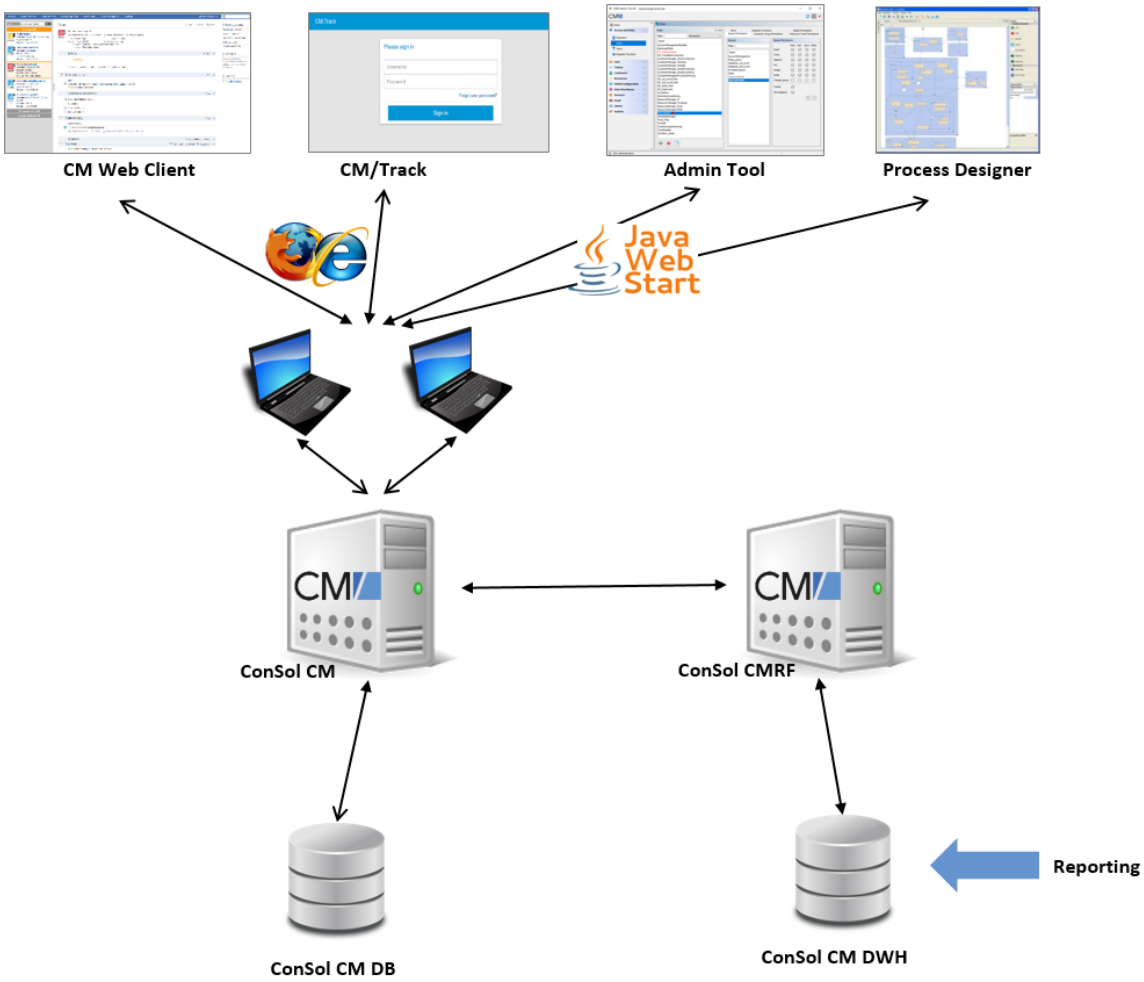


Figure 406: *ConSol CM - Infrastructure with CMRF and DWH (2 servers)*

One application server for ConSol CM and CMRF (overlay mode):

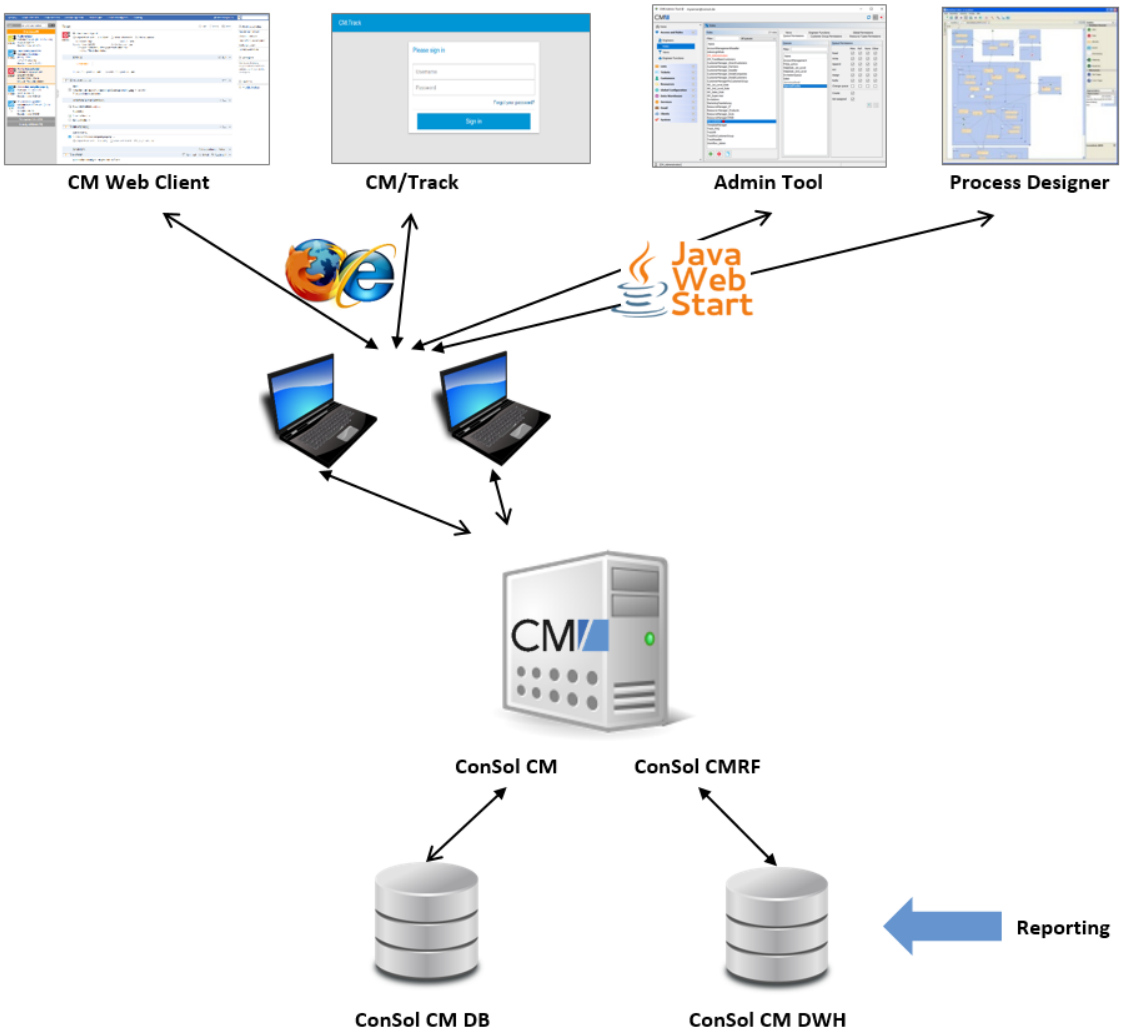


Figure 407: *ConSol CM - Infrastructure with CMRF and DWH (1 server)*

When the DWH has been established, BI (Business Intelligence) applications can be used to create reports, data cubes, and other reporting output formats. Please see the following example with the Pentaho™ BI Suite.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Separate application servers for ConSol CM and CMRF (standalone mode):

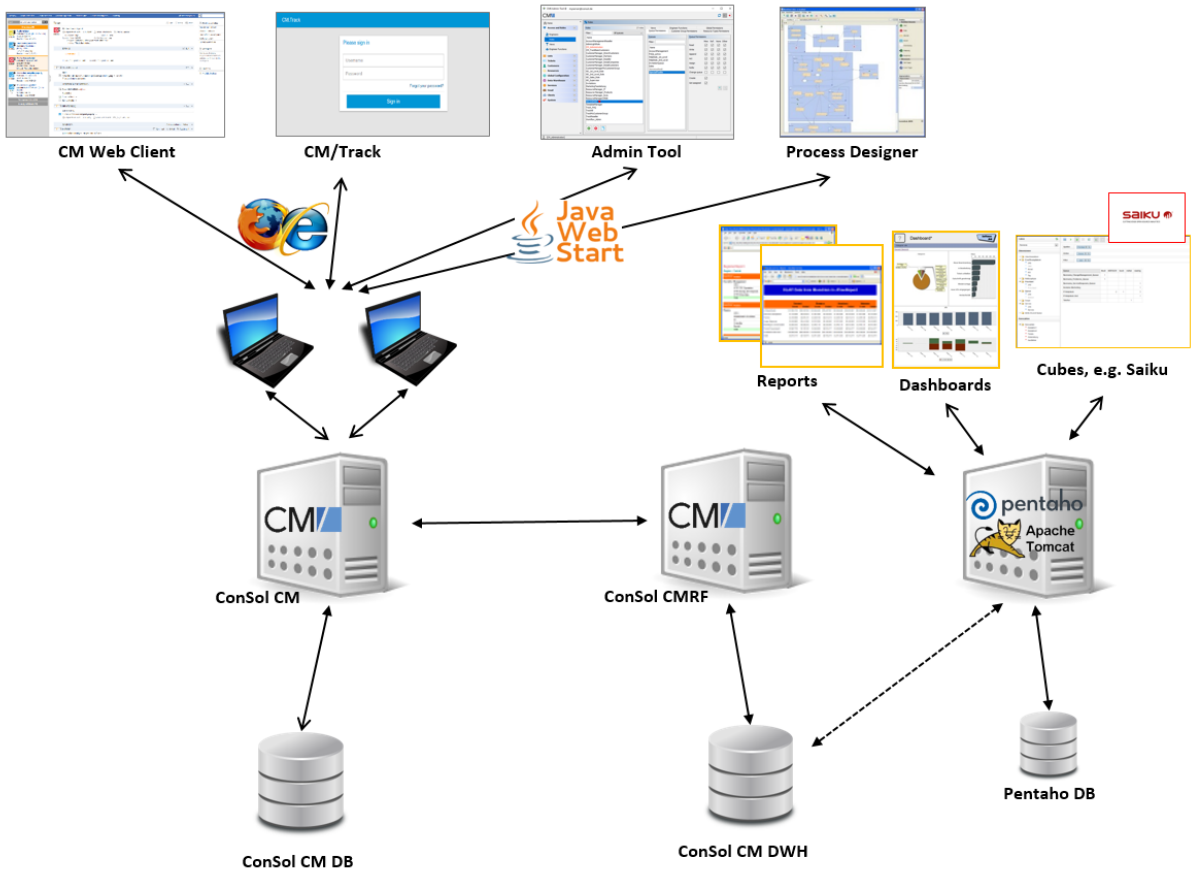


Figure 408: *ConSol CM - Reporting infrastructure (2 servers)*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

One application server for ConSol CM and CMRF (overlay mode):

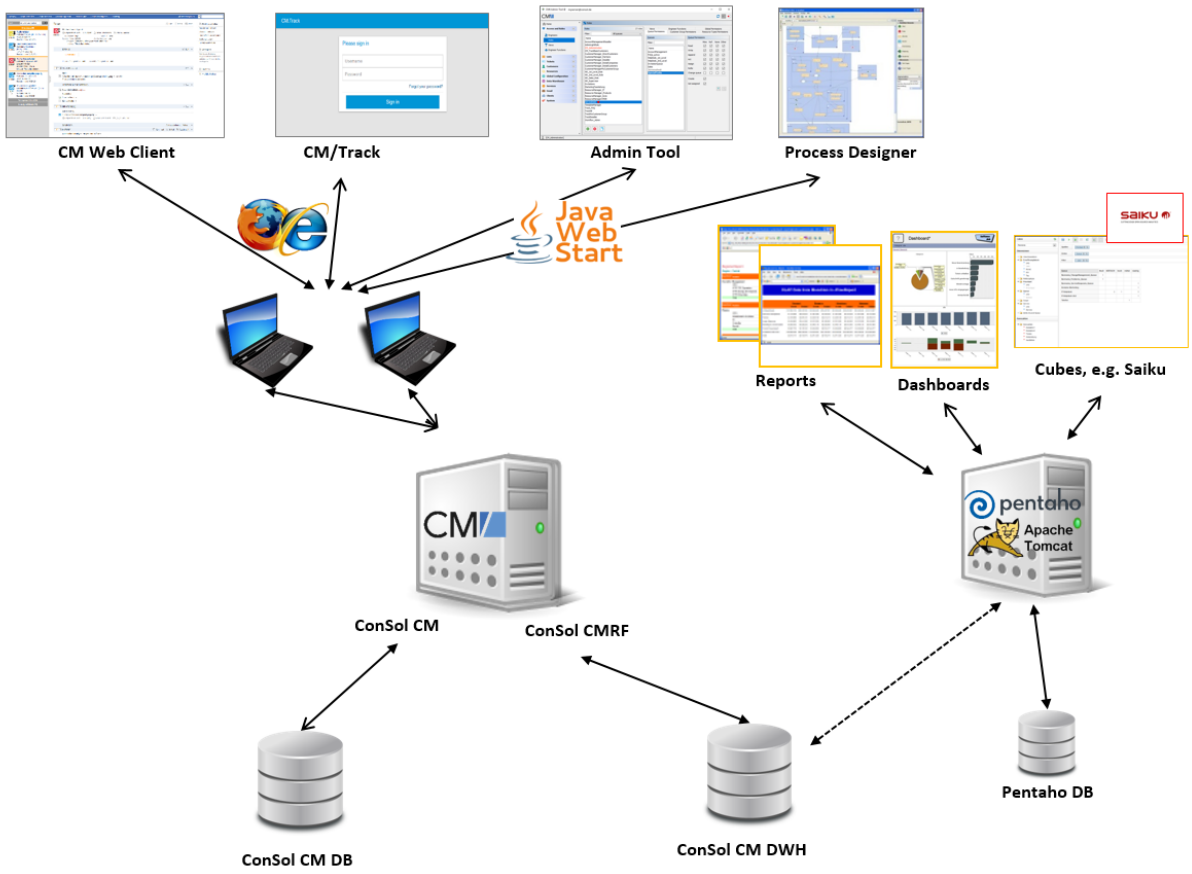


Figure 409: *ConSol CM - Reporting infrastructure (1 server)*

DWH Database

- **Oracle**
  One database scheme with one database user is used by the DWH.
- **Microsoft SQL**
  One database scheme with one database user is used by the DWH.
- **MySQL**
  One database with one database user is used by the DWH.

F.17.1.4 Components for Email Interactions

One of the core functionalities of ConSol CM is integration with mail servers. This allows ConSol CM to send and to receive emails. For the engineer, this means new tickets can easily be opened via email and the entire communication regarding a case is located in the respective ticket, including all incoming and outgoing emails.

In order to receive emails, ConSol CM connects to a mail server and retrieves emails from one or more mailboxes. ConSol CM acts like a regular email client (e.g., Thunderbird, Microsoft Outlook) and uses standard email protocols like IMAP or POP3. If you want to use the secure version, IMAPs and POPs are also supported, in which case the required certificates have to be installed on the server.

In order to send emails, ConSol CM uses an SMTP server.

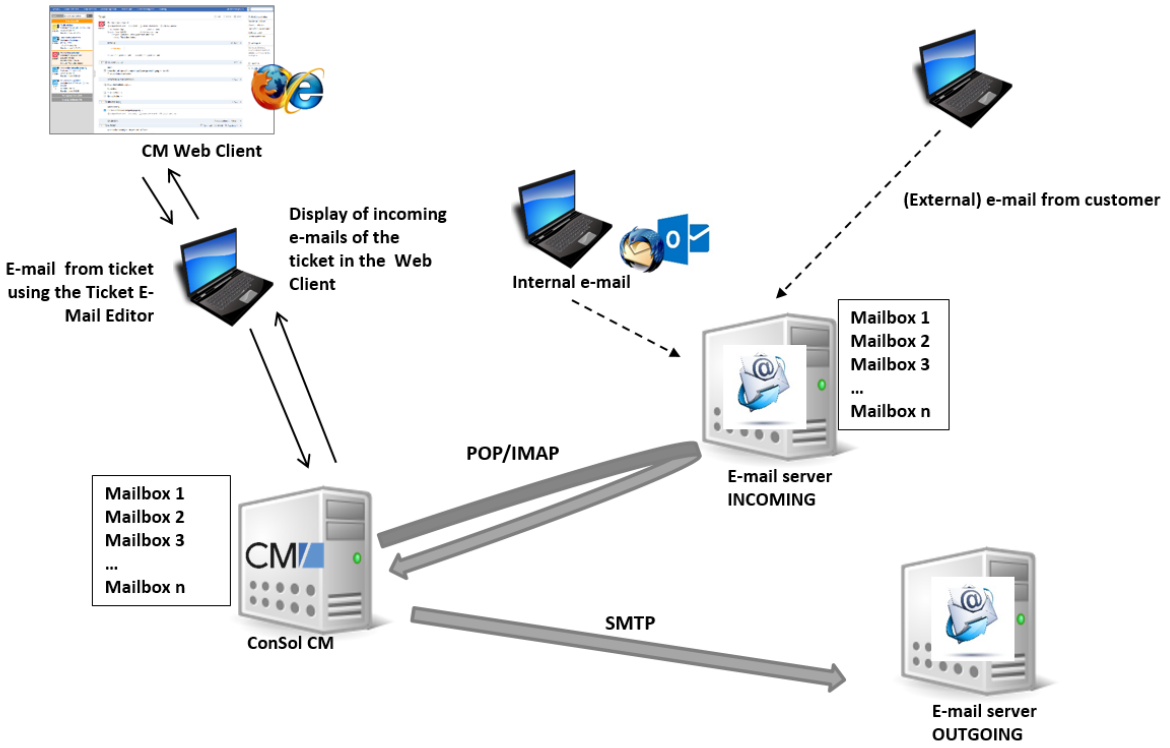


Figure 410: *ConSol CM - Mail server interactions*

F.17.1.5 Indexer

In order to perform effective searches in the database, ConSol CM builds an index for each ticket field, customer field, and resource field which should be included in a search. Furthermore, the engineer data, the ticket comments and the attachments are indexed by default. The indexes are stored in the file system. Please refer to the section ConSol CM File System Structure for an explanation of the index directory structure, and read the detailed introduction to the entire topic in the section Index.

F.17.1.6 LDAP Authentication

As standard feature, ConSol CM can use LDAP authentication in the Web Client and/or in the portal (CM/Track). Depending on the configuration of your LDAP server (e.g., Microsoft Active Directory), a user name and password might be required to establish the LDAP connection. All LDAP parameters are stored as ConSol CM system properties.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

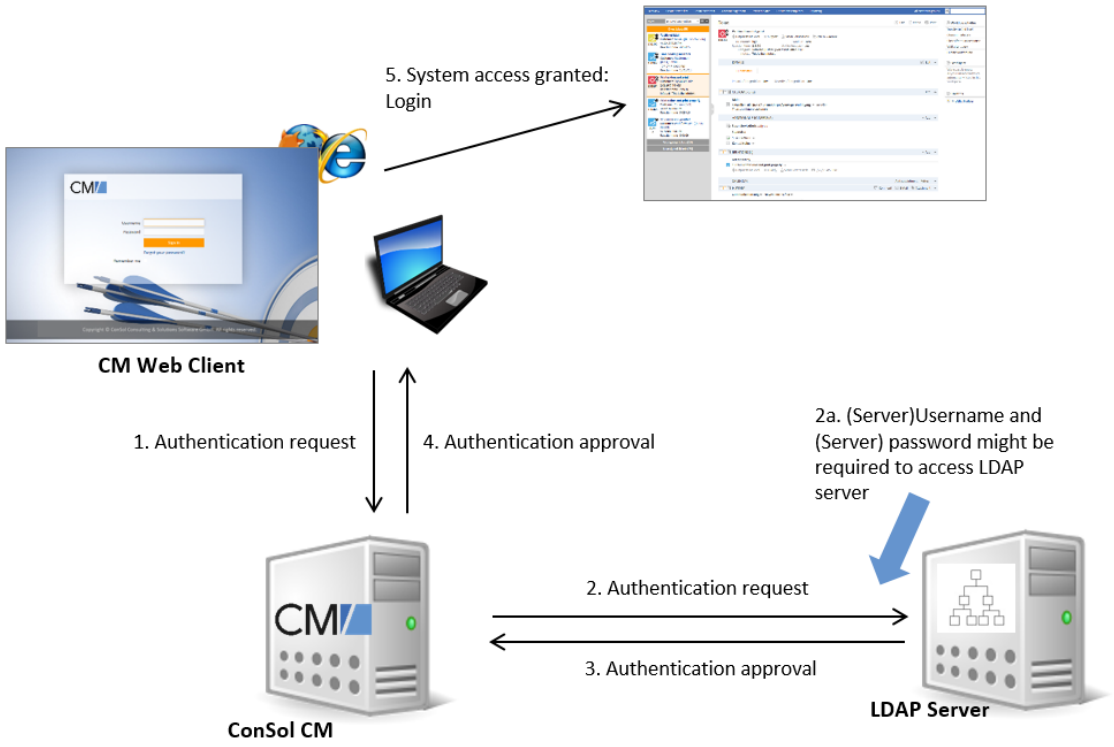


Figure 411: *ConSol CM - LDAP authentication (Web Client)*

F.17.1.7 ConSol CM File System Structure

Most of the data concerning the configuration and operation of ConSol CM is stored in the ConSol CM database. However, some data is saved in the file system in the data directory entered during system setup.

ConSol CM Data Directory

Important directories:

- *index*

  This is the directory where all the indexes are stored (see also section Index). Be sure to include it into your regular file system backup.

  - **index.0**
    In this directory, there is a subdirectory for each required index.

JBoss / Wildfly Application Server File Structure

In an JBoss EAP installation the following directories are relevant for ConSol CM, among others:

- **bin**
  Settings for all environments:
  - Configuration of the database connection and logging in the file `cm6-config.properties` or `cm6-domain-config.properties`
  - Configuration of environment variables in the file `standalone.conf` or `standalone.conf.bat`

- **modules\system\layers\base**
  Subfolders contain the JDBC drivers:
  - com\microsoft\sqlserver\jdbc\main\mssql-jdbc-7.4.1.jre8.jar
    (Microsoft SQL)
  - oracle\jdbc\main\ojdbc8.jar and oracle\jdbc\main\orai18n.jar
    (Oracle)
  - com\mysql\jdbc\main\
    (MySQL JDBC driver destination, must be installed manually)
- **standalone**
  Configuration in standalone environments:
  - **configuration**
    Configuration of the database connection and logging in the files `cm6-<DB>.xml`, `cm6-cmrf-<DB>.xml` or `cm6-dwh-<DB>.xml`
  - **data**
    Data for operation, e.g., `tx-operation` keys
  - **deployments**
    Deployed applications, for example `cm6.ear` and `cm-track.war`
  - **log**
    Log files, see section Log Files.
  - **tmp**
    Temporary data and also working copy of the application server files. When JBoss is stopped, it can be emptied, e.g. for error analysis and/or fixing.
- **domain**
  Configuration in domain (i.e. multi-server) environments
  - **configuration**
    Configuration of the database connection and logging in the files `cm6-<DB>.xml`, and `master-host.xml` and `slave-host.xml`
  - **servers/<server-name>**/log
    Log files

F.17.1.8 Logging and Log Files

Introduction

Log files are the main information source for the administrator about the activities of the system and potential problems of the system. The administrator should have a look at the log files on a regular basis. There may be problems that do not appear on the user interface, but are reported in the log file.

The standard path for the log files is `<JBOSS_HOME>/standalone/log`.

Log Files

The location of the log files and the logging behavior can be configured in the respective configuration files:

- **cm6-config.properties**, **cm6-cmrf-config.properties** (JBoss EAP / Wildfly standalone installations with or without CMRF)
- **cm6-domain-config.properties**
  (JBoss EAP cluster installation)

Log File Types

The following log files are used:

- **access.log**
  Access to and usage of the Web Client is logged. In the current configuration, a separate `access.log` file is written every day.

  How to activate the access.log

  In a default installation, the following lines in the configuration file, cm6-<DB>.xml, cm6-cmrf-<DB>.xml or cm6-dwh-<DB>.xml are commented out. By commenting them in, you can start access logging in your ConSol CM system.

```
<server name="default-server" default-host="default-host">
   <http-listener name="http" socket-binding="http" max-post-
    size="104857600" redirect-socket="https" enable-http2="true"/>
   <ajp-listener name="ajp" socket-binding="ajp"/>
   <host name="default-host" alias="localhost">
   <!-- <access-log pattern="%h %l %u %t %r %s %b %{Referer}i %{User-
    Agent}i %S %T" directory="${jboss.server.log.dir}" prefix="access"
    suffix=".log" worker="default"/> -->
     <location name="/" handler="welcome-content"/>
     <filter-ref name="server-header"/>
     <filter-ref name="x-powered-by-header"/>
     <http-invoker security-realm="ApplicationRealm"/>
   </host>
</server>
```

- **cmrf.log**
  Messages pertaining to CMRF (ConSol CM Reporting Framework), i.e., messages regarding the data transfer operations from the ConSol CM database to the CMRF database (DWH).
- **cmweb.log**
  Messages pertaining to the ConSol CM Web Client.
- **ctx.log**
  Contains messages from the Spring Framework.
- **errors.log**
  Contains only messages that have at least the log level `ERROR`.
- **index.log**
  Messages pertaining to the Indexer.
- **mail.log**
  Contains log messages from the email subsystem.
- **operationtimes.log**
  Contains timing information for requests in order to identify possible performance bottlenecks.
- **operationtimes-db.log**
  Contains information about database access times of atomic operations in order to identify possible performance bottlenecks. This logging is configured using the system properties cmas-core-server, dao.log.threshold.milliseconds and cmas-core-server, dao.log.username.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **operationtimes-rest.log**
  Contains the operation times for REST requests. The request URL and the time needed to process the request are logged on DEBUG level. The logging is configured using the system properties cmas-restapi-core, request.log.threshold.milliseconds and cmas-restapi-core, request.log.username
- **security.log**
  Contains log messages which are related to security, authorization and authentication, e.g. log entries about Kerberos authentication.
- **server.log**
  The general log file that contains all messages, by default, with at least log level `INFO`.
- **session.log**
  Contains messages about logins (session starts) and session timeouts of ***ConSol CM*** users.
- **sql.log**
  Contains log entries about SQL statements coming from hibernate if it is set to `DEBUG` level (by default it is set to `INFO`).
- **support_libs_errors.log**
  Contains errors which are thrown by support libs but are properly handled by the ConSol CM application (this method keeps the `server.log` clean).
- **timer-manager.log**
  Contains additional log messages written in log level `DEBUG` when workflow timers are activated or deactivated. Information about the escalation date is logged, too.
- **track_jersey.log**
  Contains log messages which are related to ***CM/Track***.
- **tx.log**
  Contains Spring Framework transactions related log messages.
- **unit-deletion.log**
  Contains information regarding the deletion of customers.
- **workflow.log**
  Information about activated/reinitialized/deactivated timers is logged with level `INFO` and all debug output related to the workflow engine is written to this dedicated file.

Admin Tool-Specific Log Files

Two log files report information which are specific for the Admin Tool. The CM logging configuration might have to be adapted during a system update from a CM version lower than 6.11 to 6.11.

- **audit.log**
  Reports login/logout operation and all operations which have been made using the Admin Tool:
  - configuration changes
  - creation of new objects
  - deletion of objects
  - reopening of tickets

  Reports login/logout operation and operations which have been performed using the Process Designer, e.g.,
  - deployment of a workflow

- **transfer.log**
  Reports information about the export / import of scenarios.

Log File Structure

In the default configuration, log file entries have the following syntax:

```
Date Timestamp Loglevel [Logger] Message
```

Example for a log file entry (successful start of ***ConSol CM*** in JBoss):

```
2017-10-11 13:52:44,526 INFO [reemarker.FreeMarkerConfigurer] [-]
 ClassTemplateLoader for Spring macros added to FreeMarker configuration
```

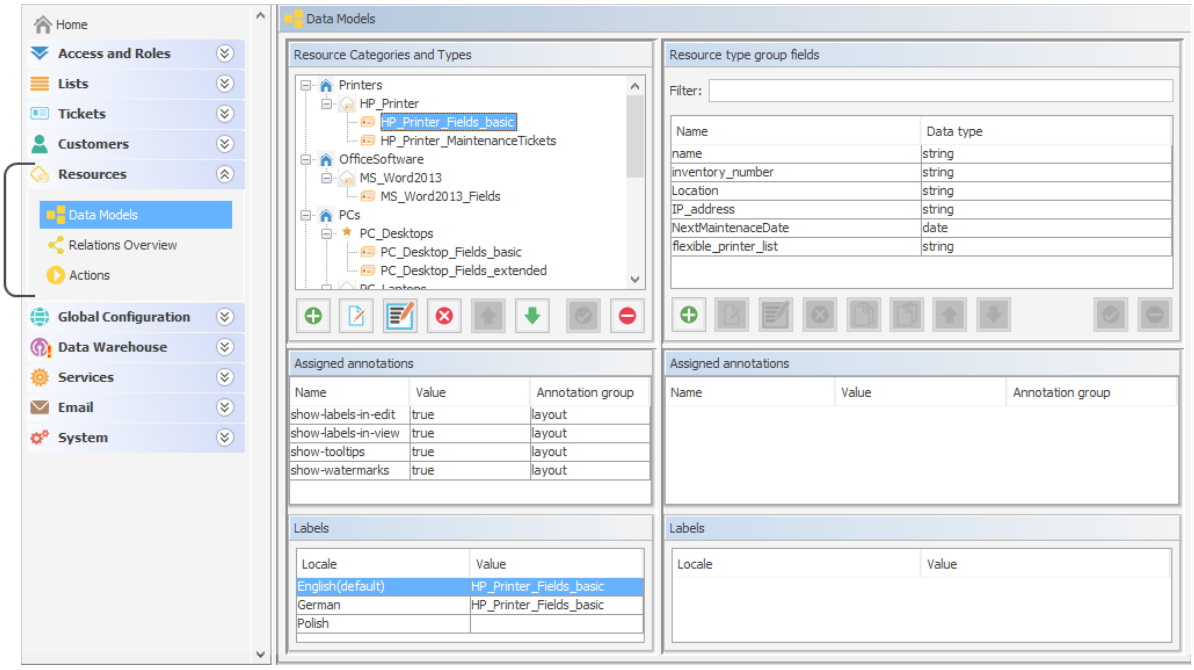
The components of the message:

- **Date:**
  October 11th, 2017
- **Timestamp:**
  13:52:44
- **Loglevel:**
  INFO
- **Logger:**
  reemarker.FreeMarkerConfigurer
  Name of a Java class, not complete (only last 30 characters), the real name would be Free-marker.FreeMarkerConfigurer
- **Message:**
  ClassTemplateLoader for Spring macros added to FreeMarker configuration

Simple messages, and those reporting a successful operation, often have only one line.

When errors are reported (log level `ERROR`), you might find stack traces in the logs. Please contact one of our ***ConSol CM*** consultants or our ***ConSol CM*** support team for help.

G - Add-Ons



This section provides information about the optional ConSol CM add-ons:

- CM/Resource Pool
- CM/Track: The Customer Portal
- CM/EBIA
- CM/Forms
- CM/Machine Learning
- CM/Archive
- CM/Doc
- CM/Phone: CTI with ConSol CM

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.1 CM/Resource Pool

Starting with version 6.10, ConSol CM offers the add-on CM/Resource Pool. Using this module, you can extend the ConSol CM database to store data objects for any type of asset or object in your company. A service enterprise might use the Resource Pool to store and manage SLAs. For an IT Service Desk, the Resource Pool might be used to implement an IT asset database. A marketing department might store design and text templates. A reseller might reflect the product portfolio in the Resource Pool. Hence, implementing the Resource Pool for your company can be a core step in the ConSol CM customization process. To learn how to work with this module, read the following sections:

For a general introduction to the CM/Resource Pool, read the section Introduction to CM/Resource Pool.

All elements in the Admin Tool which you will work with to configure the Resource Pool are explained in section CM/Resource Pool - Admin Tool Elements.

To get an impression of how the Resource Pool objects are managed by engineers working with the Web Client, see the section A Short Introduction to CM/Resource Pool Functionality in the Web Client.

One of the first steps when you start working with the Resource Pool is the set-up of the resource model. This is covered in section CM/Resource Pool - Setting Up the Basic Resource Model.

Similar to customer templates, you can define templates which control the appearance of resource data in the Web Client. These templates are explained in section CM/Resource Pool - Templates for Resource Data.

Resources can be related. Read section CM/Resource Pool - Resource Relations to learn how to define and to work with resource relations.

For every resource type, activities can be performed. They are based on resource actions which are explained in section CM/Resource Pool - Resource Actions. Some more specific information about how to implement scripts for the Action Framework are also provided in section Scripts for the Action Framework.

As per general ConSol CM standards, engineers can have certain access permissions to resources. Those permissions are based on roles. Please read section CM/Resource Pool - Assigning Permissions for Resources for details.

The Resource Pool Dashboard provides a very convenient entry point to all Resource Pool objects. The Dashboard configuration is explained in CM/Resource Pool - The Resource Dashboard.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.1.1 Introduction to CM/Resource Pool

G.1.1.1 Introduction to CM/Resource Pool

Starting with ConSol CM version 6.10, the add-on ***CM/Resource Pool*** is available. The *Resource Pool* provides a database extension which enables ConSol CM to create and store objects other than tickets or customers. For example, you could represent your IT landscape by modeling each asset (like PCs, laptops, monitors, printers) as a resource in the Resource Pool. Similar resources (e.g., HP printers) are managed in one *resource type*, and similar resource types form a *resource category* (e.g., printers). Every specific object (e.g., the printer #4711) is an instance of a resource type (e.g., resource type *HP_Printer*). Other examples for the use of CM/Resource Pool are the use for products, facilities, machines, rooms, vehicles, contracts (e.g., SLAs), or scientific samples.

CM/Resource Pool extends ConSol CM's basic objects:

1. A ticket is an instance of a process execution.
2. A real-world customer (e.g., a person who calls the service desk to open a ticket) is an instance of a customer (unit) definition.
3. **A real-world resource (e.g., a printer or an SLA) is an instance of a resource definition.**

G.1.1.2 CM/Resource Pool at a Glance

CM/Resource Pool Structure

The Resource Pool is structured based on a two-level hierarchy and comprises any number of resource categories. Each resource category represents a subtree within the Resource Pool. You can manage any number of resource categories, i.e., any number of subtrees, within the Resource Pool. In this way, you could, for example, manage the IT landscape, as well as your product portfolio, as resources, each in one or more specific resource categories/subtrees.

Relations between resources and other ConSol CM objects can mirror various constellations, e.g., a resource of resource category *printer* and of resource type *HP printer* can have a relation to a ticket which was opened for a specific printer. Or, e.g., the printer is related to all contacts which use this printer. Relations between resources are also possible. For a detailed explanation of working with resource relations, see section CM/Resource Pool - Resource Relations.

Data fields are defined for each resource type in a way similar to the field definition for ticket data (ticket fields) and customer data (customer fields). Data fields are called *resource fields* and are always managed in groups which are called *resource field groups*. The set-up of the data model for the Resource Pool, i.e., the *resource model*, is explained in detail in section CM/Resource Pool - Setting Up the Basic Resource Model.

For each resource type, resource actions can be defined, very similar to customer actions for customer objects (companies and contacts). The resource actions will trigger scripts which perform resource-specific activities, like updating the data of a resource (e.g., providing a list with all companies which are related to the printer #4711 and create a maintenance ticket for a firmware update). Resource actions are a component of the Action Framework and are explained in detail in section CM/Resource Pool - Resource Actions.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

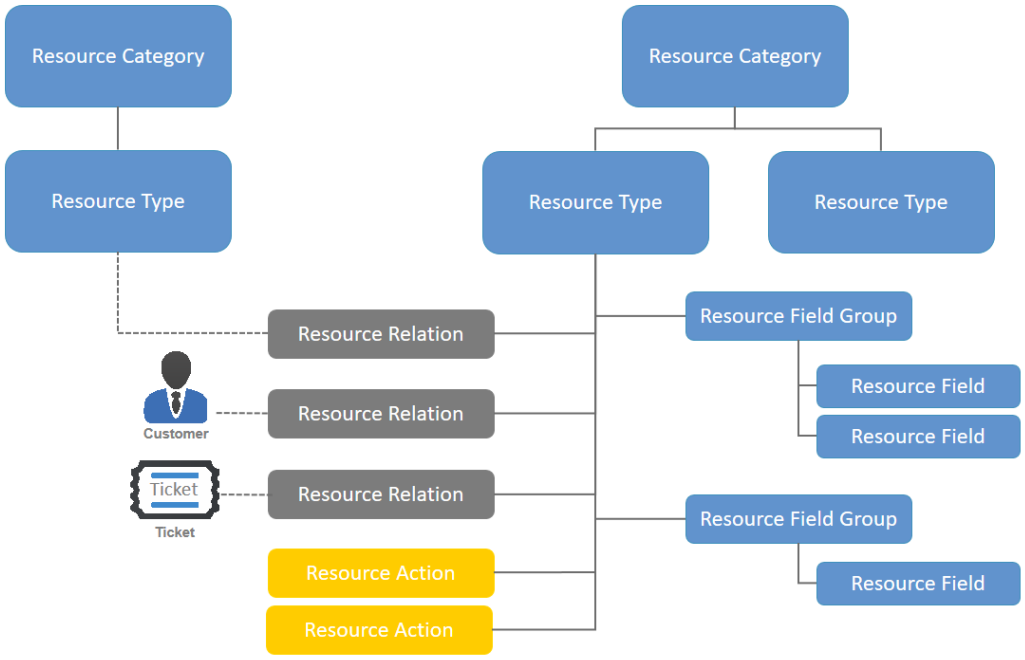


Figure 412: *CM/Resource Pool - Objects in a resource model*

Important Terms

- A **resource category**
  represents the highest hierarchy level in the Resource Pool, it can be seen as the root node of a Resource Pool subtree. For example, the Resource Pool of a company could have a resource category *Printers*. A resource category comprises one or more resource types, e.g., HP_Printers, Kyocera_Printers, Lexmark_Printers.
- A **resource type**
  represents similar resources which share the same data fields, but with differing values. For example, the resource type *HP_Printers* has the data fields (resource fields) *Model*, *IP Address*, *Inventory Number*, and *Location*. Each real-world HP printer is represented by an object of this Resource type. Lexmark printers might have the data fields (resource fields) *Model*, *IP Address*, *Location* and *SLA type*. Each real-world Lexmark printer is represented by an object of this resource type.

  > Within the Resource Pool hierarchy, there is **no inheritance**! Data fields (resource fields) are always defined as resource field groups for each specific resource type!

- A **resource field group**
  groups one or more resource fields within one resource type. On the resource page, a resource field group can be displayed as tab, just like the ticket field group for ticket data on the ticket page. Resource field groups can be annotated.
- A **resource field**
  is a data field for data of a resource type, e.g., *Model*, *IP address*, or *Location*. A resource field always is of one specific data type. Resource fields contain resource data as ticket fields contain ticket data. Resource fields can be annotated to modify their logical behavior or graphical appearance.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- The **resource model**
  is the complete data model which defines all objects within the Resource Pool. It is built using the Admin Tool, navigation group *Resources*, navigation item *Data Models*.
- A **resource relation**
  is a relation of a resource to another ConSol CM object. This can be a ticket, a customer or another resource.
- A **resource action**
  is an action defined for a resource type and available for each real-world resource object of this type. The action can be triggered automatically or manually. Manual resource actions can be selected in the Web Client on the resource page like workflow activities for tickets.
- The **resource page**
  is the page in the Web Client where all data for one real-world resource is displayed, e.g., the resource page for the HP printer #4711.
- The **resource type page**
  is the page in the Web Client which provides information about a resource type and an overview (list) of all resources of this type.
- The **Resource Pool Dashboard**
  is the overview page in the Web Client where all resource categories and resource types are displayed to which the current engineer has access.

G.1.1.3 Defining Resource Models Using the Admin Tool

To be able to create real-world resources (e.g., HP printer #4711), a resource model has to be defined, i.e., a definition of all data fields which are required for all resources of a type, e.g., of type *HP_printer*. All resource data models have to be defined using the Admin Tool. This applies to:

- Resource categories (e.g., resource category *Printers*)
- Resource types (e.g., resource type *HP Printer*)
- Resource relations (e.g., possible relation between resource type *HP Printer* and *Ticket*)
- Resource actions (e.g., resource action *Retrieve Maintenance Contract Data from ERP System*)

The following figure shows an example of a simple Resource Pool configuration with four resource categories. All steps you have to perform to configure this constellation will be explained in detail in the following sections. The Admin Tool sections involved in Resource Pool configuration are described in section CM/Resource Pool - Admin Tool Elements.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

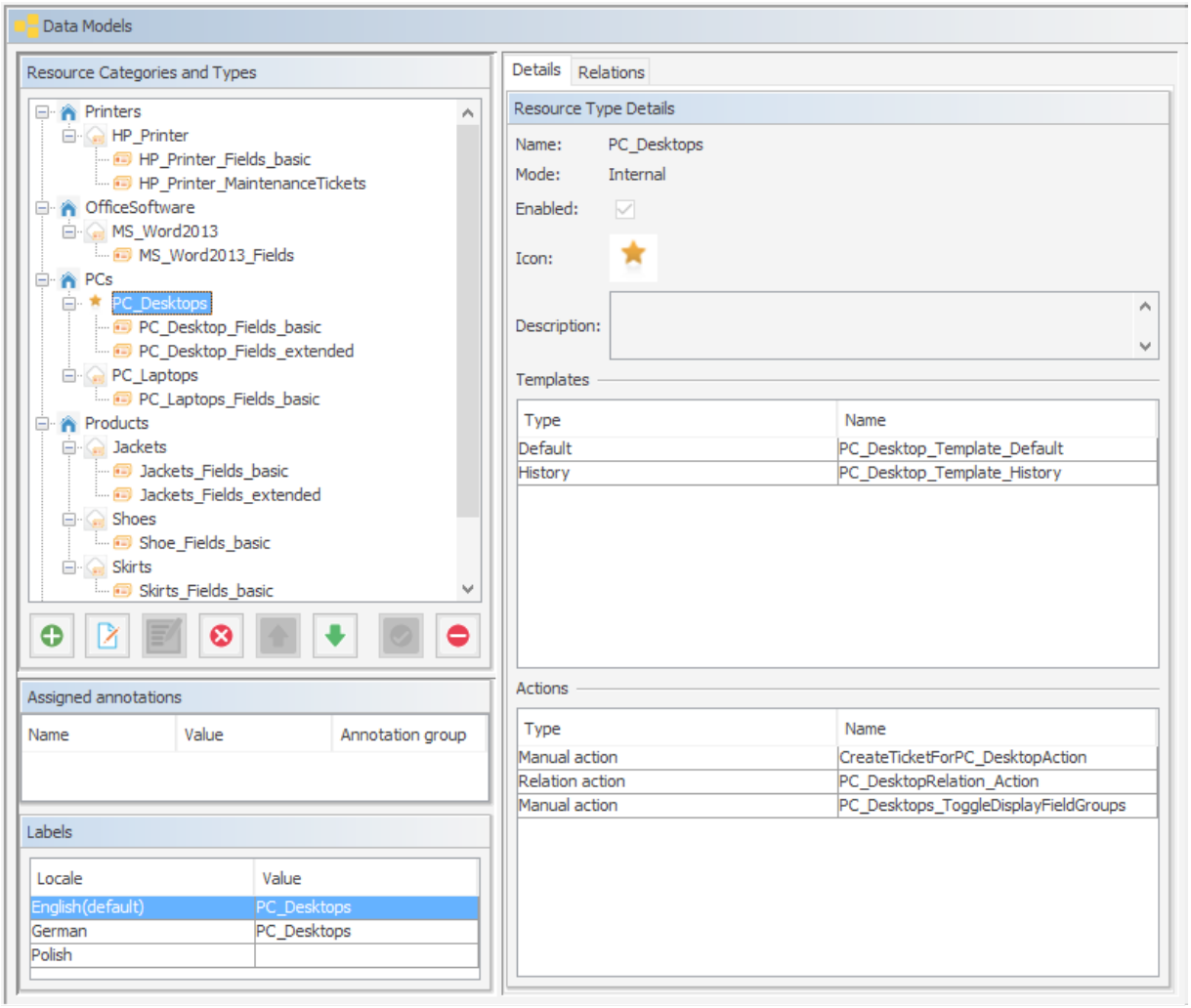


Figure 413: *ConSol CM Admin Tool - Resources, Data Models: Simple Resource Pool configuration*

Once you have defined the resources (i.e., the data model for the real resources) using the Admin Tool, the engineers can start working with real resource objects using the Web Client. This is explained briefly in section A Short Introduction to CM/Resource Pool Functionality in the Web Client. A detailed explanation is provided in the *ConSol CM User Manual*.

G.1.1.4 Licensing

CM/Resource Pool has to be licensed separately. Please ask your ConSol CM sales representative for more information.

G.1.1.5 Programming with Resource Pool Objects

The ConSol CM Java API has been extended considerably to implement the Resource Pool. The explanation of important Groovy classes and various programming examples are provided in the *ConSol CM Process Designer Manual*, CM version 6.10.

G.1.2 CM/Resource Pool - Admin Tool Elements

You will have to work with several sections of the Admin Tool to configure CM/Resource Pool:

- **Navigation group *Access and Roles***
  To assign Resource Pool permissions to roles
- **Navigation group *Lists***
  If you use MLAs and/or enums (sorted lists) in resource fields
- **Navigation group *Resources*** (see following figure)
  - **Data Models**
    Definition of the resource model, including resource relations
  - **Relations Overview**
    Contains a list of all resource relations, read-only mode, no definitions/configurations here
  - **Actions**
    Definition of resource actions
- **Navigation group *Global Configuration***
  - **Labels**
    In case you would like to modify labels of Resource Pool elements in the Web Client or define your own labels used in scripts
- **Navigation group *System***
  - **Scripts and Templates**
    - Scripts for resource actions
    - Templates for the display of resource data in the Web Client
  - **License**
    To activate a new license for CM version 6.10 with CM/Resource Pool

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

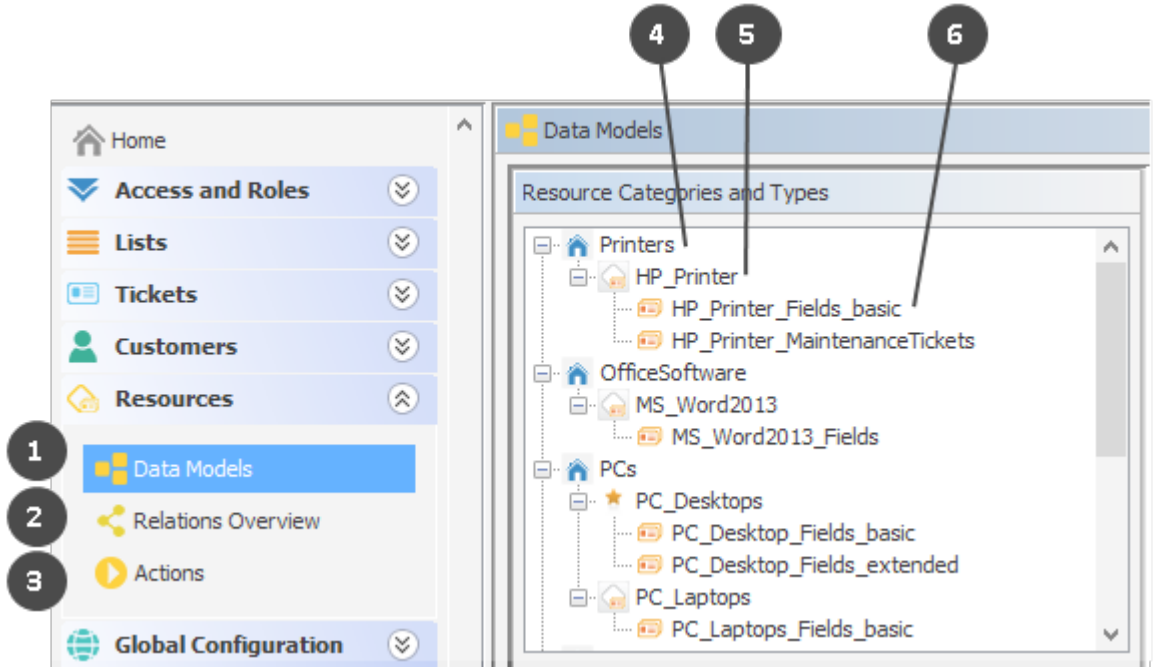


Figure 414: *ConSol CM Admin Tool - Resources, Data Models: Main elements for the configuration of CM/Resource Pool*

The main elements of CM/Resource Pool are:

- Navigation item *Data Models* to define resources and resource relations (1)
- Navigation item *Relations Overview* to display a relations list (2)
- Navigation item *Actions* to manage resource actions (3)
- Resource category (4)
- Resource type (5)
- Resource field group (6)

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

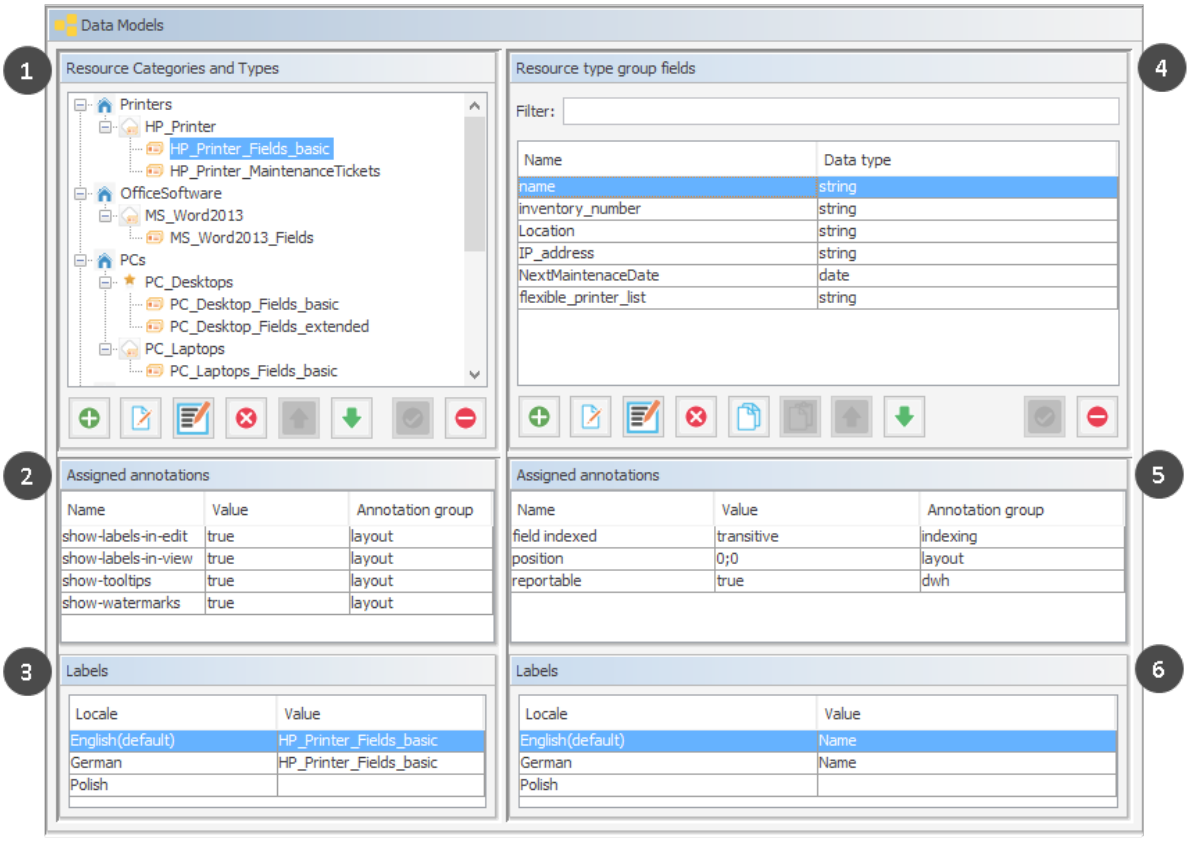


Figure 415: *ConSol CM Admin Tool - Resources, Data Models: CM/Resource Pool configuration*

The *Data Models* screen consists of six sections:

- Resource data models (1)
- Annotations for the selected resource field group (2)
- Labels for the selected resource field group (3), i.e. the localized values of the name of the selected resource field group
- Resource fields (4) of the selected resource field group
- Annotations for the selected resource field (5)
- Labels of the selected for resource field (6), i.e. the localized values of the name of the selected resource field

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.1.3 A Short Introduction to CM/Resource Pool Functionality in the Web Client

G.1.3.1 Overview of Resource Pool Operations in the Web Client

When you, as an administrator, have defined the resource data model (resource categories and the required resource types with their resource field groups and resource fields) using the Admin Tool, engineers can start working with real-world Resource Pool objects, provided they have the required access permissions. Permissions are discussed in section CM/Resource Pool - Assigning Permissions for Resources. In this section, you will get an overview of the following operations:

- Creating Resources Using the Web Client
- Working with the Resource Type Page
- Setting a Resource as Favorite
- Defining and Using Resource Relations
- Defining and Using Resource Actions
- Using the Quick Search to Find Resources
- Using the Detailed Search to Find Resources

G.1.3.2 Creating Resources Using the Web Client

In the Admin Tool, you only define the data models for the objects (developers: this is comparable to defining classes in object-oriented programming). The real objects (the instances of the defined data models) have to be defined using the Web Client where the Resource Pool Dashboard provides the required graphical user interface. The permissions to resources are managed based on resource types, using roles, according to the ConSol CM standard. An engineer only sees the resource types in the Resource Pool Dashboard if he has the required permissions. The role management for resources is explained in detail in section CM/Resource Pool - Assigning Permissions for Resources.

The following figure shows the Resource Pool Dashboard with the resource models which have been defined in the Admin Tool (see section CM/Resource Pool - Admin Tool Elements). The logged-in engineer has access permissions for all resource types.

The label for the Resource Pool in the main menu (as shown in the following figure) can be modified according to the requirements in the specific system. Examples for alternative wording could be *Asset management*, *Products*, *Machines*, or *Inventory*. Please refer to the Labels section for an explanation of how to change the label.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

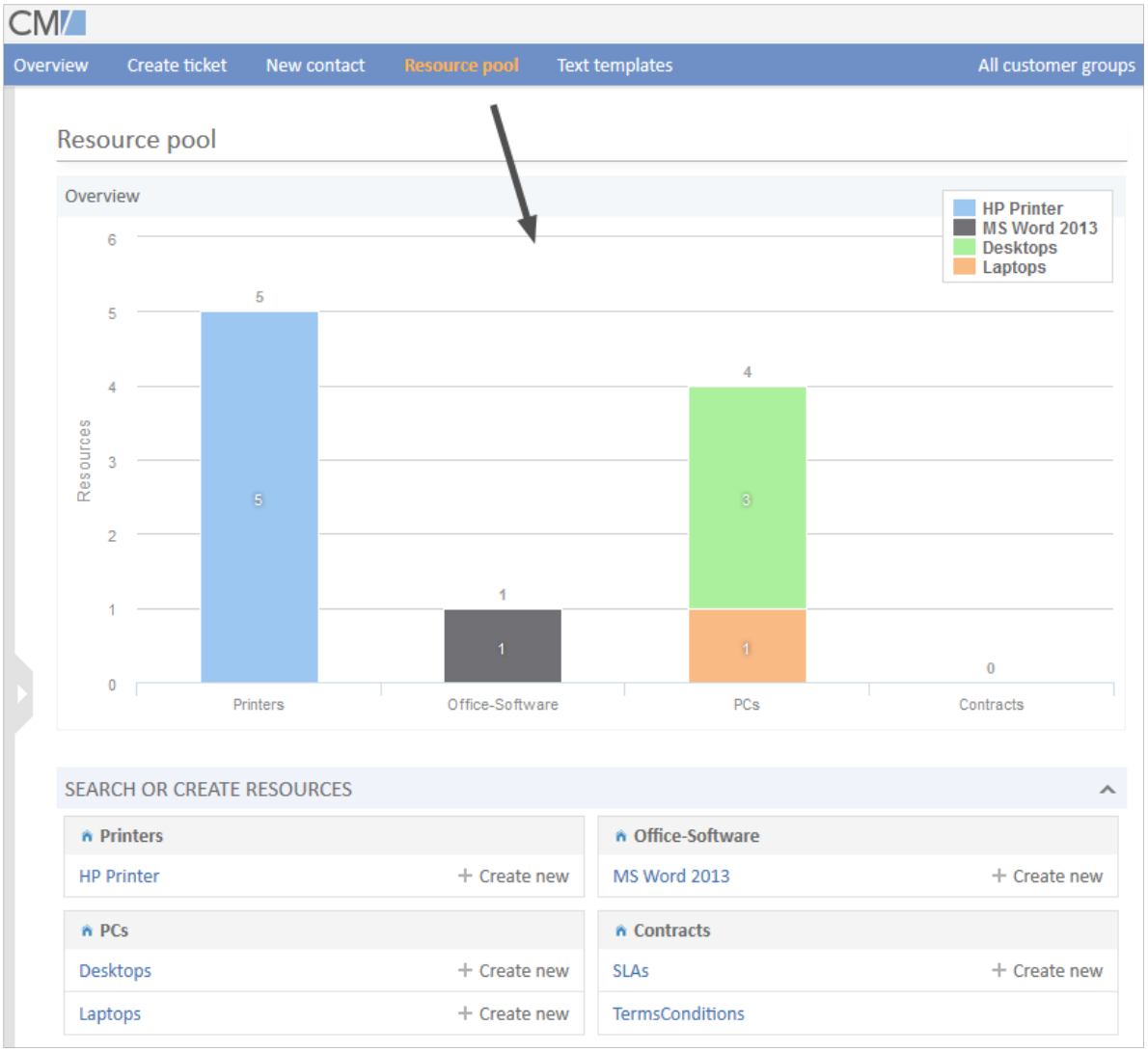


Figure 416: *ConSol CM Web Client - Resource Pool Dashboard*

Once the data models have been defined (in the Admin Tool) and you, as an engineer, have the required access permissions, you can create a real object, e.g., you can create an object for the printer #4711 which is located on the top floor.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

The following figures demonstrate the required steps.

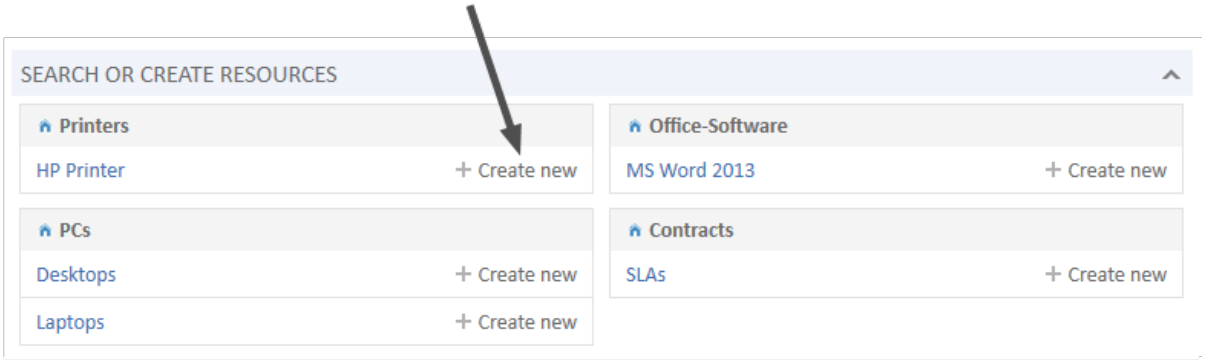


Figure 417: *ConSol CM Web Client - Resource Pool Dashboard: Create a new resource of type HP Printer, step 1*

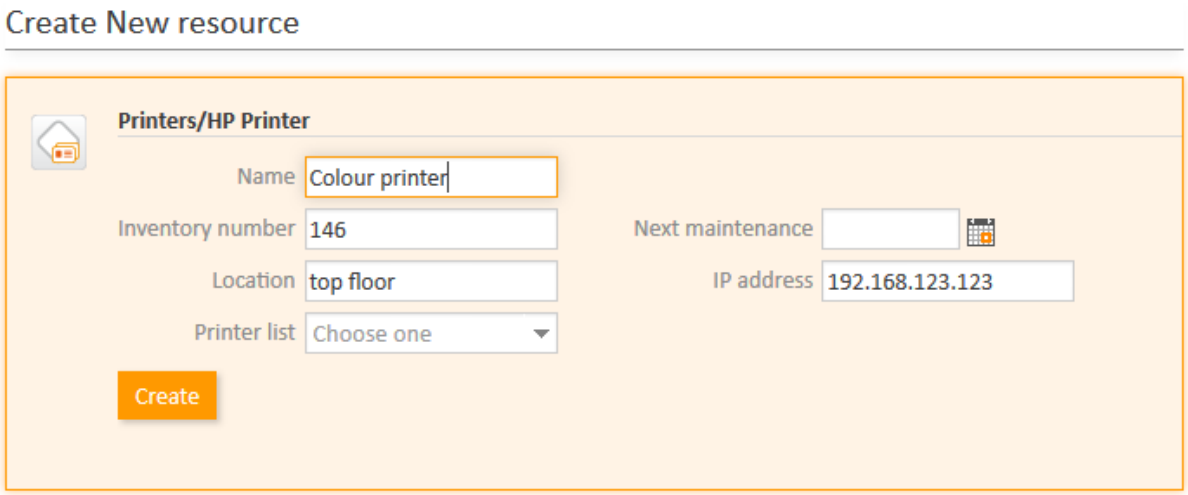


Figure 418: *ConSol CM Web Client - Create a new resource of type HP Printer, step 2*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

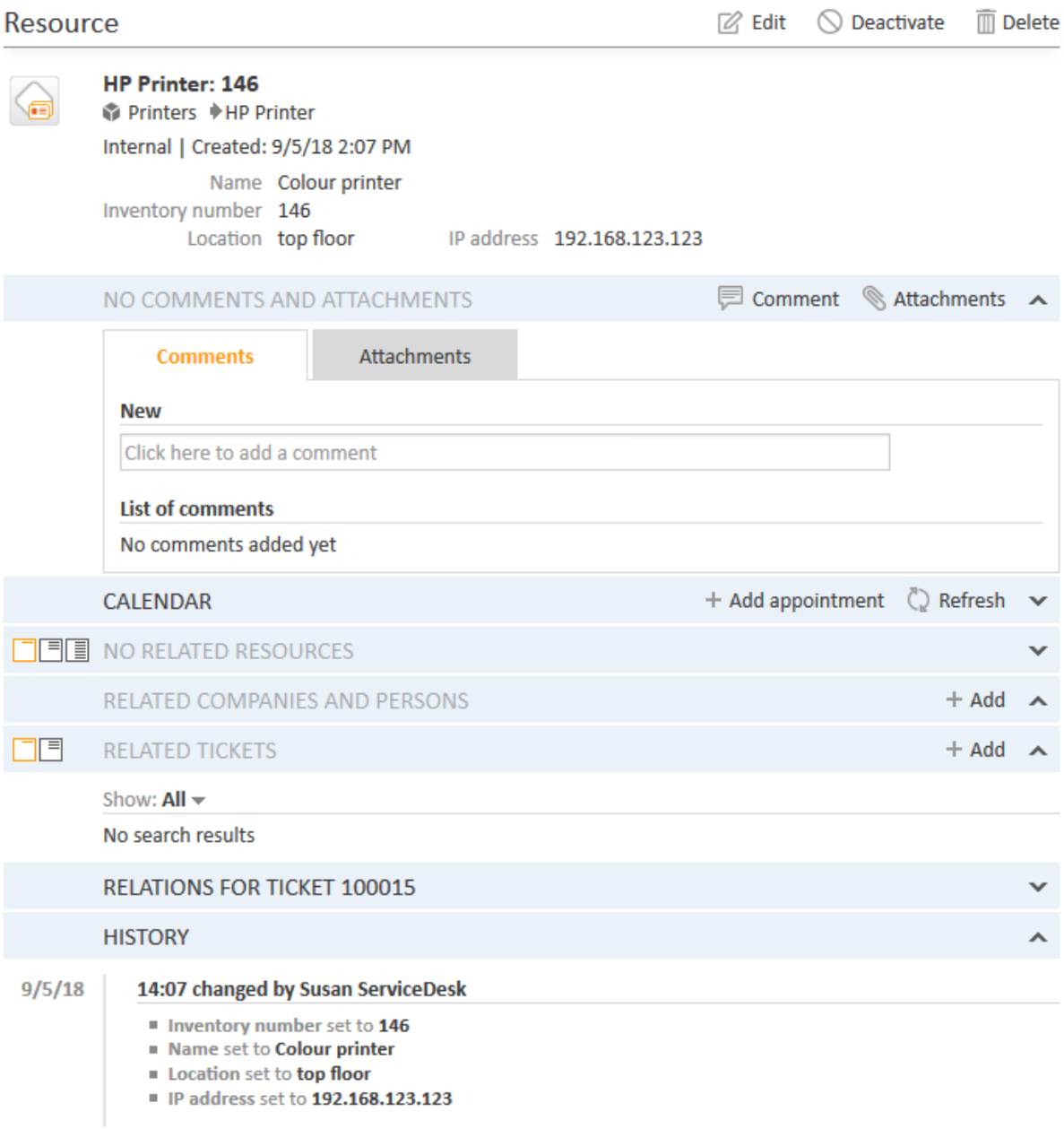


Figure 419: *ConSol CM Web Client - Resource page: Create a new resource of type HP Printer, step 3*

Now you have created a new *HP Printer* object, i.e., you have registered the printer #123 (located on the top floor). The resource page of this specific printer is displayed (see figure above).

For a detailed introduction to the work with CM/Resource Pool using the Web Client, please refer to the *ConSol CM User Manual*. Here we will only provide short explanations to show you, as an administrator, what happens when you work with the Admin Tool configuration for the ConSol CM resources.

How You, as an Administrator, Can Manipulate the Layout of the Resource Pool Dashboard Section *Search or create Resources*

- The roles of an engineer determine whether he has access permissions to a resource type. Hence, for the engineer currently logged in, only the resource types are displayed to which he has access permissions.

- The name of a resource type is displayed as a hyperlink which leads to the resource type page. This hyperlink is only active if the engineer has the access privilege READ for the respective resource type.
- The Link *Create new* for a resource type is only displayed if the engineer has the access permission CREATE for this resource type.

How You, as an Administrator, Can Manipulate the Layout of the Resource Page

- In the top section of the resource page, the localized values of the resource fields are displayed (analog to ticket data of ticket fields in a ticket). Data of resource fields may also be displayed in tabs which represent resource field groups (also analog to tabs on a ticket page which represent ticket field groups). The resource fields are displayed according to the layout you have defined. See section CM/Resource Pool - Setting Up the Basic Resource Model for details.
- The icon for each resource of the resource type can be defined in the Admin Tool.
- The template for the display of the resource name is defined for the resource type. Here on the resource page, the *Default* template is used, see section CM/Resource Pool - Templates for Resource Data for details.
- The menu entries (links) in the top right corner of the page are only visible if the engineer has the required access permissions:
  - *Edit* link: WRITE permission
  - *Deactivate* link: DEACTIVATE / ACTIVATE permission
  - *Delete* link: DELETE permission
- If resource actions are defined for resources of this type (not shown in the figure above), they will be available on the resource page as well, just like workflow activities for tickets. See section CM/Resource Pool - Resource Actions for details. However, an engineer will only see the resource actions if he has the permission ACT for the respective resource type.
- For an engineer, the section *Comments and Attachments* might not be displayed if his roles do not have the permission *Details read*. See section CM/Resource Pool - Assigning Permissions for Resources.
- If relations to other resources, contacts or companies, or to tickets have been defined in the Admin Tool, relations might be displayed in the respective sections of the resource page if real relations have been created for the resource by an engineer. See section CM/Resource Pool - Resource Relations for details.

G.1.3.3 Working with the Resource Type Page

The resource type page provides an overview of all resources of one specific resource type. You can open the resource type page by opening the Resource Pool Dashboard and clicking on the name of the resource type.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

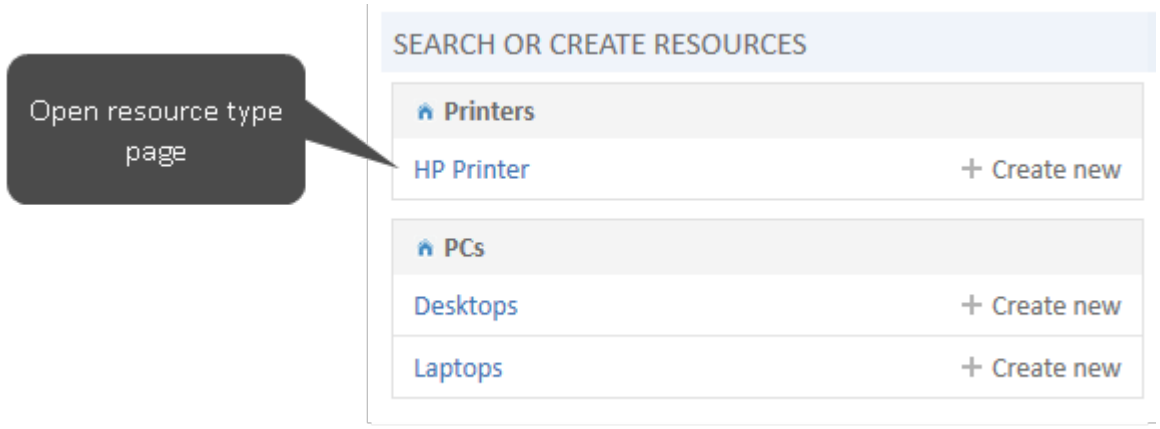


Figure 420: *ConSol CM Web Client - Opening a resource type page*

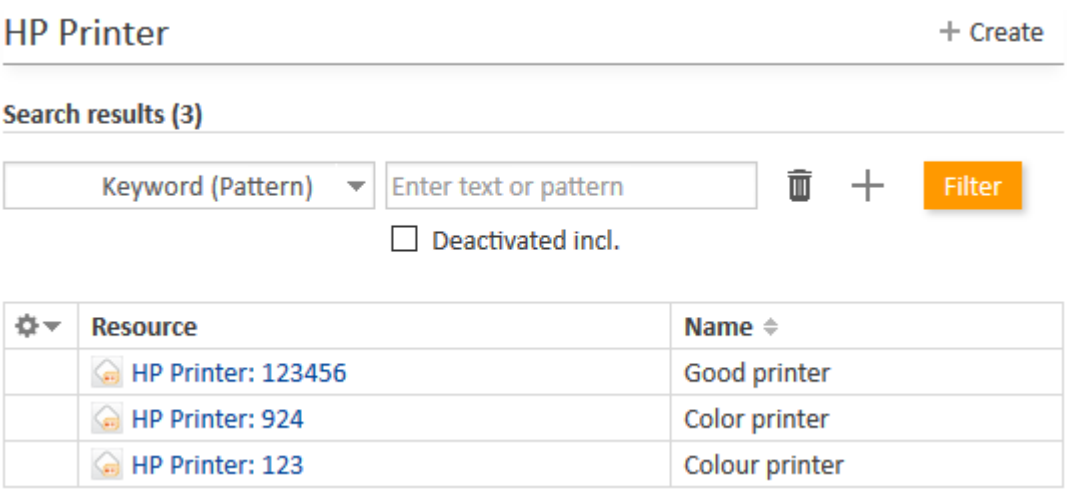


Figure 421: *ConSol CM Web Client - Resource type page*

On the resource type page, all resources of this type are listed. They can be filtered by the values of the resource fields using the search in the top section of the page.

How You, as an Administrator, Can Manipulate the Layout of the Resource Type Page

- In the drop-down menu for the search, only fields which have been indexed are available for selection. See section Search Configuration for details.
- The column headers represent the localized values of the names of the resource fields which an admin has defined. See section CM/Resource Pool - Setting Up the Basic Resource Model for details.

G.1.3.4 Setting a Resource as Favorite

Like tickets, searches and customers, resources can also be placed in the Favorites section using drag-and-drop, e.g., on the resource page.

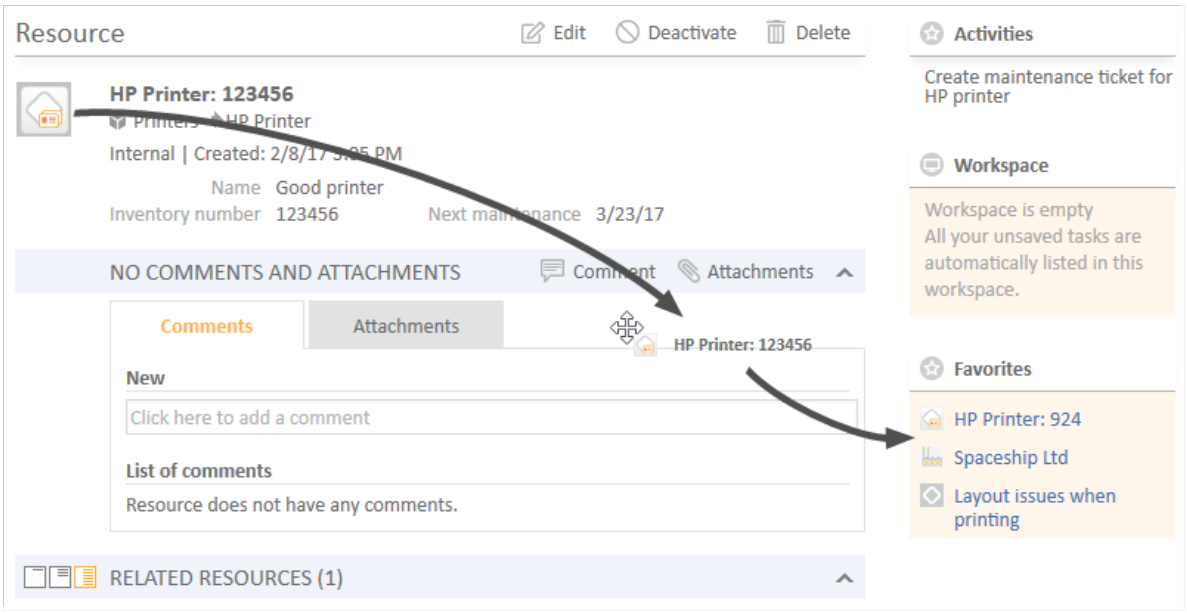


Figure 422: *ConSol CM Web Client - Resource page: Using drag-and-drop to put a resource into the Favorites section*

How You, as an Administrator, Can Manipulate the Display Format of the Resource Concerning Drag-and-Drop to Favorites

- The format for the display of the resource in the Favorites section is based on the *Default* template, as described in section CM/Resource Pool - Templates for Resource Data.
- The format for the display of the resource in the drag-and-drop box is based on the *Default* template, as described in section CM/Resource Pool - Templates for Resource Data.

G.1.3.5 Defining and Using Resource Relations

The same principle as for resource categories and resource types also applies to resource relations:

- First, the resource relation has to be defined in the data model using the Admin Tool
- Then, the real relations between resources (objects of a certain resource type) and other objects (tickets, customers or other resources) have to be created using the Web Client.

This is explained in detail in section CM/Resource Pool - Resource Relations. A short example:

To link a resource (e.g., a PC desktop) to tickets in the queue *ServiceDesk*, first, define this relation type (i.e., resource-ticket) in the Admin Tool.

Then an engineer can create a relation for a PC to a ticket, e.g., when an incident ticket involving this machine is created.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

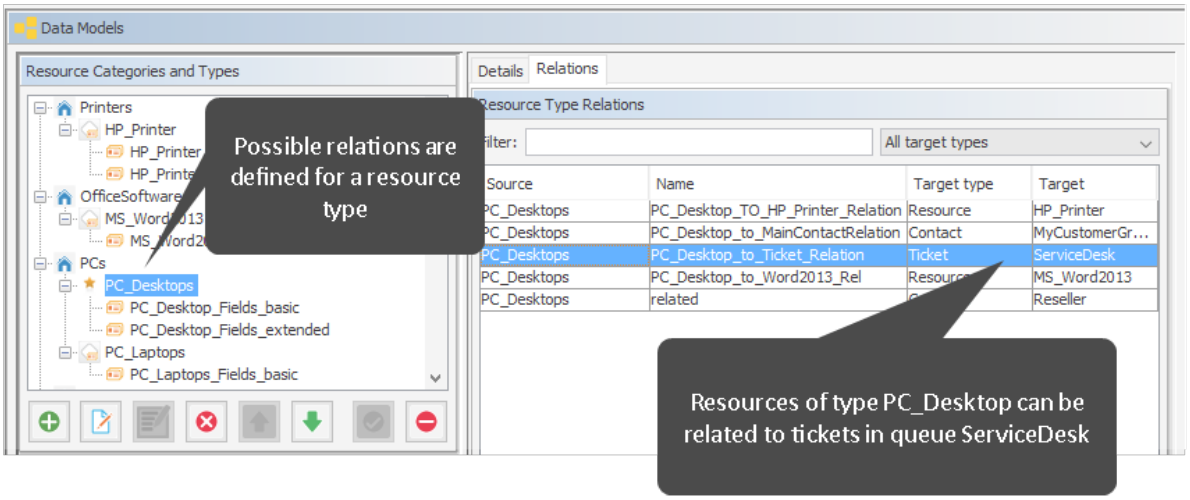


Figure 423: *ConSol CM Admin Tool - Resources, Data Models: Defining a new resource relation*

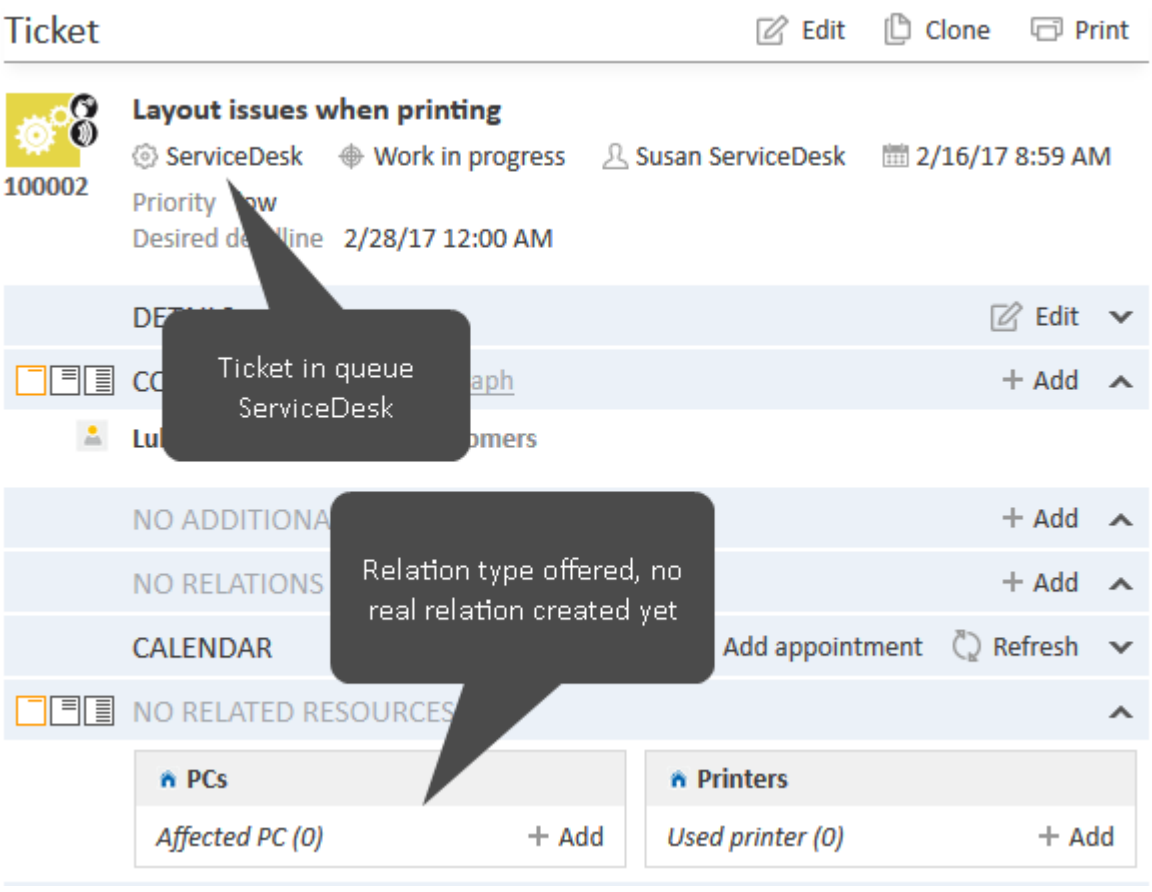


Figure 424: *ConSol CM Web Client - Ticket where a relation to a PC can be created*

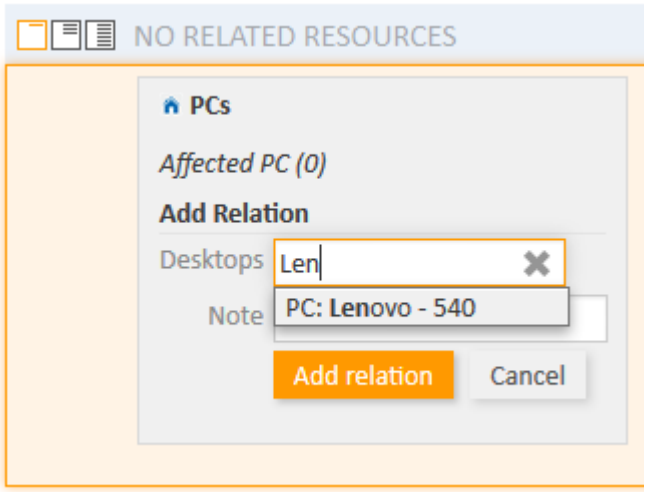


Figure 425: *ConSol CM Web Client - Creating a relation between a ticket and a PC*

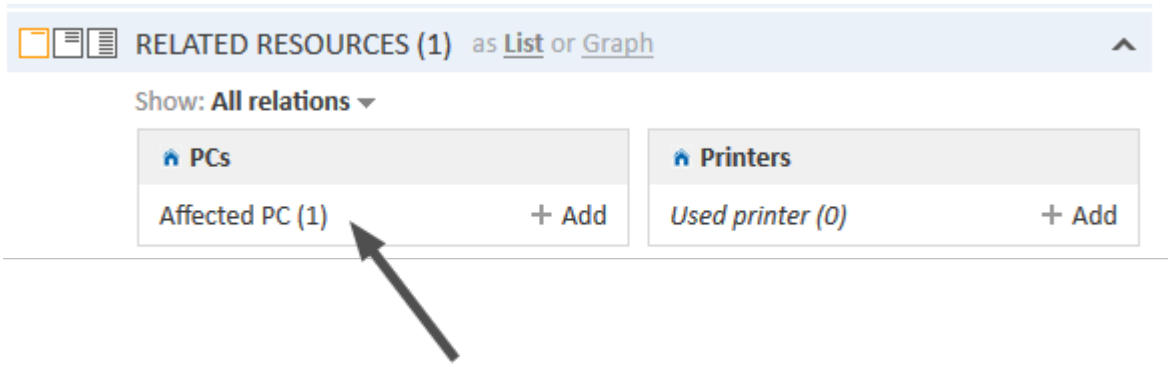


Figure 426: *ConSol CM Web Client - Ticket with relation to a PC*

When the relation has been created, the counter of the relation is set to 1. The engineer can then display details of the relation by clicking on the relation name.

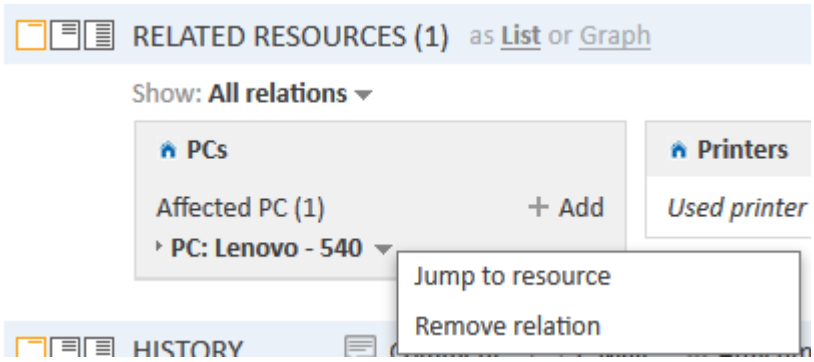


Figure 427: *ConSol CM Web Client - Details of a related resource and context menu*

The context menu of the related resource offers two options:

- **Jump to resource**
  Opens the resource page of the related resource.
- **Remove relation**
  Removes the relation of the ticket with the resource.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

If there is at least one relation, the *Show* filter is displayed, where you can choose to see all possible relations (*All relations*) or only the relations which are set for the resource (*Only assigned relations*).

How You, as an Administrator, Can Manipulate the Display of Resource Relations

- The name and description of the relation which is displayed is the localized value of the relation name and description which you defined using the Admin Tool. Please note that there is one description for the source side and one for the target side of a relation.
- A note for a resource relation can only be added in the Web Client if the check box *Note field is available* has been set for the resource relation definition in the Admin Tool.
- When one resource of a type has been added to an object (e.g., to a ticket), the *Add* option (e.g., to add another resource of the type PC to the ticket) will only be available when the cardinality of the relation is set to *many-to-one, one-to-many* or *many-to-many*.

G.1.3.6 Defining and Using Resource Actions

Resource actions also have to be defined using the Admin Tool. Each resource action is based on an Admin Tool script.

Resource actions are explained in detail in section CM/Resource Pool - Resource Actions. Here, only a short example is provided.

A resource action is always defined on the navigation item *Actions* in the navigation group *Resources* and the assigned to one or more resource type(s). For example, an action is defined which should offer the possibility to create a new ticket in the queue ServiceDesk, directly from the resource page. Thus, when a customer calls to complain about a problem with the PC, the Service Desk agent can start by opening the PC's detail page for information about this asset. If necessary, the agent can directly create a new Service Desk ticket for the customer.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

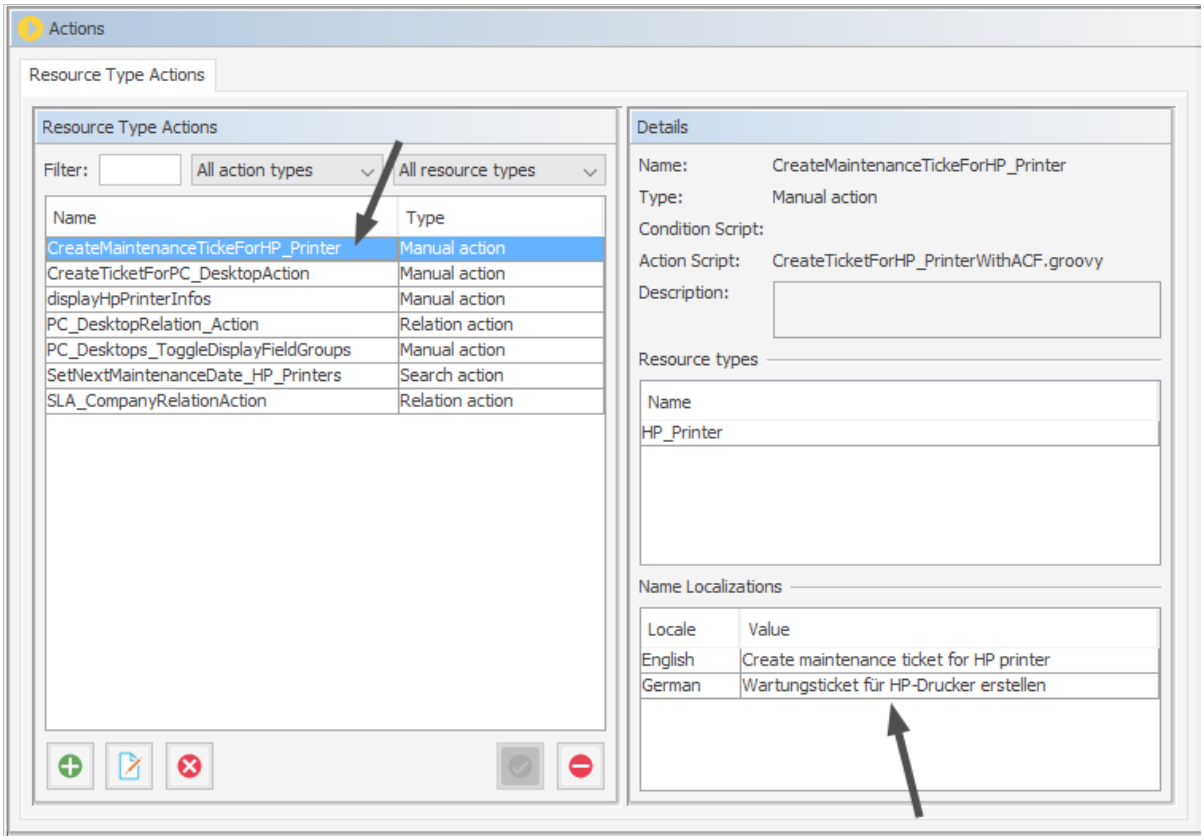


Figure 428: *ConSol CM Admin Tool - Resources, Actions: Definition of a resource action*

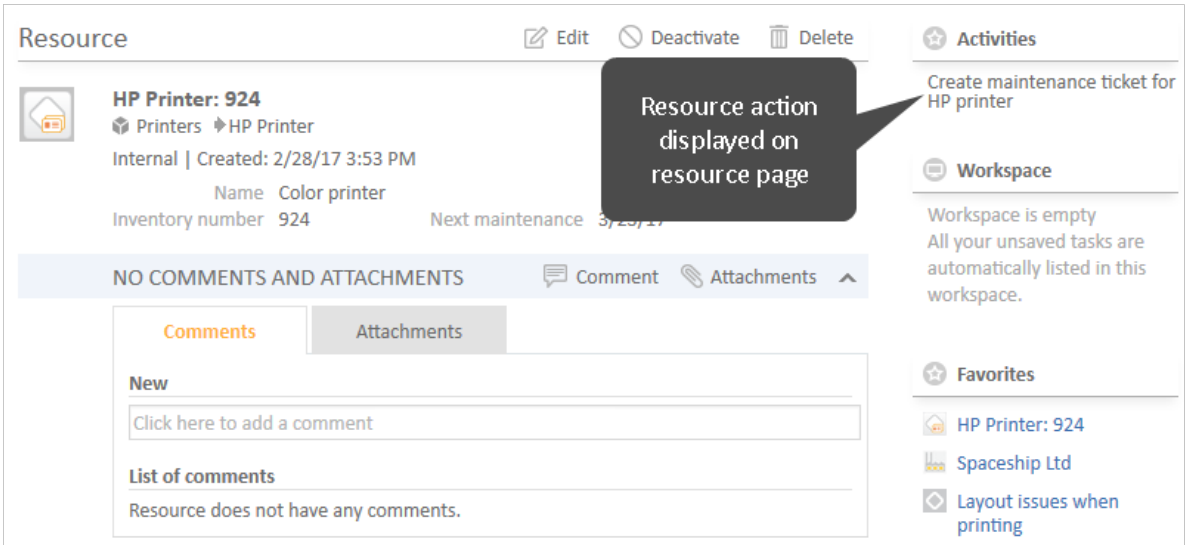


Figure 429: *ConSol CM Web Client - Resource activity (based on resource action)*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

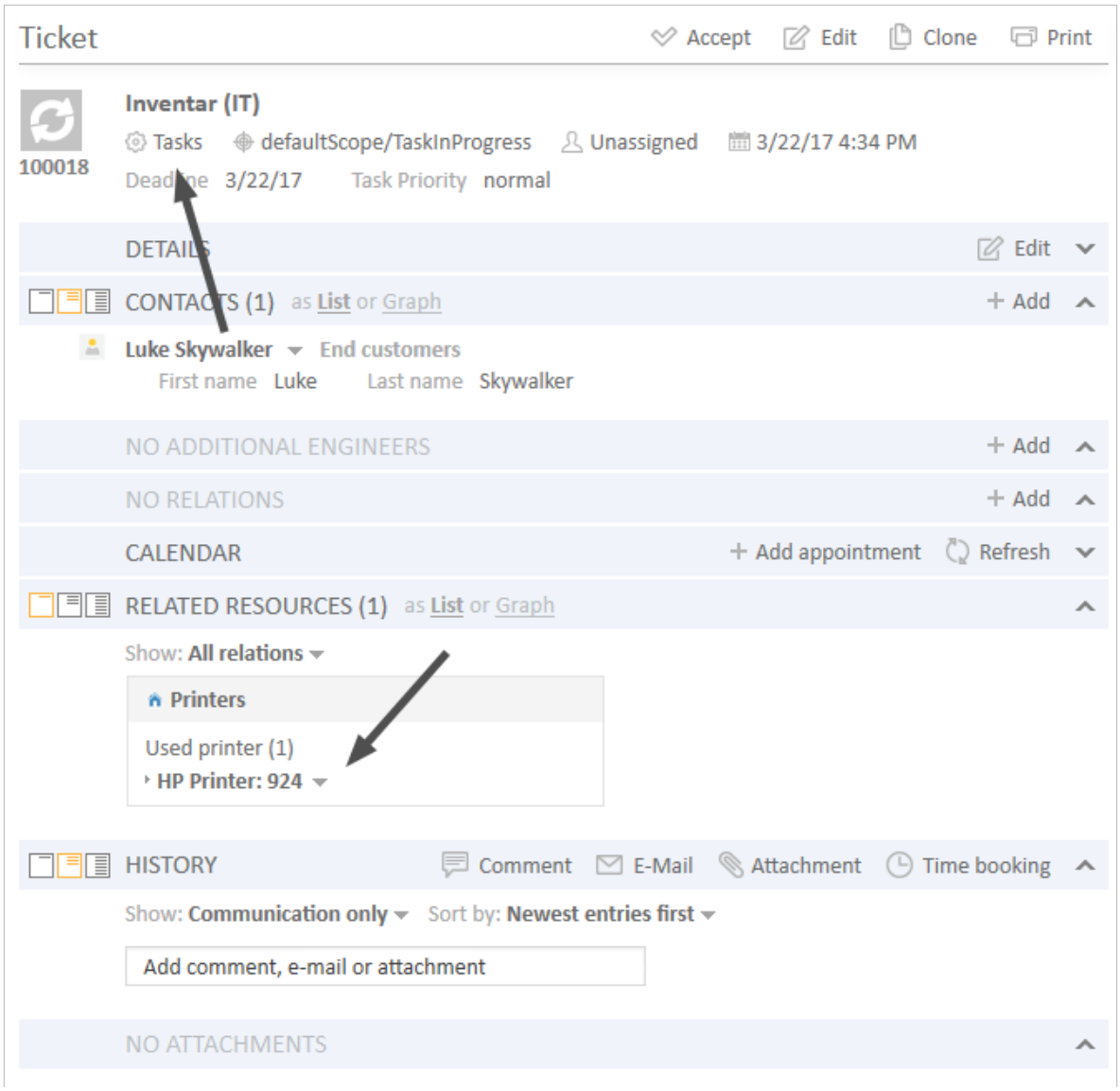


Figure 430: *ConSol CM Web Client - Task ticket created by a resource action*

How You, as an Administrator, Can Manipulate the Display of Resource Actions

- The name of the resource activity which is displayed in the Web Client is the localized name of the resource action which an administrator has defined in the *resource actions* panel. For details see section CM/Resource Pool - Resource Actions.
- The localized value of the description of the resource action will be displayed as mouse-over text for the resource activity in the Web Client.

G.1.3.7 Using the Quick Search to Find Resources

In the result list of the Quick Search, resources are displayed in a distinct section:

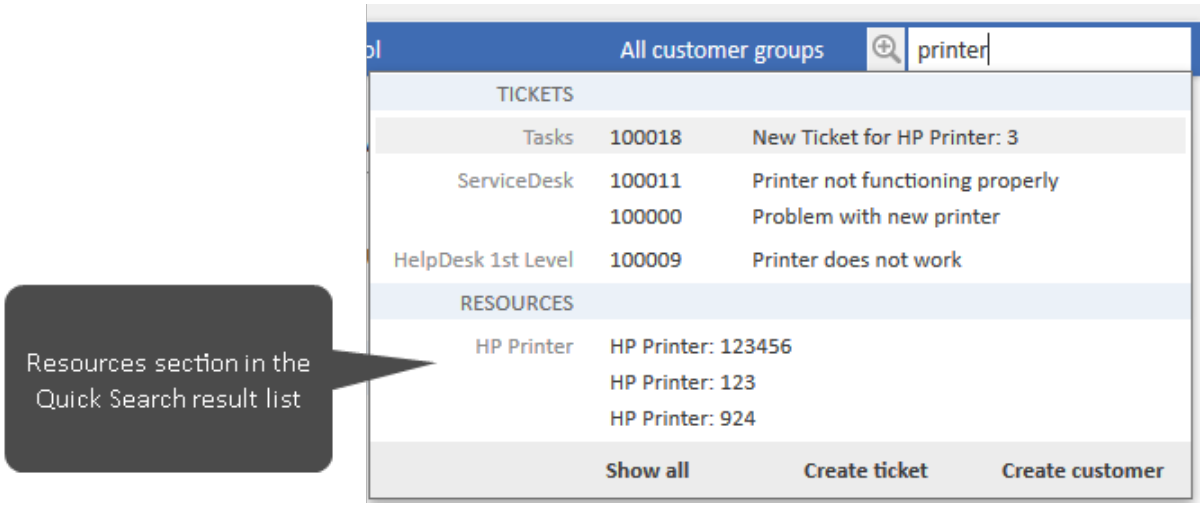


Figure 431: *ConSol CM Web Client - Result list of Quick Search with distinct resource section*

How You, as an Administrator, Can Manipulate the Display of Resources in the Quick Search

- The format for the display of the resource in the Quick Search result list section is based on the *Quick Search* template. For details see section CM/Resource Pool - Templates for Resource Data.
- The value of the system property cmweb-server-adapter, globalSearchResultSizeLimit defines the maximum number of hits displayed in the result list.

G.1.3.8 Using the Detailed Search to Find Resources

In the Detailed Search, an engineer can search for all resources of a certain type, provided that the respective resource field is indexed (the annotation `field indexed` has been set, usually to the value “transitive”).

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

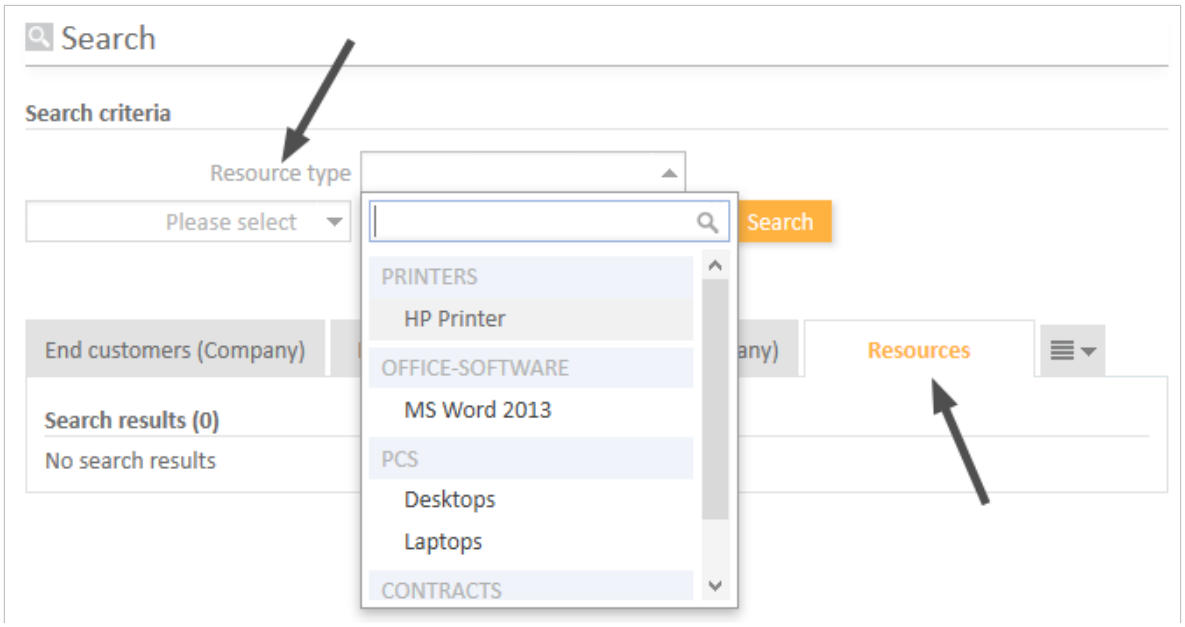


Figure 432: *ConSol CM Web Client - Detailed Search for resources*

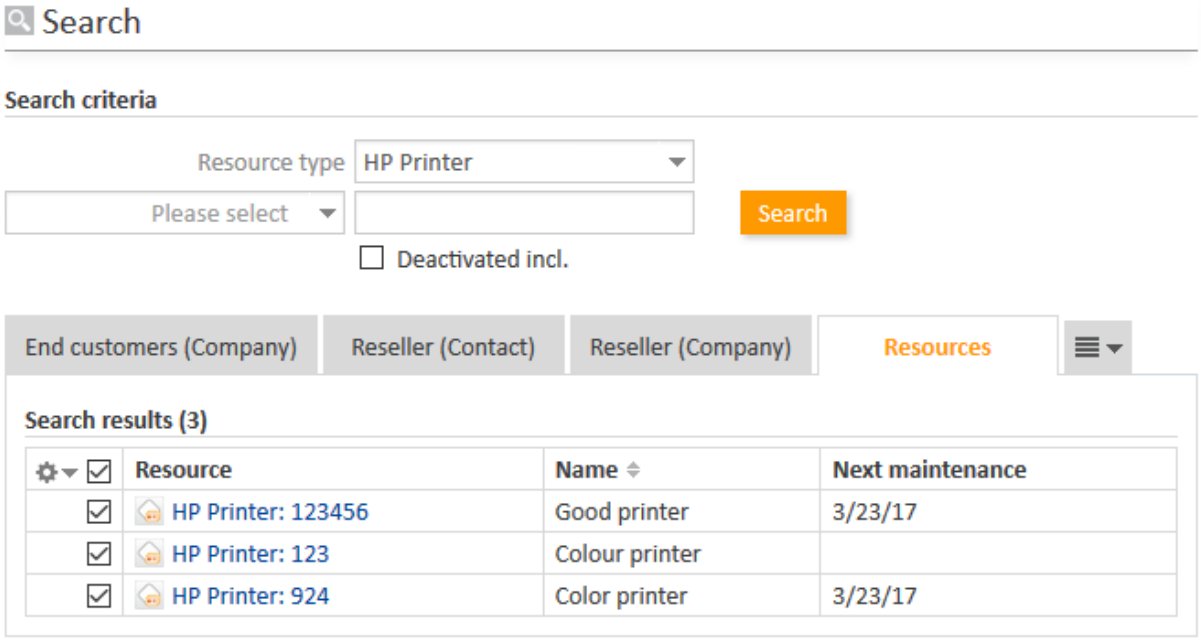


Figure 433: *ConSol CM Web Client - Result list of a Detailed Search for resources*

How You, as an Administrator, Can Manipulate the Display of Resources in the Detail Search

- On the Detailed Search page, in the drop-down menu where resource types are listed, a resource type will only appear if the respective resource field has been annotated with `field indexed` (usually with the value “transitive”).
- The value of the system property cmweb-server-adapter, searchPageSize defines the number of hits which are initially displayed

If search actions are defined for a certain resource type, the engineer will see the possible actions as Activities in the Web Client in each search result list where resources of this type are listed. Please refer to section Action Framework - Search Actions for details about this topic.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.1.4 CM/Resource Pool - Setting Up the Basic Resource Model

G.1.4.1 Introduction

This chapter will guide you through the complete set-up of a resource model. You will learn how to

- create a resource category
- create two resource types within this resource category
- create data fields for the resources, i.e., create resource field groups for the resource types
- create resource fields within the resource field groups. i.e., create the data fields
- define templates for the display format of the resource names in the Web Client

In our example, you are an administrator who has to set up a resource model for a company which wants to manage their IT assets, as well as their products, using the Resource Pool.

The complete model will look like the one in the following figure. We will show you how to define the Printers and the Office software to demonstrate the basic principles. Then you should be able to configure all remaining resources yourself.

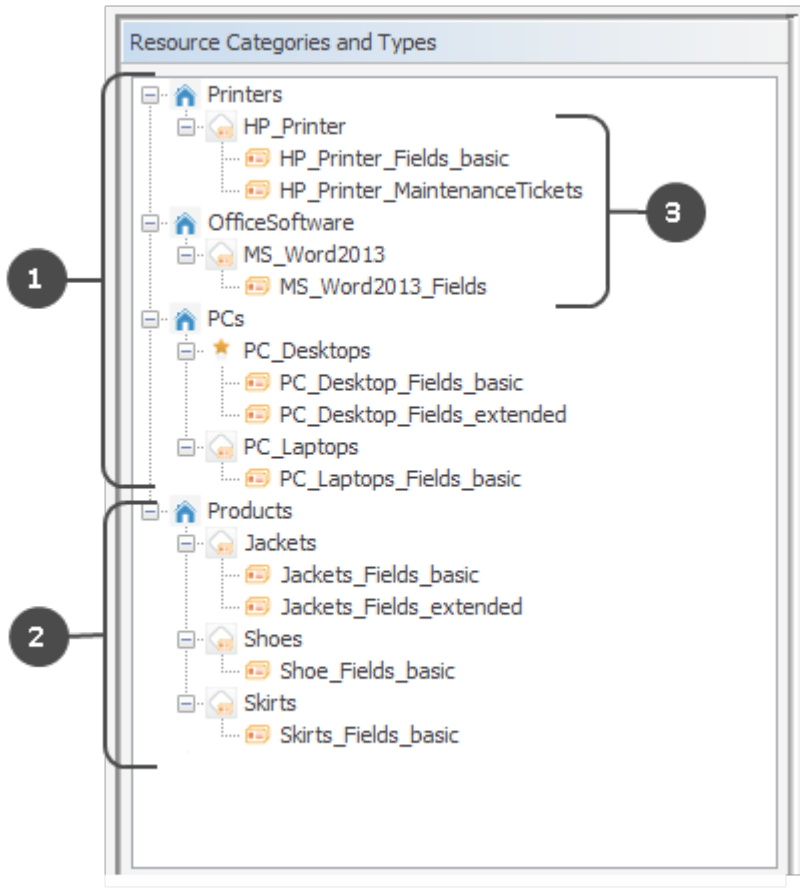


Figure 434: *ConSol CM Admin Tool - Resources, Data Models: Example of a resource model*

The above figure shows a resource pool data model with the following elements:

- Resources for IT management (1)
- Resources for product management (2)

The following example covers printers and office software (3).

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.1.4.2 Creating a Resource Category with the First Resource Type

A resource category represents the objects at the highest hierarchical level in the data model of the Resource Pool. In our example, all printers are managed in a resource category *Printers* and all office software packages are managed in a resource category called *OfficeSoftware*. The following figure shows some resource categories and resource types in the Web Client. This should provide you with an overview of the configuration in the Admin Tool and its consequences in the Web Client. Please keep in mind that the technical object names which are used in the Admin Tool are usually not the (localized) names which are displayed in the Web Client.

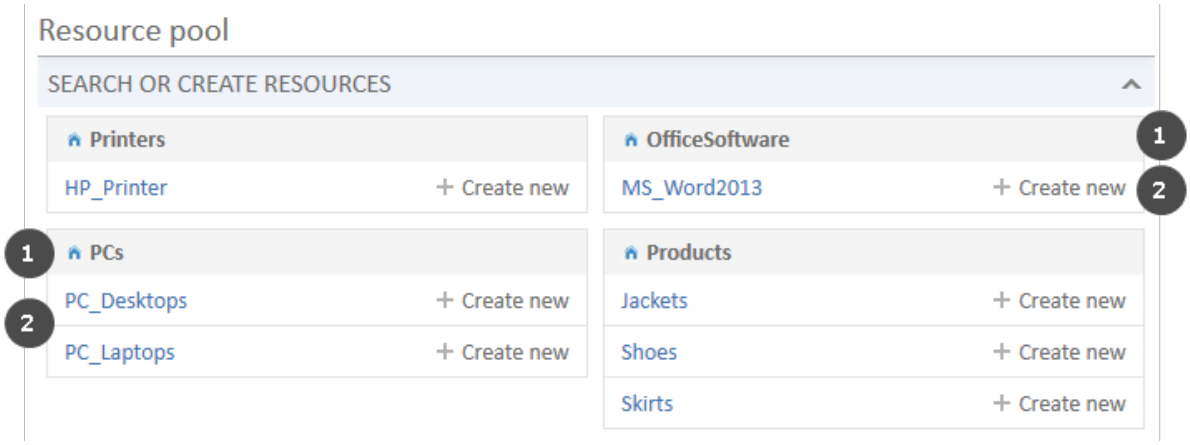


Figure 435: *ConSol CM Web Client - Resources in the Resource Pool Dashboard*

- Resource categories (1)
- Resource types (2)

The first step in a new (empty) resource model is to create a new resource category which contains one resource type. The resource type contains one resource field group.

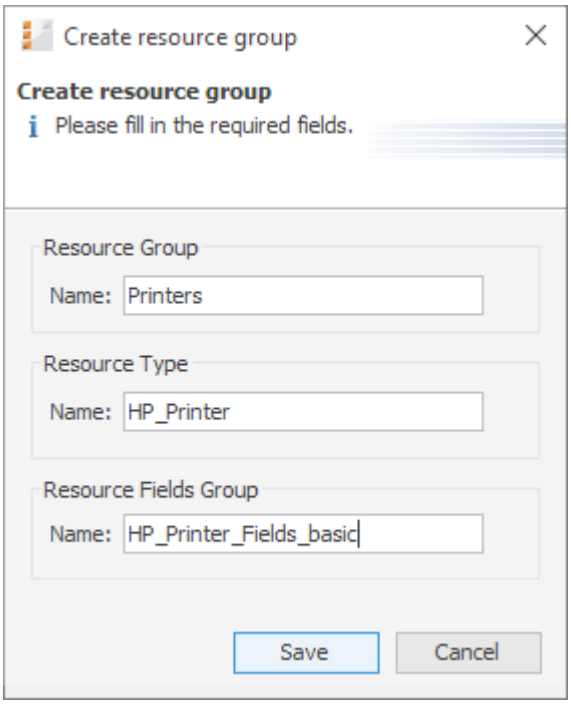


Figure 436: *ConSol CM Admin Tool - Resources, Data Models: Creating a new resource category*

To create a resource category, fill out the following fields. The localization for the names of resource category, resource type and resource field group is done using the fields on the main *Data Models* tab and is explained in section Localization of Data Fields. The descriptions are localized using the *Localize* button which is explained in section Localization of Objects in General, Type 1.

- **Resource category**, Name
  The technical name of the resource category.
- **Resource type**, Name
  The technical name of the first resource type within the resource category.
  More resource types for the resource category can be added later on.
- **Resource field group**, Name
  The name of the first resource field group (similar to a ticket field group for ticket data) within the resource type. In case the resource field group is displayed in the Group section of the resource page (annotation `show-in-group-section` = “true”), the localized name will be displayed as header of the respective tab.
  More resource field groups for the same resource type can be added later.

Click *Save* to create the resource category, type and first field group.

Once you have defined the resource category, you can add some more data for the created objects.

Editing the Resource Category

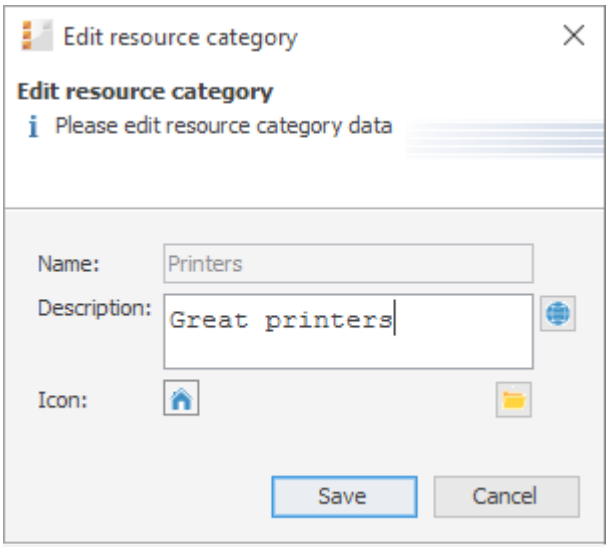


Figure 437: *ConSol CM Admin Tool - Resources, Data Models: Editing a resource category*

Mark the resource category and click the *Edit* button to edit some resource category parameters:

- **Description**
  You can enter the description directly in the pop-up window, which will become the default description, or you can use the *Localize* button to enter localized values for each language. The resource category description is displayed only in the Admin Tool, not in the Web Client.
- **Icon**
  Select one of the ConSol CM standard icons by clicking the icon, or use the file browser to upload an icon of your choice. The resource category icon will be displayed in the Web Client next to the name of the resource category. In this way, you can, for example, select a Printer icon for printers, a PC icon for PCs, etc. Allowed image file formats are: jpg, png, gif. We

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

recommend to use 32 x 32 px as image size to achieve a reasonable size of the icon on the resource type page. In tables and at other locations on the GUI where the resource is displayed, the icon is re-sized automatically.



Figure 438: *ConSol CM Web Client - Customized icon for a resource category*

Click *Save* to commit your changes.

Editing the Resource Type

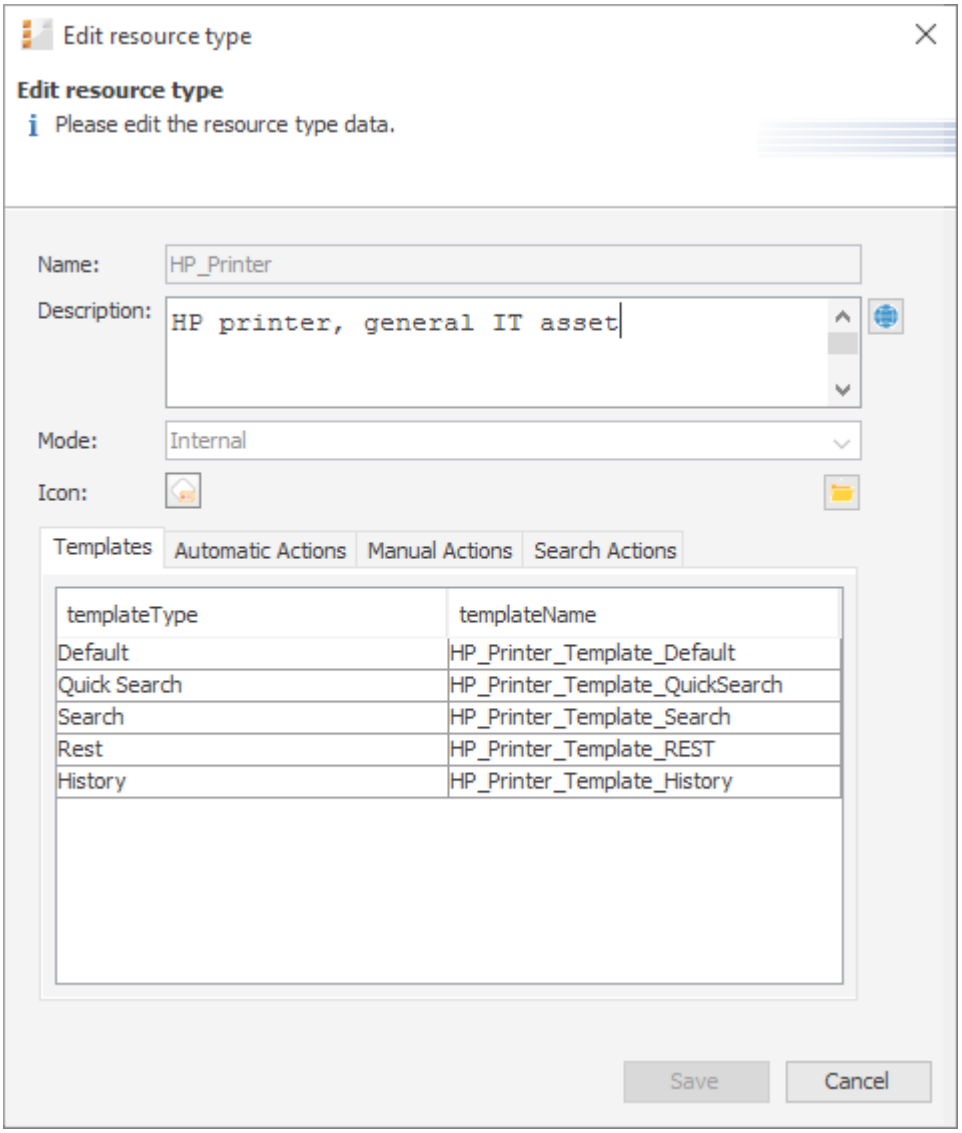


Figure 439: *ConSol CM Admin Tool - Resources, Data Models: Editing a resource type*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Mark the resource type and click the *Edit* button to edit the following resource type parameters:

- **Description**
  You can enter the description directly in the pop-up window, this will become the default description. Or you can use the *Localize* button to enter localized values for each language. The resource type description is displayed as information in the Admin Tool only, not in the Web Client.
- **Mode** (for a detailed explanation of all modes, refer to section Information about Resource Modes)
  - **Internal**
    Resource data is fully managed by ConSol CM. For details see section Internal Resources.
  - **On the fly**
    Resource data are retrieved from an external backend system whenever requested (available in ConSol CM version 6.10.3.0 and higher). For details see section On the fly Resources.
  - **Cached**
    Resource data come from an external backend system and are cached in ConSol CM. A refresh of an item can be triggered manually (available in ConSol CM version 6.10.3.0 and higher). For details see section Cached Resources.
  - **Imported**
    Resource data come from an external backend system and are stored in ConSol CM. An update requires a new bulk data import (available in ConSol CM version 6.10.3.0 and higher). For details see section Imported Resources.
- **Icon**
  Select one of the ConSol CM standard icons by clicking the icon, or use the file browser to upload an icon of your choice. The resource type icon will be displayed in the Web Client next to the name of the resource type, e.g., on the resource type page. In this way, you can, for example, select a Printer icon for printers, a PC icon for PCs etc. Allowed formats are: jpg, png, gif. We recommend to use 32 x 32 px as image size to achieve a reasonable size of the icon on the resource page. In tables and at other locations on the GUI where the resource is displayed, the icon is re-sized automatically.
- **Templates**
  Here you can define the templates for the display format of resource data in the Web Client (analog to templates for customer data). This is explained in detail in section CM/Resource Pool - Templates for Resource Data.
- **Automatic Actions**
  Here you can assign automatic resource actions to the resource type. The resource actions have to be defined first, using the *resource actions* section of the Admin Tool. See section CM/Resource Pool - Resource Actions for a detailed explanation.
- **Manual Actions**
  Here you can assign manual resource actions to the resource type. The resource actions have to be defined first, using the *resource actions* section of the Admin Tool. See section CM/Resource Pool - Resource Actions for a detailed explanation.
- **Search Actions**
  Here you can assign search actions to the resource type. The resource search actions have to be defined first using the *Scripts and Templates* section of the Admin Tool. See section Action Framework - Search Actions for a detailed explanation.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

The resource relations are also defined for a resource type. This is described in detail in the section CM/Resource Pool - Resource Relations.

Information about Resource Modes

The mode of a resource type defines whether the resource data is stored in the ConSol CM system and - if required - how the transfer from the external system to ConSol CM is managed. Once the mode has been set and resources of the respective type are present in the system, the mode cannot be changed!

In ConSol CM, you have to distinguish between *internal resources* and *external resources*. The latter comprise *On the fly Resources*, *Cached Resources* and *Imported Resources*, i.e., all resources which are not completely stored in and managed by ConSol CM.

Only internal resources can be created using the Web Client. External resources can be neither created nor modified (permanently) nor deleted using the Web Client. For all external resources the complete data model has to be defined in the Admin Tool before the first data transfer can be performed. The use of *On the fly Resources* and *Cached Resources* is based on the implementation of a specific interface. Therefore, those resources can only be used with a customer-specific ConSol CM *.ear* file. The respective development project has to be built and deployed before the first data transfer takes place. The following sections provide a detailed overview of all four resource types.

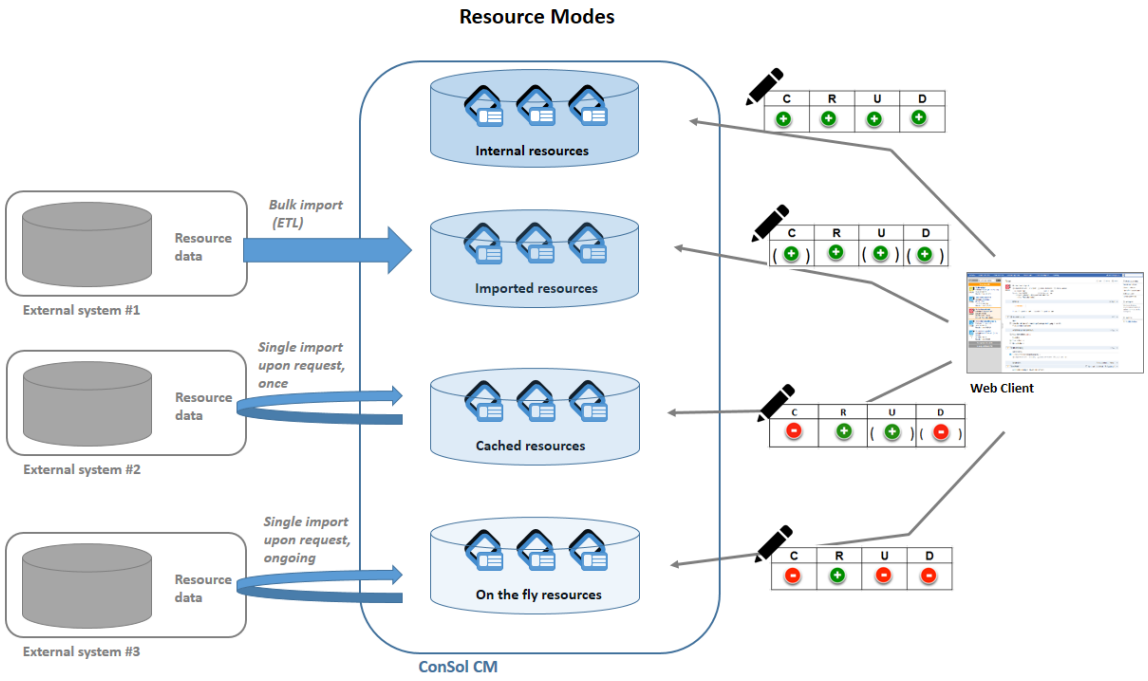


Figure 440: *Resource modes*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

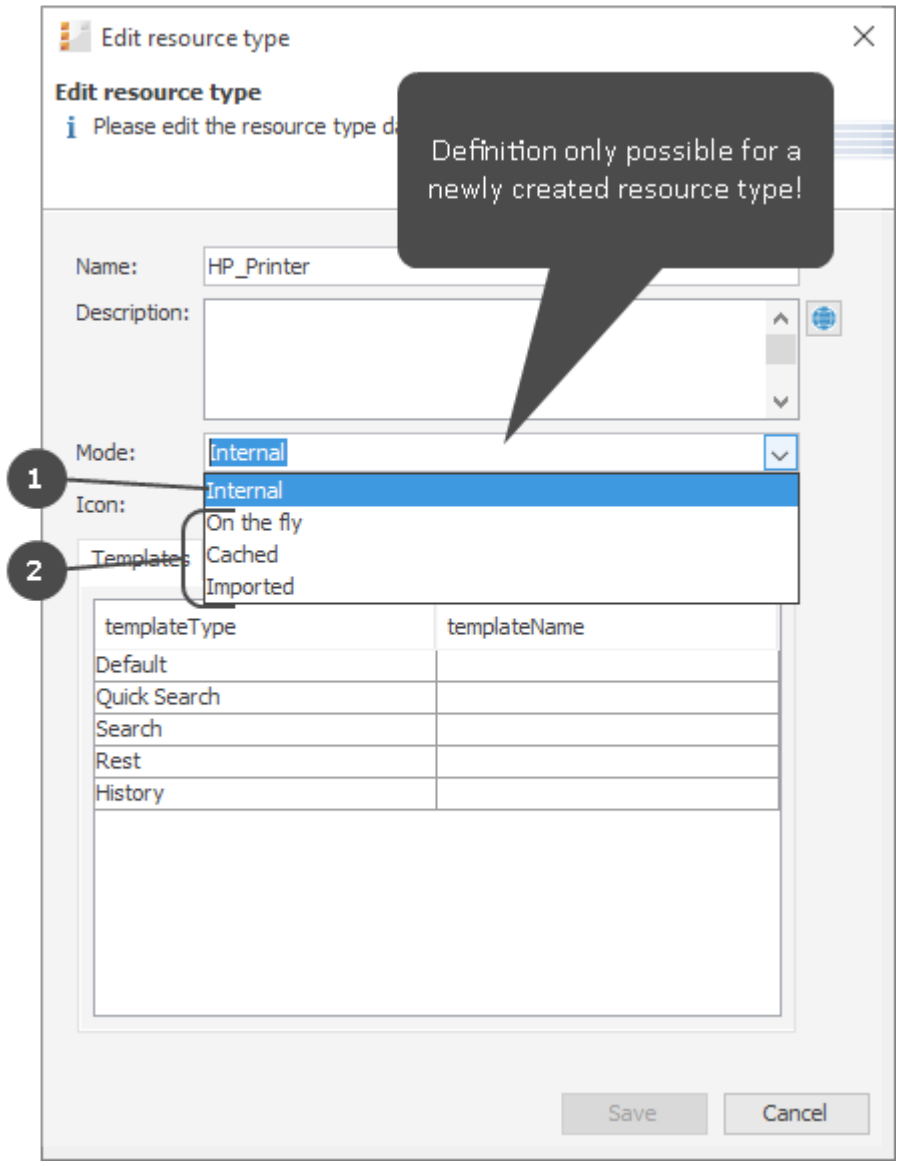


Figure 441: *ConSol CM Admin Tool - Resources, Data Models: Configuration of resource type mode*

- (1) Internal
- (2) External

Internal Resources

The resource data is fully managed by ConSol CM. Engineers who have sufficient access permissions can use the Web Client to create, modify or delete resources.

On the fly Resources

The resource data is retrieved from an external backend system whenever requested. The mode *on-the-fly* relies on a customer-specific implementation of the Java interface `ResourceExternalSource`.

An *on-the-fly* resource does not have any data which is stored in ConSol CM except for an external ID. This ID is used to retrieve data from the external system upon request.

In the Web Client:

- The Detailed Search is not available.
- In order to search CM for an on-the-fly resource, only the external ID can be used.
- There is no resource page.
- There is no resource type page
- The resource is only available/visible in the relations section of
  - tickets
  - customers
  - other resources.
- The resource data cannot be edited.

Cached Resources

The resource data originates from an external backend system. The data is transferred into ConSol CM upon the first request and is then cached in the ConSol CM database. The first request to the external system is executed when the resource is linked to another object (a ticket, a customer or another resource), i.e., when a relation is established. The mode *Cached* relies on a customer specific implementation of the Java interface `ResourceExternalSource`.

In the Web Client:

- The Detailed Search is only available for resources that have been requested at least once and are thus present in ConSol CM with their full data set.
- For resources which are already present in ConSol CM (i.e., are cached)
  - a resource page is available.
  - a resource type page is available.
  - a refresh of the resource data can be triggered manually on the Resource Page.
- In order to search ConSol CM for a *cached* resource which has not yet been requested, only the external ID can be used.
- There are no Resource Pages for resources which have not yet been cached.
- The resource is only available in the *Relations* section of tickets, customers or other resources. The resource fields are only displayed in the *Relations* section as well.
- The resource data cannot be edited.

Imported Resources

The resource data originates from an external backend system and is stored in the ConSol CM database. Once the data has been imported into ConSol CM, it is treated almost like internal data. The import into ConSol CM usually relies on an *ETL* (*Extract - Transform - Load*) job which is run once or on a certain schedule.

In the Web Client:

- The Detailed Search is fully available for imported resources.
- A resource page is available.
- A resource type page is available.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- Imported resource cannot be created using the Web Client (only via import).
- The resource data can be edited, but a new import might overwrite changes.



In order to integrate external resources into your CM system, you have to write a special Java / Groovy class which implements the interface `ResourceExternalSource` with the three methods

- PageResult<Resource> **searchByPattern**(ResourceType pType, String pPattern, int pPageSize, int pPageNumber)
- Resource **importResource**(Resource pResource)
- Resource **getByExternalId**(ResourceType pType, String pExternalId)

A detailed example for a mock class is provided in the *ConSol CM Release Notes* for version 6.10.3. You will have to implement the methods for your system to provide real data from your IT infrastructure.

Editing the Resource Field Group

A resource field group is a collection of resource fields, the data fields for resource data (analog to a ticket field group which is a collection of ticket fields for ticket data). Resource fields can never be managed as singular objects, they are always managed (e.g., faded in or out) as a resource field group.

A resource field group has the following group-specific parameters. Mark the desired resource field group and click *Edit* to edit those values (see the next section).

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Editing Resource Field Group-Specific Parameters

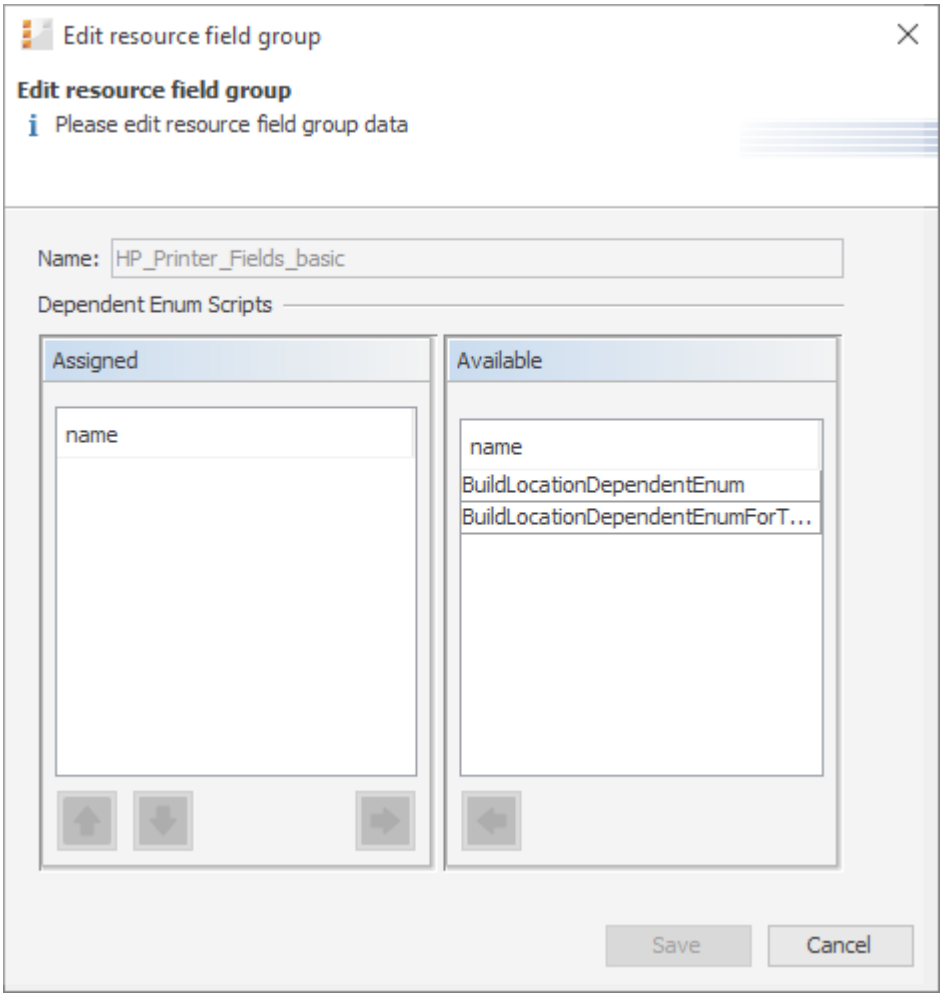


Figure 442: *ConSol CM Admin Tool - Resources, Data Models: Editing resource field group-specific parameters*

Here, you can edit the following fields:

- **Name**
  The technical name of the resource field group. This should only be changed if truly required, as changing it might have side effects on all scripts (workflow and Admin Tool) which use the resource category name. To change the term of the resource field group which is displayed in the GUI, use the mechanism which is explained in section Localization of Data Fields.
- **Dependent Enum Scripts**
  Here you can assign a Dependent Enum to the resource field group. The Dependent Enum has to be defined as an Admin Tool script first, as described in section Scripts of Type Dependent Enum.

Setting Resource Field Group Annotations

A resource field group can have annotations. Select the resource field group and click the *Annotations* button to set or unset annotations. The annotations are the same as for ticket field groups. All annotations are listed and explained in Annotations.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

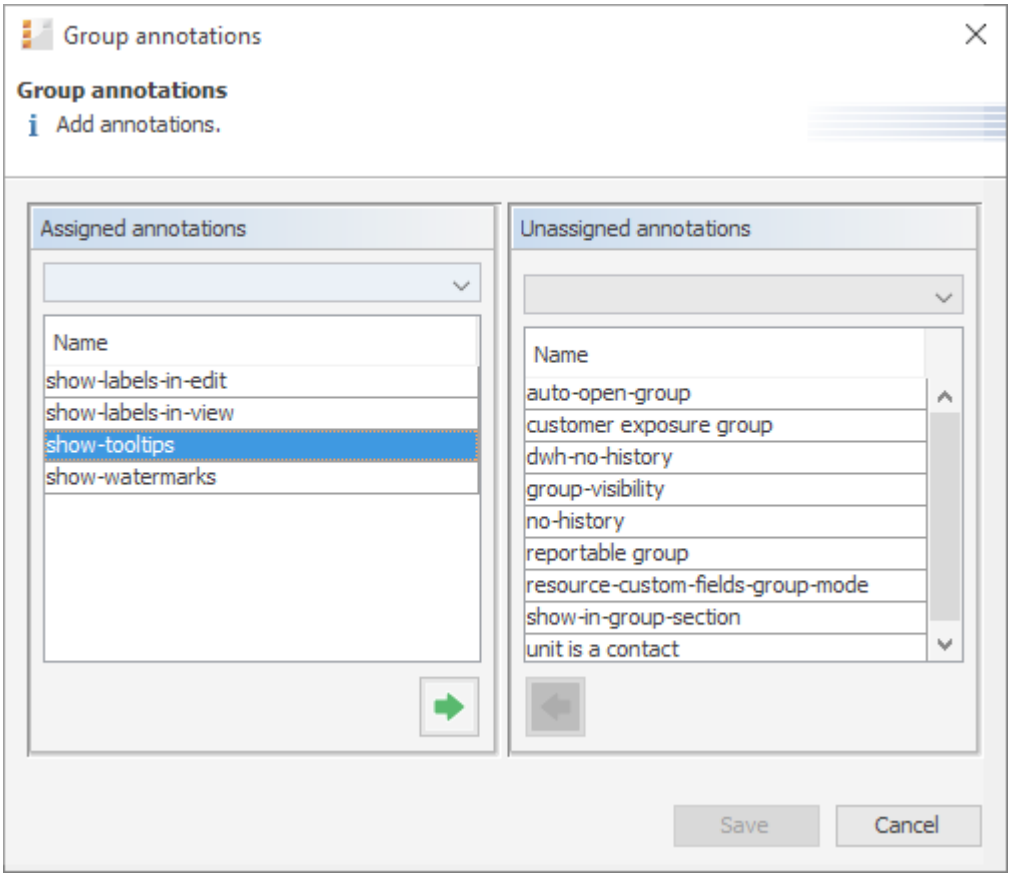


Figure 443: *ConSol CM Admin Tool - Resources, Data Models: Resource field group annotations*

Creating and Editing Resource Fields

Resource fields are containers for the data of resources (analog to ticket fields for ticket data). Resource fields always belong to a resource field group. To add or edit resource fields of a resource field group, select the group in the resource data model and enter all required data on the right-hand side of the Admin Tool.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

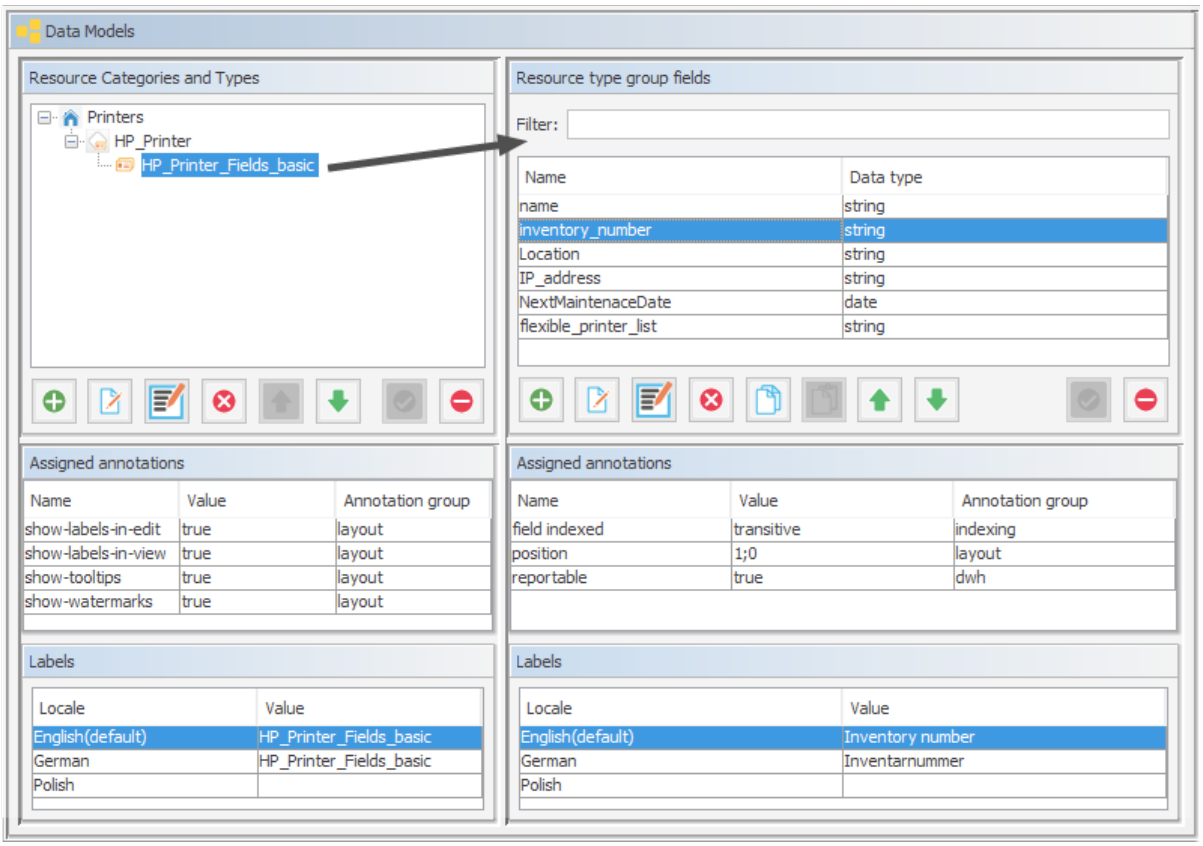


Figure 444: *ConSol CM Admin Tool - Resources, Data Models: Managing resource fields*

Creating a New Resource Field

To create a new resource field, click the *Add* button and enter the required data in the pop-up window.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

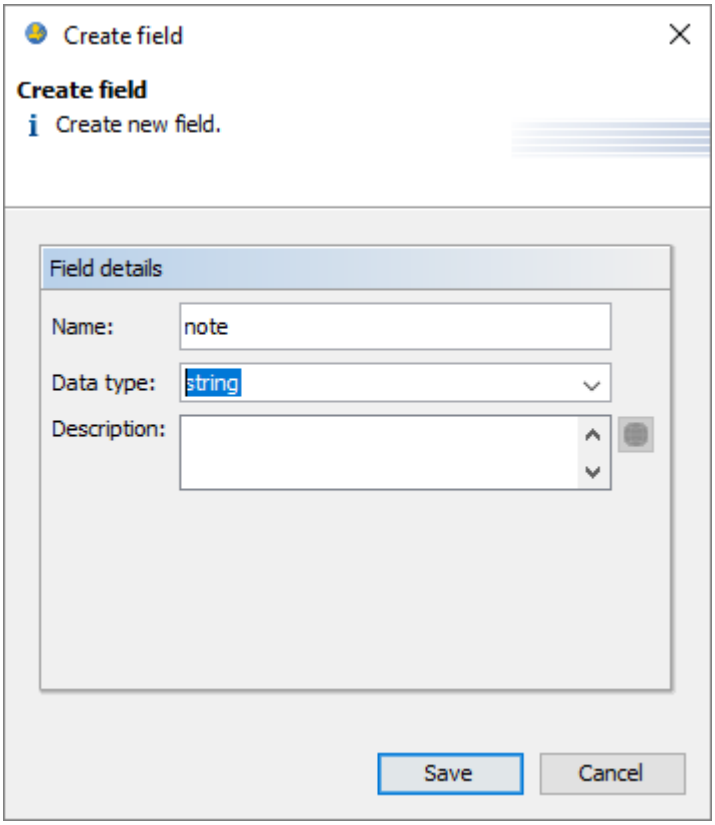


Figure 445: *ConSol CM Admin Tool - Resources, Data Models: Creating a new resource field*

Enter the following data:

- **Name**
  The technical name of the resource field. This is used in scripts and templates. The localization mechanism is explained in section Localization of Data Fields.
- **Data type**
  The data type of the resource field. You can select a data type from a drop-down list, like for ticket fields or customer fields. The following section provides detailed information about the available data types.
- **Description**
  Enter the description for the resource field. If provided, the description will be used as a tooltip in the Web Client. You can enter a localized description using the globe icon. For details, please refer to section Localization of Objects in General, Type 1.

Types of Data Fields

The following data types are available for ticket fields, customer fields and resource fields.

- **autocomplete**
  A data field which contains a text autocomplete field. This is a dynamic list which is based on a script of type *Text Autocomplete*. A detailed explanation of text autocomplete fields is given in section Scripts of Type Text Autocomplete.
- **boolean**
  Values: true/false. Depending on the annotation `boolean-type`, the value is displayed as check-box, radio buttons, or drop-down list.

> If a boolean field has not been touched, its value is “NULL”. If it is checked, its value is “true”, and if it is unchecked again, its value is “false”.

- **date**
  Format and accuracy can be set by annotations.
- **enum**
  For sorted lists. The engineer can choose one of the enum values in the Web Client. Enums and values have to be created previously within the Managing Sorted Lists: Enums. Select the desired *Enum type* and *Enum group* in the fields below.
- **list**
  A data field of this data type is the first step to creating a list (one column) or a table (multiple columns) of input fields in the Web Client.
  - For a **table** the next step will be to create another field of type *struct* (see below) to contain the input of the individual list fields (which will become the columns of the table). So, if you want to create a table you have to define a field of the type *struct* first (see below) before you can add the fields for the table columns.
  - For a **simple list**, the next step will be to create fields which belong to the list. No *struct* is required.

  For all fields belonging to a list or table you have to set the dependencies in the field *Belongs to* (see below). For example, a table field (which is a regular data field) always belongs to a *struct*, a struct always belongs to a *list.*
- **struct**
  A data field of this type defines a data structure (line of a table) which groups one or multiple fields. It is the second step to building a table after you have created a field of the type *list*. Add the fields for the columns of the table in the next step. The dependencies have to be set for each field in the *Belongs to* field (see below), i.e., a *struct* always belongs to a *list*.

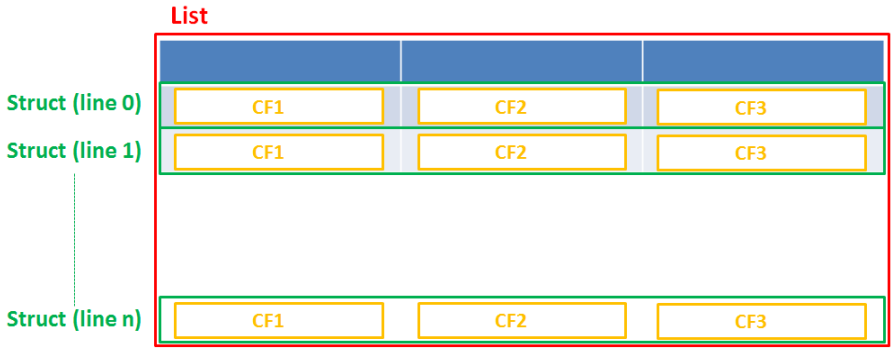


Figure 446: *Scheme: List of structs*

Technically spoken, the list is an array which contains a map (= key:value pairs) in each field.



Figure 447: *List of structs, technical principle*

- **number**
  For integer values.
- **fixed point number**
  For numbers with a fractional part, e.g., currencies. You have to enter the total number of digits (*Precision*) and the number of digits that fall to the right of the decimal point (*Scale*) in the respective fields below.
- **string**
  For up to 4000 alphanumeric characters.

  ⚠ Restriction when using an Oracle database: at most 4000 bytes can be saved in UTF encoding. Starting with Oracle12c.

- **long string**
  For large objects.

  ⚠ For *long strings* the limit depends on the database system used for ConSol CM: MS SQL Server: 2 GByte; MySQL: 4 GByte; Oracle: 16 - 64 GByte (depending on page size of tablespace).

- **short string**
  For up to 255 alphanumeric characters.

  ⓘ For *string* fields, you can use specific annotations to fine-tune the field definition. For example, a *string* field can be defined to contain a URL which will automatically be displayed as hyperlink or can be the hook for an autocomplete list. Please read the following section.

- **contact data reference**
  Special data type used internally for referencing the contacts associated with a ticket. This data type is not displayed but only used internally in the CM system.
- **MLA field**
  This data type is used for fields that contain hierarchical lists with a tree structure called *MLA* (Multi Level Attributes). The name of the field is the name of the new MLA that has to be defined within the MLAs. The group of the field has to be referenced when the MLA is created.

⚠ The data type you choose on creating a data field cannot be changed afterwards!

Using Annotations to Fine-Tune Data Fields

String fields can be used for various purposes, from simple input fields to text boxes, URLs or passwords. The configuration is done using annotations, see Annotations. The most commonly used annotation are listed in the following section.

How can I ...

... insert a **text box** instead of a single line?

Value for the annotation `text-type`: “textarea”

The size of the text box can be adjusted, displayed as standard text box depending on web browser. Use the `field-size` annotation in case a specific size of the text box is required.

... hide the input of the fields for **passwords**?

Value for the annotation `text-type`: “password”

Only dots are displayed. This annotation does **not** define the field to contain a password. It only defines the display mode. Use the `password` annotation to define a string field to contain the CM/Track password.

... display a **hyperlink**?

Value for the annotation `text-type`: “url”

The input is displayed as a hyperlink in view mode. The string has to match a specific URL pattern:

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

The first part of the string is the link (url), the second part is the name which should be displayed.

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

... display a **file link**?

Value for the annotation `text-type`: “file-url”

The input is displayed as a link to a file on the file system. The web browser has to allow/support those links.

Example: Enabling file:// URLs in a Firefox browser

Add the following lines to either the configuration file `prefs.js` or to `user.js` in the user profile. On a Windows system usually in a folder like `C:\Users\<USERNAME>\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");

Alternatively a Firefox browser add-on like ***Local Filesystem Links*** can be installed for better access to the referenced files and folders.

The link is also displayed as a tooltip.

The URL is correctly formed if the following conditions are met:

- It starts with `file:` followed by regular slashes:
  - three slashes "///" for files on the same computer as the browser (alternatively "//localhost/")

    - two slashes followed by the server name followed by another slash for files on file servers accessible from the computer running the browser
- These are followed by the full path to the file ending with the file name.
- The path on Microsoft Windows systems is also written with forward slashes instead of backslashes.
- The drive letter of a local path on Microsoft Windows systems is noted as usual, for example `C:`.
- Paths with spaces and special characters like "{, }, ^, #, ?" need to be percent encoded ("%20" for a space for example) for Microsoft Windows systems.

Example URLs:

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

See also the explanation about `file-url` in the section text-type

... define a **label**?

Value for the annotation `text-type`: “label”

This is a read-only field which is displayed in label styling. Use the `label-group` annotation to link the label and the input fields which belong together. Please take a look at the annotations for labels (`show-label-in-edit`, `show-label-in-view`) before implementing special label fields.

... define a **title**?

Value for the annotation `text-type`: “title”

This is a read-only field which is displayed in title styling, i.e. in a bold font which is larger than regular labels.

... define a field for the **valid email addresses**?

Value for the annotation `email`: “true”

The field may only contain valid email addresses. The input is validated according to the standard email format <name>@<domain>.

... define a scripted autocomplete list?

Value for the annotation `text-type` = “autocomplete”

Optional: value for the annotation `autocomplete-script` = <name of the respective script>

A scripted autocomplete list is used to show a drop-down list which is filled dynamically using the input the engineer has provided so far. For example, when the user types "Mil", the possible values "Miller", "Milberg", and "Milhouse" are displayed as a list and the engineer can select the one required for the field. The values are based on a result set which can contain strings, engineers, tickets, customers or resources.

The logic to create the result set needs to be implemented in a script, see Scripts of Type Text Autocomplete.

... define a **rich text** field?

Create a field of the type *long string*.

Value for the annotation `string-content-type`: “html-inline” (field content is directly displayed on the GUI in view mode) or “html-link” (field content is displayed in a pop-up window after clicking the *Show content* link in view mode)

A rich text field can contain images and formatted text. In the edit mode, it features a basic editor.

... define a tooltip for a data field?

Value of the annotation `show-tooltip` = “true”

When creating or editing a data field, you can enter a text in the *Description* field. This text is used as a tooltip in the Web Client. You can enter a localized description using the globe icon. For details, please refer to section Localization of Objects in General, Type 1.

> This does not apply to date fields and fields containing phone numbers and URLs, as they have special tooltips.

... optimize the data field layout?

Use three columns to display the data fields.

Display data fields which most likely have a value on the left in order to avoid gaps.

Add the annotations `position` = 0;0 and `colspan` = 3 to all tables. Remove the `colspan` annotations from other types of data fields.

Show the labels for all data fields (annotation `show-label-in-view` should be `true`).

Setting Resource Field Annotations

The characteristics of a resource field, e.g., the visibility, the position in the Web Client, and whether the field should be indexed or reportable, are defined using annotations. You might know this principle from your work with ticket field annotations.

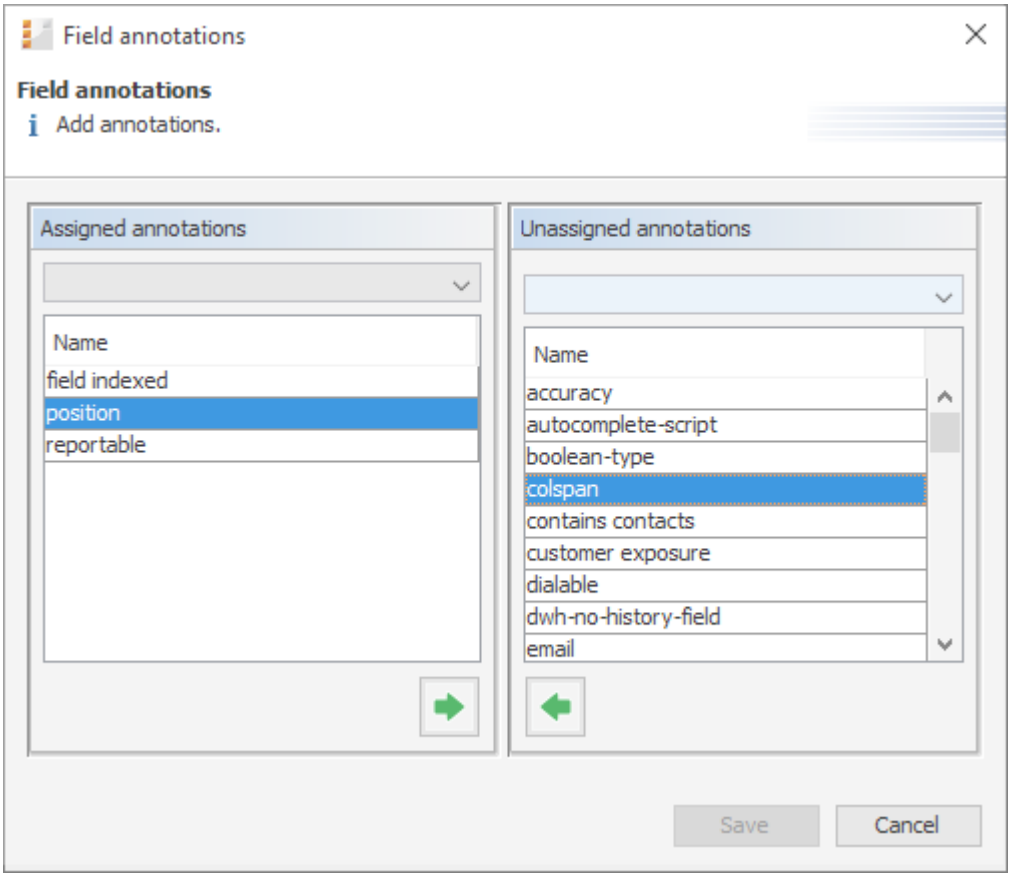


Figure 448: *ConSol CM Admin Tool - Resources, Data Models: Defining resource field annotations*

The annotations which are available are the same as for ticket fields and are listed and explained in detail in Annotations.

The `position` annotation marks the position of the field on

- the form which is used to create a new resource. See section A Short Introduction to CM/Resource Pool Functionalities in the Web Client, Resource Create Form.
- the resource page which is explained in section A Short Introduction to CM/Resource Pool Functionalities in the Web Client, Resource Page.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

As with ticket layout, you can place the resource fields in the GUI by using the matrix principle and setting the position accordingly:

0;0	0;1	0;2
1;0	1;1	1;2
2;0	2;1	2;2
n;0	n;1	n;2

Figure 449: *Example matrix for position annotation*

G.1.4.3 Copying Resource Fields

You do not necessarily have to create every data field using the pop-up menu. You can also copy an existing data field. This might be a ticket field, customer field, or resource field. Select the respective data field in the data field group and click the *Copy* button. Then navigate to the data field group where the copied field should be placed. This might be a ticket field group, a customer field group, or a resource field group. You can copy a field from one object type (e.g. ticket) to another (e.g. customer). Click into the list of data fields of the target field group and click the *Paste* button.

Every type of field can be copied, from simple string fields over enums to entire lists of structs. To copy a list of structs or a list of fields, mark and copy only the list, the child objects (e.g., structs with fields) will be copied automatically.

These things will be copied:

- almost all annotations with their values
- all localized values

These things will not be copied:

- the following annotations
  - *ldapid*: This annotation can be only once in a data model. Duplication would cause an invalid model.
  - *username*: This annotation can be only once in a data model. Duplication would cause an invalid model.
  - *password*: This annotation can be only once in a data model. Duplication would cause an invalid model.
  - *position*: The value for this annotation must be unique, so the annotation is duplicated, but the value is removed from the copy.
  - *ticket-list-position*: This annotation should be used very specifically and, thus, it should be present only for very few fields. Duplicating it would potentially multiply it although it should not be used for the majority of copies.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Naming of the copied field(s):

- Copy to other field group: field name is used
- Copy to same field group: <field name>_copy is used

Besides the Copy/Paste buttons you can also use the keyboard shortcuts:

- Copy: CTRL-C
- Paste: CTRL-V

G.1.4.4 Using Scripted Field Visualization for Resource Fields

Using scripted field visualization, you can enhance the display of data in resource fields. Please see section Scripts of Type Field Visualization for details.

G.1.4.5 Creating Further Resource Types within a Resource Category

To add a new resource type within a resource category, mark another resource type of the desired resource category in the list and click the *Add* button. Then fill in all required data as described in the section Editing the Resource Type.

G.1.4.6 Creating Resource Field Groups and Resource Fields

To create a new resource field group within a resource type, select another resource field group of the desired resource type in the list and click the *Add* button. Then enter all the required data as described in the section Editing the Resource Field Group.

G.1.4.7 Creating Further Resource Categories

To create a new resource category (OfficeSoftware in our example), select another resource category in the list and click the *Add* button.

Enter all the data as for the first resource category, see section Editing the Resource Category.

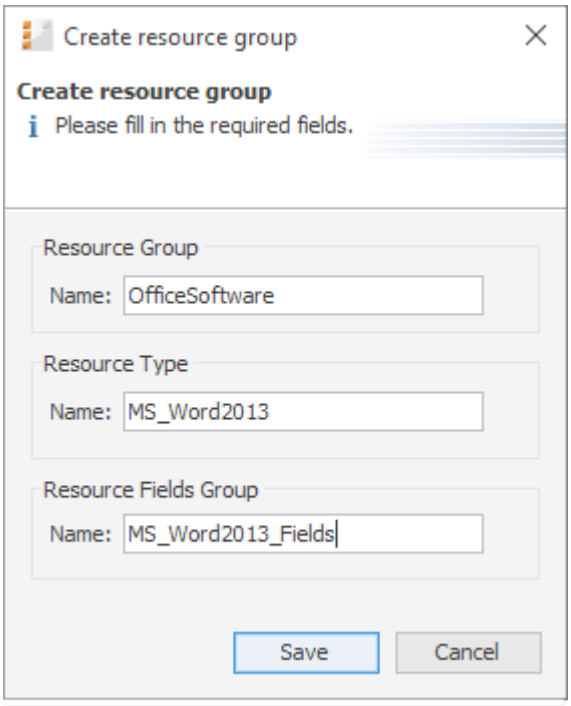


Figure 450: *ConSol CM Admin Tool - Resources, Data Models: Adding a new resource category*

To fill the new resource category with resource types and resource fields, proceed as described in the previous sections.

G.1.4.8 Example for a Complete Resource Model

In our example, the following resource model has been created. It will be used as a basis for later examples in this manual.

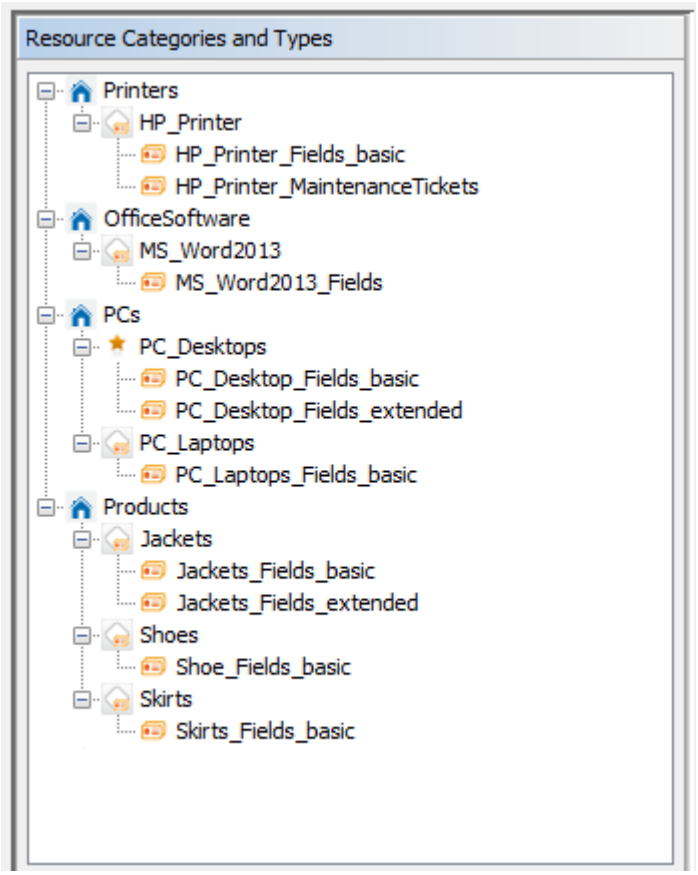


Figure 451: *ConSol CM Admin Tool - Resources, Data Models: Example for a resource model*

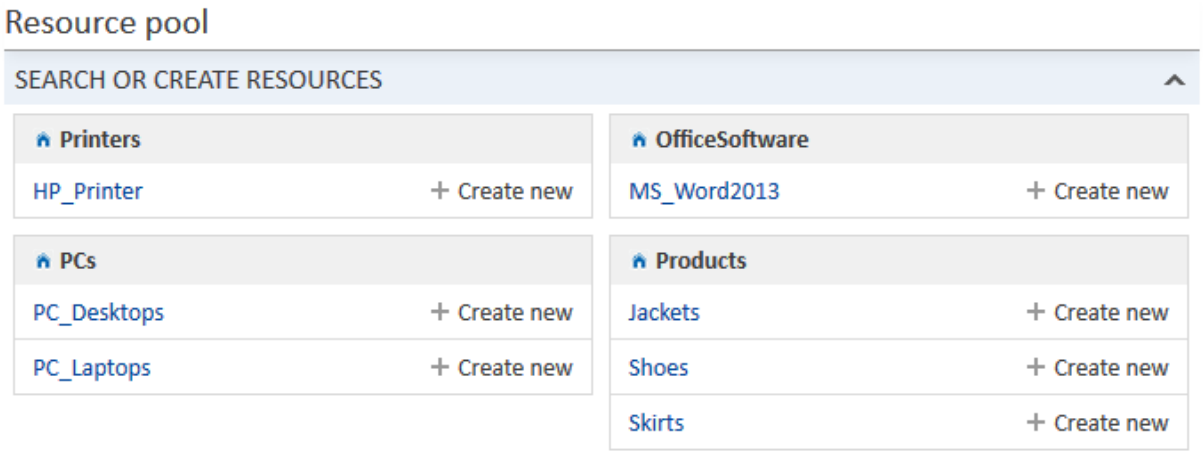


Figure 452: *ConSol CM Web Client - Example for a resource model (in Resource Pool Dashboard)*

G.1.4.9 Defining Resource Relations

This is explained in detail in the section CM/Resource Pool - Resource Relations.

G.1.4.10 Defining Resource Actions

This is explained in detail in the section CM/Resource Pool - Resource Actions.

G.1.4.11 Assigning Access Permissions for Resources to Engineers Using Roles

This is explained in detail in the section CM/Resource Pool - Assigning Permissions for Resources.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.1.5 CM/Resource Pool - Templates for Resource Data

G.1.5.1 Introduction to Using Templates for the Display of Resource Data

In the ConSol CM Web Client, resource data sets are displayed in a short form, the so called display name of the resource, at various locations. The display name is based on the template for the respective location. In this way, it is possible to use a different display name for each location, for example to adapt to the available screen space. Templates are defined by resource type.

The configuration of the display name is based on the following principle:

- A template for a specific location in the Web Client is assigned to a resource type in the resource model definition (navigation group *Resources*, navigation item *Data model*).
- The referenced template must be stored in the Admin Tool, navigation group *System*, navigation item *Scripts and Templates*. The name of a template is user-defined, but it needs to match the referenced template name in the resource type definition.

In the following paragraphs, the syntax and coding for templates and all possible template types are explained.

Please see Admin Tool Templates for a more generic description of how to use templates.

The following figure shows an example of a resource template to define the display name on the resource page.

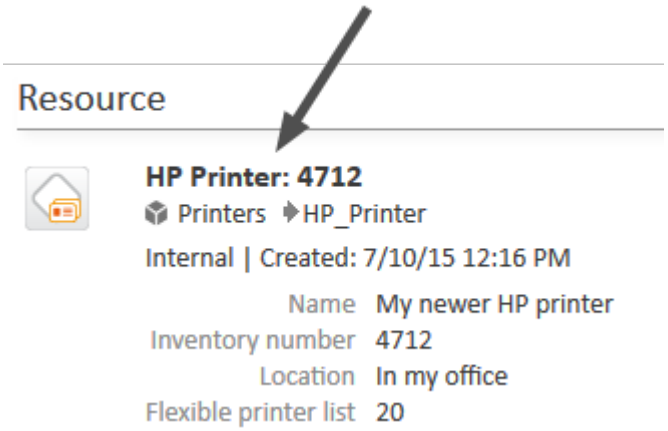


Figure 453: *ConSol CM Web Client - Example of a resource template (resource page)*

G.1.5.2 Coding Templates

General Principle

The templates are written in ***FreeMarker*** notation. For detailed information, please refer to the FreeMarker web site.

Within the templates, you work with three object types:

- the current resource (referenced as `resource`)
- the technical name of the resource field group
- the technical names of the resource fields

⚠ The resource templates must be single-row! They must not contain line breaks!

If you need the display name of a resource as determined by a template in scripts, you can use the method `getDisplayName()` from the class `ResourceService`. The following example shows how to print the resource name as determined by the default template in German to the log files. It can be used for example in a resource action, where the `resource` object is available.

```
log.info resourceService.getDisplayName(resource, ResourceTemplateType.DEFAULT, new
 Locale("de"));
```

Examples for Templates

The following example shows a template for resources of the resource type *PC*.

```
PC: ${resource.getFieldValue("PC_Desktop_Fields_basic","name")!} -
 ${resource.getFieldValue("PC_Desktop_Fields_basic","modell")!}
```

How to Handle Missing Values

You can use if statements in combination with the `??` operator or the `?has_content` operator to check if an object exists or a field has a value for the given customer or resource.

```
<#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 example 91: *Example of a customer template with if statement (has to be written in one line!)*

Alternatively, you can specify a default value using the `!` operator to avoid exceptions due to missing values. The default value can be empty.

```
${ResellerCompany.getFieldValue("ResellerCompanyData","company_name")!} -
 ${ResellerCompany.getFieldValue("ResellerCompanyData","company_number")!}
```

Code example 92: *Example of a customer template with empty default value (has to be written in one line!)*

How to Set the Format of Numbers

Freemarker allows to set a number format pattern.

```
<#setting number_format="0.#"/>${customerModelCompany.getFieldValue("groupName",
 "numberValueField")!}
```

Code example 93: *Setting a number format, so only one decimal place is displayed*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Localizing Enum Values in Templates

It is possible to display localized enum values in templates by using the `localize()` method. This method retrieves the localized enum value as defined in the Admin Tool to display it in the Web Client. Otherwise, the technical enum value would be displayed.

The following example shows a template for a resource. The resource field *SLA_country* is an enum containing a list of countries. The `localize()` method is used to display the country name in the respective browser locale.

```
${resource.getFieldValue("SLA_Fields_basic","SLA_Name")!} (${localize
 (resource.getFieldValue("SLA_Fields_basic","SLA_country"))!})
```

Abbreviating Values in Templates

It is possible to abbreviate the values retrieved by a template using the `abbreviate()` method. In this way, long values are shortened, so all fields are displayed even if the first field contains a long value.

The `abbreviate()` method needs two parameters:

- The value which should be abbreviated.
- The number of characters to be displayed. Please note that this is the number of all characters which are displayed, i.e., including the three dots. Thus, for example, to display five letters and three dots, use “8” as a parameter. The minimum value is “4”.

The following example shows the template for a company and how the abbreviated company name is displayed in the ticket list.

```
${abbreviate(ResellerCompany.getFieldValue("ResellerCompanyData","company_
 name"),8)!} - ${ResellerCompany.getFieldValue("ResellerCompanyData","company_
 number")!}
```



Figure 454: *ConSol CM Web Client - Abbreviated value in ticket list*

G.1.5.3 Template Types

The following template types can be defined for the different locations where resources are displayed in the Web Client:

- Default
- Quick Search
- Search

- REST
- History

Default

This template must always be defined. If it is not, there will be an error in the Web Client (*unknown* is displayed as name of the resource). The template is used for all locations in the Web Client for which no other templates have been defined and as REST template for CM/Track in case there is no dedicated REST template. The templates which are described in the subsequent sections override the default template for the specific locations.

The template is always used for the display name of the resource on the resource page, the display name when the resource is dragged, and the resource name in the Workspace and Favorites.

Quick Search

This template defines the display name of the resources in the results of the Quick Search.

Search

This template defines the display name of the resources in search results in autocomplete fields or suggestion lists. It is used, for example, when creating a relation to a resource.

REST

This template defines the display name of a resource when accessed using the REST API. When retrieving a resource data set using the REST API, the display name is included in the `mark` tag of the output.

History

This template defines the display name of the resource in the ticket, customer and resource history. It is used for example in the history entries which are created when a relation to or from a resource was added.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.1.6 CM/Resource Pool - Resource Relations

G.1.6.1 Introduction

A resource can have relations to other ConSol CM objects. In contrast to customer relations, no hierarchy levels are defined. All relations are simple references. A resource can have zero or more relations to the following types of ConSol CM objects:

- **Ticket**
  E.g., an incident ticket has a ticket-resource relation to the printer which caused a problem.
- **Company**
  E.g., a company has a company-resource relation to a specific SLA.
- **Contact**
  E.g., a contact has a contact-resource-relation to the terminal server access rights.
- **Customer** (i.e., company **or** contact of a customer group)
  E.g., a resource has a resource-customer relation to a customer group to reflect that customers in this customer group are all potential receivers of a newsletter.
- **Resource**
  E.g., a laptop has a resource-resource relation to all connected/configured printers.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

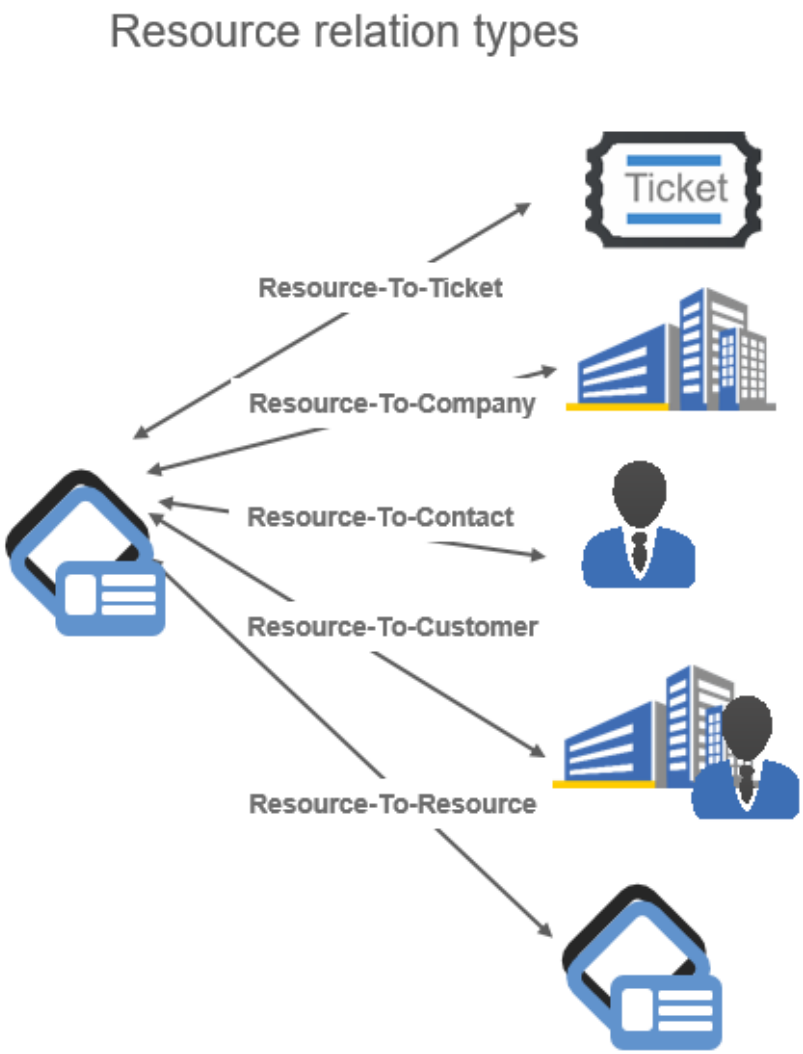


Figure 455: *ConSol CM resource relation types*

For a resource relation, the multiplicity (cardinality) has to be defined, i.e., for each relation, how many objects of the selected type may participate in a relation must be defined.

The following configurations are possible for the resource relation multiplicity (cardinality):

- **One-to-one**
  Each resource can only be related to a single target object.
- **One-to-many**
  Each resource can be related to many different target objects (e.g., one laptop can have resource-resource relation to one, two, or more printers).
- **Many-to-one**
  Many resources can be related to one single target object.
- **Many-to-many**
  Many resources can be related to many different target objects.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

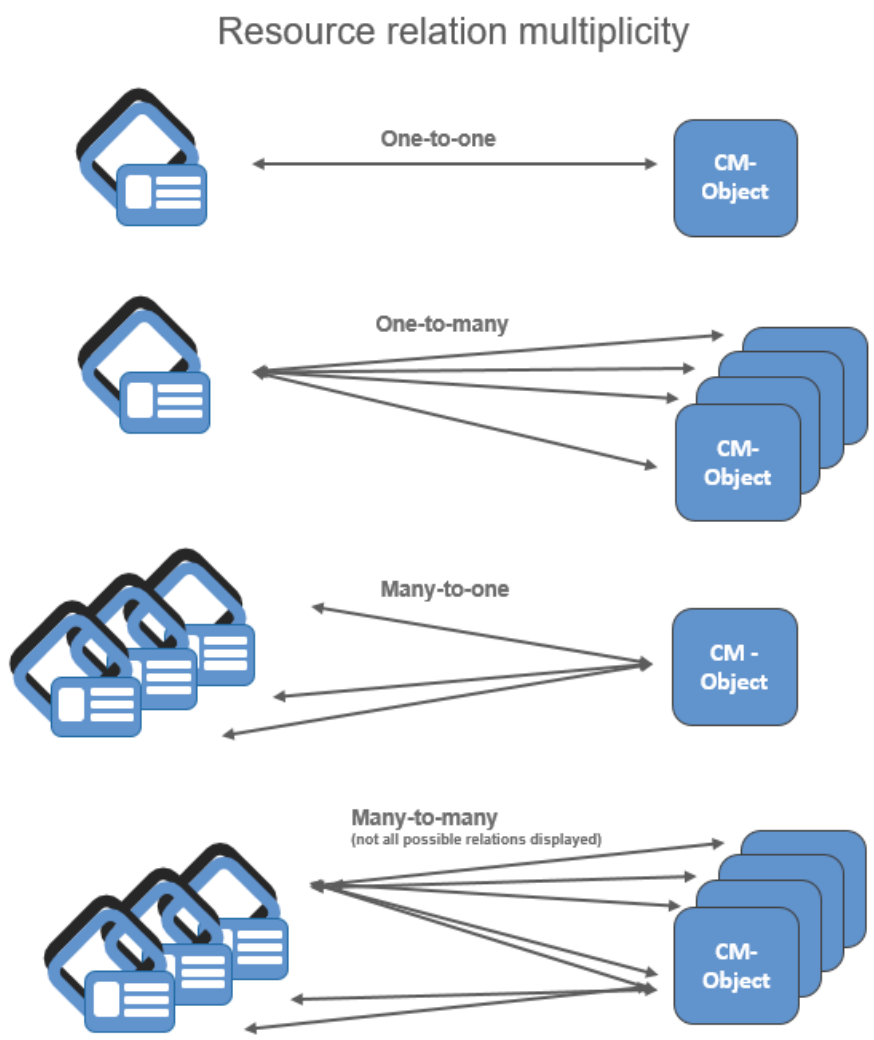


Figure 456: *ConSol CM resource relation multiplicity*

G.1.6.2 Creating Resource Relations in the Data Model Using the Admin Tool

Resource relations are always defined for a resource type, e.g., for the resource type *HP_Printer*.

To create new resource relations or to edit resource relations, open the resource type panel in the Admin Tool (navigation group *Resources*, navigation item *Data Models*).

> Please note that you cannot create or edit resource relations using the navigation item *Relations Overview*. This panel is only used for the display of relations and for defining the order of relations for display in the Web Client!

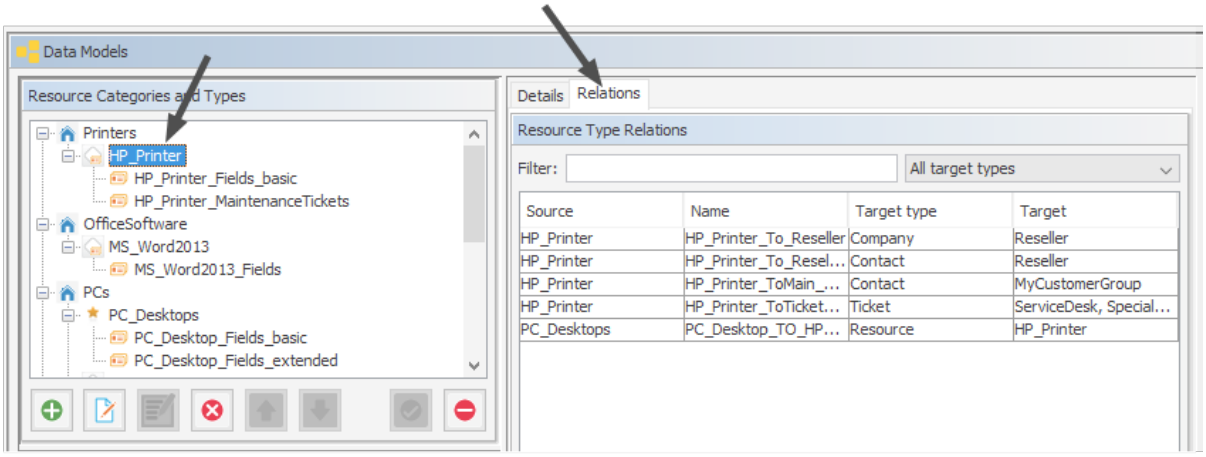


Figure 457: *ConSol CM Admin Tool - Resources, Data Models: Creating and editing resource relations*

To define a new resource relation, click the *Add* button and fill in the required fields in the pop-up window.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

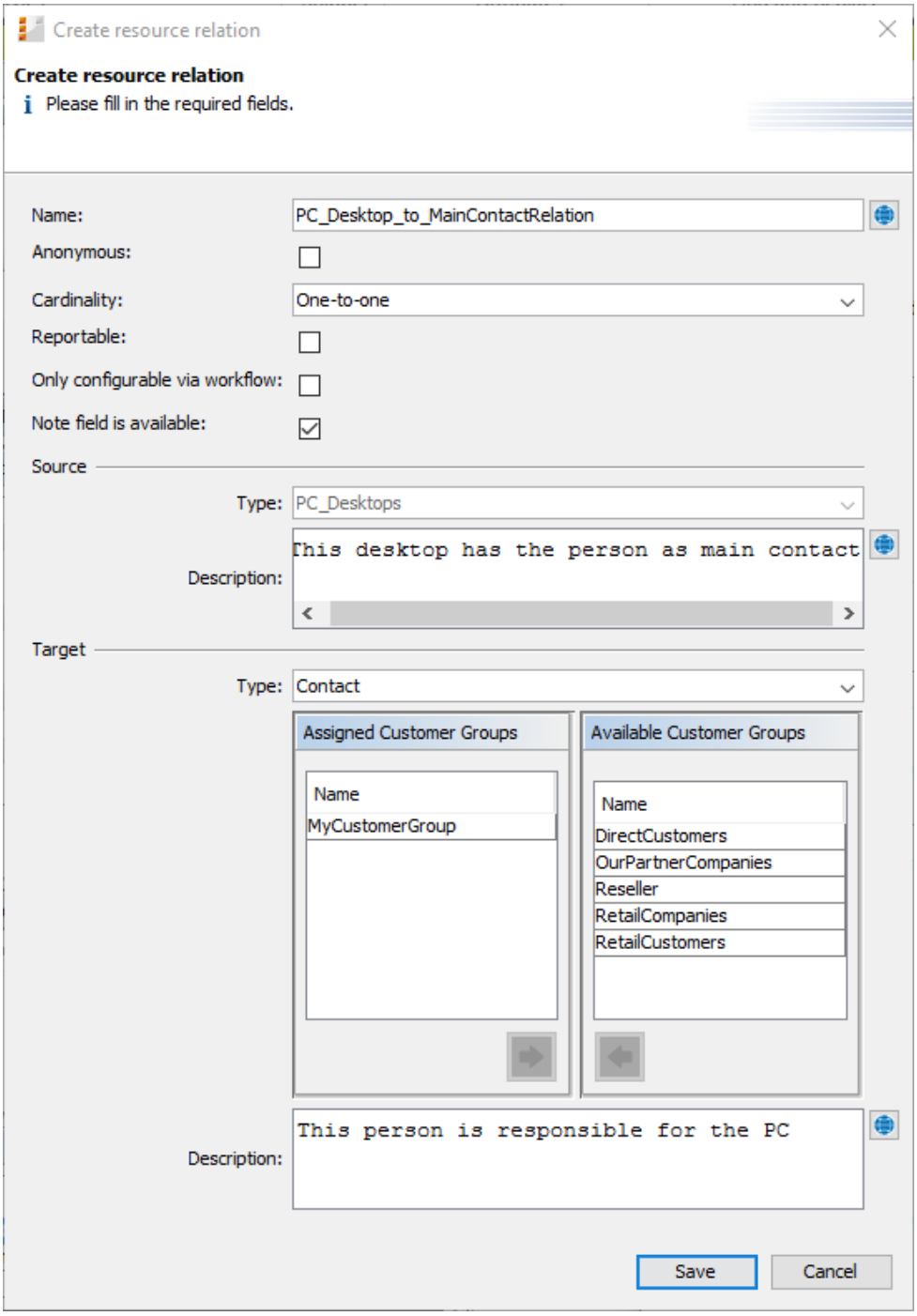


Figure 458: *ConSol CM Admin Tool - Resources, Data Models: Creating a new resource relation*

- **Name**
  The technical name of the resource relation. You can localize it using the *Localize* button. This name will be displayed in the Web Client when a resource relation of this type has been established. For a detailed explanation of the localization mechanism, please refer to section Localization of Objects in General, Type 1.
- **Anonymous**
  If this checkbox is selected, *related* will be displayed as the description for the relation in the

Web Client. Use this checkbox if you do not want to have a description displayed. (Since you cannot leave the name field empty, the only other way would be to display the technical name of the relation.)

- **Multiplicity**
  Select the multiplicity (cardinality) for the relation (see the explanation of cardinality above).
- **Reportable**
  If this checkbox is checked, the relations of this type will be transferred to the DWH.
- **Only configurable via workflow**
  If this checkbox is checked, the relation can only be set / removed using workflow scripts. The relation will not be available in the respective menus in the Web Client.
- **Note field is available**
  If this checkbox is checked, a note field will be displayed for each resource when a new resource relation is established (see the following two figures)

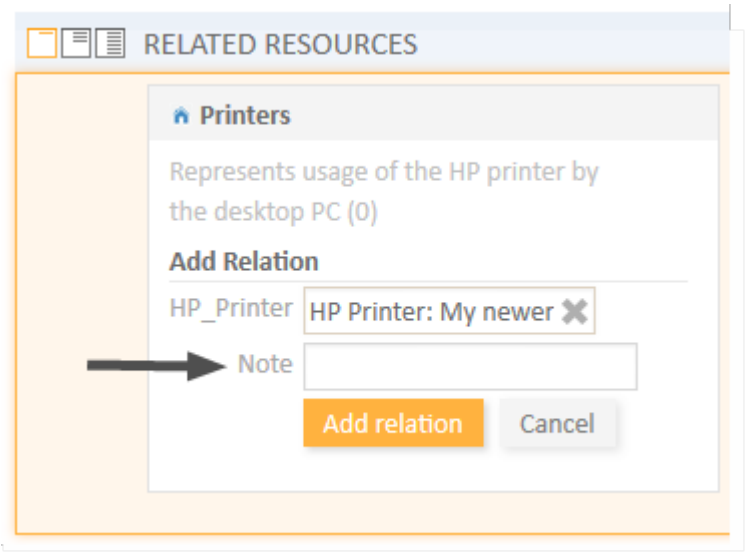


Figure 459: *ConSol CM Web Client - Establishing a resource-resource relation with Note field*

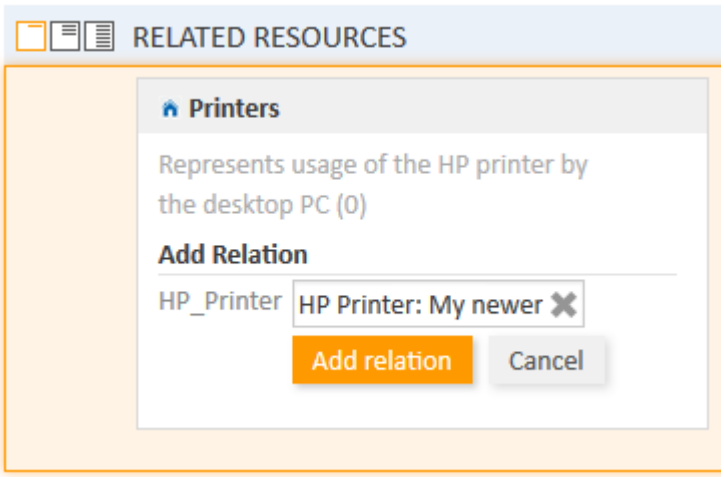


Figure 460: *ConSol CM Web Client - Establishing a resource-resource relation without Note field*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Source**
  - **Type**
    The current resource. The value is pre-selected and cannot be modified.
  - **Description**
    The description of the relation. It will be displayed in the Web Client as a header for any relation of this type on the source side. The description can be localized using the *Localize* button.For a detailed explanation of the localization mechanism, please refer to section Localization of Objects in General, Type 1.
- **Target**
  - **Type**
    The object type of the target of this relation. For an explanation, please see the section about relation types.
  - **Assignment table**
    You can assign objects of the selected type to the relation here, i.e., the resource relation will only be available for these objects. Depending on the selected target object type, there will be different objects which can be selected in the assignment table:
    - Target type **Resource**: assigned/available resource types
    - Target type **Customer**: assigned/available customer groups
    - Target type **Contact**: assigned/available customer groups
    - Target type **Company**: assigned/available customer groups
    - Target type **Ticket**: assigned/available queues
  - **Description**
    The description of the relation. It will be displayed in the Web Client as a header for any relation of this type on the target side. The description can be localized using the *Localize* button.

G.1.6.3 Relations Overview

The navigation item *Relations Overview* provides a list of all defined relations and provides the possibility to define the display order of the relations in the Web Client.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

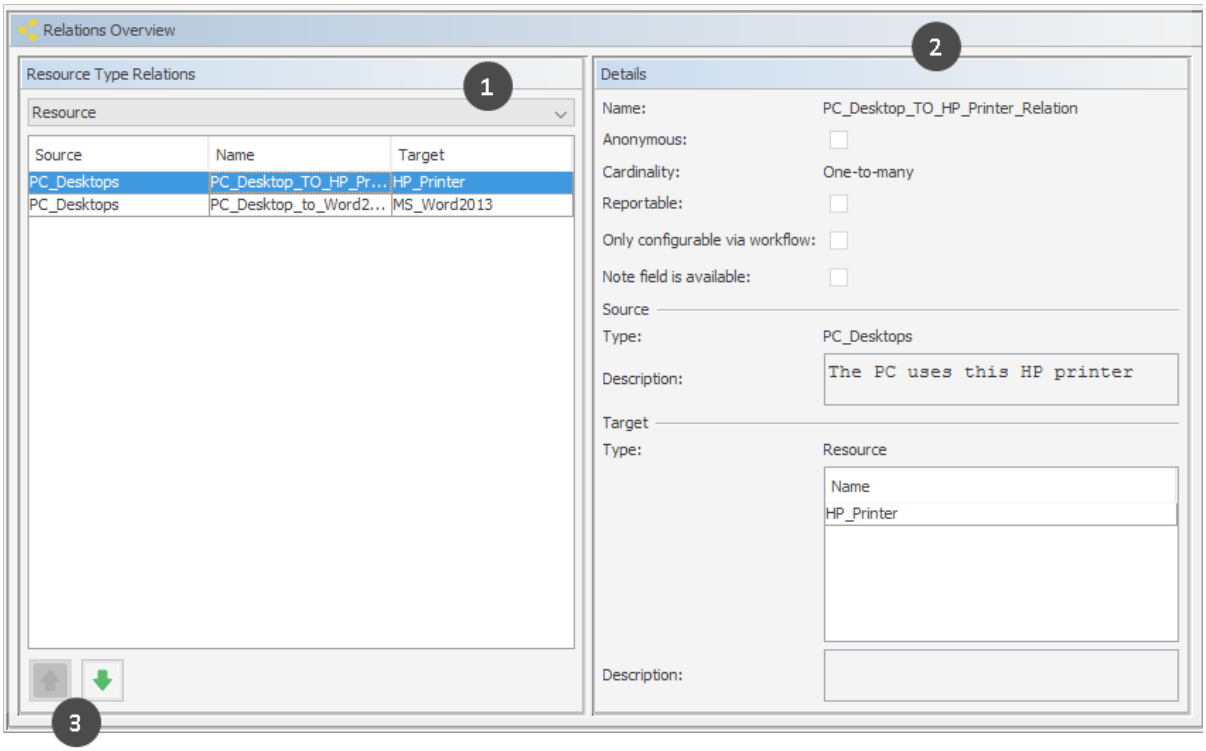


Figure 461: *ConSol CM Admin Tool - Resources, Relations Overview*

Use the filter (drop-down menu) in the top row to select the target object type (1). Only relations with this target object type will be displayed in the list. Please refer to the section about relation types for details about possible target object types.

Please note the filter above the list. You have to select one resource relation type:

- Resource
- Customer
- Contact
- Company
- Ticket

Only relations of this type will be displayed in the list.

Two functions are available in the *Relations Overview*:

- **Display relation details**
  Select a relation. Its details will be displayed on the right-hand side in read-only mode (2).
- **Modify display order**
  Select a relation and use the *Up* and *Down* arrows to move it up/down within the list. This will define the position of the relation in the list when more than one relation is available for selection in the Web Client (3).

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.1.6.4 Creating Resource Relations Using the Web Client

Resource relations can be established on several pages in the Web Client, depending on the source object.

- Ticket-to-X relations can be created on the ticket page.
- Resource-to-X relations can be created on the resource page.
- Contact-to-X relations can be created on the contact page.
- Company-to-X relations can be created on the company page.

For a short explanation of working with resource relations, see section A Short Introduction to CM/Resource Pool Functionality in the Web Client. A detailed explanation of all resource-related Web Client functions is provided in the *ConSol CM User Manual*.

G.1.6.5 Configuration of Resource Relations in the Web Client

Some parameters related to the display mode of resource relations can be configured using Page Customization. These are:

- The default limit for the number of resource relations before the table filter control elements are displayed.
- The sorting strategy for resource relations.

For details about parameters affecting customer (unit) pages, please see section UnitResourceRelationof the Page Customization chapter.

For details about parameters affecting the resource page, please see section TicketRelation of the Page Customization chapter.

For details about parameters affecting the ticket page, see section resourceRelations of the Page Customization chapter.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.1.7 CM/Resource Pool - Resource Actions

G.1.7.1 Introduction

Resource actions are a component of the ConSol CM Action Framework. Resource actions are actions which can be performed for a resource, i.e., an object which is stored in the Resource Pool. The actions can be performed automatically by the system or manually, triggered by an engineer who has the required permissions. You might want to apply resource actions for use cases like the following:

- Create a maintenance ticket for a printer.
- Find all companies which use a certain SLA.

You can use the following types of resource actions:

- **Automatic** actions which are performed by the system after one of the following resource operations:
  - CREATE
  - UPDATE
  - DELETE
  - RELATION
  - SEARCH
- **Manual** actions which are performed by the engineer using *Activities* links on a resource page in the Web Client (similar to *Workflow activities* for tickets). Manual actions are executed for the resource which is displayed.

⚠ Please keep in mind that only engineers who have at least one role with the following access permissions for the respective resource type are allowed to use the resource actions, i.e., only then will the Activities link be displayed in the Web Client:

- Act

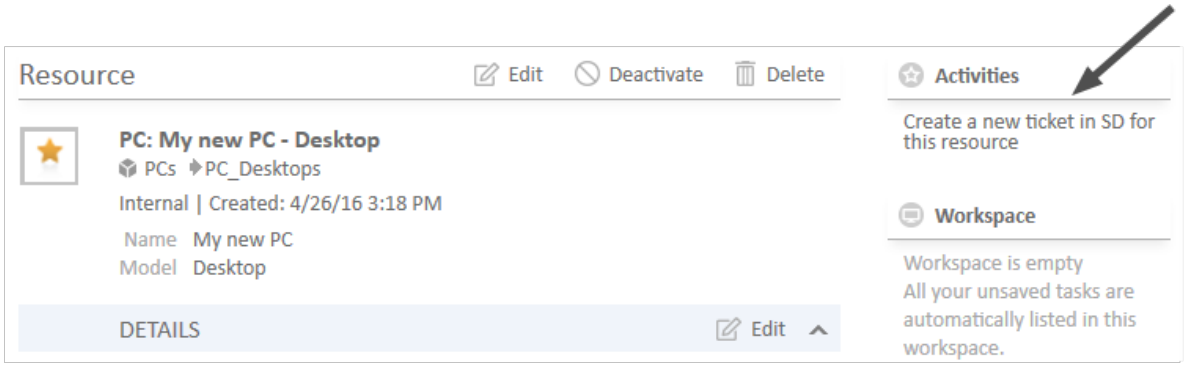


Figure 462: *ConSol CM Web Client - Resource action (on resource page)*

Resource actions are defined as Groovy scripts which are stored in the *Script and Template* section of the Admin Tool.

The execution of resource actions can be controlled using condition scripts, i.e., you can implement a condition script which is executed before the action script of the resource action. The action script is only executed if the condition script returns “true”.

So there are two types of scripts you have to deal with when you use ConSol CM resource actions:

- **Resource action script**
  Defines the action which should be performed.
- **Resource condition script**
  Defines one or more conditions for the execution of the action script. Has to return “true” or “false”. If “false” is returned the action script is not executed. If this is a manual action, it is not displayed as *Activity* in the Web Client.

When you want to implement a resource action you have to proceed in three steps:

1. Create a script for the resource action (either an action script only or an action script and a condition script).
2. Create the resource action(s) which use(s) the script(s).
3. Assign the resource action(s) to the resource type(s) where they should be available.

In the following sections, all three steps are explained in detail.

G.1.7.2 Creating Resource Actions Using the Admin Tool

Step 1: Write the Resource Action Script

Create a new Admin Tool script of type *Resource action*. If required, create another script of type *Resource condition*.

For a detailed explanation of Admin Tool scripts in general, please refer to section Admin Tool Scripts.

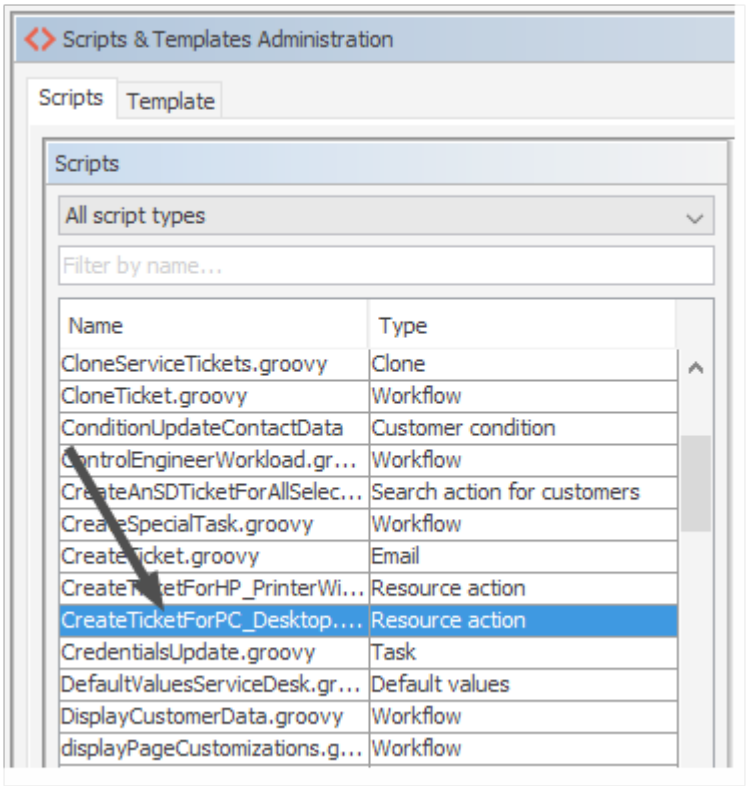


Figure 463: *ConSol CM Admin Tool - System, Scripts and Templates: Resource action script*

```
// 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!

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) {
   client.failure("cmweb.rp.resource.action.no_contact_set")
} else {
   def cont_rel = cont_list[0]
   maincont = cont_rel.getTargetUnit()

   ticketService.createWithUnit(newtic,maincont)
   log.info '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)
   def resRelationDef = resourceRelationDefinitionService.getByCriteriaUniqueResult
    (resRelationDefCriteria)
   def resRelation = new ResourceTicketRelation(resRelationDef, resource, newtic)
   resourceRelationService.create(resRelation)

   // go to new ticket
   client.goToTicket(newtic)
}
```

Code example 94: *Resource action script for PC_Desktops to create a new Service Desk ticket for responsible PC contact*

Step 2: Create Resource Action(s) Which Use(s) the Script

To create, edit, or delete resource actions, open the navigation item *Actions* in navigation group *Resources* in the Admin Tool.

To create or add a new action click the *Add* button and fill-in the required data in the pop-up window (the pop-up window is the same for adding and for editing a resource action).

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

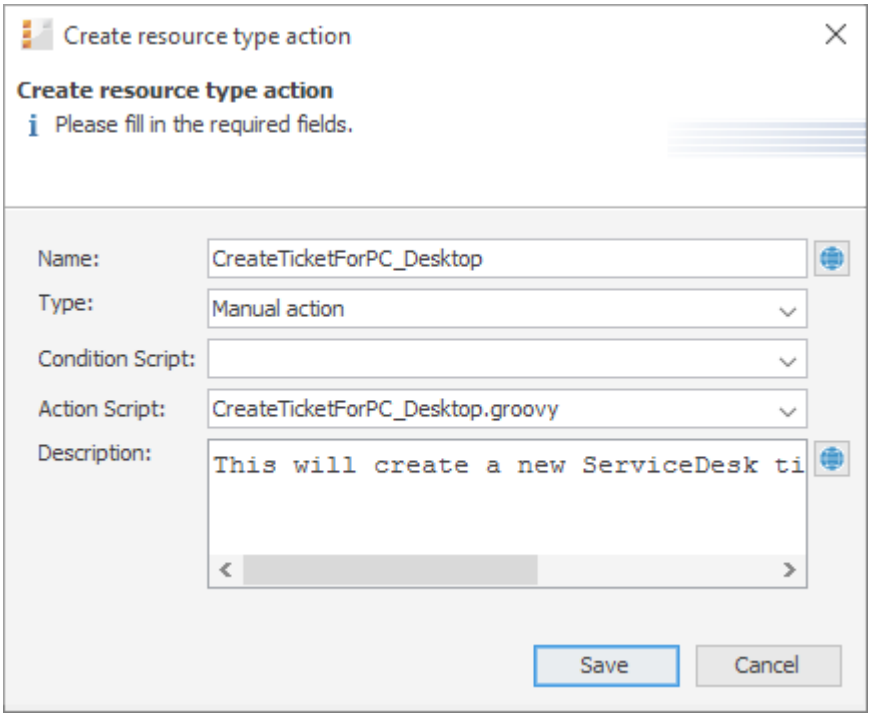


Figure 464: *ConSol CM Admin Tool - Resources, Actions: Creating a resource action*

- **Name**
  The technical name of the resource action. You can localize the value by using the *Localize* button. The localized name will be displayed in the Web Client. For a detailed explanation of the localization mechanism, please refer to section Localization of Objects in General, Type 1.
- **Type**
  The resource action type. Select one of the following types. Once a type has been defined, it cannot be modified afterwards (when you edit an existing resource action).
  - **Create**
    This action will be executed automatically when the resource is created.
  - **Update**
    This action will be executed automatically when the resource is updated, i.e., when the data has been modified (either manually or automatically) and is saved again.
  - **Delete**
    This action will be executed automatically when the resource is deleted.
  - **Relation**
    This script will be executed automatically when a relation to or from a resource of this type is
    - created
    - deleted

    (The script will not be executed when the comment of a relation is changed.)
  - **Manual**
    The resource action is displayed as *Activity* in the Web Client and can be executed only manually. If a Resource Condition Script is implemented, the activity will only be displayed in the Web Client if the condition script has returned “true”.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

  - **Search**
    The resource action is a search action. Actions of this type are explained in section Action Framework - Search Actions.

- **Condition Script**
  In case a condition script should be executed before the action script, the name of the condition script must be entered here. Use the drop-down menu to select from a list with all scripts of type *Resource condition* which are stored in the Scripts section of the Admin Tool. The action script will only be executed if the condition script returns “true”. If there is no condition, just leave this field empty.
- **Action Script**
  The name of the action script which should be executed. Use the drop-down menu to select from a list with all scripts of type *Resource action* which are stored in the *Scripts* section of the Admin Tool.
- **Description**
  Enter the description which should be displayed as mouse-over in the Web Client (for manual actions only).

Save the action. Then you can assign it to resource types. Please see following step.

Step 3: Assign Resource Actions to Resource Types

To assign pre-defined Resource Execution and/or resource condition scripts to resource types, the respective manual and/or automatic actions have to be assigned to a resource type. Open the navigation item *Data Models* in navigation group *Resources* in the Admin Tool. Select the resource type you would like to edit and click the *Edit* button to open the pop-up window where you can assign the resource actions. An action might contain only a resource action script, or a resource condition script and a resource action script.

In the following example (next figure), a manual resource action is assigned to the resource type *PC_Desktop*.

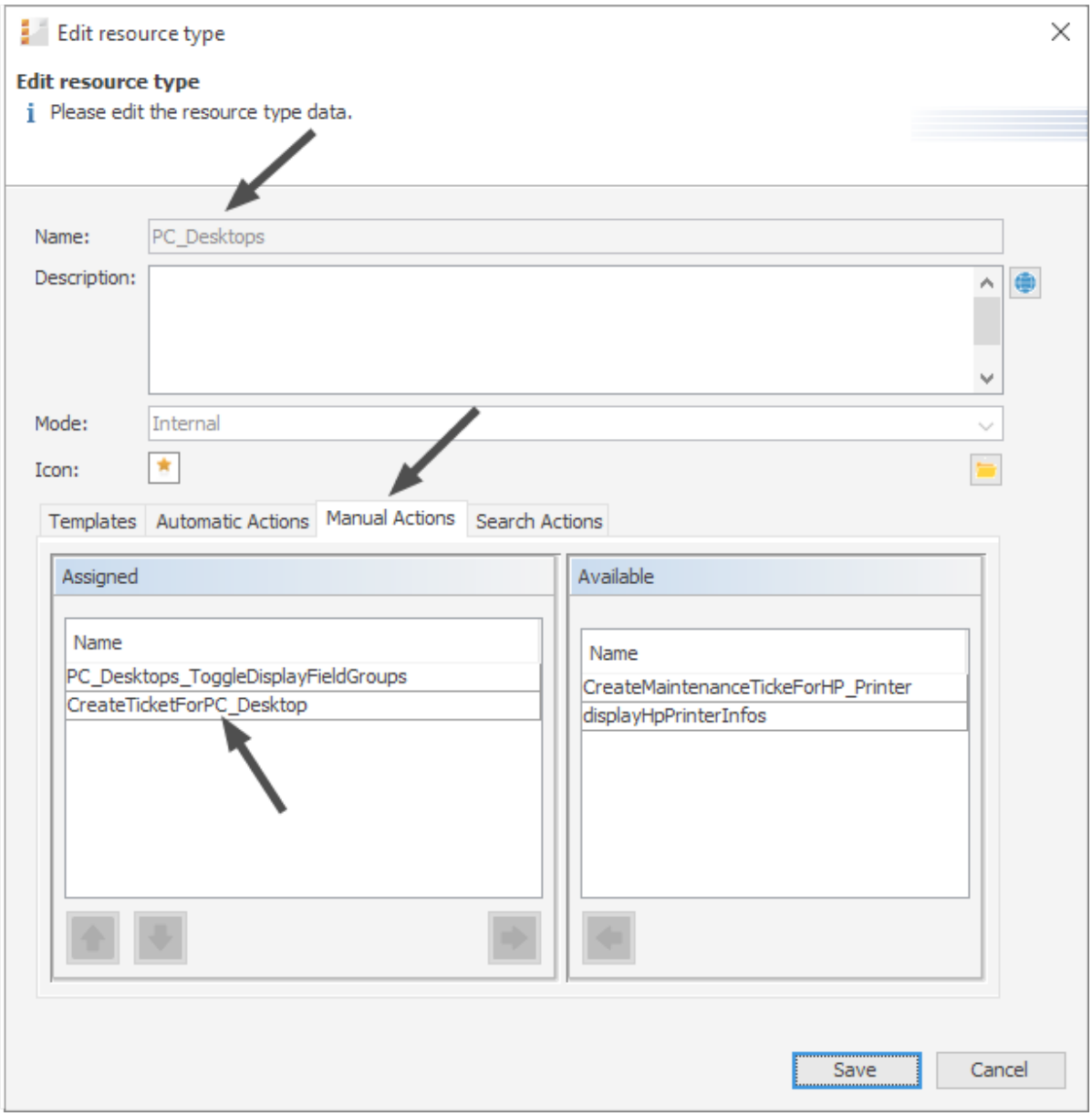


Figure 465: *ConSol CM Admin Tool - Resources, Data Models: Assigning manual resource actions to a resource type*

You can assign resource actions of the following action types:

- **Automatic Actions**
  Those actions will be executed automatically when the respective type of event (Create, Update, Delete, Relation) has been triggered. Select an action for each type which is required. Only the actions with the correct type (which have been defined as resource actions, see step 2) are listed in each drop-down menu. For example, for an automatic action of type *Create*, only resource actions of type *Create* (which have been defined in navigation item *Actions*) will be available.
- **Manual Actions**
  Those actions are displayed as activities on the resource page in the Web Client and have to be executed manually. An activity is only displayed if either no Resource Condition Script is present or if the respective resource condition script has returned “true”.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Search Actions**
  See section Action Framework - Search Actions.

G.1.7.3 Using Resource Actions in the Web Client (as an Engineer)

As an engineer (user), two resource action types are relevant for you because they are available as activities in the Web Client:

- **Manual**
  Manual actions are offered in the Web Client similar to workflow activities for a ticket. Please see *Example 1* in the next section.
- **Search**
  See section Action Framework - Search Actions.

The *CREATE*, *UPDATE*, *RELATION*, and *DELETE* actions run in the background.

G.1.7.4 Programming Resource Execution and Resource Condition Scripts

Please read the section about Scripts for the Action Framework for a general introduction about important principles, classes, and methods for execution and condition scripts.

G.1.7.5 Examples for Resource Actions

Example 1: Simple Manual Action

Use case: The engineer should be able to create a new Service Desk ticket directly from a resource page of a PC. The new ticket should be related to the resource (PC). The main contact of the new Service Desk ticket should be the person who is responsible for the PC. This is implemented as *resource-contact* relation in the resource type *PC_Desktops*. To implement the resource action, perform the following steps.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Write the resource action script:

```
// 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!

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) {
   client.failure("cmweb.rp.resource.action.no_contact_set")
} else {
   def cont_rel = cont_list[0]
   maincont = cont_rel.getTargetUnit()

   ticketService.createWithUnit(newtic,maincont)
   log.info '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)
   def resRelationDef = resourceRelationDefinitionService.getByCriteriaUniqueResult
    (resRelationDefCriteria)
   def resRelation = new ResourceTicketRelation(resRelationDef, resource, newtic)
   resourceRelationService.create(resRelation)

   // go to new ticket
   client.goToTicket(newtic)
}
```

Code example 95: *Resource action script for PC_Desktops to create a new Service Desk ticket for responsible PC contact*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Create a resource action based on the script:

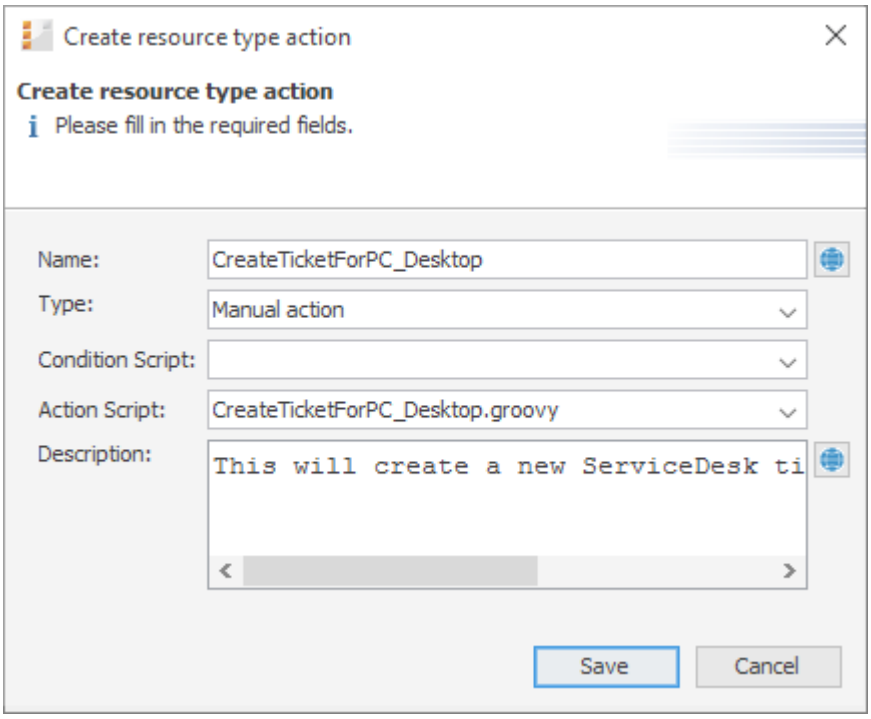


Figure 466: *ConSol CM Admin Tool - Resources, Actions: Creating a new resource action*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Assign the action to the correct resource type:

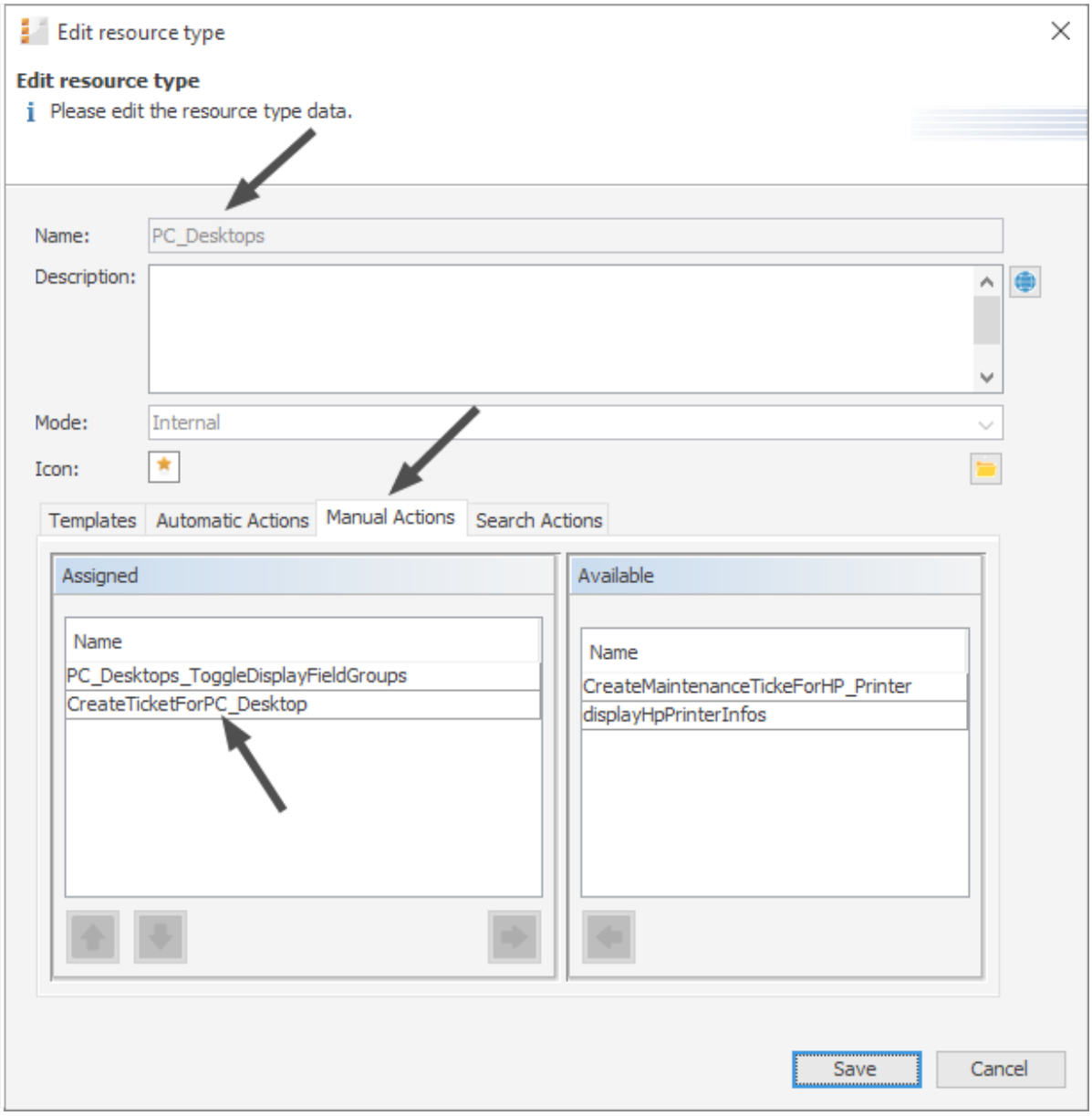


Figure 467: *ConSol CM Admin Tool - Resources, Data Models: Assign the resource action to the correct resource type*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Check the functionality using the Web Client:

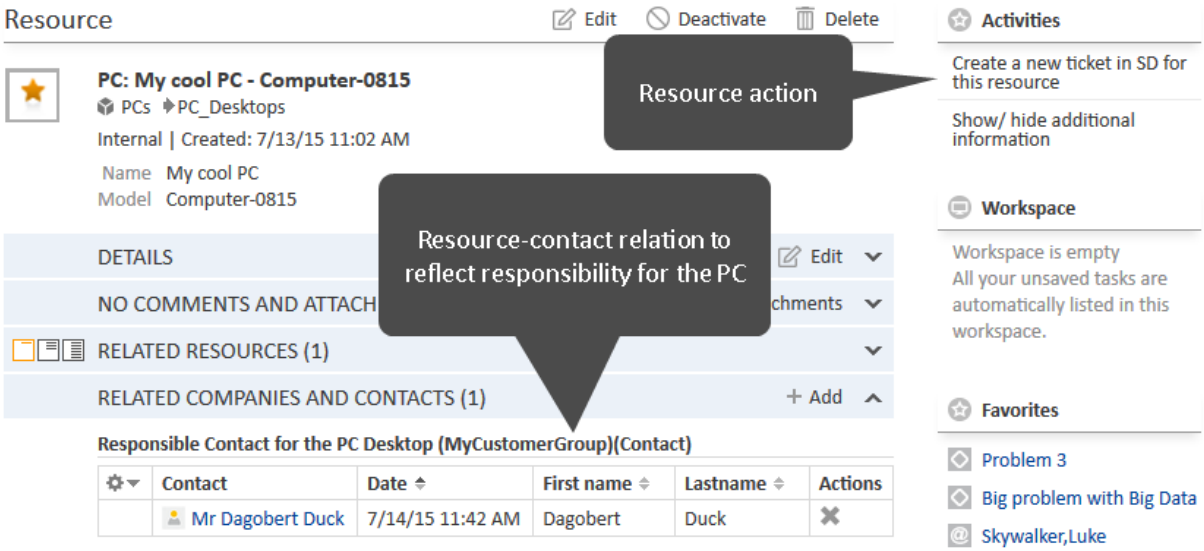


Figure 468: *ConSol CM Web Client - Resource page with the resource action*

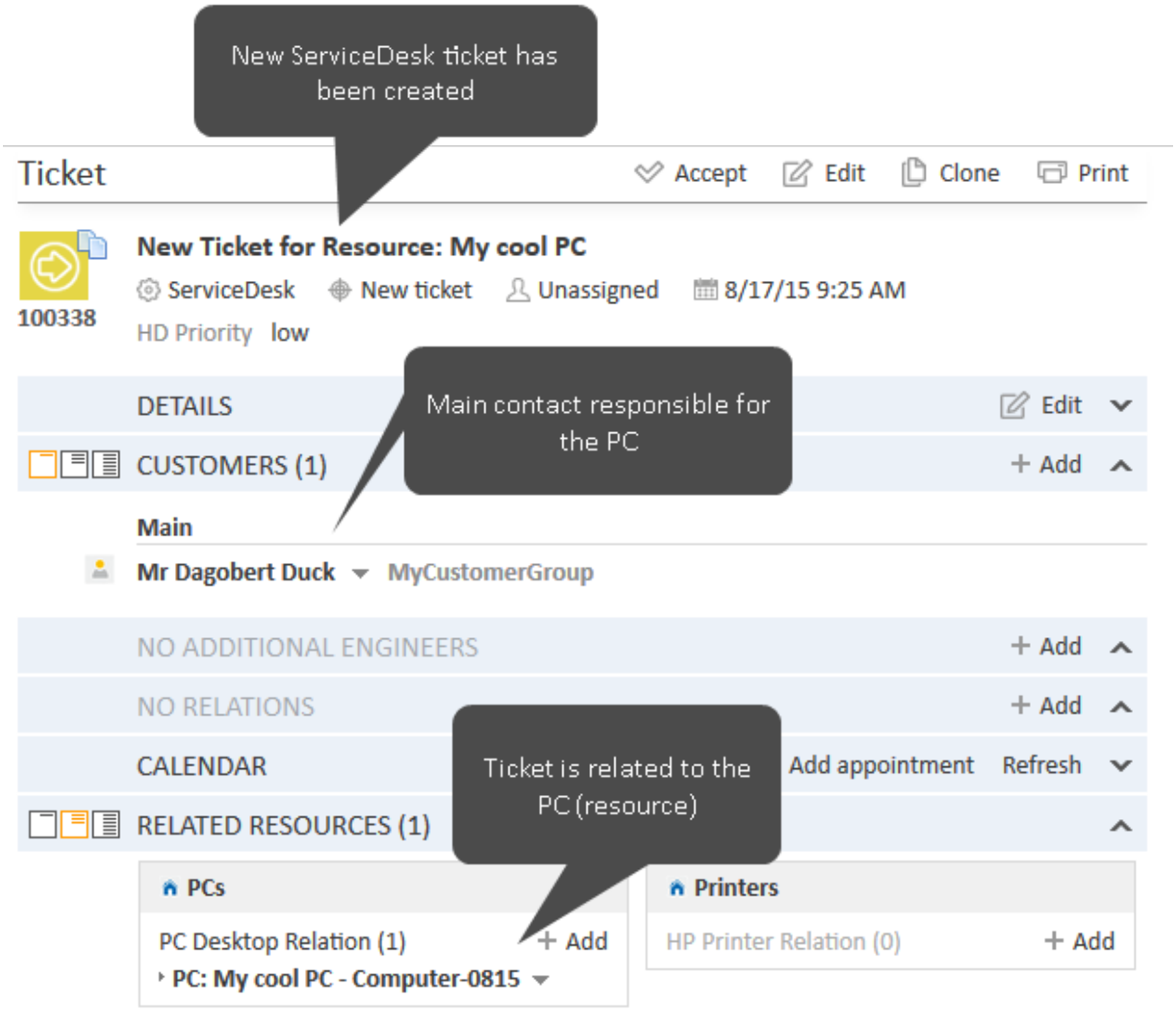


Figure 469: *ConSol CM Web Client - New Service Desk ticket created by the resource action*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

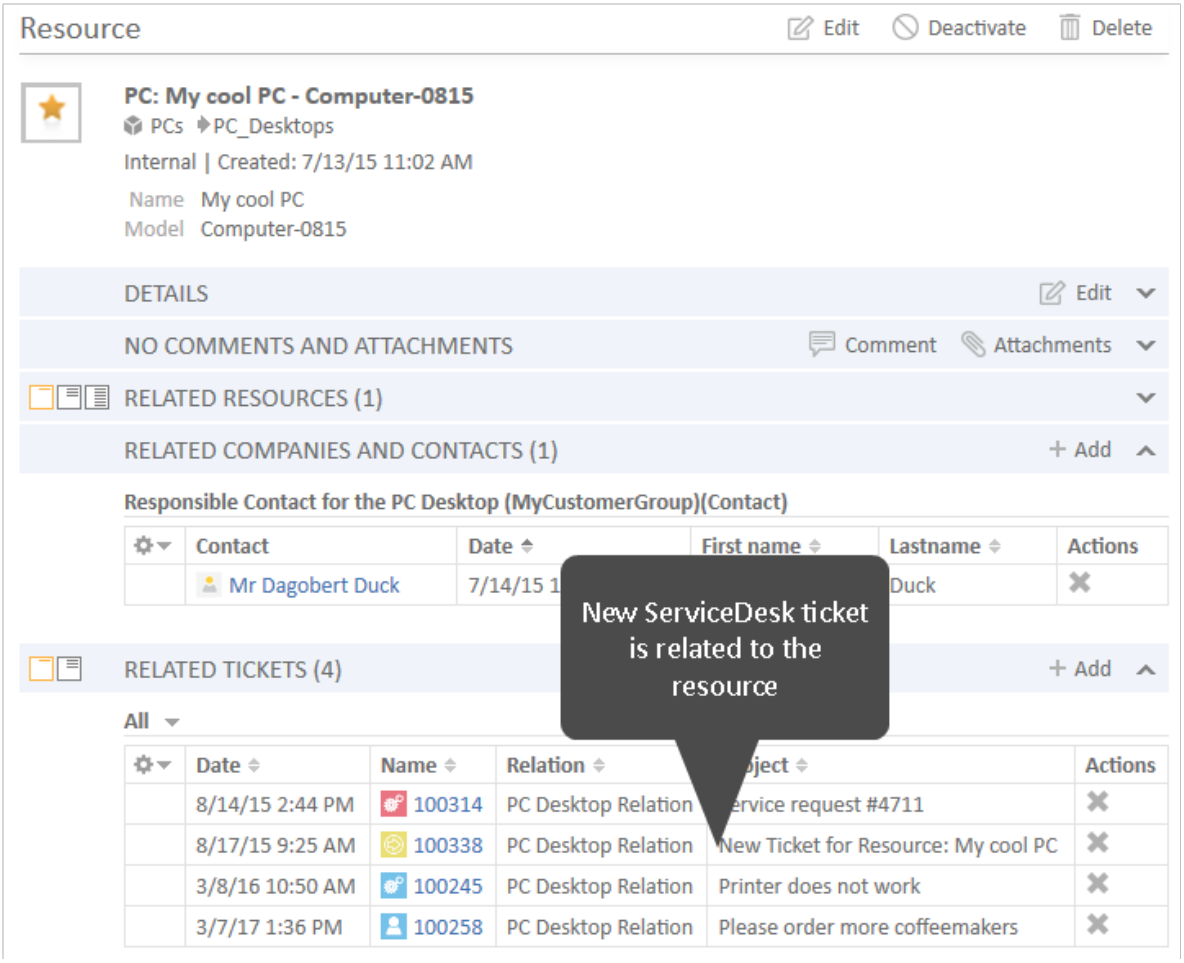


Figure 470: *ConSol CM Web Client - Resource page with one or more ticket relations (new Service Desk ticket)*

Example 2: Open a Maintenance Ticket from a Resource Using an ACF

The Action Framework offers the possibility to open an ACF (Activity Control Form) when a ticket is created. The ACF is used to gather data for the following workflow activity, i.e., the ticket can be created and moved through the first workflow step very easily. The following example demonstrates this functionality. A maintenance ticket should be created for a resource (an HP printer). During the (sub-) process, an ACF should be displayed to ask for data which will then be used during the next workflow activity.

To implement the resource action, perform the following steps.

Write the resource action script:

```
// 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!

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) {
   client.showErrorMessage("Add a relation to a customer first.")
} else {
   def cont_rel = cont_list[0]
   maincont = cont_rel.getTargetUnit()
   ticketService.createWithUnit(newtic,maincont)
   log.info '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)
   def resRelationDef = resourceRelationDefinitionService.getByCriteriaUniqueResult
    (resRelationDefCriteria)
   def resRelation = new ResourceTicketRelation(resRelationDef, resource, newtic)
   resourceRelationService.create(resRelation)

   // go to new ticket, but fill ACF before
   def executionContext = controlFormDefinitionService.getExecutionContext(newtic,
    "defaultScope/TaskInProgress/Aufgabe_annehmen")

   // set ticket number in list in resource:
   def newtic_id = newtic.id.toString()
   def newtic_name = newtic.name

   resource.add("HP_Printer_MaintenanceTickets.MaintenanceTicketsList",new Struct
    ().set("MaintenanceTicketID", newtic_id).set("MaintenanceTicketName", newtic_
    name)
   )

   if (!executionContext) {
      client.showErrorMessage("action.fail.wrong.activity")
   } else {
      // if ticktes are in the list in the resource, the field group should be
       visible

      def groupName = "HP_Printer_MaintenanceTickets"
      def fieldGroupDefinition = fieldDefinitionService.getGroupByName(groupName)

      if (fieldGroupDefinition == null) {
         throw new IllegalArgumentException("There is no group definition with name
          '" + groupName + "'.")
      }
```

```
      resource.getGroupsConfiguration().setProperty(fieldGroupDefinition,
       GroupPropertyType.VISIBLE, "true")

      // 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 example 96: *Resource action script which opens a ticket and uses an ACF*

Create a resource action based on the script.

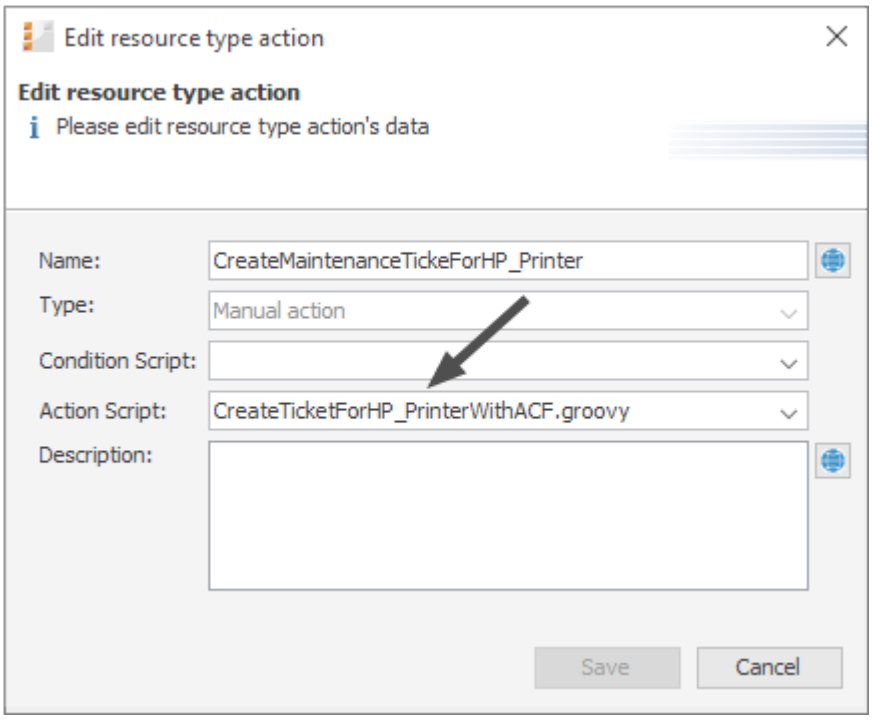


Figure 471: *ConSol CM Admin Tool - Resources, Actions: Create the resource action for the HP Printer maintenance ticket*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Assign the action to the correct resource type:

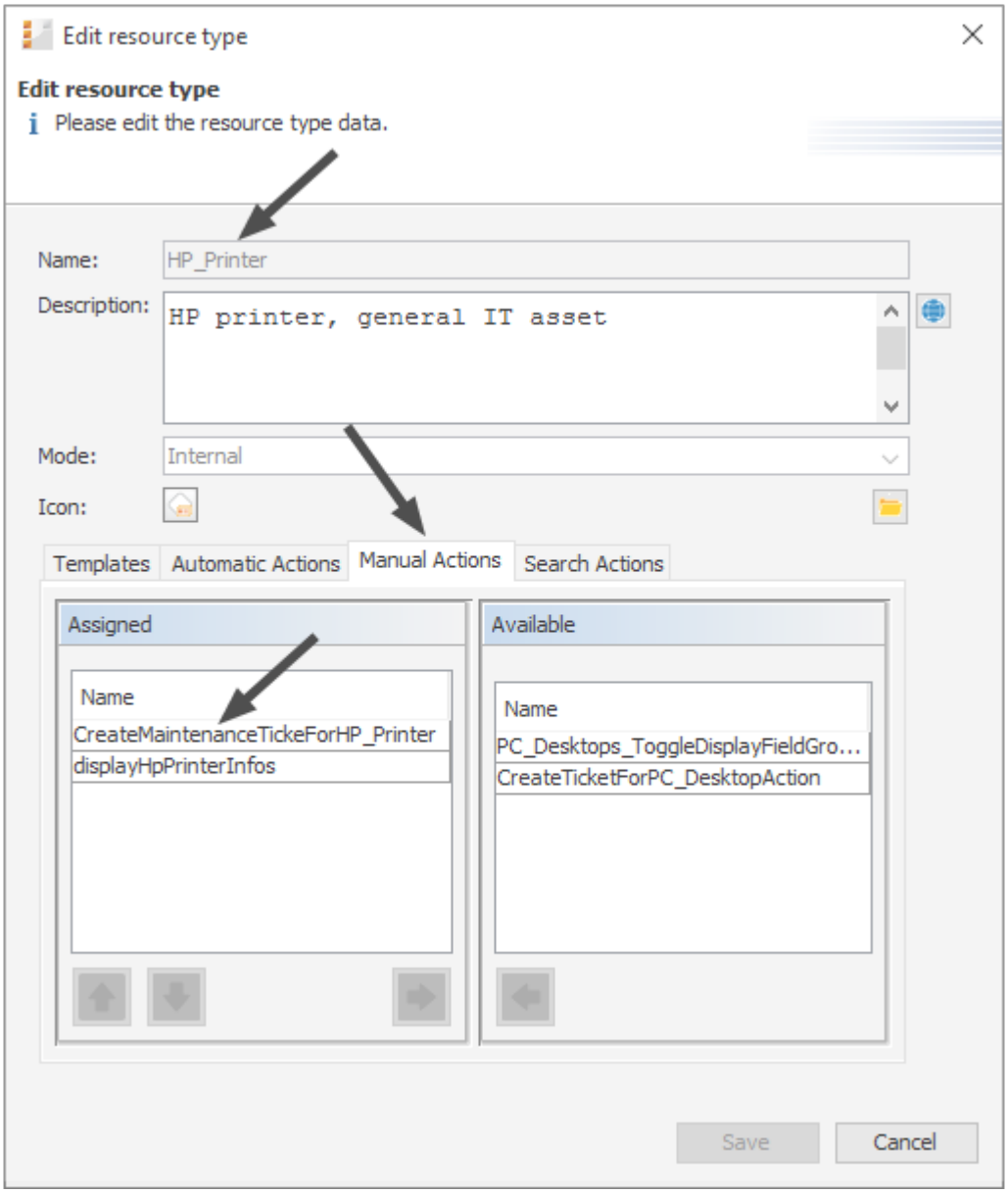


Figure 472: *ConSol CM Admin Tool - Resources, Data Models: Assigning the resource action to the correct resource type (HP Printer)*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Check the functionality using the Web Client:

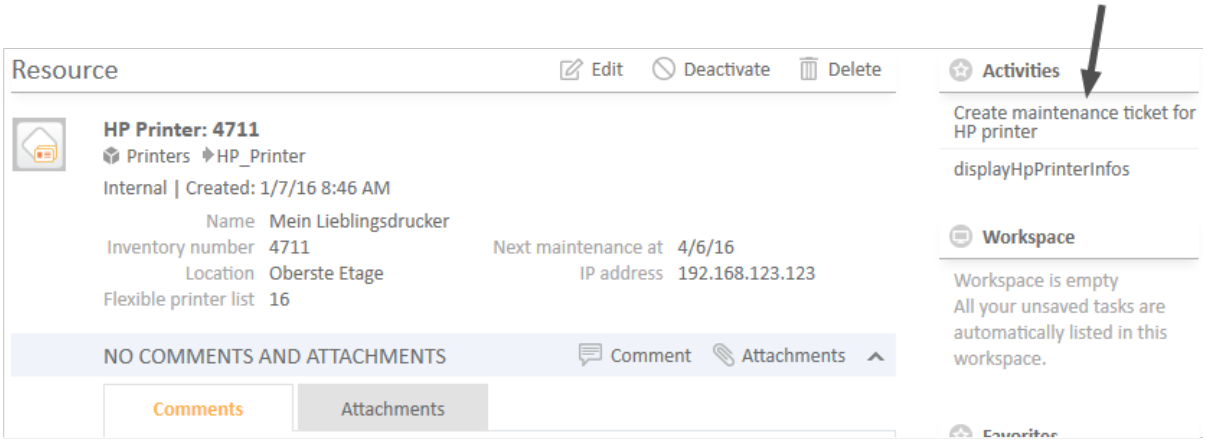


Figure 473: *ConSol CM Web Client - Resource action for HP Printer*

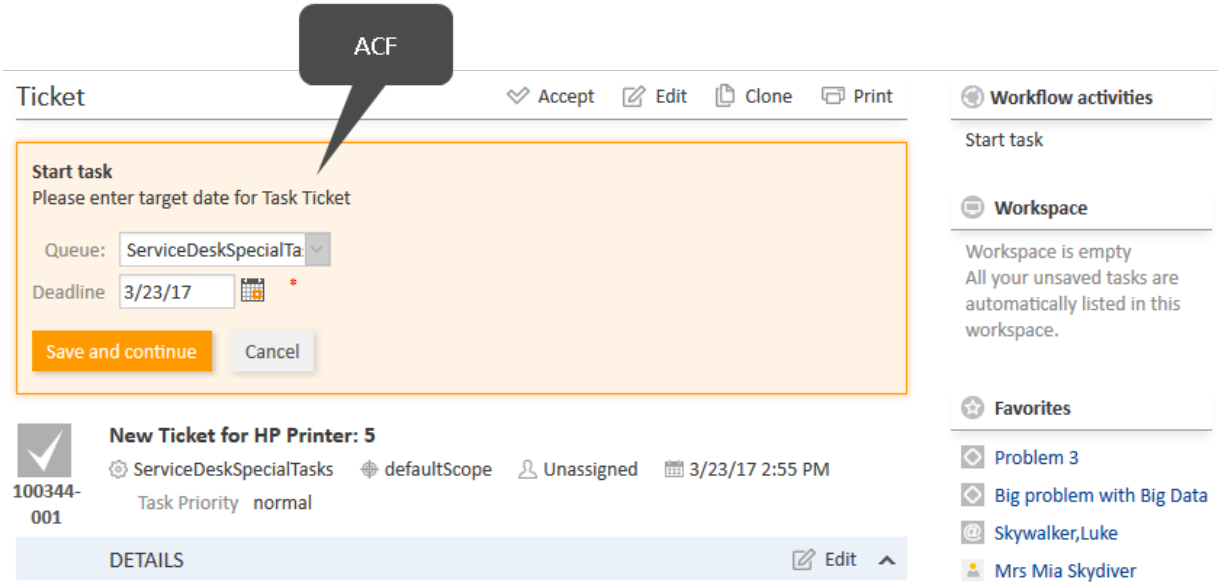


Figure 474: *ConSol CM Web Client - New maintenance ticket for resource (HP Printer), ACF*

Working with the Changes Object in Resource Update Actions

Starting with CM version 6.10.5.4, it is possible to monitor the changes which have been performed during a resource update action. (The same applies to unit Update actions, explained in section Working with the Changes Object in Customer Update Actions).

To find out which changes have been performed use the object of class `ResourceChanges` in resource actions.

Please remember, the Update script will be executed:

- in an explicit *Update* action
- when comments or attachments are added
- when comments or attachments are removed

There are two methods of the `ResourceChanges` object which provide information about the changed data:

- `getCustomFieldChangeInfo()`
  provides information about changes of resource data (in resource fields)
- `getContentChangeInfo()`
  provides information about changes in the resource history (comments, attachments)

Since the method return parameters contain rather complex components, we recommend to read the API doc of the `ResourceChanges` class. The following code provides an example for a script where a `resourceChanges` object is used.

```
/**
* 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 example 97: *Resource Update script where changes are monitored and printed out to server.log*

When for a resource of type *PC_Desktops*, the content of the two resource fields *modell* and *name* are modified, the following text is printed into the `server.log` file.

```
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 example 98: *Log output from the script above*

G.1.8 CM/Resource Pool - Assigning Permissions for Resources

G.1.8.1 Introduction to Permissions Concerning Resources

According to the ConSol CM basic principle, engineers can only access resources if they have the required permissions, i.e., if he has at least one role which has the required permissions. Resource permissions are granted at the resource type level, e.g., members of the role *ResourceManager_IT* can access the resource types *HP_Printer*, *MS_Word2013*, *PC_Desktops*, and *PC_Laptops*.

G.1.8.2 Assigning Resource Permissions Using the Admin Tool

To manage resource permissions, open the tab *Resource Types Permissions* (navigation group *Access and Roles*, navigation item *Roles*) in the Admin Tool.

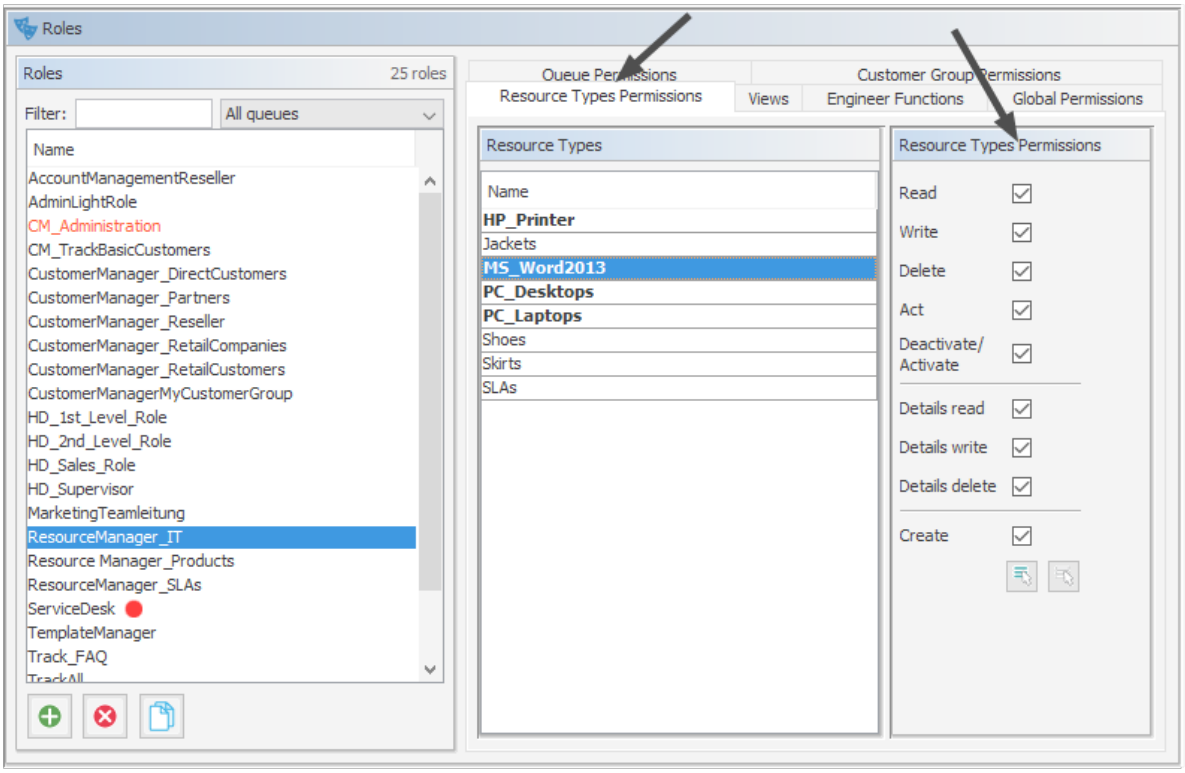


Figure 475: *ConSol CM Admin Tool - Access and Roles, Roles: Assigning resource permissions*

For a detailed description of the permissions, see section Tab Resource Types Permissions.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.1.9 CM/Resource Pool - The Resource Dashboard

The resource dashboard is displayed when the user clicks the *Resource pool* link in the main menu. The name of the link can be configured using a label, see Labels.

By default, the dashboard contains a bar chart with the number of resources of the different resource types in each resource category. The bar chart is configured in the script `resourcesDashboard.groovy`. Below the chart, there is a box for each resource category which contains a link to access the list of resources for each resource type and a link to create a new resource of the respective resource type.

As for other objects, only the resources for which the user has permissions are displayed.

The following example shows the resource dashboard:

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

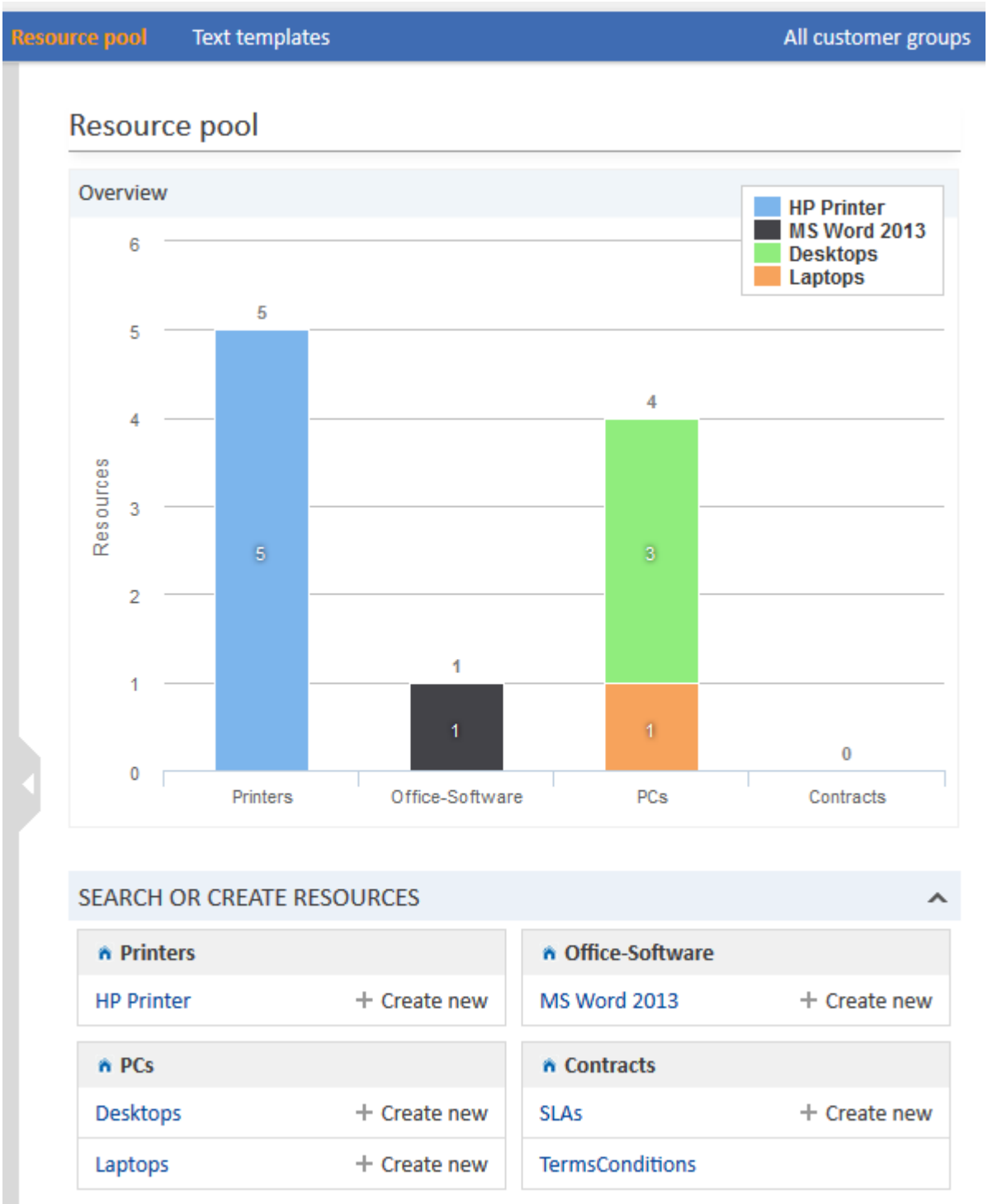


Figure 476: *ConSol CM Web Client - Resource dashboard with chart*

You can configure the dashboard by removing the bar chart or adding additional widgets. The configuration is done in the page customization. See Dashboards for a detailed explanation of the available widget types and required steps.

G.2 CM/Track: The Customer Portal

G.2.1 Overview

G.2.1.1 Introduction

The customer portal ***CM/Track*** allows customers to log in to the ConSol CM system. Like the ConSol CM Web Client, CM/Track is a web-based application, i.e., the customer only needs a standard web browser for access to the portal.

Technically, the data for CM/Track is retrieved using the REST API.

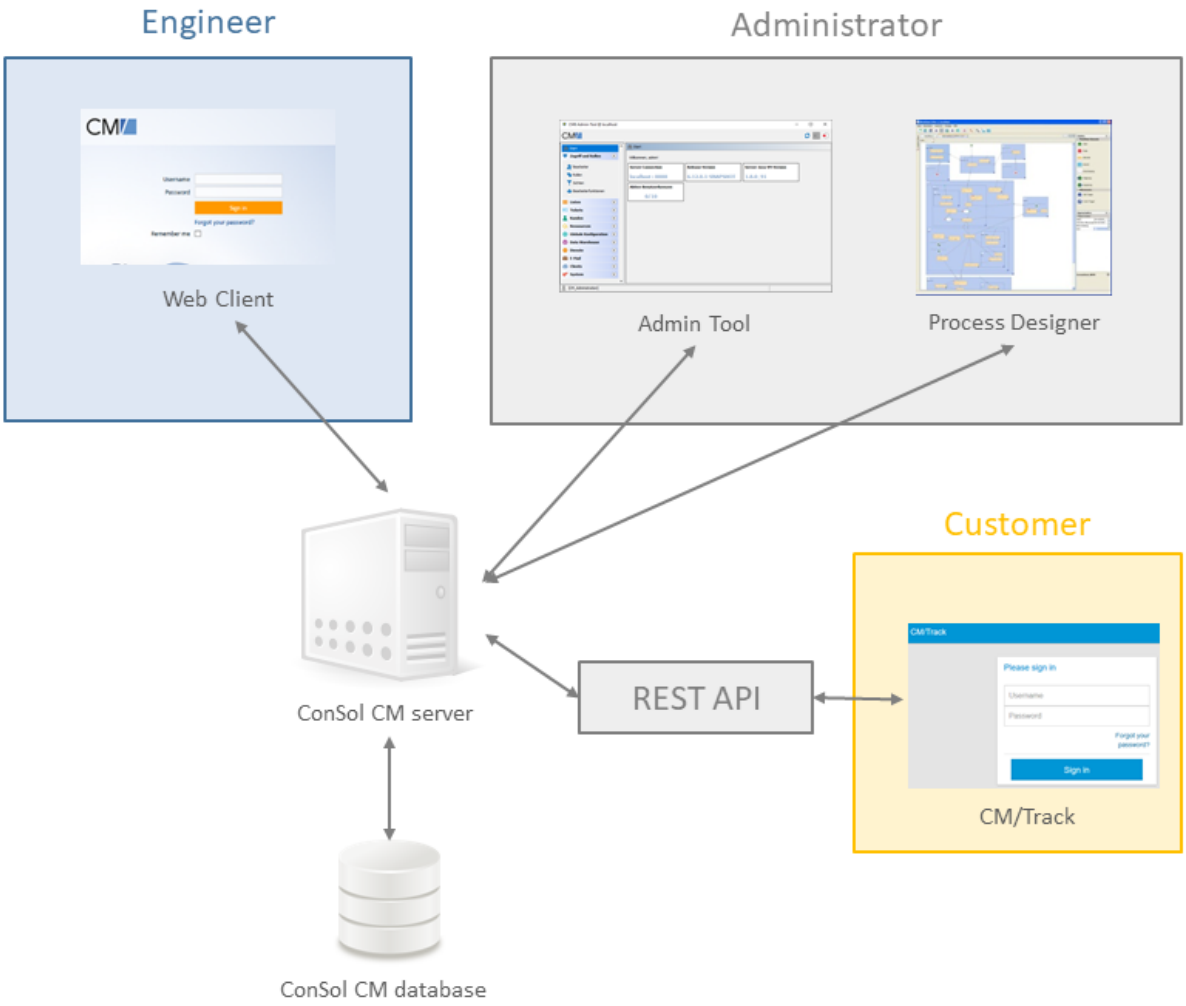


Figure 477: *ConSol CM system architecture with CM/Track*

In a standard environment, a customer can perform the following operations via CM/Track:

- See a list of his tickets.
- See a list of all tickets of his company (if this has been configured).
- Add comments and/or attachments for a ticket.
- Open/Create a new ticket.
- Search the FAQs for solutions.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.2.1.2 CM/Track Versions

The current version of CM/Track is **CM/Track V2**. The support for CM/Track V1 was phased out with ConSol CM version 6.11.2.2.

> ⚠ If you run CM/Track V1 and perform an update to CM version 6.11.2.2 you must migrate to CM/Track V2, which includes adapting V2 in the same way V1 was adapted, if you do not use the standard flavor of CM/Track. Please note that there is **no automatic update** from V1 to V2, since the two are separate web applications which are deployed in the application server!

G.2.1.3 Detailed Information about CM/Track-Related Topics

- Customer authentication methods in CM/Track:
  see section: Authentication Methods for Customers in CM/Track.
- General system access to CM/Track for customers:
  see section CM/Track: System Access for CM/Track Users (Customers).
- Managing the availability of ticket fields in CM/Track:
  see section CM/Track: Data Availability For Customers
- Using the portal for FAQ:
  see section CM/Track: FAQs in CM/Track.
- Configuring the Welcome Page of CM/Track:
  see section Configuration of the Welcome Page

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.2.2 Installing CM/Track

This version needs Java 8.

CM/Track can be deployed in the application server where CM is running or in a separate servlet container (currently supported: Tomcat). The latter allows to run CM/Track as a separate portal which could be operated in another security context than CM, e.g., in a DMZ. If you deploy CM/Track in a separate Tomcat servlet container but on the same machine as JBoss, please avoid port conflicts! JBoss as well as Tomcat use the 8080 port as default.

Perform the following steps:

1. Deploy the `.war` file:
   - In case the standard version of CM/Track is used, you have to deploy the standard `.war` file, usually named `cmtrack-v2-distribution-<CM_VERSION>.war` (example: `cmtrack-v2-distribution-6.12.0.1.war`).
   - In case a customer-specific `.war` file is used, you have to deploy this specific `.war` file.
   - Deployment together with CM:
     Just deploy the `.war` file mentioned above. You do not need to change the file name.
   - Deployment in a Tomcat servlet container:
     Rename the `.war` file to `track.war`.
2. By default, CM/Track will look for the CM server / REST API on `http://localhost:8888`. In order to change this default setting, you need to perform one of the following steps:
   - Set the respective system variable for the REST API in the file `standalone.conf` (Linux) or `standalone.conf.bat` (Windows)

     Linux:

     ```
     export CM_REST_URL=<PROTOCOL>://<CM_SERVER>:<CM_PORT>/restapi
     ```

     Windows:

     ```
     set CM_REST_URL=<PROTOCOL>://<CM_SERVER>:<CM_PORT>/restapi
     ```

   - Add the following Java option to the start script:

     ```
     -Djersey.cm6.url=<PROTOCOL>://<CM_SERVER>:<CM_PORT>/restapi
     ```

3. Restart the CM server during deployment.
4. Test the deployment by calling the following URL:

   ```
   <PROTOCOL>://<CM_SERVER>:<CM_PORT>/track/
   ```

The login screen should be displayed.

G.2.3 Authentication Methods for Customers in CM/Track

G.2.3.1 Available Authentication Methods

There are three possible authentication modes:

- Against the ConSol CM database
  This is called *DATABASE mode*, see Database Authentication for Customers
- Against an LDAP server
  This is called *LDAP mode*, see LDAP Authentication for Customers in CM/Track
- Against an LDAP server and the ConSol CM database
  The order can be configured. This is called *Mixed mode*, see Mixed Authentication Mode

G.2.3.2 Defining the Authentication Method

The authentication mode is specified by the system property cmas-core-security, contact.authentication.method. A change of this property does not require a server restart, and is propagated to all cluster nodes.

The possible values (see also section System Properties) and their respective system behaviors are:

- **DATABASE**
  Attempt a database login if the customer has a database password. I.e., the login and password are stored in the ConSol CM database and are thus managed by the ConSol CM engineers, or indirectly by the customers themselves when they reset their password. The customer can reset his own password, see section Password Reset Template for Customers Using CM/Track.
- **LDAP**
  Try authentication using the available LDAP server(s), if an LDAP ID is provided. I.e., the password is stored in the LDAP directory and cannot be changed via ConSol CM, neither by the customer nor by an engineer.
- **LDAP,DATABASE**
  First attempt authentication using the available LDAP server(s), if an LDAP ID is provided. On failure, try a database login if the customer has a database password.
- **DATABASE,LDAP**
  First attempt a database login if the customer has a database password. On failure try authentication using the available LDAP server(s) if an LDAP ID is provided.

The values are case insensitive, and commas and whitespace are ignored.

G.2.3.3 Database Authentication for Customers

Database authentication is activated by setting the system property cmas-core-security, contact.authentication.methodto “DATABASE” (default value).

There are two steps which you need to perform to set up database authentication for customers using CM/Track:

- Create customer fields for the user name (login) and password in the Admin Tool (see Defining the Customer Fields for CM/Track Login and Password)
- Enter the user name and password for the actual customers in the Web Client (see Granting Access to CM/Track for Customers)

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

When database authentication is used, you can allow your customers to change their own passwords, see Configuring CM/Track for Password Reset by Customers.

Defining the Customer Fields for CM/Track Login and Password

The fields for login and password for a customer are regular customer fields at the contact level. Please see section Setting Up the Customer Data Model for an introduction to customer field management and GUI configuration for customer data.

Edit the customer data model in the Admin Tool (navigation group *Customers*, navigation item *Data Models*). If there are two levels you need to edit the fields of the contact level. Create the following two fields:

- Create a field for the login with the annotation `username` = “true”.

  > ⚠ Assigning the annotation `username` to a customer field is only possible, if there is no previous assignment of this annotation. Otherwise it will be prohibited. When assigning it, a warning dialog must be confirmed before it is executed, since it can be a longer running operation. Unassigning the annotation must be confirmed as well, because it cannot be undone: Unassignment deletes the user name values unrecoverably from internal storage.

- Create a field for the password with the annotation `password` = “true”. The annotation `text-type` = “password” guarantees that only stars/dots are displayed in the Web Client, not the clear text password.

  > ⚠ The annotation `password` requires confirmation when assigned.
  >
  > In case of an update from CM versions lower than 6.11 to 6.11 and up: When this annotation is set, the system reads the plain text passwords from the original field values, encrypts them and saves the encrypted values to the internal storage. Then the original field values are deleted and thus the plain text value cannot be recovered anymore.
  >
  > When trying to unassign the `password` annotation the operation must be confirmed as well, since the encrypted passwords are deleted from the internal storage. After the annotation unassignment the password information is completely lost and cannot be recovered at all.

When a scenario from a CM version lower than 6.11 is imported into a system with CM 6.11 (or higher), a transformation of user names and passwords is performed automatically. This is described in detail in section Transformation of User Name and Password Fields During Import into CM 6.11 .

Granting Access to CM/Track for Customers

The engineer working with the Web Client can then assign a user name, initial password, and a CM/Track user profile to every customer who should have access to the portal CM/Track. The user name has to be unique. This is checked by the system. You cannot enter a name a second time if this has already been assigned to another customer. The password is stored as encrypted string in the CM database. This means that an engineer can set a new password, e.g., when a customer calls and asks for this, but it is never possible to read the password from the system.

> You, as an administrator, can define if the CM/Track user names should be case sensitive. Use the CM system property cmas-core-security, policy.track.username.case.sensitive. This is a boolean variable. When it is set to "true", the CM/Track user names are case sensitive. Please make sure that the database collation which is in use supports case sensitive strings!

The following example shows the customer data of an example contact in the ConSol CM Web Client. You reach this screen by opening a contact data set in edit mode.

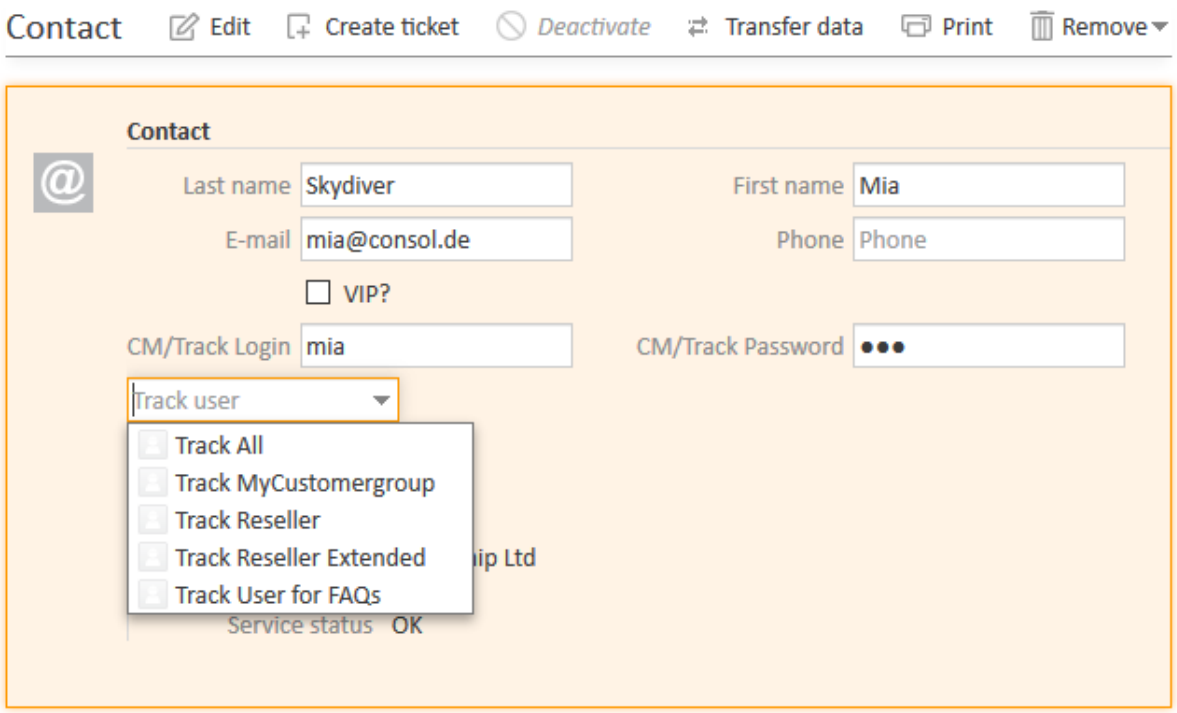


Figure 478: *ConSol CM Web Client - Contact page: CM/Track user data*

Configuring CM/Track for Password Reset by Customers

CM/Track can be configured to offer a hyperlink for customers where a customer can reset his password. This is based on the template `track-password-reset-template`. Please refer to section Password Reset Template for Customers Using CM/Track for a detailed explanation. The password reset in CM/Track is only possible when the DATABASE mode is used. It is not possible when LDAP authentication is in operation. See section Authentication Methods for Customers in CM/Track for the portal for an explanation of all possible authentication modes.

Please note that the `From`address of the email which is sent to a customer who has requested a new password can be set using the CM system property cmas-core-security, password.reset.mail.from.

G.2.3.4 LDAP Authentication for Customers in CM/Track

Introduction to ConSol CM LDAP Authentication

ConSol CM offers LDAP authentication for CM/Track as a standard feature, i.e., instead of managing the passwords for the ConSol CM customers in the ConSol CM database, they can be retrieved from an LDAP server (like e.g., a ***Microsoft Active Directory*** server).

When customers want to log in to CM/Track, they enter their user name and password and press *Enter*. Behind the scenes, the ConSol CM server sends a request with the customer's user name and password and asks the LDAP server whether those credentials are correct.

If the credentials are correct, the approval is sent back to the ConSol CM server and the customer is logged into CM/Track.

> Please keep in mind that the LDAP connection is only used to authenticate the customer (confirm the identity). The authorization (i.e., the assignment of access permissions in the system) is done via the assignment of a CM/Track user profile in the Web Client. The CM/Track user profiles are managed in the engineer and role administration in the Admin Tool.

Please see also the following picture for an explanation of the CM/Track authentication process using LDAP.

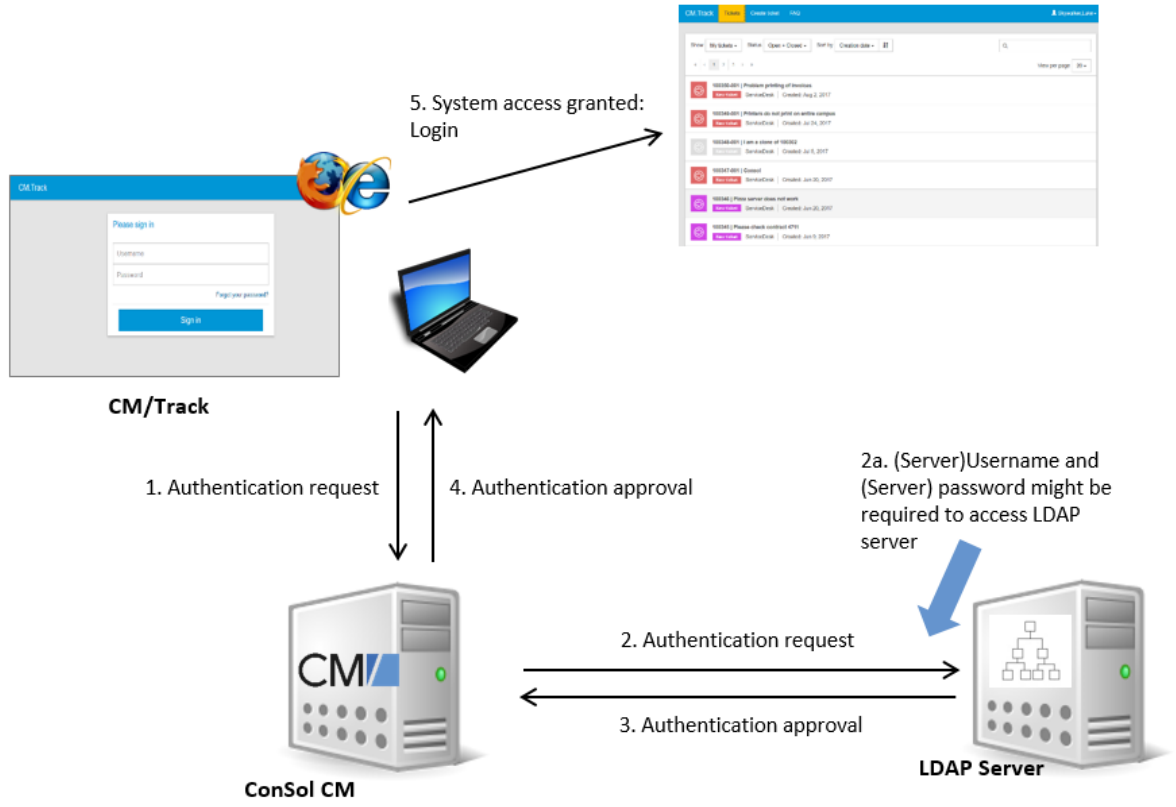


Figure 479: *ConSol CM - LDAP authentication process for CM/Track*

Configuring LDAP Authentication

LDAP authentication is activated by setting the system property cmas-core-security, contact.authentication.method to “LDAP”.

Then you have to set the required values in the system properties (navigation group *System*, navigation item *System Properties*, see System Properties).

The LDAP servers can be defined using the following system properties from the module `cmas-core-security`.

{name} is a string that you can choose to distinguish LDAP servers. It must always be set, even if only one LDAP server is configured. You should use a simple string for the {name}, not containing any keywords, like *internal* or *external*, and which does not contain special characters.

- cmas-core-security, contact.authentication.method
  LDAP
- ldap.contact.{name}.providerurl
  The property value is the address of the LDAP server in the form *ldap[s]://host:port*.
- ldap.contact.{name}.userdn
  The value is the user DN used to look up the contact DN by the LDAP ID. An anonymous account is used if the value is not set.
- ldap.contact.{name}.password
  The property contains the password to look up the contact DN by the LDAP ID. An anonymous account is used if the value is not set.
- ldap.contact.{name}.basedn
  This represents the base path to search for the contact DN by the LDAP ID, e.g., “ou=accounts,dc=mycompany,dc=de”.
- ldap.contact.{name}.searchattr
  The property value stands for the attribute to search for the contact DN by the LDAP ID, e.g., “uid”.

Initially, these system properties might not be present in your CM system. Just add them manually. Changes to any of the above system properties do not require a server restart and are propagated to all cluster nodes. The use of the placeholder {name} allows configurations to define several different LDAP servers.

- ldap.initialcontextfactory
  This is a predefined global property. If it is not set, “com.sun.jndi.ldap.LdapCtxFactory” is used as its value.

Authentication attempts against LDAP servers are made until first success, where the server order is determined by their {name} values (ascending alphabetical order of the values).

Mixed Authentication Mode

Set the system property cmas-core-security, contact.authentication.method depending on the desired order of authentication instances:

- **LDAP,DATABASE:**
  First attempt authentication using the available LDAP server(s), if an LDAP ID is provided. On failure, try a database login if the customer has a database password.
- **DATABASE,LDAP:**
  First attempt a database login if the customer has a database password. On failure try authentication using the available LDAP server(s) if an LDAP ID is provided.

The CM system will first contact the instance which is mentioned first, than the second one. For example, when the contact authentication method is set to “LDAP,DATABASE” and the customer (contact) uses the password which is only valid in the database, the login will succeed.

In `server.log` the following message will be displayed:

```
LDAP login failed: [LDAP: error code 49 - Invalid Credentials]; nested exception is
 javax.naming.AuthenticationException: [LDAP: error code 49 - Invalid Credentials]
```

Logging of LDAP Login Attempts in CM/Track

All LDAP errors encountered are logged without a stack trace using loggers with the following prefix:

- `com.consol.cmas.core.security.contact`

The stack trace of LDAP errors is not logged because failed login attempts on the first LDAP server would clutter logs if a following login on the second LDAP server succeeded.

Using LDAPS (LDAP over SSL)

Introduction

Per default, when an LDAP client accesses an LDAP server, the information is transferred in clear text. In case you want the user name and password to be transferred to the LDAP server in encrypted form, you have to set up the LDAP authentication using LDAPS.

Preparations

You have to configure the CM server machine (Java) in a way that can use certificates. One way to do this for a Linux environment is described in the following section.

1. Retrieve the certificate:

   ```
   openssl s_client -connect dc2.mydomain.com:ldaps
   ```

2. The answer will contain a section which starts with `"---BEGIN CERTIFICATE "` and ends with `"END CERTIFICATE ---"`. Copy this section to a file, e.g., `/tmp/certificate2_dc2_mydomain_com.txt`
3. Import the certificate to the truststore of your machine, e.g., `/home/mydirectory/mytruststore`

   ```
   $JAVA_HOME/bin/keytool -import -alias <arbitrary> -trustcacerts -keystore
    /home/mydirectory/mytruststore -file/tmp/certificate2_dc2_mydomain_com.txt
   ```

   You have to enter (set) a password.
4. Enter the truststore in the ConSol CM config file in JAVA_OPTS:

   ```
   -Djavax.net.ssl.trustStore=/home/mydirectory/mytruststore -
    Djavax.net.ssl.trustStorePassword=<see above>
   ```

LDAPS Configuration in the ConSol CM Admin Tool (System Properties)

Configure the ConSol CM server as shown in the following example:

- 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

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- 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

Depending on the LDAP server configuration, use one of the following values for the 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

Setting Up Customer Accounts for LDAP

There are two steps which you need to perform to set up LDAP authentication for customers using CM/Track:

- Set the required annotation for the customer field which should hold the LDAP ID in the Admin Tool.
- Enter the LDAP IDs for the actual customers in the Web Client.

When LDAP mode is used, the customer field which is used for the CM/Track user name (login) has to have two annotations:

- `username` = true
- `ldapid` = true

G.2.4 CM/Track: System Access for CM/Track Users (Customers)

In the following chapter you will find detailed information about how to configure your ConSol CM system to grant access to CM/Track (the ConSol CM portal) to your customers.

> CM/Track is a ConSol CM add-on which has to be purchased separately.
>
> Please note that for every CM/Track user (i.e., user profile), a ConSol CM license is required. Since numerous customers can log in to CM/Track using one user profile, you do not need a ConSol CM license for every customer. Please refer to section License Management for details.

G.2.4.1 Precondition

CM/Track is not part of every default shipment of ConSol CM. When you (your company) have purchased the license, you will receive a .war file which has to be deployed in the application server. Some minor modifications then have to be made in CM configuration files in the application server in order to operate CM/Track. This is all are explained in the *ConSol CM Setup Manual*.

The default function set of CM/Track provides basic functionalities (e.g., viewing a ticket list, creating a new ticket, seeing ticket details) and the pages have a CM standard layout. In order to use CM/Track as a powerful portal for customer access to the system, the layout should be adapted to a company's CD (corporate design), a process called *skinning*. The forms and lists which are displayed for the customer might be modified and/or extended. Please contact our consulting team or your account manager if you would like to adjust CM/Track for your company in an optimal way.

> To enable access to CM/Track, communication between ConSol CM and CM/Track has to be allowed using the following system properties cmas-restapi-core, csrf.request.filter.enabled, cmas-restapi-core, csrf.domain.white.list, and cmas-restapi-core, csrf.domain.allow.none.

G.2.4.2 CM/Track Technical Background

The portal CM/Track is based on the ***REST API*** of ConSol CM. Please refer to the *ConSol CM REST API Documentation* for details.

G.2.4.3 General Principle of System Access via CM/Track

A customer who wants, or should, have access to your ConSol CM system using the portal CM/Track has to have a login and a password. Both can be initially provided by the engineer who edits the customer data using the ConSol CM Web Client, or the values can be imported automatically into the database.

The fields for the login and password of customers are customer fields which are defined like any other customer field and which have special annotations. If you are not familiar with customer fields, please refer to section Customer Field Management and GUI Design for Customer Data.

The access permissions of the customer are defined by assigning a CM/Track user profile to the customer's account. The CM/Track user profiles are managed by the ConSol CM administrator using the Admin Tool.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.2.4.4 Defining the User Profiles/Access Permissions for CM/Track

As one of the first steps, you have to define CM/Track user profiles, i.e., profiles of access permissions to CM/Track. A CM/Track user profile is defined like a regular engineer (please see section Engineers for details), but is marked as *Track*.

The following example shows some CM/Track user profiles in the Admin Tool. You reach this screen by opening the navigation group *Access and Roles*, navigation item *Engineers*.

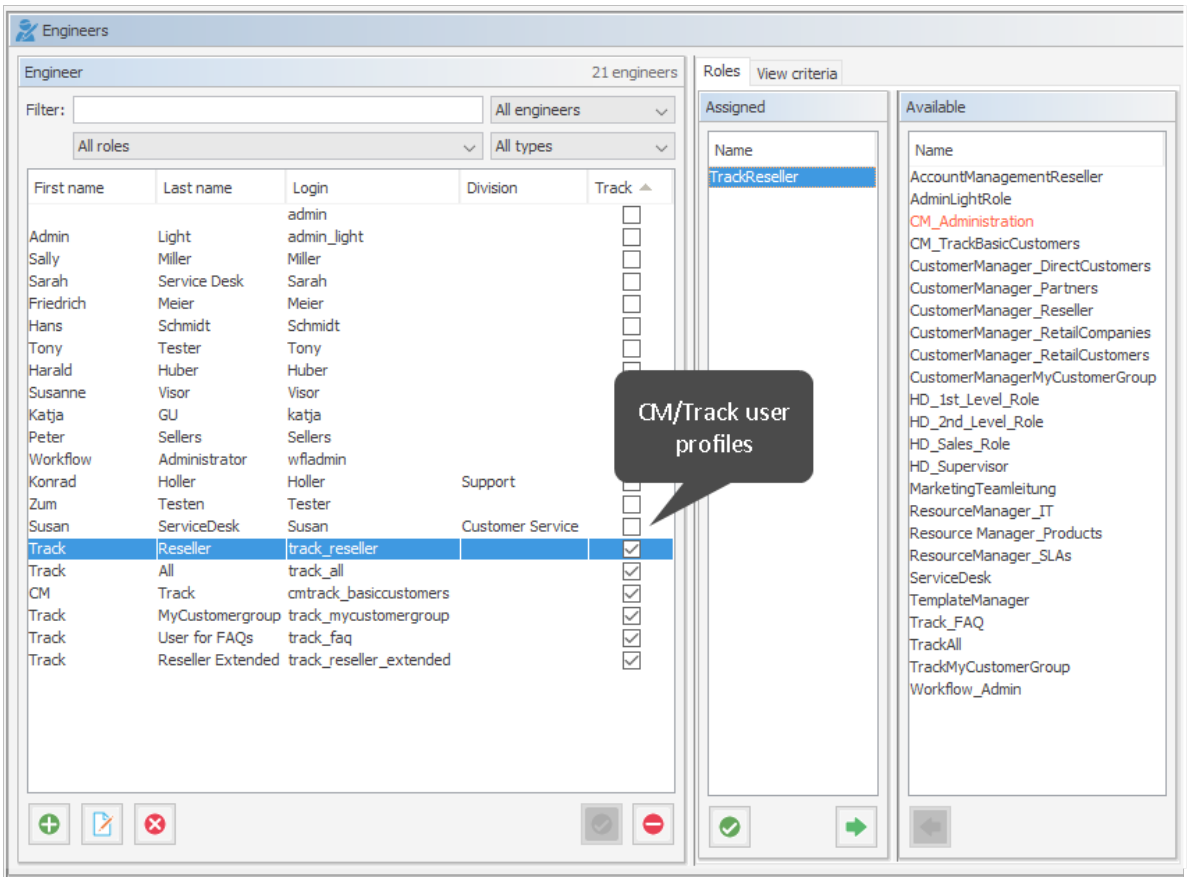


Figure 480: *ConSol CM Admin Tool - Access and Roles, Engineers: User profile name for CM/Track*

One or more roles are then assigned to the user profile to define the access permissions to queues and customer groups. For example you can set up a user profile (engineer) *track_reseller* that has the role *TrackReseller*. This role has read/write/append permissions for the queues *FAQs_active*, *Helpdesk_1st_Level*, *Helpdesk_2nd_Level*, *ServiceDesk*, and *SpecialTasks* and has customer group permissions for one customer group.

Please note that

- always queue **and** customer group permissions have to be granted to allow ticket access via CM/Track for customers.
- you must assign matching queue and customer group permissions, i.e., assign queues where the respective customer groups have been assigned (see section Queues).
- read permissions for customer groups will be sufficient in most (standard) cases, since it is not possible to edit customer data using the portal.
- write permissions for the “own” customer group are required for the customer to be allowed to reset his password.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

In our example, the role *TrackReseller* has read and write access to the *Reseller* customer group. You reach the following screens by opening the navigation group *Access and Roles*, navigation item *Roles*. In your system, it might be required to create different CM/Track roles with access to different customer groups. For a detailed introduction to role administration, please refer to section Roles.

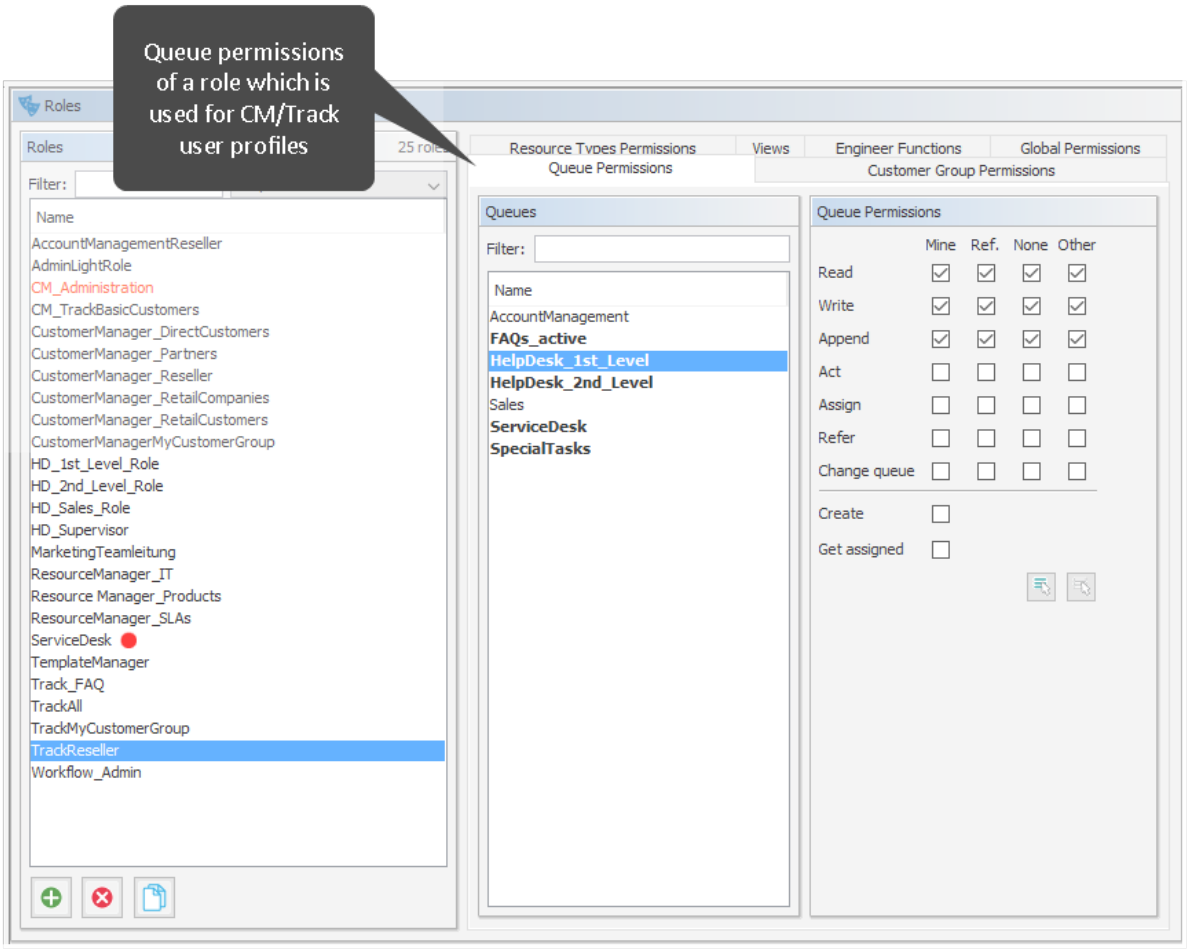


Figure 481: *ConSol CM Admin Tool - Access and Roles, Roles: User profile for CM/Track, queue permissions*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

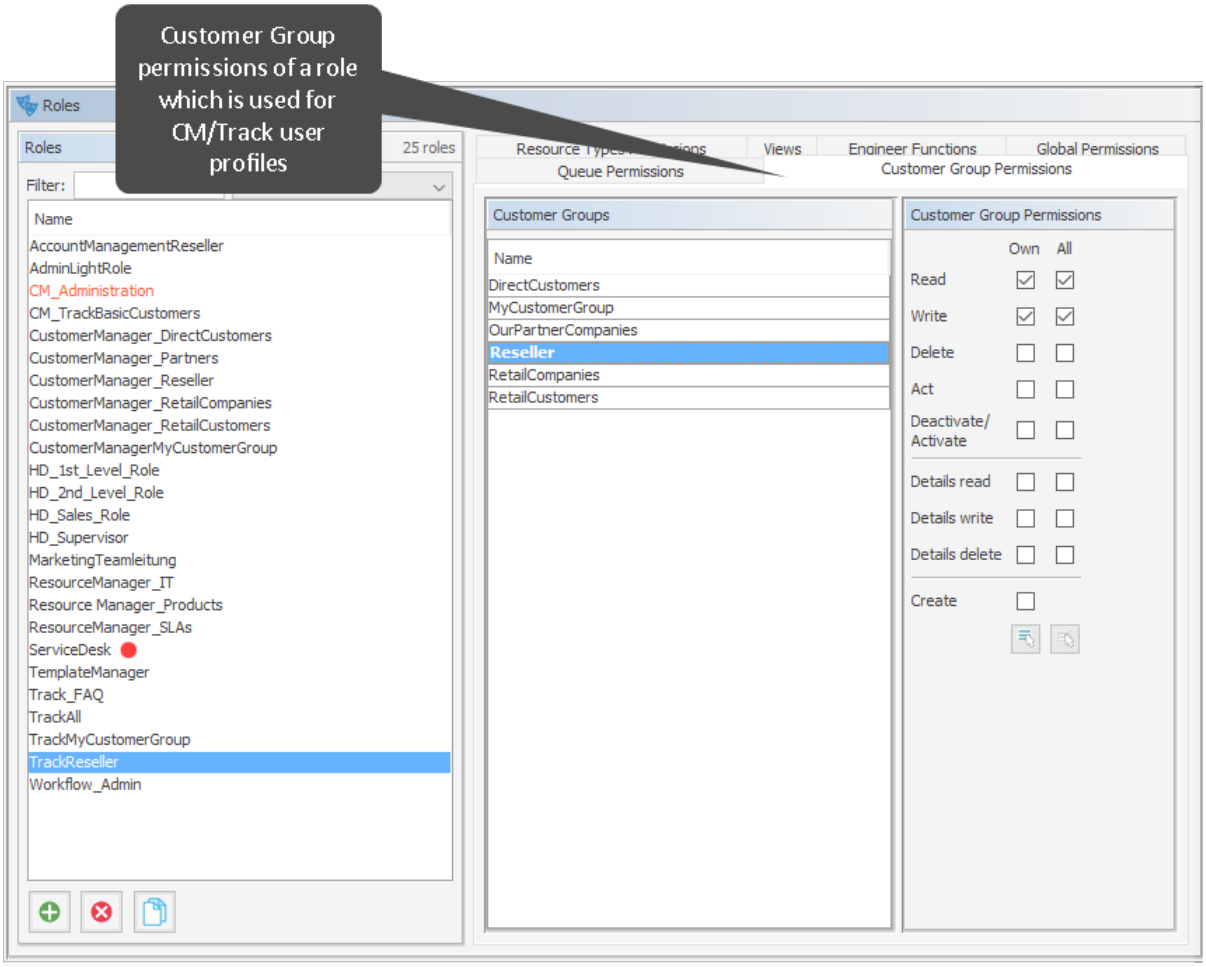


Figure 482: *ConSol CM Admin Tool - Access and Roles, Roles: User profile for CM/Track, customer group permissions*

With this approach, a customer with the CM/Track user profile *TrackReseller* can only see and add comments to tickets from those queues. Another user profile might have access to *Sales* tickets and/or to an *FAQ* queue.

G.2.4.5 Defining the User Assignment Mode

The user assignment mode is defined for each customer group and defines the system behavior concerning the way of assigning a CM/Track user profile to a contact. Open the customer group edit panel: in the navigation group *Customers*, navigation item *Customer Groups*, select a group and click the *Edit* button. In the last line of the panel, the CM/Track user assignment can be made.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

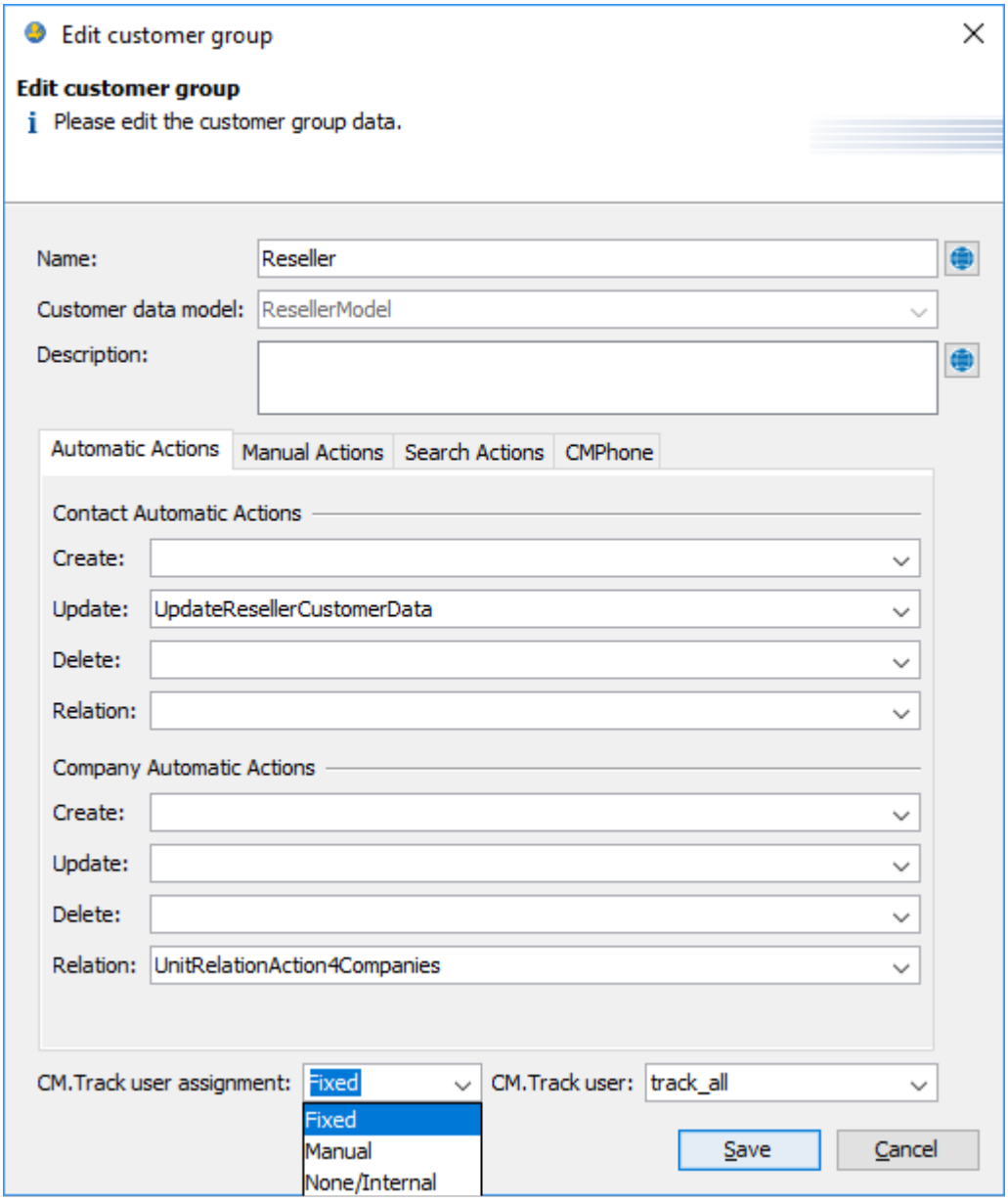


Figure 483: *ConSol CM Admin Tool - Customers, Customer Groups: Defining the CM/Track user profile mode for a customer group*

Three modes are available for the assignment mode of the CM/Track user profiles:

- **Fixed**
  A CM/Track user profile (i.e., a ConSol CM engineer object which has been defined as CM/Track user profile, see section Defining the User Profiles/Access Permissions for CM/Track) is selected in the pull-down menu *CM/Track user*. This CM/Track user profile is used for all new contacts created in this customer group. The data fields which are potentially available in the Web Client to assign a CM/Track user profile to a contact (see section Granting Access to CM/Track for Customers) are not displayed in the Web Client.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

> ⚠ Please be aware that the assignment mode “Fixed” cannot be changed anymore as soon as the customer group has contacts. If you select the *Fixed* mode for a customer group which already has some contacts, the CM/Track user profile of these contacts is **not** changed automatically. They need to be migrated to the new user profile using a task script.

- **Manual**
  Default. The assignment of a CM/Track user profile is done manually by an engineer in the Web Client as described in section Granting Access to CM/Track for Customers. The user profile can also be set using the REST API.
- **None/Internal**
  The association of a CM/Track user profile is not allowed in any client and can only be done by script, if desired. There will be no choice available in the Web Client, and an attempt to change it via REST API will also return a status message METHOD NOT ALLOWED.

G.2.4.6 Defining the Customer Fields for CM/Track Login and Password

The fields for login and password for a customer are regular customer fields at the contact level. Please see section Setting Up the Customer Data Model for an introduction to customer field management and GUI configuration for customer data.

Edit the fields which contain the customer data (if there are two levels: **not** the company level, but the contact level!) as demonstrated in the following example. You reach the following screen by opening the navigation group *Customers*, navigation item *Data Models*.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- One field for the login has to be created, annotation `username` = “true”.

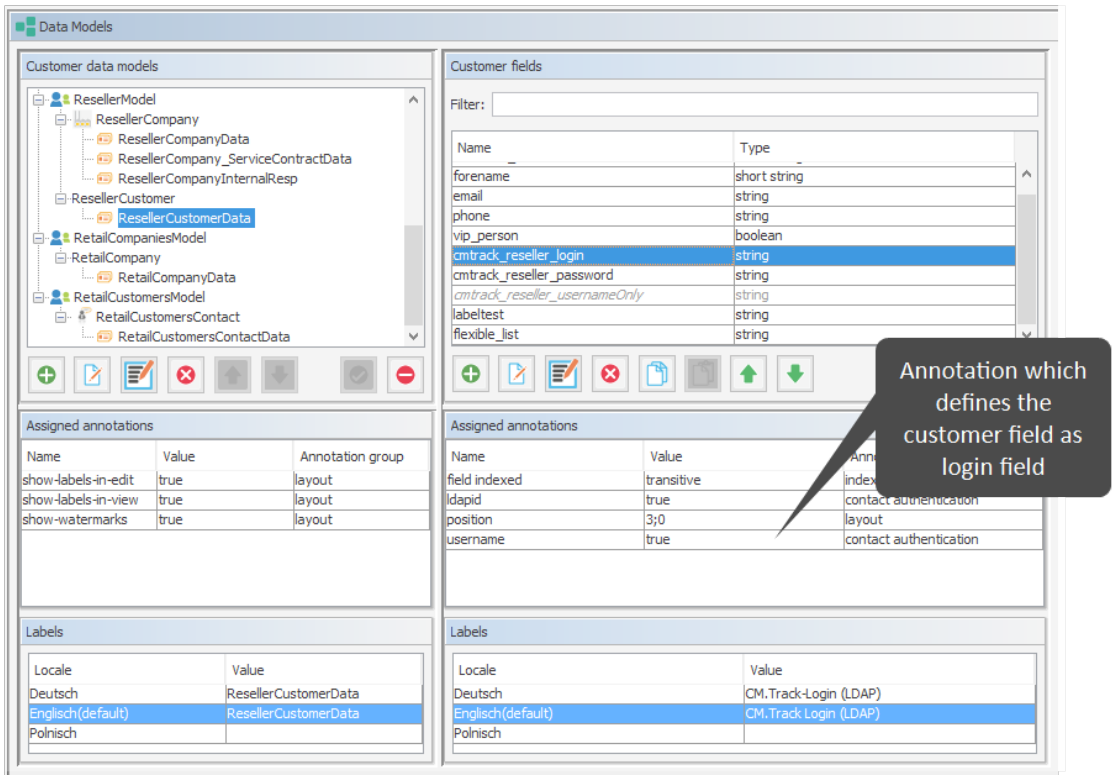


Figure 484: *ConSol CM Admin Tool - Customers, Data Models: CM/Track - Annotation 'username' for login*

> ⚠ Assigning the annotation `username` to a customer field is only possible, if there is no previous assignment of this annotation. Otherwise it will be prohibited. When assigning it, a warning dialog must be confirmed before it is executed, since it can be a longer running operation. Un-assigning the annotation must be confirmed as well, because it cannot be undone: Un-assignment delete the username values unrecoverably from internal storage.

- One field for the password has to be created, annotation `password` = “true”. The annotation `text-type` = “password” guarantees that only stars/dots are displayed in the Web Client, not the clear text password.

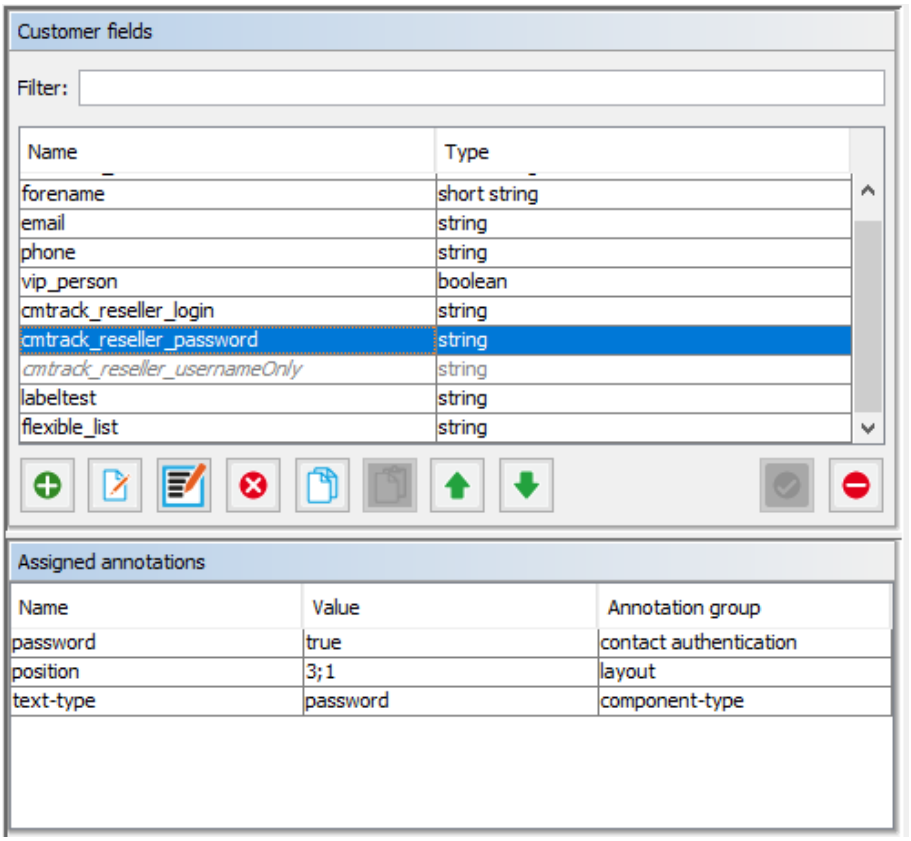


Figure 485: *ConSol CM Admin Tool - Customers, Data Models: CM/Track - Annotation for password*

> ⚠ The annotation `password` requires confirmation when assigned.
>
> In case of an update from CM versions lower than 6.11 to 6.11 and up: When this annotation is set, the system reads the plain text passwords from the original field values, encrypts them and saves the encrypted values to the internal storage. Then the original field values are deleted and thus the plain text value cannot be recovered anymore.
>
> When trying to un-assign the password annotation the operation must be confirmed as well, since the encrypted passwords are deleted from the internal storage. After the annotation unassignment the password information is completely lost and cannot be recovered at all.

When a scenario from a CM version lower than 6.11 is imported into a system with CM 6.11 (or higher), a transformation of user names and passwords is performed automatically. This is described in detail in section Transformation of User Name and Password Fields During Import into CM 6.11 .

G.2.4.7 Granting Access to CM/Track for Customers

For all customer groups where the CM/Track user profile assignment mode is set to “Manual”, the engineer working with the Web Client can then assign a user name, initial password, and a CM/Track user profile to every customer who should have access to the portal CM/Track. The user name has to be unique. This is checked by the system. You cannot enter a name a second time if this has already

been assigned to another customer. The password is stored as encrypted string in the CM database. This means that an engineer can set a new password, e.g., when a customer calls and asks for this, but it is never possible to read the password from the system.

> You, as an administrator, can define if the CM/Track user names should be case sensitive. Use the CM system property cmas-core-security, policy.track.username.case.sensitive. This is a boolean variable. When it is set to "true", the CM/Track user names are case sensitive. Please make sure that the database collation which is in use supports case sensitive strings!

The following example shows the customer data of an example contact in the ConSol CM Web Client. You reach this screen by opening a contact data set in edit mode.

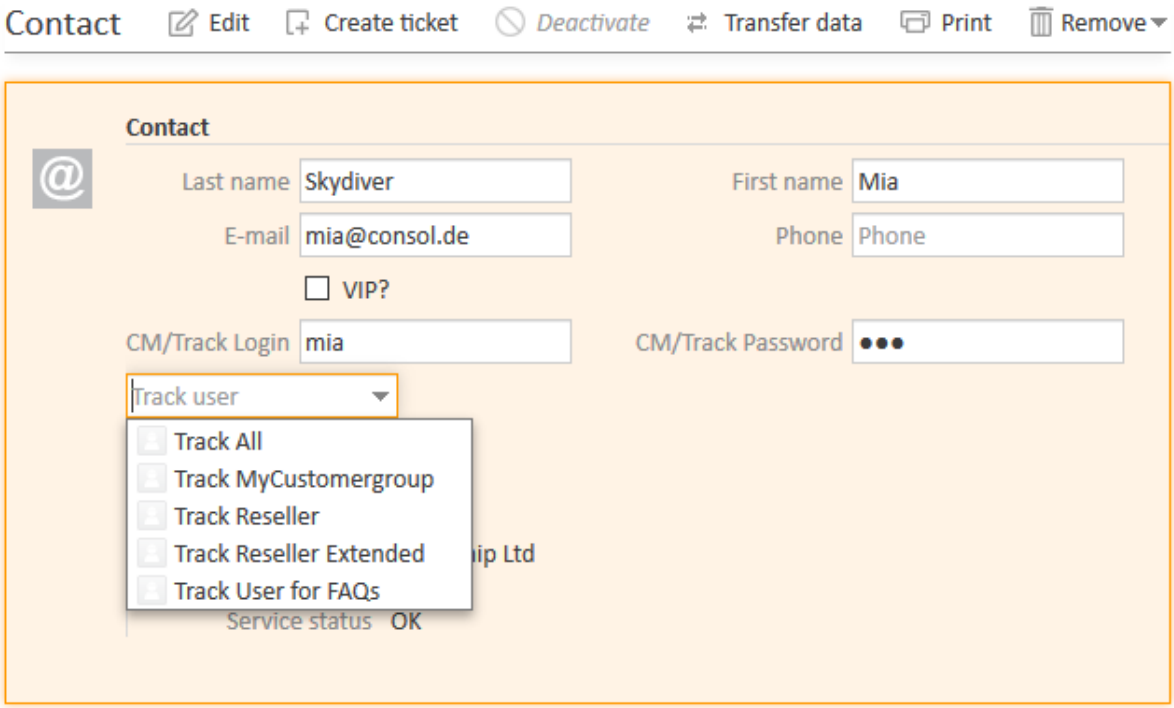


Figure 486: *ConSol CM Web Client - Contact page: CM/Track user data*

G.2.4.8 Customer Login to the System

Then customers can log in to the system and see their tickets. Please refer to the *ConSol CM User Manual*, section *CM/Track* for a detailed explanation on how to work with ConSol CM as a customer.

There are two mechanisms for performing user authentication:

- simple authentication
- LDAP authentication

Please refer to section Authentication Methods for Customers in CM/Track for details.

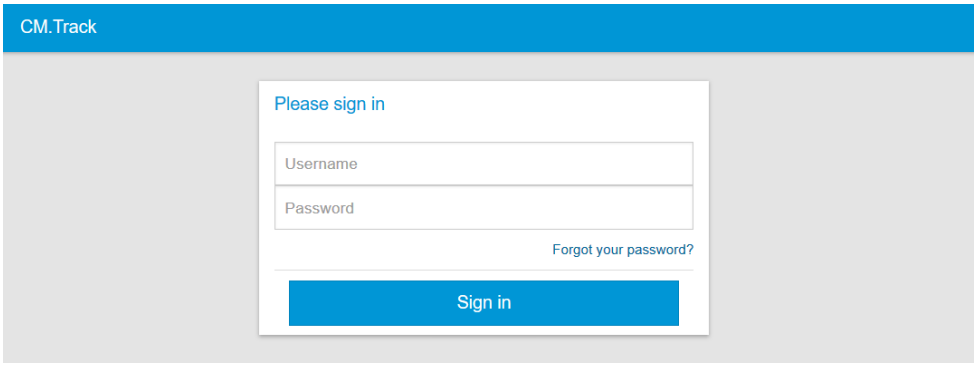


Figure 487: *ConSol CM/Track - Customer login*

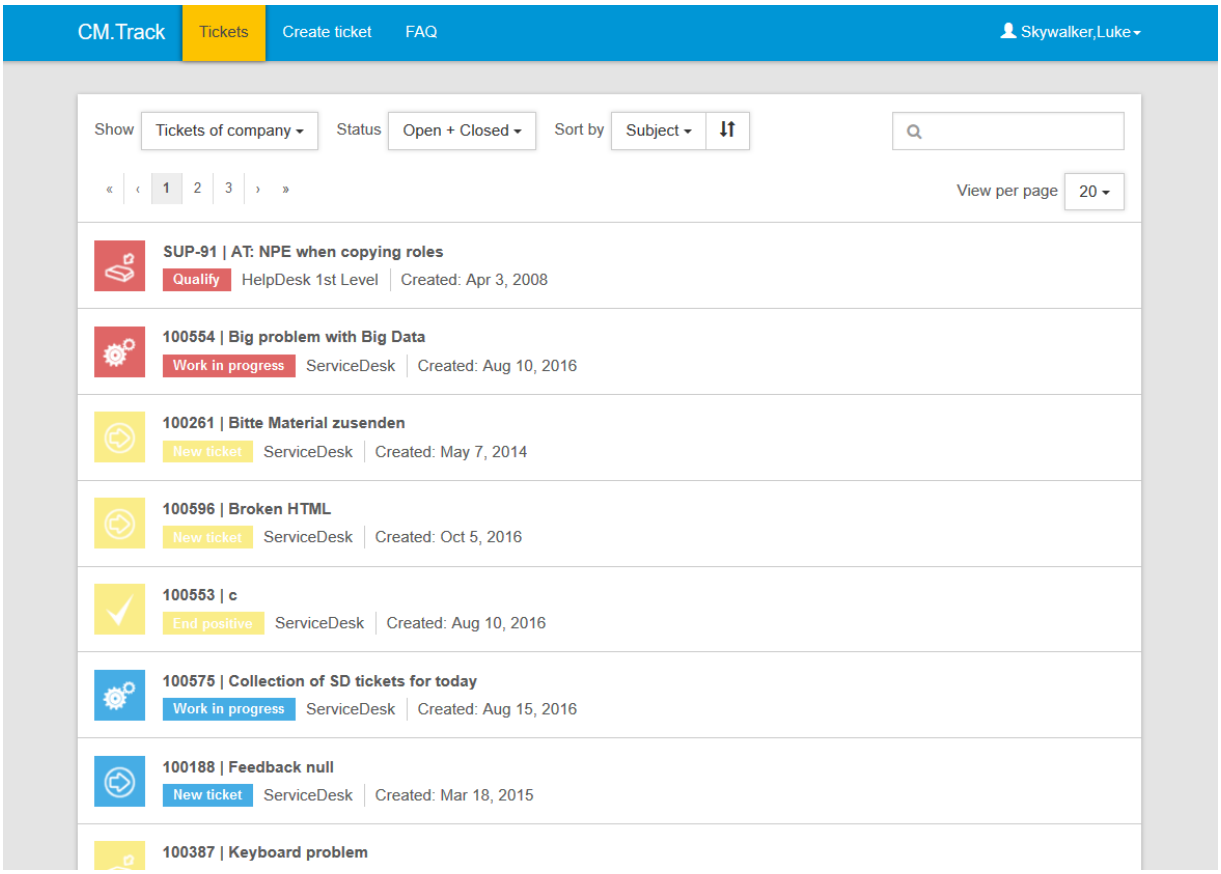


Figure 488: *ConSol CM/Track - Ticket list*

G.2.4.9 Extended Customer Permissions to See Company Tickets

In some cases it might be required that customers log in to the ConSol CM portal CM/Track and have to have access not only to their personal tickets but to all tickets of their company. In this case, the role for the CM/Track user (user profile) should be assigned the permission *Access tickets of the own company* under *Track User Permissions*. Please refer to the section Roles for a detailed explanation.

G.2.4.10 Configure CM/Track for Password Reset by Customers

CM/Track can be configured to display a hyperlink for customers where a customer can reset his password. This is based on the template `track-password-reset-template`. Please refer to section Password Reset Template for Customers Using CM/Track for a detailed explanation. The password reset in

CM/Track is only possible when the DATABASE mode is used. It is not possible when LDAP authentication is in operation. See section Authentication Methods for Customers in CM/Track for the portal for an explanation of all possible authentication modes.

Please note that the `From` address of the email which is sent to a customer who has requested a new password can be set using the CM system property cmas-core-security, password.reset.mail.from.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.2.5 CM/Track: Data Availability For Customers

G.2.5.1 Introduction

When configuring CM/Track for your customers, you need to decide which part of the information stored in the customers' tickets you want to share with the customers and which information should be kept internal. There are two aspects which need to be handled for this purpose:

- Ticket history entries: comments, emails and attachments
- Ticket fields

G.2.5.2 Visibility of History Entries

The visibility of ticket history entries is configured using text classes, see Classes of Text. All entries which are marked with a text class which is *customer readable* are visible for the customer when he views a ticket. This applies to:

- comments
- emails
- attachments

When the engineer wants to share a comment, email or attachment with the customer, he needs to set a text class which is customer readable to the respective history entry.

> ⚠ The default text class is not customer readable. This means that comments written by the customer in CM/Track and emails which are exchanged with the customer are **not** visible in CM/Track unless an engineer selects another text class for the respective history entry. Usually, there are some situations where you want to make history entries available directly, e.g. if the customer adds a comment, this comment should be displayed in the ticket history immediately. You need to adapt the workflows so that a suitable text class is assigned automatically to this kind of history entries.

G.2.5.3 Visibility of Ticket Fields

The visibility of ticket fields in CM/Track can be configured for ticket field groups and for single ticket fields by using annotations. In addition, the visibility is affected by system properties and the active client configuration.

The following sections describe the different places where the configuration is made and their combination.

System Properties

The value of cmas-rest-api, security.fields.customer.exposure.check.enabled determines whether the ticket field annotations are checked or not.

- **true**: The customer can only see the ticket fields which are explicitly annotated as visible in CM/Track.
- **false**: The customer can see all ticket fields in CM/Track.

Annotations

The annotations which are assigned to ticket fields are only considered if the system property `security.fields.customer.exposure.check.enabled` is set to "true". They can be assigned to ticket field groups or ticket fields:

- **Ticket field groups:**
  Annotation `customer exposure group` with the following values:
  - **full**: The fields in this ticket field group are displayed in CM/Track and their values can be modified.
  - **read**: The fields in this ticket field group are displayed in CM/Track.
- **Ticket fields:**
  Annotation `customer exposure` with the following values:
  - **full**: The ticket field is displayed in CM/Track and its value can be modified.
  - **read**: The ticket field is displayed in CM/Track.
  - **none**: The ticket field is not displayed in CM/Track.

The value of the annotation on field basis overwrites the value of the group annotation. In this way, you can fine-tune the visibility of single fields within a group.

The annotations `group-visiblity` (for ticket field groups) and `visibility` (for single ticket fields) will work in any case and are stronger than the `customer exposure group` and `customer exposure` annotations, i.e. when a group or field is annotated with `(group-)visibility` = “none”, it will not be displayed, no matter what `customer exposure (group)` annotation might been set.

Client Configuration

The JSON client configuration also influences which fields are displayed in CM/Track. The setting `expandable` in the layout options, determines whether all fields with the corresponding annotations are displayed or not.

- **true**: Ticket fields which are not explicitly mentioned in the JSON layout attribute but which are annotated as customer exposed will be displayed.
- **false**: Only ticket fields which are explicitly mentioned in the JSON layout attribute are displayed.

> Fields which are mentioned in the JSON layout attribute but not annotated as customer exposed are not available in CM/Track.

See Design and Configuration of REST-based ConSol CM Client GUIs.

Combinations

The following table shows the effect of the different combinations on a certain field.

- **System property**: value of `security.fields.customer.exposure.check.enabled`
- **Client configuration**: value of `expandable`
- **Annotation**: assignment of `customer exposure` to the field, if not assigned, assignment of `customer exposure group` to the field group

System property	Client configuration	Annotation	Effect
false	false	*	The ticket field is only displayed if it is mentioned in the layout attribute of the client configuration.

System property	Client configuration	Annotation	Effect
false	true	*	The ticket field is displayed.
true	*	no	The ticket field is not displayed.
true	false	yes	The ticket field is only displayed if it is mentioned in the layout attribute of the client configuration.
true	true	yes	The ticket field is displayed.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.2.6 CM/Track: FAQs in CM/Track

G.2.6.1 Introduction to FAQs in CM/Track

If you use CM/Track as a portal where your customers can access their tickets or the tickets of their company, you might consider offering an FAQ (Frequently Asked Questions) search to this clientele. This has proven very helpful in help desk or service desk environments where customers can check if the problem they face has occurred before and if there is a known solution. They only need to contact the service desk and/or open a new ticket if they do not find any help in the FAQ, saving time for both customer and the service team. It might also be employed in other environments where you would like to offer this service.

In ConSol CM, every FAQ is treated as a ticket. Any queues which should be available as FAQ queues via CM/Track have to be defined as special FAQ queues because customers are usually allowed to see only their own tickets or tickets from their company, but FAQ tickets do not belong to any specific customer. They can be accessed by every customer who logs in with a user profile that has access to the FAQ queues. Here, only read access has to be granted.

G.2.6.2 Configuring the ConSol CM System to Allow FAQ Search in CM/Track

As a first step you have to create an FAQ workflow (please see the *ConSol CM Process Designer Manual* for details) and create an FAQ queue that is marked as a queue for frequently asked questions (checkbox *FAQ*). In addition, you have to assign at least one class of text which has the option *Customer readable* selected (see Classes of Text for details) to the queue. The comments which should be visible to the customers in CM/Track must be marked with this class of text in the Web Client.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

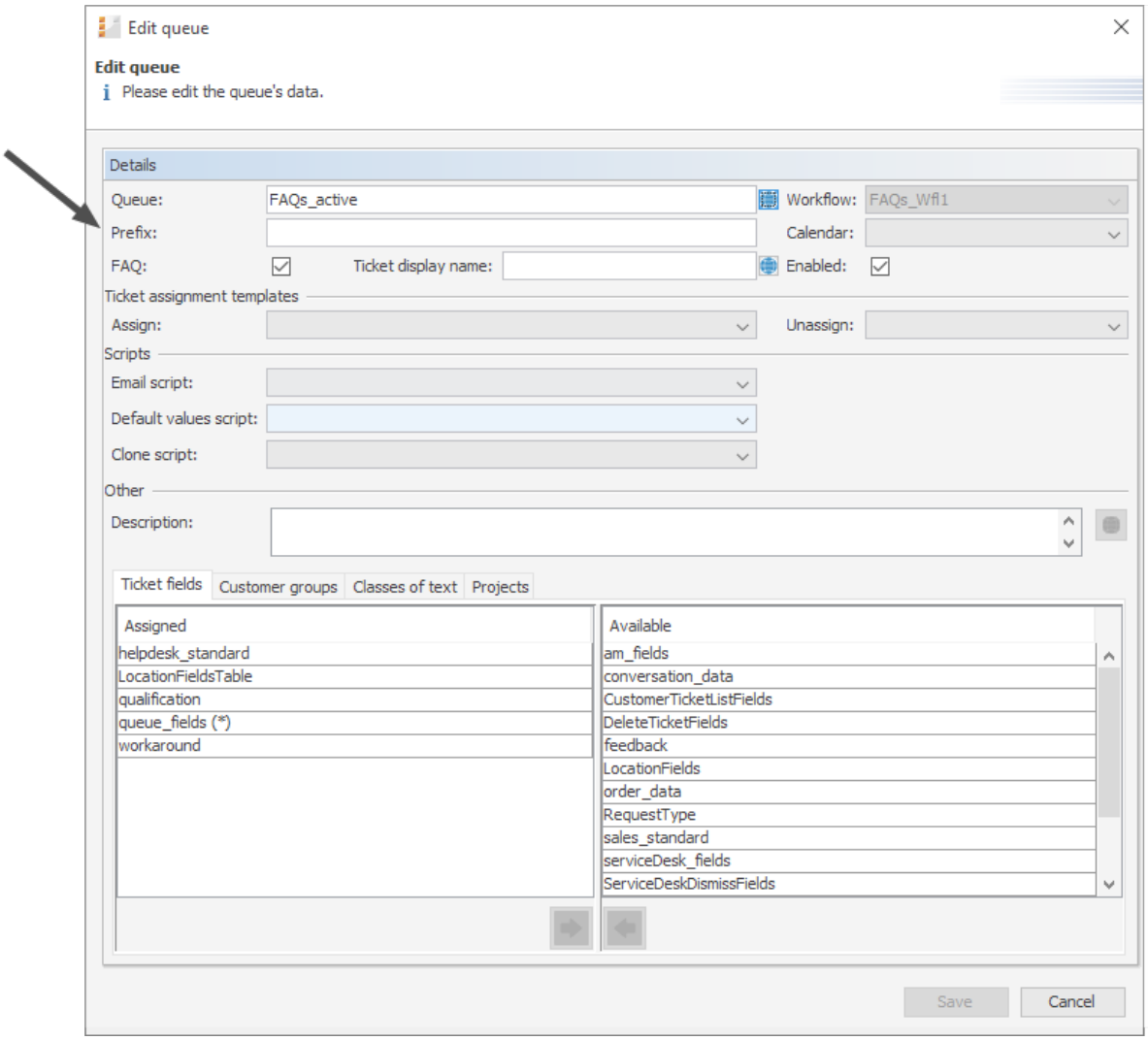


Figure 489: *ConSol CM Admin Tool - Global Configuration, Queues*

Then a role has to be defined which can access the FAQ queue in read-only mode. Please keep in mind that this role also needs read access to the customer group under which you have located the FAQ queue tickets.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

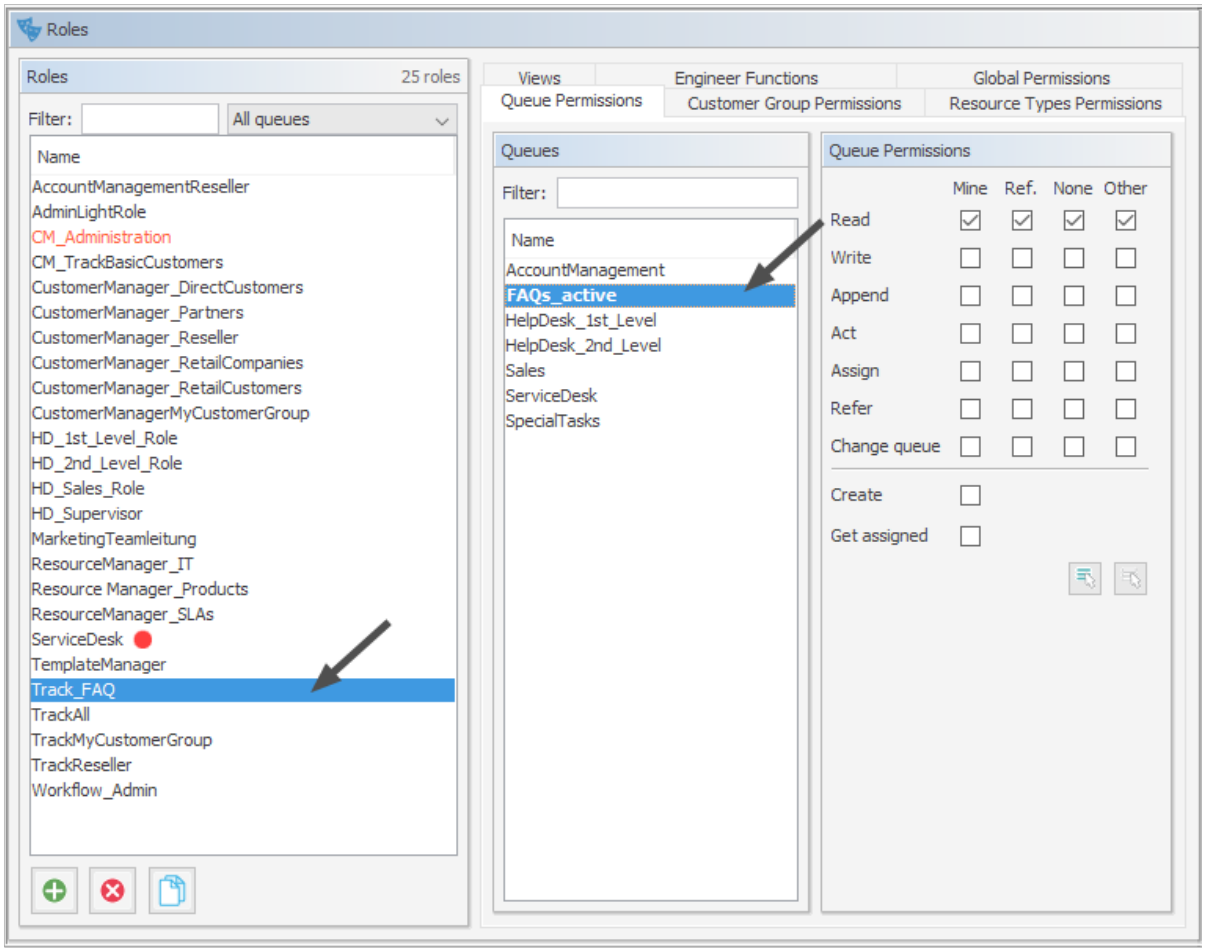


Figure 490: *ConSol CM Admin Tool - Access and Roles, Roles: Role for FAQ*

Then this new role has to be assigned to the user (profile) which is used as CM/Track access user (see section CM/Track: System Access for CM/Track Users (Customers)).

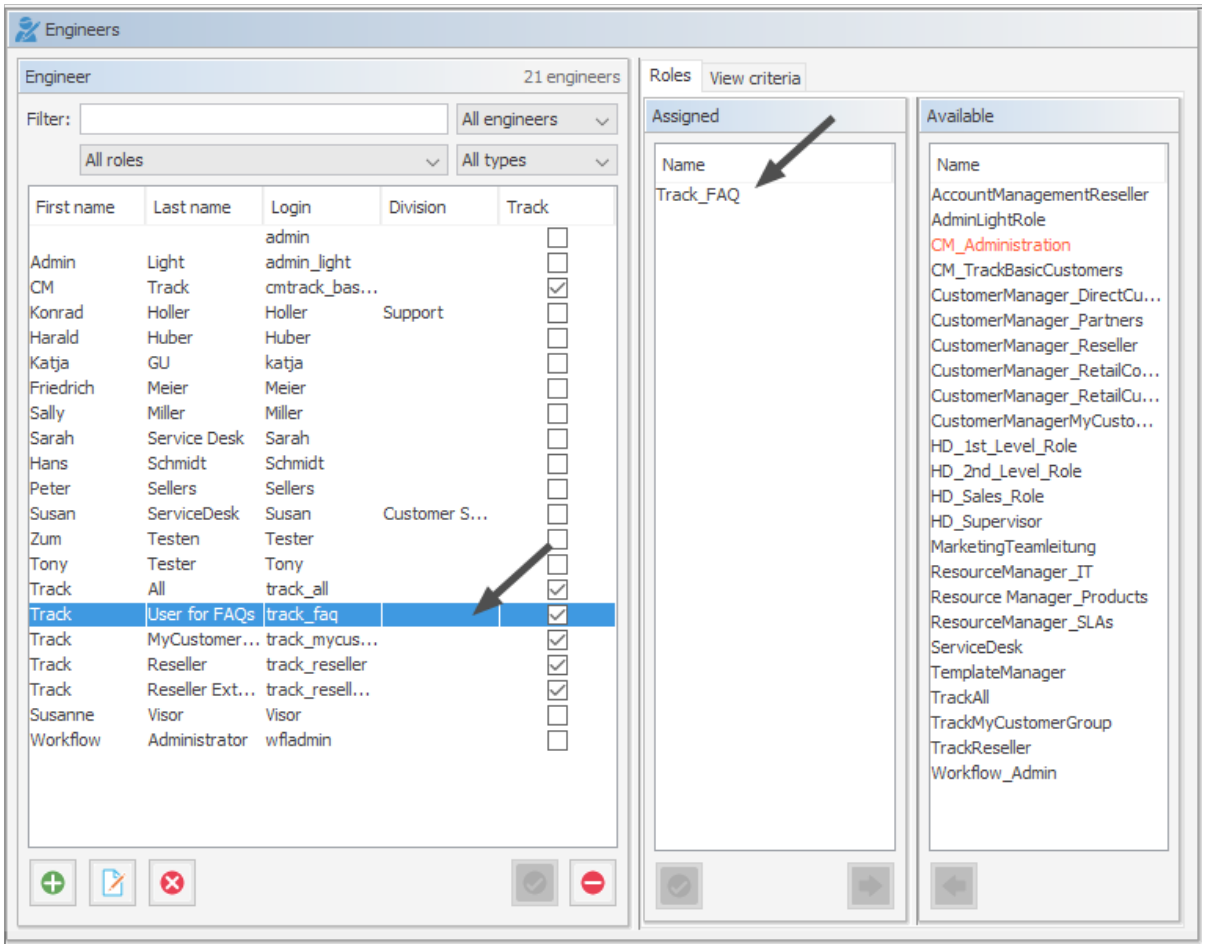


Figure 491: *ConSol CM Admin Tool - Access and Roles, Engineers: Engineer for FAQ*

G.2.6.3 FAQ Search in CM/Track from a Customer's Point of View

All queues which are marked as *FAQ* are automatically offered as FAQ queues in the portal (see following figure). No further access configuration is required.

Initially, the FAQ queue selector is set to *All FAQ Lists*, thus all tickets from all FAQ queues are displayed. A customer can narrow down the FAQ ticket list by using the string filter option or by selecting a particular queue using the queue selector. By opening one ticket from the list, the fields of the tickets are displayed.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

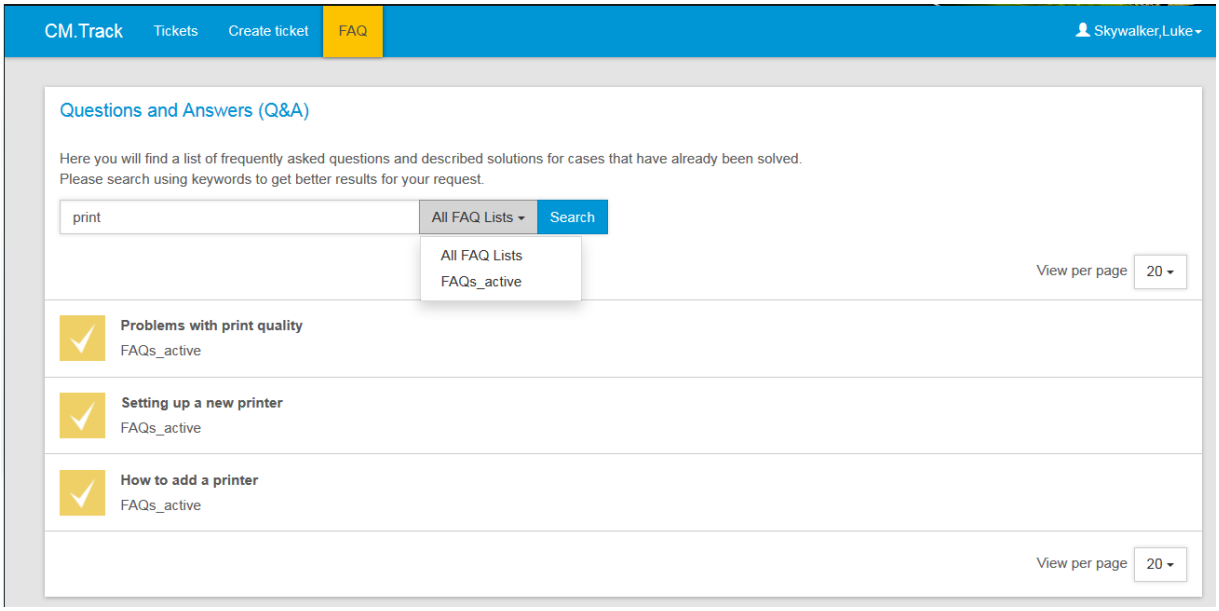


Figure 492: *ConSol CM/Track - Example for FAQ search (1)*

G.2.6.4 More Complex Solutions for Managing FAQs

Using Two FAQ Queues: FAQ Management and Active FAQs

Instead of using only one FAQ queue, two queues might be used. One can be an FAQ management queue where tickets can be placed manually or be transferred from help or service desk queues. An FAQ manager checks the FAQ and edits the ticket if required. Then the ticket is placed in the queue for active FAQs. Only this active queue is marked as FAQ in the queue management, thus only tickets in this queue can be accessed by customers. After a certain period of time, or when the FAQ manager decides the FAQ should no longer be available, it is transferred back to the FAQ management queue. It may later be re-activated or closed.

Setting Up Two (or More) Parallel FAQ Environments Using Track Users

By creating more than one FAQ queue (or a pair of FAQ queues) and creating the respective CM/Track user profiles, it is possible to provide FAQs for different customer groups. For example, for one customer group technical help desk questions and answers are provided, whereas for the other customer group support and update information is provided. Of course, there can also be a CM/Track user profile which has access to both FAQ environments.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.3 CM/EBIA

G.3.1 Introduction to CM/EBIA

CM/EBIA is a ConSol CM add-on which allows to create reports directly in the Web Client. The created reports are shared with the regular users as dashboard widgets.

The following figure illustrates how CM/EBIA integrates with ConSol CM:

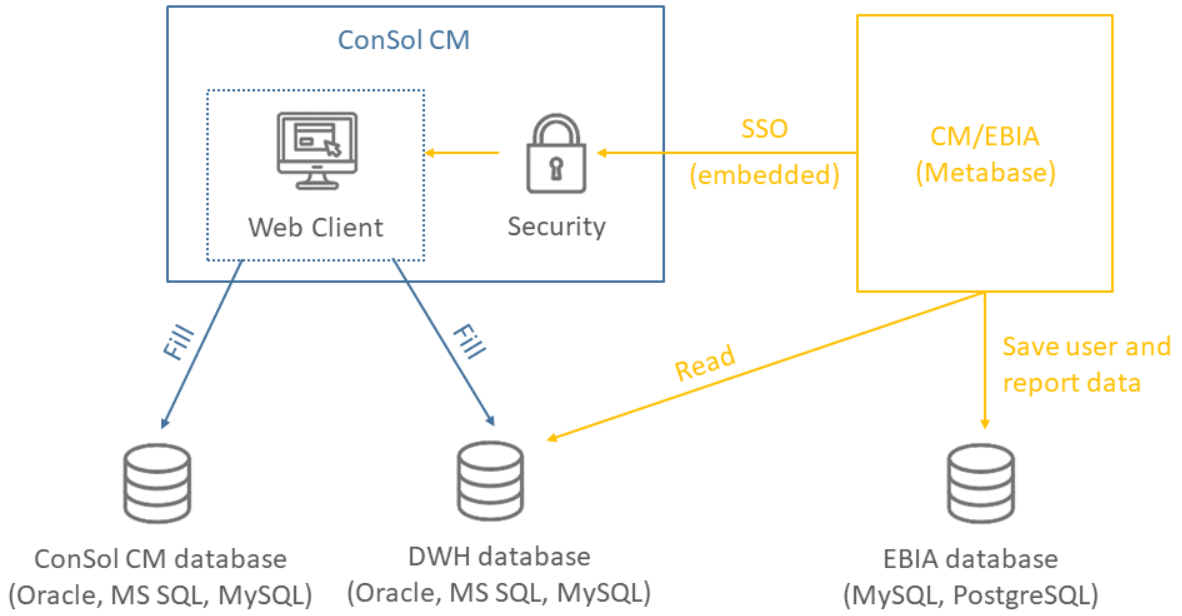


Figure 493: *Architecture of ConSol CM with CM/EBIA*

G.3.1.1 Concepts, terms and definitions

Concept	Other terms	Definition
CM/EBIA		Name of the ConSol CM add-on which allows to create reports directly in the Web Client
Metabase		Third-party application used in CM/EBIA
question		A chart or table which visualizes certain data
dashboard		A logical collection of questions on one page, enriched with global filtering, headlines and text
collection		A folder-like structure holding charts, dashboards and pulses. Collections can be guarded by role permissions
pulse		A message which contains one or several questions. It is sent in regular intervals to the users to inform them about current metrics.

Concept	Other terms	Definition
widget		An item displayed on the ConSol CM Web Client dashboard. Widgets can contain charts and tables backed by ConSol CM's runtime data by default. With CM/EBIA, a widget can also contain CM/EBIA questions and dashboards backed by DWH data, which allows much more powerful analytics.

G.3.1.2 Purpose and usage

The ConSol CM add-on CM/EBIA allows to create reports directly in the Web Client. CM/EBIA uses a licensed enterprise version of the third-party application Metabase, which is integrated via SSO. It can be accessed using the menu item *Analytics*, which is displayed for all the users with general administrator permissions or CM/EBIA administrator permissions. The link *Open full application* within the embedded CM/EBIA application allows to open CM/EBIA in its own tab to access the complete functionality, including administration, of CM/EBIA.

The created reports are shared with the regular users as dashboard widgets.



CM/EBIA needs to be licensed separately.

G.3.2 Available reports in CM/EBIA

CM/EBIA allows to define reports which analyze data from a database. There are four types of reports:

- **Question:**
  A question is a single chart or table which visualizes certain data.
- **Dashboard:**
  A dashboard consist of several questions. You can add headlines, text and filters to make the dashboard more appealing.
- **Pulse:**
  A pulse is a message which contains one or several questions. It is sent by email or Slack in regular intervals to the users to inform them about current metrics.
- **Collection:**
  A collection can include questions, pulses and dashboards. Access to collections can be limited by permissions, which can be set in the CM/EBIA administration.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.3.3 Basic tasks

G.3.3.1 Creating reports in CM/EBIA

Please proceed as follows to create a report:

1. Click *Ask a question*.
2. Choose the mode:
   - **Simple question**: Allows to create a question based on a single table. You can filter and group the data using the graphical user interface. Ideal for beginners.
   - **Custom question**: Allows to create a question based on several tables. You can join the tables and then filter and group the data using the graphical user interface.
   - **Native query**: Allows to create a question based on a SQL query. For experts only.
3. Follow the instructions on the graphical user interface to create the question. You can use the following features:
   - **Filter**: Filter by the content of certain columns to use only a subset of the table data in your report.
   - **Summarize**: Use counts, sums, averages, etc. on certain table columns in your report.
   - **Visualization**: Choose a type of chart for the report. CM/EBIA automatically suggests appropriate types.
   - **Settings**: Fine-tune the chart by modifying the data, display and drill-through options.
4. Upon saving the report, you can directly add it to a dashboard.
5. Edit the dashboard which should be shared with the users by adding several questions, a headline and text boxes to illustrate its contents.

Once the dashboard is finished, you need to create a widget to share it with the end users.

G.3.3.2 Sharing reports as dashboard widgets

You can publish reports on the Web Client dashboard using generic widgets (see HTML widgets).

Use the methods of the class `metabaseService` to reference reports in the widget visualization script:

- **getDashboard(String pId, String pAttributes)**
  Returns an `iframe` tag with the dashboard by ID
- **getQuestion(String pId, String pAttributes)**
  Returns an `iframe` tag with the question by ID
- **getCollection(String pId, String pAttributes)**
  Returns an `iframe` tag with the collection by ID
- **getMetabaseIframe(String pReturnToURI, String pAttributes)**
  Returns an `iframe` tag with the CM/EBIA application according to certain parameters (can be used to access the full CM/EBIA application)
- **getMetabaseURL(String pReturnToURI)**
  Returns the URL with the authentication token, which is used as the value of the `src` parameter in the `iframe` tag

The following parameters are used in the above methods:

- **pId**
  Dashboard, question, or collection identifier.

  > You can see the identifier of questions and dashboards in the URL when opening the CM/EBIA application in its own tab (link *Open full application* on the *Analytics* page).

- **pAttributes**
  `iframe` tag attributes, see https://developer.mozilla.org/en-US/-docs/Web/HTML/Element/iframe#Attributes
- **pReturnToURI**
  The part of the URL that is used for `iframesrc` (e.g., "/dashboard/ 5", "/question/7")

The following example shows how to use the method `getDashboard()` to render a CM/EBIA dashboard in the widget.

```
def render(String pParams) {
return """
   ${metabaseService.getDashboard("3", 'height="700px"')}
""" as String
}
```

Please see Scripts of Type Widget Visualizationfor details about widget visualization scripts in general.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.3.4 Advanced tasks

G.3.4.1 Installing and setting up CM/EBIA

⚠ CM/EBIA needs to be licensed separately.

The first step to use CM/EBIA is to install and set up the Metabase application:

1. Download the Metabase JAR file from https://-downloads.metabase.com/enterprise/latest/metabase.jar.
2. Start the Metabase application with the following start command:

   ```
   java -jar metabase.jar
   ```

   You can set the following variables:

   - **-DMB_SESSION_COOKIE_SAMESITE**: “None”, “Lax” or “Strict”. If Metabase is installed in a different domain than ConSol CM, you must use “None”.
   - **-DMB_JETTY_PORT**: The port of the Metabase application. The default port is 3000.
3. Open the Metabase application. It runs on port 3000 by default. Create the administrator account. You can add the database connection in this step or later.
4. Go to *Enterprise*, click *Activate a license* and enter the license key provided by ConSol.
5. Go to *Settings -> General*. The value of the ConSol CM system property cmas-analytics, meta-base.url needs to match the value in *Site URL*.
6. Go to *Authentication -> JWT*. Set *JWT Authentication* to *Enabled*. Enter the URL of the Web Client in *JWT Identity Provider URI*, e.g. “https://localhost:8080/cm-client”. Paste the value in *String used by the JWT signing key* to the ConSol CM system property cmas-analytics, meta-base.secret.key.
7. Go to *Embedding in other Applications*. Enter the URL of the ConSol CM system in the field *Embedding the entire Metabase app*.

The next step is to add the databases. These are the databases whose data is used for creating analysis in CM/EBIA. Usually, this is the DWH database with some preconfigured cubes.

⚠ It is also possible to connect the ConSol CM production database to CM/EBIA. This is not recommended, so you must check with the ConSol CM support first.

Please proceed as follows to add a database:

1. Open the CM/EBIA administration.
2. Go to *Databases* and click *Add database*.
3. Enter the information of your database and click *Save*.

G.3.4.2 Managing permissions

There are two regular ConSol CM permissions which grant access to the CM/EBIA application:

- Administrate system (full access)
- EBIA full access

One of these two permissions is required to see the *Analytics* link and create questions and dashboards in CM/EBIA.

Regular users, e.g. the users who see the dashboard which contains the reports, are created automatically in CM/EBIA in the user group *All users*. By default, they get access to the complete database, i.e. they can execute SQL queries and drill into the data. If this is not desired, you need to change the permission settings within CM/EBIA:

1. Go to the administration page and open the *Permissions* page.
2. In *Collection permissions*, set the permission *View collection* in the *All users* column.
3. In the *Data permissions*, revoke data access and SQL queries in the *All users* column.



The users can still see the dashboard widgets but they cannot view the underlying data or modify the dashboard contents. This configuration is recommended in most cases. You can grant *EBIA full access* permissions to those user who should be allowed to create questions.

G.3.4.3 Configuring the display of CM/EBIA administration

The CM/EBIA page can be configured using the page customization. The respective section is called *analyticsPage*. It has three attributes:

- **adminLinkVisible:**
  Determines whether the *Admin* link is displayed. Default: true.
- **iframeHeight:**
  Determines the height of the iFrame with the CM/EBIA application. Default: 1000
- **navigationLinksVisible:**
  Determines whether the links *Home*, *Create question* and *Open full application* are displayed. Default: true

G.3.4.4 Fine-tuning CM/EBIA

The following settings, which can be made in the CM/EBIA administration, have been proven useful.

Section *General*:

- Do not change the settings *Report Timezone*. Use the database default instead.
- In *Friendly Table and Field Names*, use the setting *Only replace underscores and dashes with spaces* to stop CM/EBIA from trying to interpret and beautify column names.

Section *Localization*:

- For German environments: change the date, time, decimal number format and currency.

Section *Data Model*:

- Edit the display names of your cubes and most used tables and columns to provide more user-friendly labels. This affects both the labels used for the tables and table columns when creating questions and the labels displayed to the end users.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.4 CM/Forms

G.4.1 Introduction to CM/Forms

CM/Forms is a ConSol CM add-on which enables the creation of web forms. Web forms can be filled out by unregistered users, i.e. user who have not logged in or user who have not yet been created in the Web Client or CM/Track.

G.4.1.1 Concepts, terms and definitions

Concept	Other terms	Definition
webhook		Mechanism to allow communication between ConSol CM and external applications
submission		Data which was entered in the web form by a user and sent to ConSol CM by submitting the web form

G.4.1.2 Purpose and usage

Web forms are managed in the menu item *Forms* of the Web Client. Administrator permissions are required to work with web forms.

A web form consists of two components: the form itself, which includes the general configuration and the form fields, and a script to process the submitted data. This can be either an integration scripts, which is used together with a webhook, or a task script, which is used when the submissions are recorded in the ConSol CM database.

Web forms can be useful in a variety of use cases:

- **Registration forms**
  Create a web form which allows your users to register themselves for CM/Track.
- **Feedback forms**
  Collect feedback from your customers and directly associate the feedback with a case.
- **Application forms**
  Create a form where candidates can enter their data and upload a CV.
- **Wizards for case creation**
  Provide an assistant which guides the user through the process of creating a case.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.4.2 Available settings for web forms

G.4.2.1 Basic form data

The basic form data are shown in the upper section of the web form:

- **Name**: Mandatory. The internal name for the web form.
- **Public name**: Optional. The public name of the web form. You can click the *Generate* button to fill it with a unique string consisting of random characters. If the form is published on the ConSol CM instance, the public name is part of the URL used to access the web form.
- **Publish type**: Mandatory. Determines how the web form is made available. See Publishing a web form. Possible values are:
  - *DENIED*: Client requests to the form are not possible.
  - *CM*: The form is published on the ConSol CM instance. The URL is: `<CM server>/cm-client/cf/<public name>`. It can be used in an iFrame.
  - *OWN*: The form is published in its own instance. It is necessary to have an external client application which serves the form. The use of an iFrame is not possible.
  - *FULL*: The form is published on the ConSol CM instance, and can be also published on its own instance at the same time. This is the combination of the publish types CM and OWN.
- **Condition script**: Optional. The name of the condition script (script of the type *Web form condition*). The script checks if the form can be used by the client. It is executed for GET and POST requests. The script has to return the response “200” for the form to be displayed. The following variables are available in the script:
  - `remoteIp`: IP address of the client requesting the form
  - `formName`: name of the form
  - `formPublicName`: public name of the form
- **Submission recording**: Optional. Select this option if the data which is submitted using the web form should be stored in the ConSol CM database table `cmas_custom_form_submission`. This allows to track the user submissions and to process submissions in a batch, e.g., using a task. See Data processing.
- **Webhook**: Optional. The webhook which is notified when a user submits a form. See Data processing. There are two options:
  - ConSol CM webhook: Enter the name of the webhook.
  - External webhook: Enter the complete URL.
- **Version**: Mandatory. The version of the web form. Click the *Increment* button to increment the version. The version can be used if significant changes are made after first publishing the form. The version is saved in the submission, so that processing scripts can handle changes to the form.
- **Active**: Optional. Select this checkbox if the web form should be active, i.e., it can be used according to the selected publish type.

G.4.2.2 Form design

The lower section contains the designed form. The ***form.io*** library is used for designing the form. The available form components are displayed on the left and the form is displayed on the right.

The *Form type* setting determines the basic structure of the form:

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Form**
  Use this option if the web form should have one page.
- **Wizard**
  Use this option if the web form should have several pages.

You can drag-and-drop the desired components to the form on the right side. The configuration dialog for the component is opened automatically, see Fine-tuning a web form.

> You can find more information in the official documentation of the ***form.io*** library (see Form Components, Layout Components and Data Components).

G.4.2.3 Data processing

There are two options for processing the data submitted in the web form:

- **Webhook**
  Precondition: A webhook is configured for the web form.
  1. The submitted data is sent to ConSol CM using the webhook.
  2. The integration script of the webhook processes the data.
  3. The data is saved in the ConSol CM database, e.g., in a case or contact.

  > You need to enter the name of the webhook in the *Webhook* field of the basic data, see Basic form data.

  > If the web form is published on the same instance as ConSol CM (publish type "CM"), the IP of the ConSol CM instance has to be allowed by the webhook. The webhook needs to be secured by IP address. The use of secret tokens is not possible for CM/Forms. See Webhooks for further information on how to create a webhook.

- **Database**
  1. The submitted data is saved directly to the ConSol CM database in the table `cmas_custom_form_submission`.
  2. The data is processed at a later time, for example, using a task script which is executed periodically.

  > You need to enable the checkbox *Submission recording* in the basic data, see Basic form data, and create the task script for processing the data, see Creating a Task.

The following figure illustrates data processing using a webhook (blue data flow) and submission recording (yellow data flow).

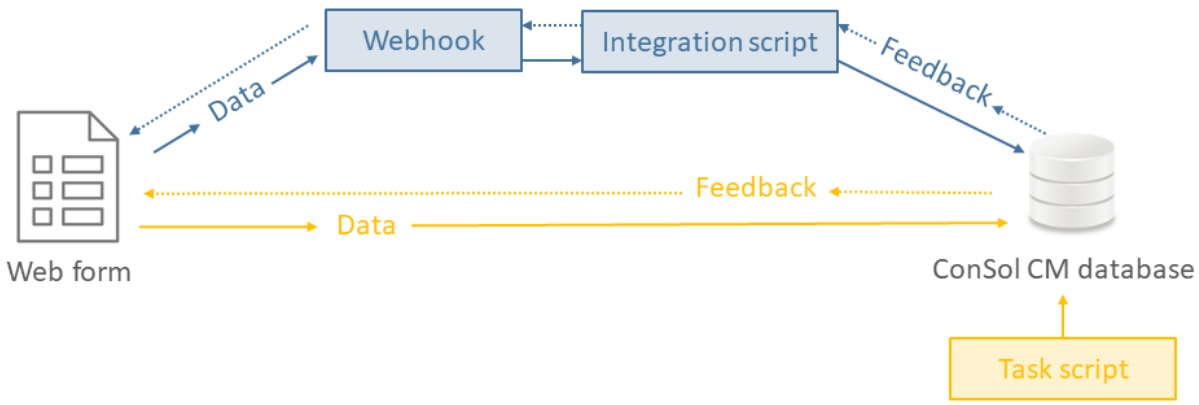


Figure 494: *Data processing for web forms*

It is possible to use both submission recording and the webhook for one web form. The acceptance of the submission depends on the selected option:

- **Submission recording + webhook**
  The submission is accepted if the webhook returns “200”. Otherwise, the submitting user sees an error message. Nevertheless, the submission is saved to the database even if the webhook returns an error.
- **Only webhook**
  The submission is accepted if the webhook returns “200”. Otherwise, the submitting user receives an error message.
- **Only submission recording**
  The submission is accepted if the data is saved to the database.

⚠ If only submission recording is used, the web form is not notified about the result of the processing performed by the task script, i.e., the user who submitted the web form will not know if his request could be processed successfully.

G.4.2.4 Publishing a web form

Depending on the selected publish type, see field *Publish type* in the Basic form data, and the desired usage of the web form, there are different options for making the web form available.

Common use cases are:

- Publish type “CM”, internal usage within the company
- Publish type “CM”, web form served from an URL provided by ConSol CM, external access possible
- Publish type “CM”, web form embedded in a website using an iFrame, external access possible
- Publish type “OWN”, web form served from a custom-made application, external access possible

The following figure shows a typical setup for the third use case:

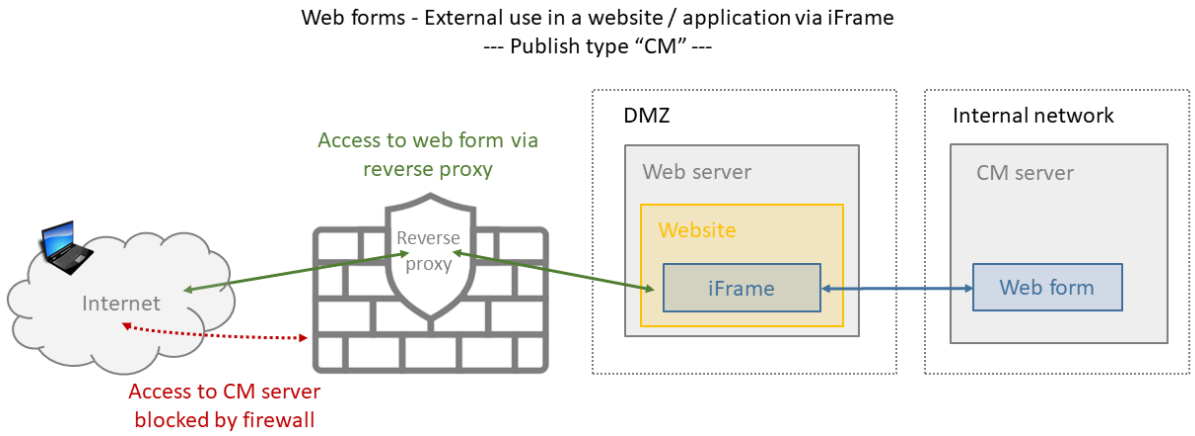


Figure 495: *Publishing a web form for external access embedded in a website*

> **Note regarding publish type “OWN”**
>
> A custom-built application is required to serve web forms if the publish type is “OWN”. This application is **not** part of the ConSol CM distribution.

> Please see section *Operating ConSol CM behind a proxy (e.g., Apache HTTPD)* of the *ConSol CM Setup Manual* for information about the required proxy settings.

G.4.3 Basic tasks

You can perform the following actions on web forms:

- **Create a new web form**

  Click the *Create new custom form* button above the table with the web forms and make the required settings, see Available settings for web forms.

  > If the new web form is similar to an existing web form, you can create a copy of the web form by selecting the *Clone* option in the form's context menu. You need to provide a unique name for the web form.

  The following steps are required to create a new web form:

  1. Fill out the basic form data, see Basic form data.
  2. Design the form, see Form design.
  3. Decide how the form data should be processed in ConSol CM, see Data processing.
  4. Decide how the form should be published, see Publishing a web form.

  > Depending on the publish type, you might need to create a webhook, see Creating a webhook.

- **Edit a web form**

  Select the *Edit* option in the context menu of the web form and update the form as desired, see Available settings for web forms. If the web form has already been published, the changes become effective immediately.

- **Enable a web form**

  Select the *Enable* option in the context menu of the web form to enable the form. Web forms need to be enabled and published to be used.

- **Disable a web form**

  Select the *Disable* option in the context menu of the web form to disable the form. Disabled web forms cannot be filled in.

- **Delete a web form**

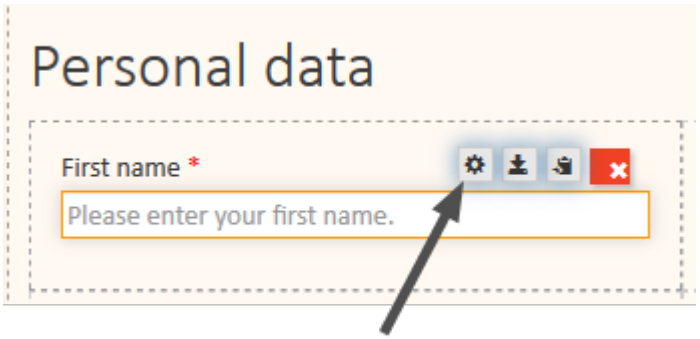
  You can delete a web form by selecting the *Delete* option in the context menu of the web form.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.4.4 Advanced tasks

G.4.4.1 Fine-tuning a web form

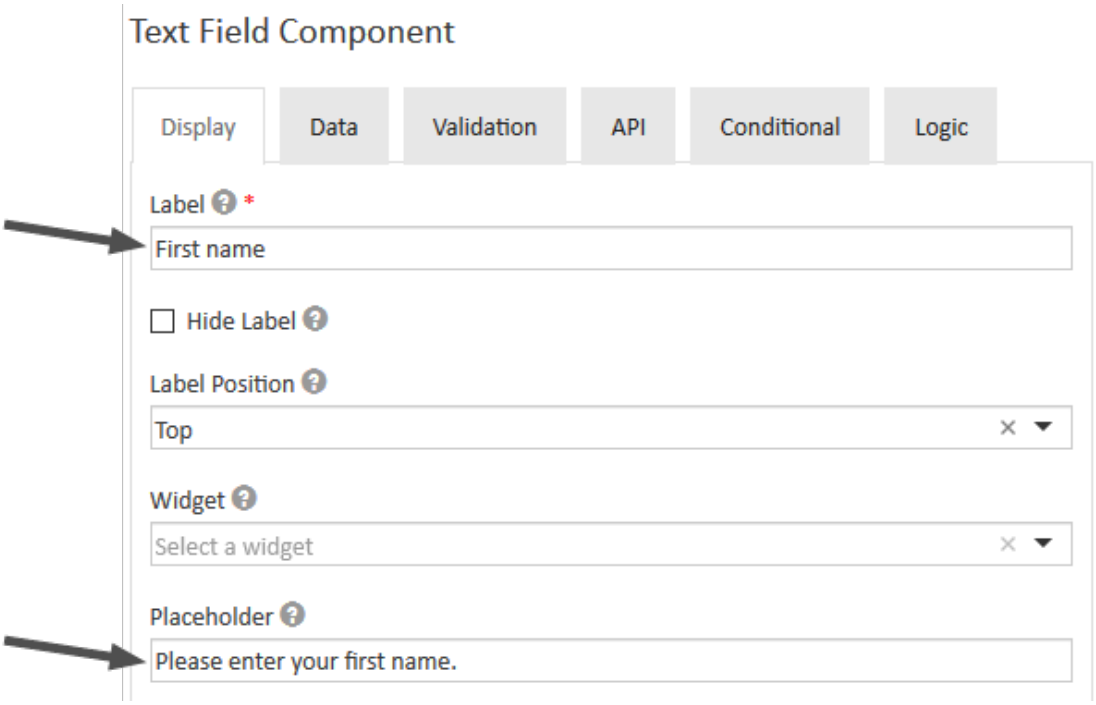
There is a large number of settings you can use for a form field. You can configure the field either directly when creating it, or afterwards by clicking the *Edit* icon:



There are several aspects which you usually need to consider when designing a web form:

- **Labels and placeholders:**

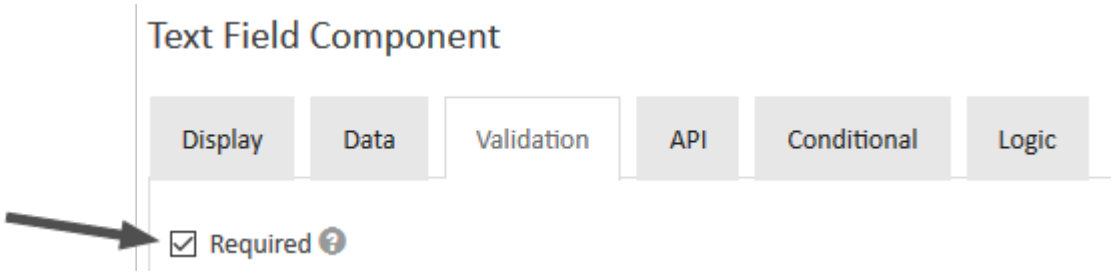
  Set the label and placeholder:



- **Required fields:**

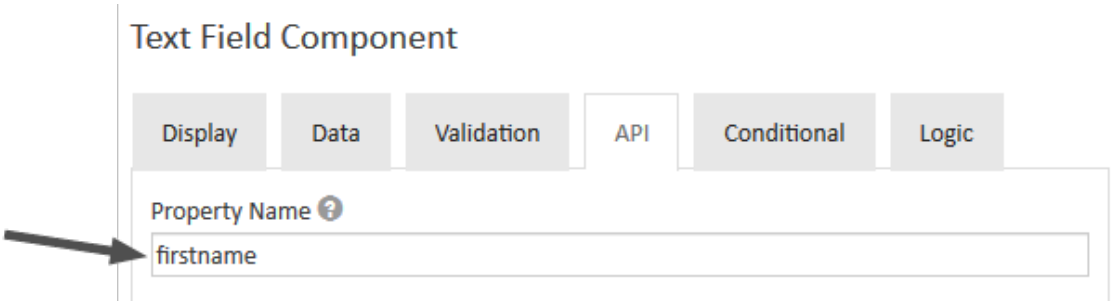
  Set required fields from the data model to required fields in the web form.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH



- **API keys:**

  You need to use the API key to reference the submitted values in the script which processes the form data.



- **Assignment of the CM/Track user profile:**

  If you use a web form to allow the users to register for CM/Track on their own, the CM/Track user profile needs to be assigned to the created contact automatically.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.5 CM/Machine Learning

G.5.1 Introduction to machine learning in ConSol CM

CM/Machine Learning is a ConSol CM add-on which allows to use your existing data to automate processes. With machine learning, you can increase the productivity by detecting and automating routine requests. Manual steps can be omitted and you can gain new insights about your existing data.

The following figure illustrates text classification with machine learning:

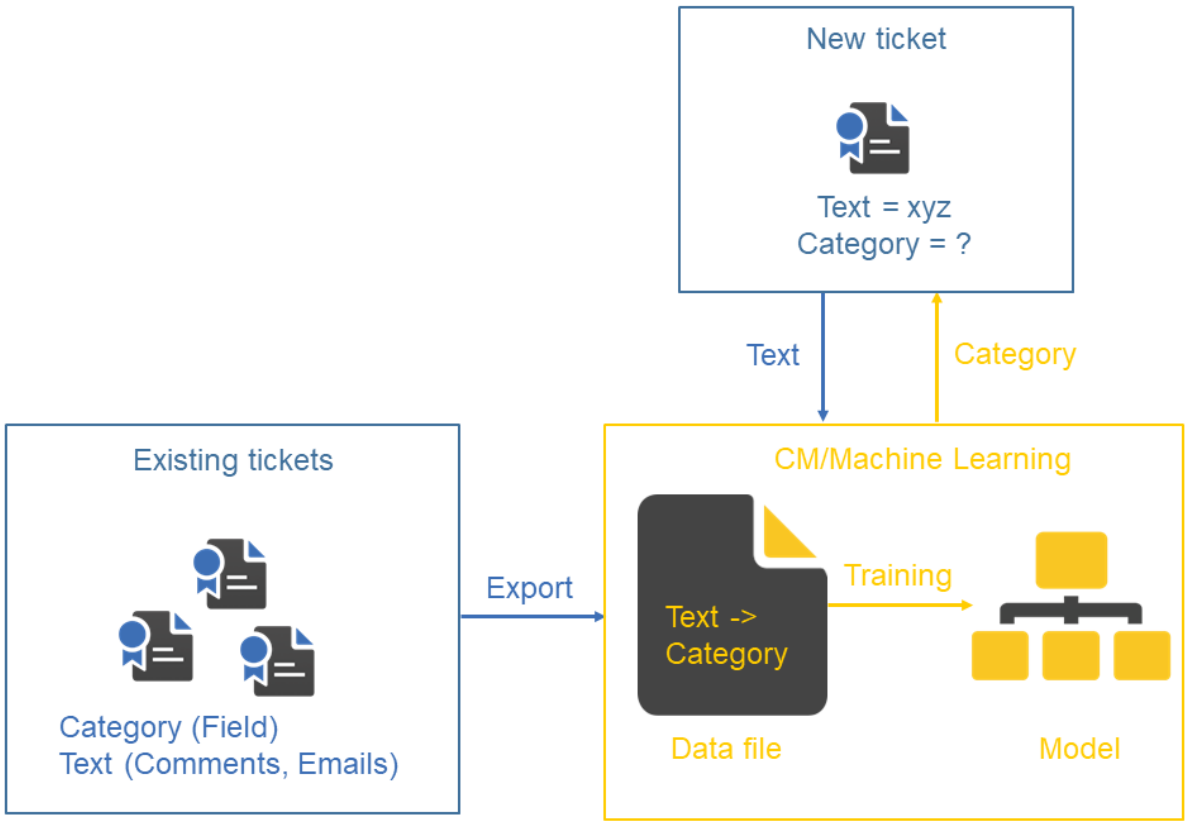


Figure 496: *Principle of text classification with machine learning*

Using machine learning for text classification consists of the following steps:

1. Determine the field used as category. This is a data field of the type ENUM.
2. Export the existing tickets. This creates a data file which contains one entry for each ticket. Every entry consists of the ticket text, i.e. comments and emails, and the value of the category field.
3. CM/Machine Learning transforms the data file into a model which can be queried later.
4. New tickets, which should be classified, already have some ticket text, e.g. the initial comment or email, but their category is unknown. In order to determine the category, the machine learning model is queried to make a prediction of the category based on the available ticket text.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.5.1.1 Concepts, terms and definitions

Concept	Other terms	Definition
CM/Machine Learning		Name of the ConSol CM add-on providing machine learning features
machine learning	ML	Procedure to extract information from existing data and apply it to new data in order to automate processes
training		Process which extracts information from existing data and stores this information in a model that can be queried later
model		Binary file which contains the training result and can be queried
confidence		Value between 0 and 1 which indicates how confident the algorithm is that its prediction is true. This depends on the number of categories, the quality of the training data and the used algorithm. The confidence is typically used to define threshold values for automation actions based on predictions.

G.5.1.2 Purpose and usage

CM/Machine Learning allows to automatically classify tickets based on the existing data. The classification can include the language, sentiment or category of the tickets. The category can be any ticket data field from the data model which you defined, e.g. the category, priority or product of a ticket.

CM/Machine Learning runs locally within your ConSol CM system. There is no data transfer to external providers.

⚠ CM/Machine Learning needs to be licensed separately.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.5.2 Available machine learning features

CM/Machine Learning includes three features:

- **Language detection:**

  Find out the language in which texts are written.

  Example: When you receive an email, detect the language and open the ticket in the queue of the team which works with this language.

- **Sentiment detection:**

  Find out the sentiment (positive, neutral or negative) of texts. Only available for German.

  Example: When you receive an email with negative sentiment of an important customer, you can set a higher priority for the ticket.

- **Ticket classification:**

  Train the machine learning engine with your existing tickets and automatically classify new tickets.

  Example: Set a ticket field with a category based on the initial comment. Determine the queue in which the ticket should be created. Detect routine requests and handle them automatically.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.5.3 Process for introducing machine learning

The process for using machine learning consists of two stages:

1. **Evaluation:**
   First, the existing ConSol CM system is analyzed to determine its potential for machine learning. Once some possible use cases have been found, an evaluation of the costs and gains is performed and an approximate ROI is estimated.
2. **Implementation:**
   The implementation depends on the used features. The language and sentiment detection algorithms can be used as is, i.e. they do not need to be trained for your ConSol CM system. If you use text classification, the most powerful machine learning feature, you need to perform a training and create a model:
   a. **Prepare the training data**: Determine which data fields and texts should be used for training and export the data.
   b. **Train the model**: Split the training data, so that part of the training data is not used for training but reserved for testing the obtained model. Then you create a task script to train the model based on the training data. Usually, several iterations are performed, i.e. after training the model, the training data is fine-tuned based on the results of the tests.
   c. **Integrate machine learning**: Once you have a satisfying model, you can integrate machine learning into the workflows. The first step is usually to prefill data but have the users review them before taking any action. If the precision of the predictions is sufficient, the review step can be omitted to automate the respective processes.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.5.4 Advanced tasks

G.5.4.1 Using language detection

The most important method for working with language detection is `createLanguageDetector()` in the class `nlpService`. It can be used to obtain the language of a text (string). The language is returned as ISO 639-3 code.

The following example shows a script which uses language detection:

```
LanguageDetector languageDetector = nlpService.createLanguageDetector();
DetectedLanguage detectedLanguage = languageDetector.detectLanguage("hello world");
String language = detectedLanguage.getLanguage();
switch (language) {
   case "eng":
      // do something
      break;
   case "deu":
      // do something
      break;
}
double confidence = detectedLanguage.getConfidence();
```

G.5.4.2 Using sentiment detection

The most important method for working with sentiment detection is `createSentimentDetector()` of the class `nlpService`. It can be used to obtain the sentiment of a text (string). The returned sentiment is NEGATIVE, NEUTRAL or POSITIVE. Sentiment detection is supported only for German.

The following example shows a script which uses sentiment detection:

```
boolean html = true;
NormalizeLanguage normalizeLanguage = NormalizeLanguage.GERMAN; // only GERMAN
 language is supported
SentimentDetector sentimentDetector = nlpService.createSentimentDetector
 (normalizeLanguage, html);
DetectedSentiment detectedSentiment = sentimentDetector.detectSentiment
 ("<p>text</p>");
Sentiment sentiment = detectedSentiment.getSentiment();
switch (sentiment) {
   case NEGATIVE:
      // do something
      break;
   case NEUTRAL:
      // do something
      break;
   case POSITIVE:
      // do something
      break;
}
double confidence = detectedSentiment.getConfidence();
```

If you want to track the execution steps of the algorithm, you can call the `detectSentiment` method with an additional boolean parameter:

```
boolean recordDetails = true;
DetectedSentiment detectedSentiment = sentimentDetector.detectSentiment
 ("<p>text</p>", recordDetails);
String details = detectedSentiment.getDetails();
```

G.5.4.3 Using text classification

The most important classes for working with ticket classification are `MLTicketTextClassifierService` and `MLTicketTextClassifierSpec`. You need to perform the following steps:

1. Create the training specification (class `MLTicketTextClassifierSpec`). You can train the model either to determine the queue of a ticket or the value of a ticket field of the type ENUM.
   - Use the method `classifyToQueueName` to train the model to determine the ticket queue, or use the method `classifyToEnumFieldValueName` to decide which ticket field of the type ENUM is the category for which the model should be trained.
   - Decide which tickets should be used for training the model, e.g. tickets in a certain queue, tickets which have a certain value set in a given ENUM field, or tickets for which a certain activity was not yet executed.
   - Decide which kind of content entries should be used for training the model, e.g. only comments, only incoming or outgoing emails.
   - Limit the number of processed tickets and content entries.
2. Train the model (class `MLTicketTextClassifierService`). Use the method `train` to create a `bin` file containing the training results. You need to provide the created training specification.
3. Query the model to classify new tickets (class `MLTicketTextClassifierService`). Use the method `predict` to obtain the predicted classification. You need to provide the model and the ticket text.

> Please see the ConSol CM API documentation for details about the available methods and settings.

The following example shows how to train a model for the field `helpdesk_standard.priority`.

```
// this method will replace old ticket_priority.bin MLFile with new one
MLTicketTextClassifierTrainingSummary summary = mlTicketTextClassifierService.train
 ("ticket_priority.bin",
  MLTicketTextClassifierSpec
     .classifyToEnumFieldValueName("helpdesk_standard", "priority")
     .limitNumberOfTicketsPerEnumFieldValue(10000)
     .limitNumberOfContentEntriesPerTicket(1)
     .trainUsingTexts()
     .trainUsingOnlyClosedTickets()
     .trainUsingOnlyTicketsFromQueues("ml_test")
  );

log.info("Training summary :" + summary)
```

The following example shows how to query the created model:

```
mlTicketTextClassifierService.predict("ticket_priority.bin", "This is the initial
 email sent by the customer to create a ticket.")
```

It is possible to export the data which is used for training to a CSV file, which contains the category and the text. You can inspect and edit the file before using it to create a model.

The following example shows how to generate a CSV file containing the data used for training:

```
// export file from source system / production system
mlFileService.exportWithMetaData(dataset.getId(), new FileOutputStream
 ("/tmp/ticket_priority.csv.jar"));
mlFileService.delete(dataset.getId());

// use created file on training system (note that the data set may change name if
 file ticket_priority.csv already exists)
MLFile dataset = mlFileService.importWithMetaData(new FileInputStream("/tmp/ticket_
 priority.csv.jar"));
```

G.5.4.4 Fine-tuning CM/Machine Learning

There are two system properties which allow to configure the machine learning module:

- cmas-core-server, ml.evict.unused.after.hours
  The number of hours after which a machine learning model is evicted from cache if it is not used. The ConSol CM server checks every hour for machine learning models to evict. The default value is 24.
- cmas-core-server, ml.learn.worker.threads
  The number of threads that a machine learning algorithm can use while training its machine learning model. The default value is 1.

G.6 CM/Archive

G.6.1 Introduction

CM/Archive is a ConSol CM add-on which allows to archive tickets from ConSol CM. The tickets are stored in a MongoDB database and removed from the CM database and the DWH, if desired. Using CM/Archive, it is possible to reduce the database size and related costs, while still storing the tickets for regulatory compliance and future reference.

It can be configured if the whole ticket should be deleted or some part of the ticket should remain in the Web Client. In the latter case, the ticket history and attachments are removed from the ConSol CM database but the basic ticket data remains.

The following figure illustrates how CM/Archive integrates with ConSol CM:

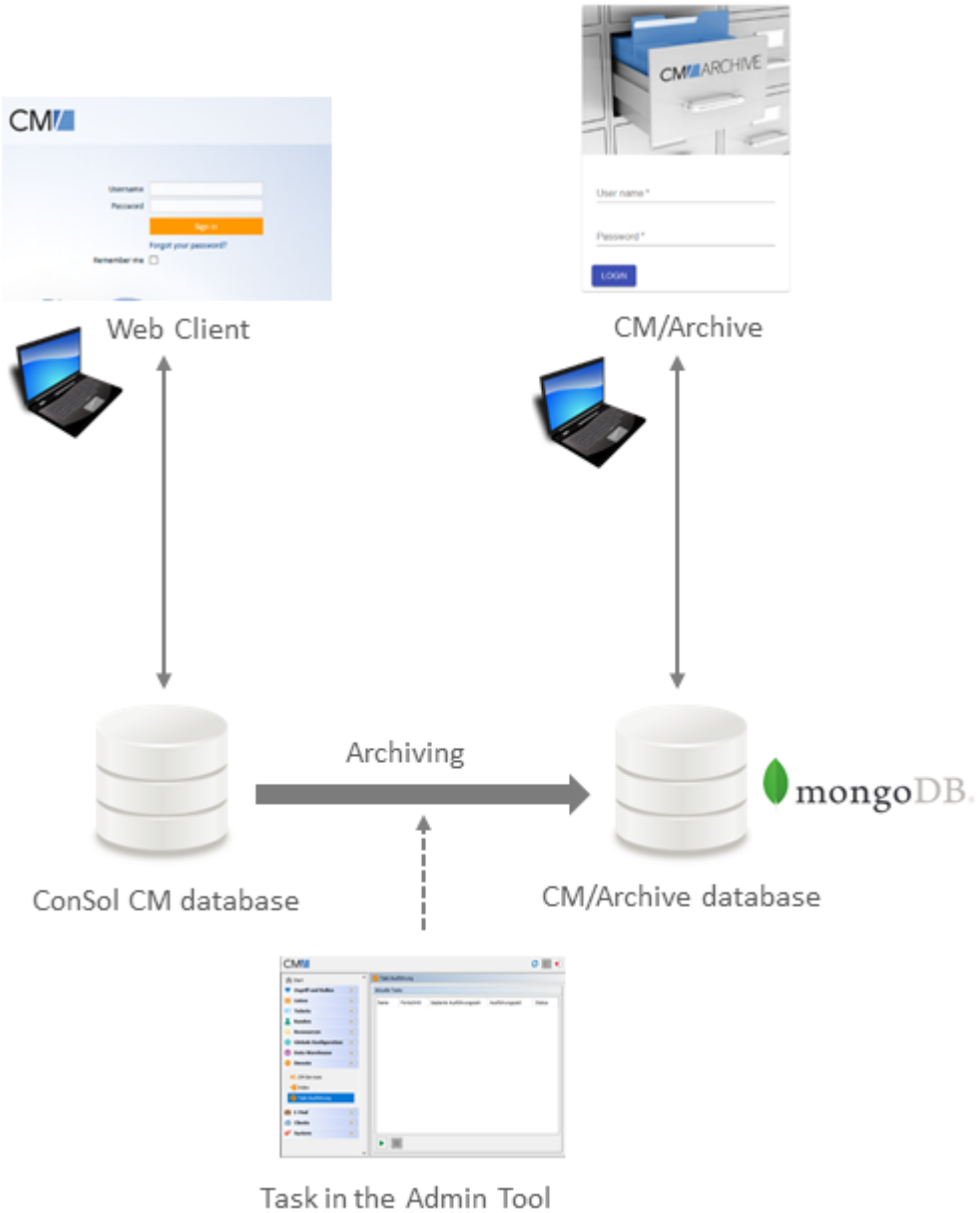


Figure 497: *Architecture of ConSol CM with CM/Archive*

G.6.2 Setting up CM/Archive

The basic process for using CM/Archive is:

1. Install CM/Archive. This is a separate Java application. The installation and setup is explained in detail in the *ConSol CM Setup Manual*.
2. Configure CM/Archive, see Setting the System Properties for CM/Archive.
3. Create a role with archive permissions in the Admin Tool. In addition to the archive permissions (see Archive Permissions) the engineers need read permissions to the queues which the tickets belong to. Any of the read permissions for the queue (*Mine*, *Ref.*, *None*, or *Other*) is sufficient to view tickets from the queue in CM/Archive.
4. Assign the archive role to the engineers who should work with CM/Archive, see Tab Roles - Assign Roles to an Engineer Account.
5. Create a script of the type *Task* with the archiving logic, see Creating the Task Script to Archive Tickets.
6. Execute the task with the created script, see The Task Execution Framework (TEF).

The following sections explain the steps which are specific for CM/Archive in detail.

G.6.2.1 Setting the System Properties for CM/Archive

Some system properties have to be set in the Admin Tool to use CM/Archive.

- Set the URL of CM/Archive using cmas-archive-core-server, archive.uri.
- Configure the authentication tokens of CM/Archive using cmas-auth-server, access.-token.signing.key and cmas-auth-server, client.archive.secret.
- Set the validity period of the authentication tokens using cmas-auth-server, client.archive.access.token.validity.seconds and cmas-auth-server, client.archive.refresh.token.validity.seconds.
- Set cmas-archive-core-server, archive.enabled to determine if GDPR-compliant deletions or anonymizations performed on the ConSol CM database should be applied to CM/Archive as well.

G.6.2.2 Creating the Task Script to Archive Tickets

Before you create the task script, you need to determine the following aspects of the scope of archiving:

- Which tickets should be archived? This can be decided based on different criteria using the available API. For example, it is possible to archive tickets which were closed 5 years ago, or tickets which belong to a certain customer.
- Should the archived tickets be completely removed from the Web Client or should the basic ticket data remain?
  -> Use the method `deleteByIds` to completely delete the ticket, and the method `deleteHistoryEntries` to delete only the ticket history. Both methods belong to `TicketService`.
- If the basic ticket data should remain in ConSol CM, which items of the ticket history should be archived?

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

-> Set the desired value (“ALL” to delete all entries, “ATTACHMENT” to delete only attachments, “LOG” to delete only history entries) in the parameter `pHistoryEntryTypeToDelete` in the method `deleteHistoryEntries`.

- Should the tickets remain in the DWH?
  -> Use the flag `pDwhAware` in the method `deleteByIds`.

For the archiving itself, the method `archiveTicket` from `ArchiveService` is used.



Archiving tickets, i.e. creating tickets in the CM/Archive database and deleting tickets from the ConSol CM database, is done in two separate methods. Therefore, it is possible to split the archiving in two steps in order to check if the tickets are present in CM/Archive before deleting them for good from the CM database.

Please bear in mind that archiving tickets is an irrevocable action which cannot be undone. Archiving tickets has the following implications:



Tickets **cannot** be transferred from CM/Archive back to ConSol CM.

Archived tickets cannot be changed in any way. This means that they cannot be edited, reopened or used as target of relations.

If the archived tickets should remain in the DWH, it is recommended to use partial archiving. Otherwise, the tickets remain in the DWH, but are deleted from the DWH if it is recreated.

Archived tickets only remain available in CM/Track and via REST API if partial deletion is used, i.e., if part of the ticket remains available in the Web Client as well.

The following example shows a task script to archive a single ticket without deleting it from the Web Client. Only the log entries in the ticket history are deleted, comments and attachments remain.

```
import com.consol.cmas.archive.common.model.TicketAo
import com.consol.cmas.common.model.ticket.Ticket
import com.consol.cmas.common.service.TicketService

def onInitialize(taskDescriptor) {
   taskDescriptor.setTxTimeout(24 * 60 * 60)
}

def onExecute(taskDescriptor) {
   //Single ticket to archive
   def id = 100001
   Ticket ticket = ticketService.getById(id)
   if (ticket) {
      archiveService.archiveTicket(ticket)
      log.info("Ticket '$ticket.name' has been archived")
   } else {
      log.info("Ticket (id=$id) doesn't exist")
   }

   //partial deletion, only logs
   ticketService.deleteHistoryEntries(id, HistoryEntryTypeToDelete.LOGS);
}

def onError(taskDescriptor) {}
def onCancel(taskDescriptor) {}
```

Code example 99: *Task script for archiving a single ticket*

You can delete tickets from CM/Archive by using the following method:

```
archiveService.deleteTicket(ticket.getTransferKey())
```

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.6.3 Display of Archived Tickets

If the tickets are only partially archived, they can still be found in the Web Client. The basic ticket data remains as well as relations to customers, tickets and resources. The ticket header includes *(ARCHIVED)* and the link *Open in archive application* which can be used to open the ticket in CM/Archive. The ticket history shows an entry that the ticket has been archived.

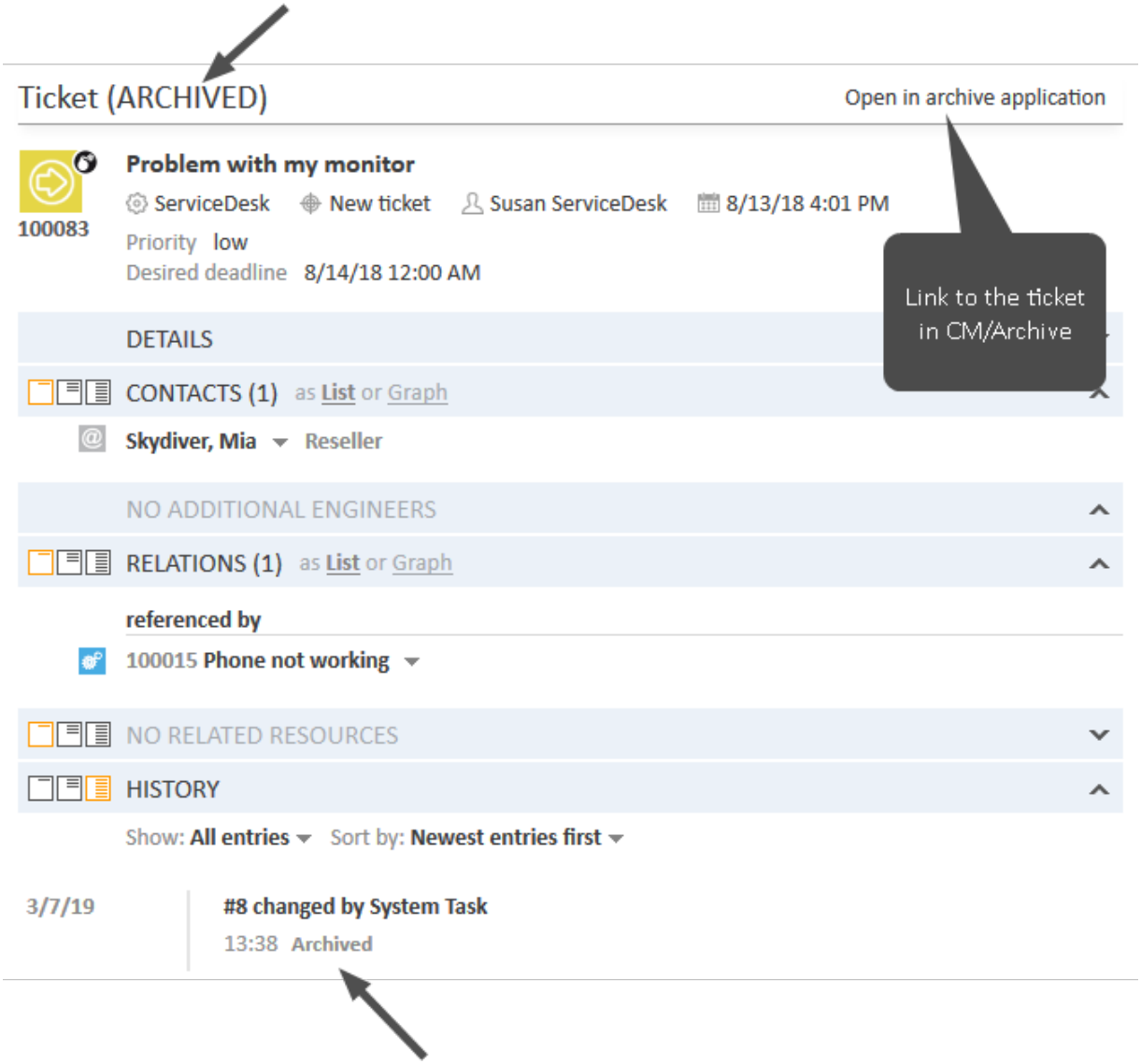


Figure 498: *ConSol CM Web Client - Archived ticket which was partially deleted from the Web Client*

All the archived tickets can be viewed in CM/Archive. The user interface of CM/Archive enables the following actions:

- View statistics (1) - special permissions are required
- Search for a ticket (2)
- Change the GUI language (3) - the available languages are English, German and Polish
- Log out (4)

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

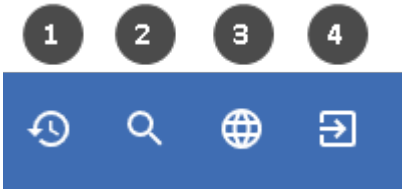


Figure 499: *Icons for actions in CM/Archive*

The following actions are only possible on the ticket details page:

- Delete the ticket (5) - special permissions are required
- Create a PDF with the ticket content (6)



Figure 500: *Icons for actions in CM/Archive (ticket details page)*

The archived ticket consists of several sections which contain the following information:

- **Header**
  Ticket number, subject, queue, scope, workflow, assigned engineer
- **Ticket fields**
  Ticket fields (only fields containing values). The fields from the *Details* section of the ticket are displayed in tabs.
- **Contacts**
  Ticket customers (main and additional customer). You can click the customer to open a pop-up window with the customer data fields. The fields from the *Details* section are displayed in tabs.
- **Referenced tickets**
  Related tickets. If the referenced ticket was also archived, you can click the ticket relation to open it.
- **Referenced resources**
  Related resources
- **Attachments**
  Attachments of the ticket. You can click an attachment to open it.
- **History**
  Ticket history, including comments and emails

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.7 CM/Doc

CM/Doc is a ConSol CM add-on that allows you to create document templates. These templates can be used to create documents directly from the business management process. CM/Doc supports Microsoft Word documents and OpenOffice documents.

Please see section CM/Doc for a detailed description of this add-on.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.8 CM/Phone: CTI with ConSol CM

This chapter discusses the following:

G.8.1 Introduction to CM/Phone 870

G.8.2 CM/Phone Setup 872

G.8.3 Configuration of CM/Phone in the Admin Tool 873

G.8.1 Introduction to CM/Phone

CM/Phone is a ConSol CM add-on which has to be licensed separately. Each client where CM/Phone is installed and active consumes one license of the type REST, see License Management, REST.

CM/Phone is a Windows client application for the integration of telephony systems using the ***TAPI 3*** protocol. TAPI is part of any Windows operating system and provides generic telephony functions. The CM/Phone client has to be installed on each Windows client which should use the CTI (Computer Telephony Integration) functionality with ConSol CM.

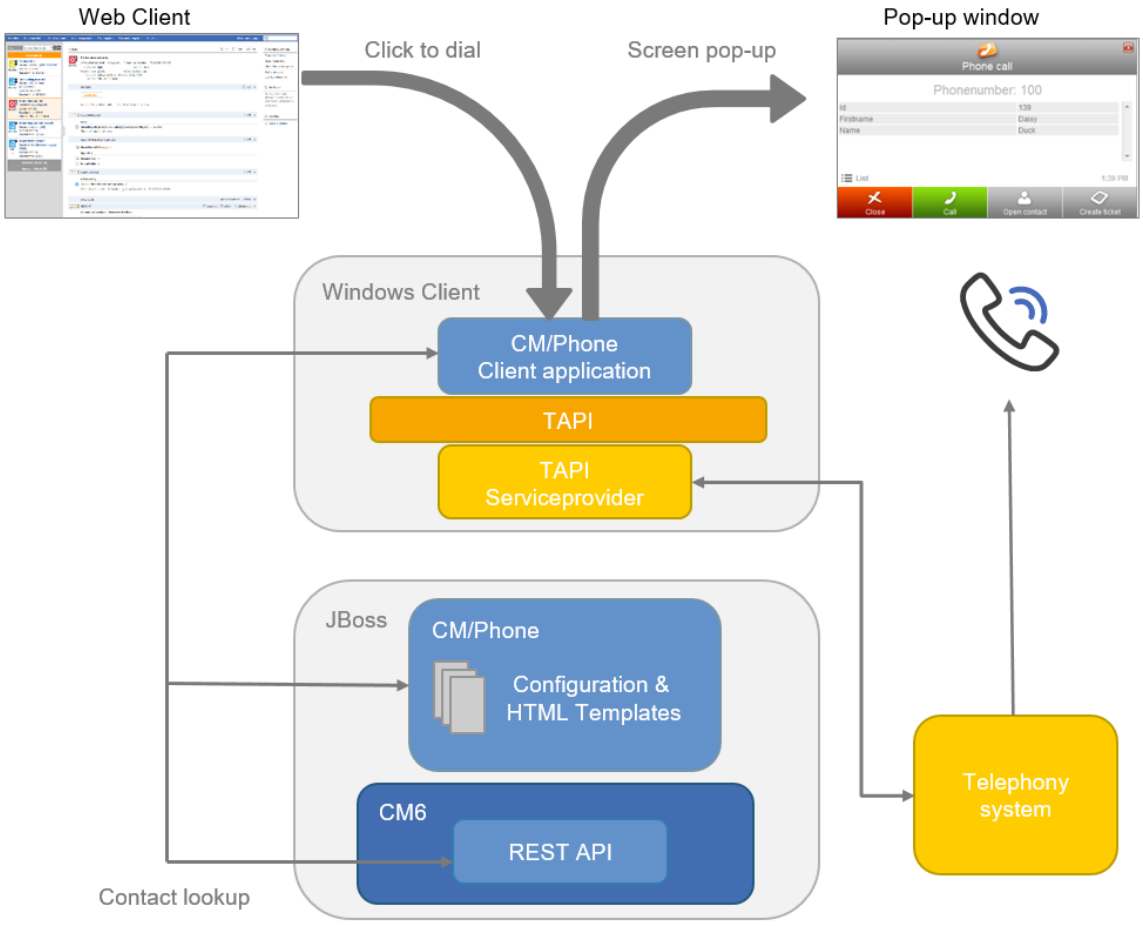


Figure 501: *ConSol CM/Phone - Basic principle*

The engineers can download the CM/Phone application for Windows from the *Downloads* section of their user profiles. Before installing a new CM/Phone version, they need to manually uninstall the previous version.

G.8.1.1 Incoming Calls

The CM/Phone client monitors the telephone handset (i.e., the selected TAPI device, `address` or `line`) for incoming calls. When an incoming call has been registered, a pop-up window is displayed with the phone number of the caller. The ConSol CM customer database is searched for matches for this customer. If one or more matches have been found, a customer list is displayed for selection. Engineers can then decide if they want to create a ticket for the customer or if they want to have the customer page displayed. If no corresponding customer data matches the phone number, just the calling number is displayed and the option *Create customer* is shown.

Please note that a user can only see the customer data in the CM/Phone pop-up window which is allowed by the user's permissions. Others will be filtered out and will thus not be visible.

The pop-up window is based on HTML template files which are located in the CM/Phone folder on the ConSol CM server. These templates are loaded by the CM/Phone client application during startup. The information displayed in the pop-up window (customer fields from the customer data model) can be customized by editing the template files (see *ConSol CM Setup Manual*).

The following options can be selected in the pop-up window if exactly one customer matches in the CM database:

- **Open customer**
  Opens the customer page (contact/company) in the Web Client (alternatively *Create customer* will be listed if the caller is unknown in ConSol CM).
- **Create ticket**
  Opens the *Create ticket* page for this found (or new) customer in the Web Client.
- **Call back**
  Will be available in the case of a missed call.
- **Close**
  Closes the CM/Phone pop-up window.

In case the customer is not yet present in the ConSol CM system, the caller's phone number will be used to fill in the phone number field in the customer data (customer fields) annotated as `dialable`. This will be done for new customers and newly created tickets. Should multiple fields be annotated as `dialable`, the first one will be pre-filled. If the user has access to multiple customer groups, the respective `dialable` phone number fields of each customer group will be pre-filled.

G.8.1.2 Outgoing Calls

The engineer can start an outgoing call directly by clicking on a phone number (e.g., in the customer data) in a customer field which has been annotated as `dialable`. The CM/Phone application is started automatically by the browser and the phone number is passed to the telephone system as a command line parameter. The CM/Phone application creates an outgoing call via TAPI and quits immediately.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.8.2 CM/Phone Setup

Please refer to the *ConSol CM Setup Manual* for a detailed explanation about how to install CM/Phone on the CM server and clients. Here, only the CM/Phone configuration in the Admin Tool will be described.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

G.8.3 Configuration of CM/Phone in the Admin Tool

In the Admin Tool you have to perform the following steps to configure CM/Phone:

- Set the annotations for the customer fields which contain phone numbers.
- Configure the Admin Tool templates for customer data for each customer group.
- Configure the phone number format for each customer group.
- Set the system properties.
- Optional: Change the dialing prefix for outgoing calls.

G.8.3.1 Set Annotations for Customer Fields Containing Phone Numbers

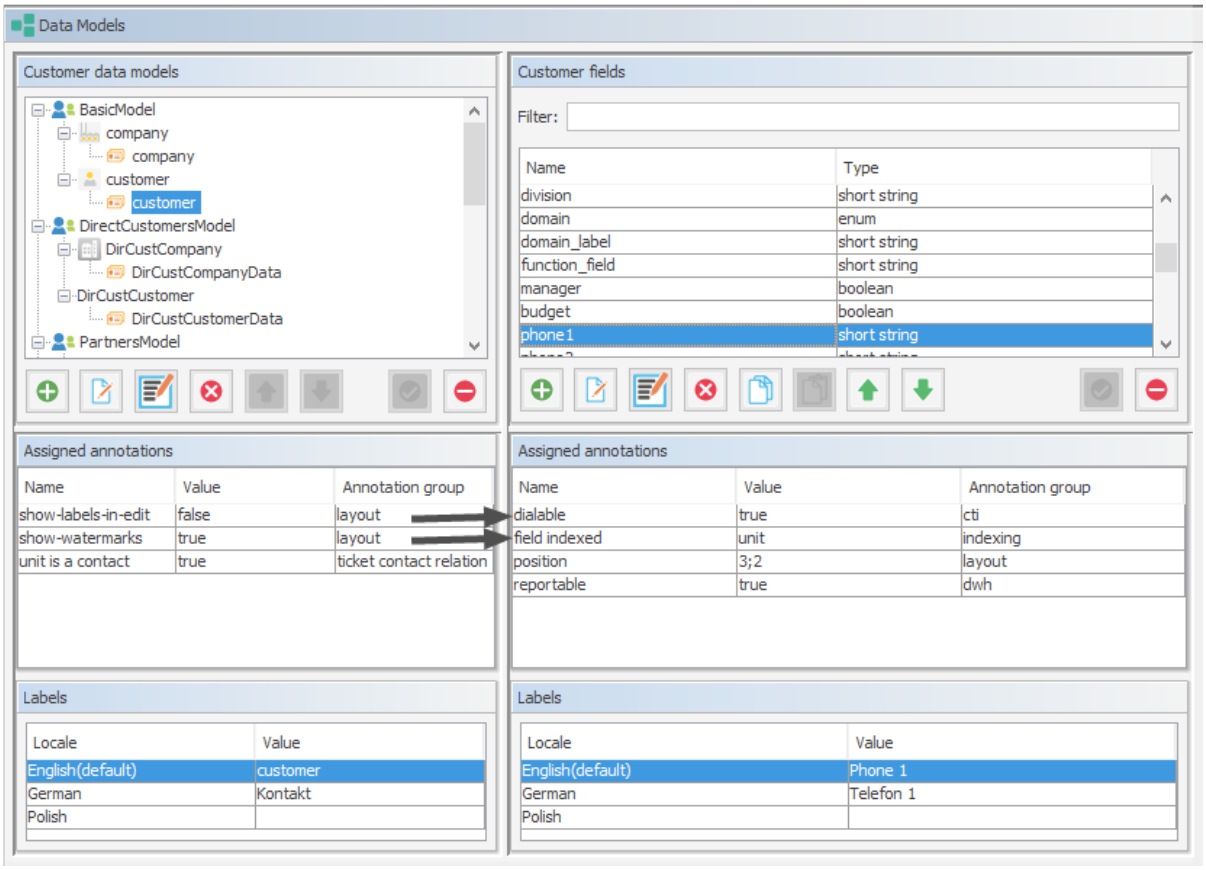


Figure 502: *ConSol CM Admin Tool - Customers, Data Models: Annotations for customer fields with phone numbers*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

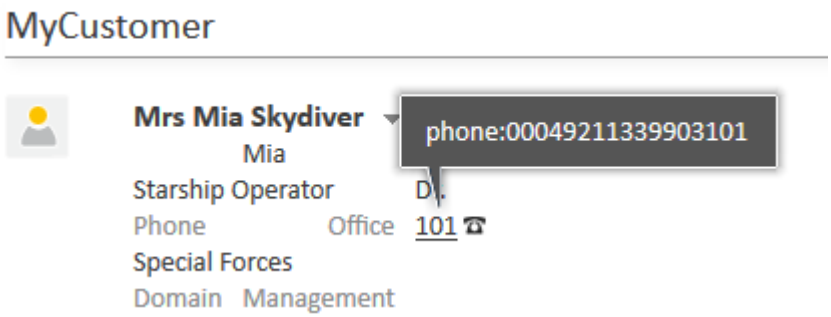


Figure 503: *ConSol CM Web Client - Dialable number when using CM/Phone*

Two annotations are required for customer fields which contain phone numbers:

- **dialable = “true”**
  This configures the phone numbers as dialable links in the Web Client. This is for outgoing calls.
- **field-indexed = “local”**
  This makes the field searchable which is important for the customer look-up. This is for incoming calls.

G.8.3.2 Configure the Templates for Customer Data (for Incoming Calls)

The customer data model configuration includes two CM/Phone-specific types of templates:

- **CMPhone customer details**
  For the display of customer data for a single customer (contact or company, the definition can be made on contact and/or on company level)
- **CMPhone customer list**
  For the display of a list of customers (contact or company, the definition can be made on contact and/or on company level)

They are used for defining how CM/Phone should render incoming call information. The first one is used for exactly one customer matching the phone number and the second one is used for multiple matches, so that the engineer may select the desired customer.

You have to perform two steps:

1. Write the templates and store them in the *Scripts and Templates* section of the Admin Tool.
2. Assign the templates to customer data models (navigation group *Customers*, navigation item *Data Models)*.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

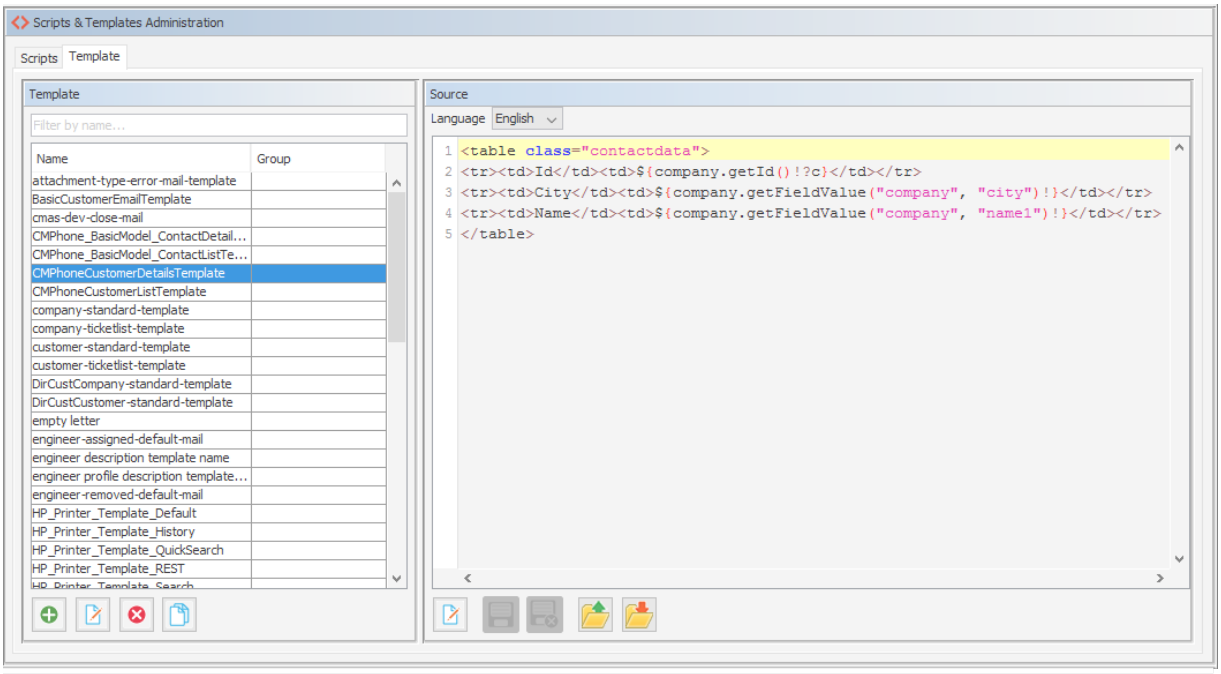


Figure 504: *ConSol CM Admin Tool - System, Scripts and Templates: Example template for rendering customer data for display in CM/Phone*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

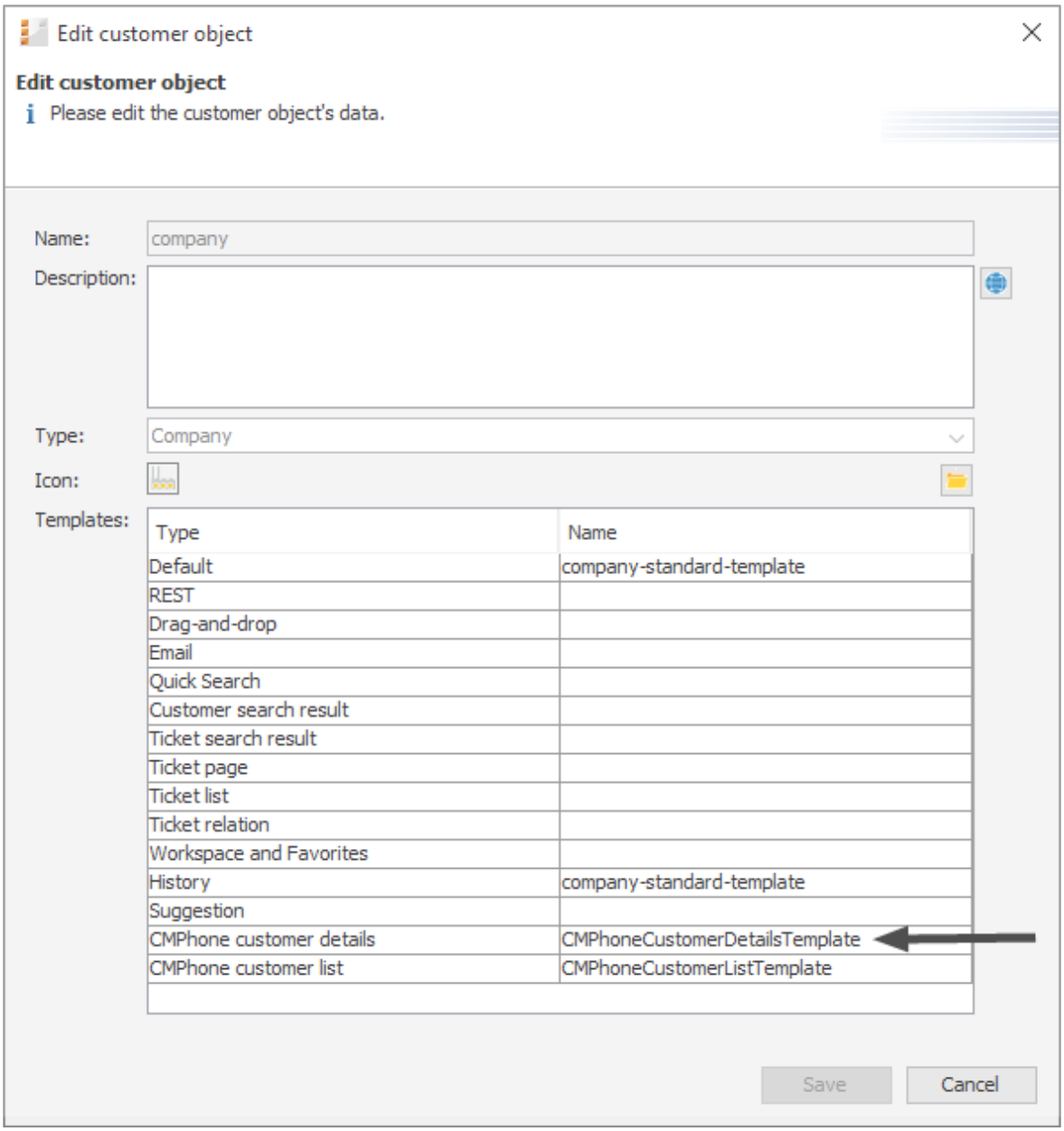


Figure 505: *ConSol CM Admin Tool - Customers, Data Models: Assignment of CM/Phone templates for customer data to customer groups*

G.8.3.3 Configure the Phone Number Format (for Incoming and Outgoing Calls)

The format defined here is used to transform phone numbers (from the respective customer group) to a common canonical form. The engineer can enter a phone number in any format with or without prefixes, e.g., as company internal number. To avoid problems with interpreting such numbers there is a dedicated configuration per customer group which is used when a user submits a phone number for a particular customer. The patterns/elements of the different formats which can be interpreted as a phone number in the fields marked as `dialable` can be defined in detail in the Admin Tool.

The navigation item *Customer Groups* in navigation group *Customers* has to be selected after logging in to the Admin Tool for this configuration. On the navigation item *Customer groups*, the desired customer group has to be selected for editing. After clicking the *Edit* button below the list of customer groups, the edit dialog opens, containing a new tab titled *CMPhone*.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

On the *CMPhone* tab of the *Edit customer group* dialog there are fields in which you can enter phone number prefixes for different scopes and number patterns for several phone number types.

The fields for configuration values are:

- **Country prefix**
  The international country prefix for extending national phone numbers, without prefixes like “0” or “+”. Such a prefix is not allowed here!
  The country prefix is required in order to check whether or not an outgoing call is within the same country. Some phone providers do not handle canonical (so, theoretically, correct) numbers for domestic calls, and the country prefix has to be removed from the number in such cases.
- **Area prefix**
  The local city/area prefix for extending local phone numbers. Please note that this also does not include general prefixes like “0” or “1”, so the entry for Munich in Germany would be “89”, not “089”!
- **Company prefix**
  The phone number of the company as used in (local) calls without extensions. Adding an extension number to this prefix would allow a local call from outside the company to this extension.
- **Subscriber pattern**
  This regular expression (RegEx) describes a number pattern used to identify whether the number provided is a full subscriber number (potentially including an extension) which would allow for a local call.
- **Internal pattern**
  The regular expression (RegEx) in this field defines the pattern to classify extensions if only a phone extension is entered.
- **Mobile pattern**
  This regular expression (RegEx) is used to identify a number entered as a mobile/cell phone number in the country, which would be valid to make a national call to a mobile phone.

For example, for all numbers (12, 33990312, 21133990312) from above points the result should be always the full canonical number: 4921133990312. For mobile numbers, a country prefix will also be added, so the result will be: 49600289906. If the engineer enters a full number starting with “+” or “0” then the configuration is skipped - ConSol CM assumes no number conversion is required.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

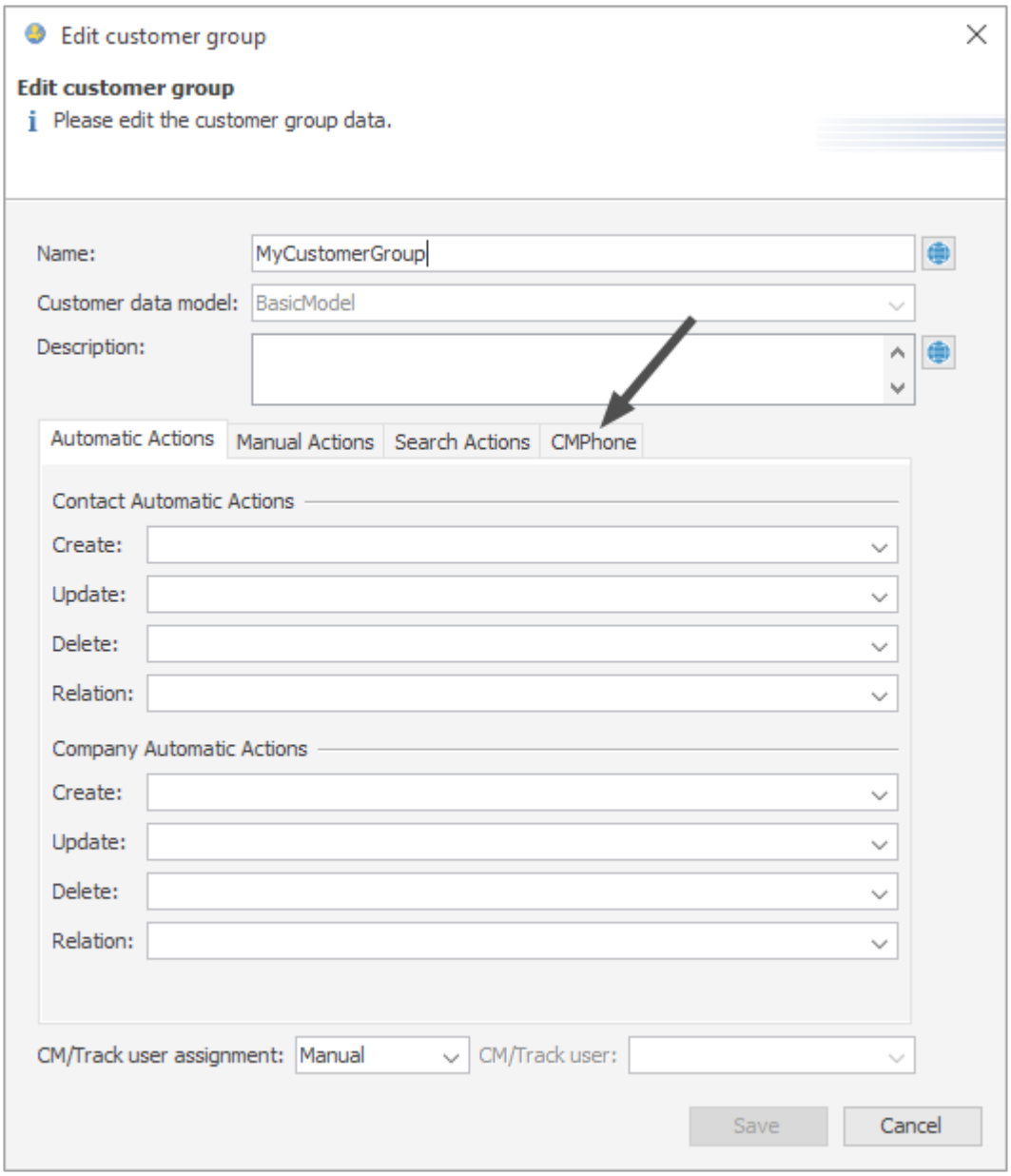


Figure 506: *ConSol CM Admin Tool - Customers, Customer Groups: Configuration of phone number formats for a customer group*

These prefix values are defaults for extending phone numbers which are not fully qualified. They can always be overridden by entering a fully qualified phone number.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

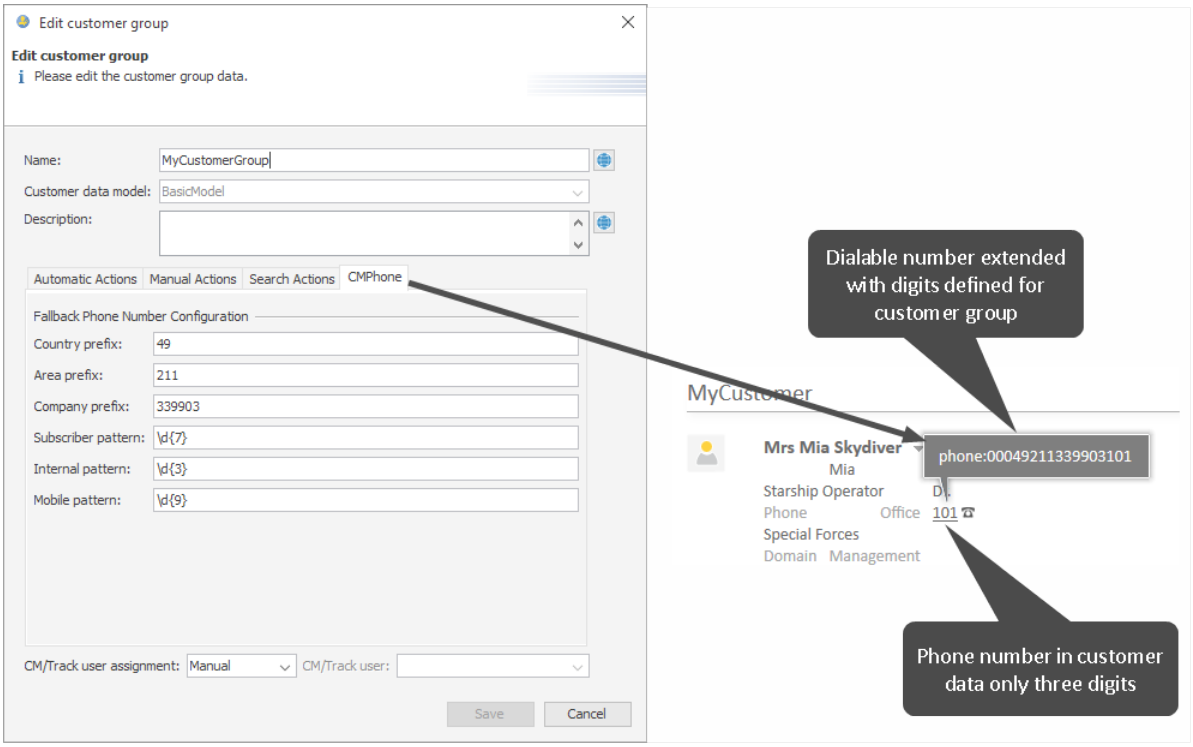


Figure 507: *CM/Phone - Use of customer group-specific number configuration for dialable numbers (outgoing calls)*

The patterns are used to guess the type of a phone number which is not fully qualified. The guessed type determines its use and necessary additions for connecting a call. For this purpose, after removing unnecessary characters, a number is checked to determine whether it is already fully qualified. If not, it is matched against these patterns. For exactly one match, a valid number is constructed and used. If two matches are area code and mobile number, these are combined with the country prefix to create the number to be dialed. In all other cases the supplied original number cannot be used for making a connection.

G.8.3.4 Set the System Properties

There are three new properties in ConSol CM, to be set in the Admin Tool, which are relevant for CM/Phone. The correct configuration for these is essential for proper usage of phone numbers for connection calls. The properties are elements of the module `cmas-core-server`:

- local.country.prefix
  String. This is the local country code. The value is an international country code, like “49” for Germany. Default value is “49”.
- internal.line.access.prefix
  This is a prefix that the company’s telephony system uses for outside lines, if required. So, if a “0” or a “9” needs to be dialed in order to make a call to any number outside the company, this value needs to be configured here. Default value is “0”.
- external.line.access.prefix
  This is the general prefix to dial before an area code to get a long-distance connection in the country. For example, in Germany it is a “0” that needs to be prepended to the area code. Default value is “0”.

These properties are all optional, so they have to be added manually, if needed.

G.8.3.5 Change the Prefix for Outgoing Calls



This step is optional!

Usually the prefix `phone:` is set before the number for outgoing calls for interaction with the TAPI. If another prefix (e.g., `tel:`) is required, this can be configured in the ***Windows Registry***. Please ask your ConSol CM consultant for advice.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

H - Appendix

This section contains several appendices:

- Annotations
- System Properties
- Administrator and Notification Email Addresses
- Default Java Imports
- List of Code Examples
- Trademarks
- Glossary

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

H.1 Annotations

There are two types of annotations: field annotations and group annotations. Field annotations are applied to a single ticket, customer or resource field. Group annotations are applied to a ticket, customer or resource field group. Please see:

- List of Field Annotations
- List of Group Annotations

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

H.1.1 List of Field Annotations

This chapter describes the following field annotations grouped by annotation type:

personal-data 885
groupable 885
sortable 885
autocomplete-script 885
leave-trailing-zeros 886
readonly 886
string-content-type 886
visibility 886
visualization 887
visualize-when-empty 887
boolean-type 887
enum-in-search-type 887
enum-type 888
list-type 888
text-type 888
ldapid 889
password 889
username 889
dwh-no-history-field 889
reportable 890
field indexed 890
phonetic 890
colspan 890
field-group 891
fieldsize 891
label-group 891
label-in-view 891
order-in-result 892
position 892
rowspan 892
show-label-in-edit 892
show-label-in-view 892
show-tooltip 892
show-watermark 893

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

ticket-list-colspan 893
ticket-list-position 893
ticket-list-rowspan 893
no-history-field 894
dialable 894
resource-color 894
customer exposure 894
contact search result column 895
contains contacts 895
enum field with ticket color 895
accuracy 895
email 896
format 896
matches 896
maxLength 896
maxValue 896
minLength 897
minValue 897
required 897
visibility configuration 897

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

H.1.1.1 anonymize (type)

personal-data

- **Type**: anonymize
- **Description**: Can be assigned to ticket and contact fields. Determines that the field holds personal data. Contact fields with this annotation will be deleted when the contact is anonymized. Ticket fields with this annotation will be deleted when the main customer of the ticket is anonymized. See Example 8: Removing Customer Data for information about how to anonymize a contact.
- **Values**:
  - *true / false*: Set “true” if the field contains personal data.

H.1.1.2 cmweb-common (type)

groupable

- **Type**: cmweb-common
- **Description**: Enables grouping of the ticket list by this field. Please see section Grouping for a detailed explanation.
- **Values**:
  - *true*: Used only with ENUM data fields. Remove the annotation if you want to disable grouping.

sortable

- **Type**: cmweb-common
- **Description**: Used to enable sorting of the ticket list by this field. Please see also the detailed explanation in section List of Field Annotations
- **Values**:
  - *true*: Used for data fields of type DATE or of type ENUM. Remove the annotation if you want to disable sorting.
    For ENUM fields: Works only if order index is set for all values of the ENUM field.

H.1.1.3 common (type)

autocomplete-script

- **Type**: common
- **Description**: Specifies the script (type *Text autocomplete*) used for the scripted autocomplete field.
- **Values**: <name of the script>

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

> **Note:**
>
> This annotation is part of the way of configuring autocomplete fields which should only be used in ConSol CM versions up to 6.11.1.0. Starting with version 6.11.1.1, a new way of implementing scripted autocomplete lists is available. See section *Scripted Autocomplete Lists* in the *ConSol CM Administrator Manual*.
>
> Existing fields which use the old method will remain functional as of version 6.11.1.1. These fields will not be modified during an update. The previously used method `onEditDisplayEntered` still works for existing fields. Nevertheless, it is not needed anymore and will be removed in a future version of ConSol CM.

leave-trailing-zeros

- **Type**: common
- **Description**: Used for the display of fixed point numbers.
- **Values**:
  - *true / false*: Trailing zeros in the fractional part are not cut off when value is “true”.

readonly

- **Type**: common
- **Description**: Used to indicate that the ticket field cannot be modified.
- **Values**:
  - **all**: The field is always read-only.
  - **edit**: The field is read-only when creating or editing a ticket, customer or resource. It can be edited in forms.
  - **form**: The field is read-only in forms. It can be edited when creating or editing a ticket, customer or resource.
  - **unset**: The field is always editable. This is the default behavior if the annotation is not assigned.

string-content-type

- **Type**: common
- **Description**: Specifies the type of display for a rich text field in view mode.
- **Values**:
  - **html-inline**: The content of the rich text field is displayed directly on the GUI.
  - **html-link**: The content of the rich text field is displayed in a pop-up window which the user can access by clicking the *Show content* link.

visibility

- **Type**: common
- **Description**: Defines when the field is visible.

- **Values**:
  - *edit*: Field will be displayed in edit mode.
  - *view*: Field will be displayed in view mode.
  - *none*: Field is not visible.
  - If any other, or no value, is set then the field will always be visible.

visualization

- **Type**: common
- **Description**: Specifies the script (type *Field visualization*) used to render the content of the data field. For a detailed explanation, please refer to the section *Scripts of Type Field Visualization* .
- **Values**: <name of the script>

visualize-when-empty

- **Type**: common
- **Description**: Determines if the visualization script of a data field with the annotation `visualization` should be executed even if the field is empty. By default, the visualization script is only executed if the field contains a value.
- **Values**:
  - *true / false*: Set “true” if the field visualization script should be executed even if the field is empty.



This annotation is applied for scripted field visualization both in the Web Client and CM/Track.

H.1.1.4 component type (type)

boolean-type

- **Type**: component-type
- **Description**: Definition of the layout of a boolean field.
- **Values**:
  - *checkbox* (default): Field that can be checked (set to “false ”by default).
  - *radio*: 2 radio buttons (yes/no) for selection (only one can be active).
  - *select*: Drop-down field with 2 values (yes/no).

enum-in-search-type

- **Type**: component-type
- **Description**: Defines whether an ENUM field used in a search accepts searching over multiple values.
- **Values**:
  - *single* (default) / *multiple*: Accepts searching over multiple values if value “multiple” is set.

enum-type

- **Type**: component-type
- **Description**: Layout definition of list display
- **Values**:
  - *select* (default): Drop-down list for selection.
  - *radio*: List of radio buttons to select (only one option can be active).
  - *autocomplete*: Drop-down list for selections where the field is an input field used to filter the list.

list-type

- **Type**: component-type
- **Description**: Disables the add and/or delete options for data fields of type LIST or STRUCT.
- **Values**:
  - *fixed-size*: It is not possible to add or delete fields/rows.
  - *non-shrinkable*: It is not possible to delete fields/rows.
  - *non-growable*: It is not possible to add fields/rows.

text-type

- **Type**: component-type
- **Description**: Defines the possible types of a STRING field.
- **Values**:
  - *text* (default): Single-line input field
  - *textarea*: Multi-line input field
  - *password*: Input field for passwords.
    Password will be displayed as ******* in view mode.
  - *label*: Read-only label, no input is possible.
  - *title*: Read-only label styled as a headline, no input is possible.
  - *autocomplete*: The field will be used as autocomplete list. Please see the detailed explanation in section Scripts of Type Text Autocomplete.
  - *url*: The input will be displayed as a hyperlink in view mode. If no protocol is provided, *http://* is added automatically in front of the entered string. If a protocol, e.g. “http”, “https”, “mailto”, “file”, or “ftp” is used, the URL is rendered as is. The display text for the URL can be entered after a whitespace.
    Example: “http://consol.de ConSol”
  - *file-url*: Input will be displayed as a link to a file on the file system. The web browser has to allow/support those links! See section Using Annotations to Fine-Tune Data Fields on how to achieve this. The link will also be displayed as tooltip.

    The URL is correctly formed if the following conditions are met:

    It starts with `file:` followed by regular slashes:

- three slashes “///” for files on the same computer as the browser (alternatively “//localhost/”) or
- two slashes followed by the server name followed by another slash for files on file servers accessible from the computer running the browser.

These are followed by the full path to the file ending with the file name. The path on Microsoft Windows systems is also written with forward slashes instead of backslashes.

The drive letter of a local path on Microsoft Windows systems is noted as usual, for example C:. Paths with spaces and special characters like “{, }, ^, #, ?” need to be percent encoded (“%20” for a space for example) for Microsoft Windows systems.

Example URLs:

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

See also the explanation in the section Using Annotations to Fine-Tune Data Fields.

H.1.1.5 contact authentication (type)

ldapid

- **Type**: contact authentication
- **Description**: Used in a customer field group of type *contact*, for the customer field which contains the LDAP ID for CM/Track authentication.
- **Values**: Indicates that this field will be used as an LDAP ID in the authentication process. Data type string is required.
  Since the definition is made at the customer group level, the LDAP authentication can be run in mixed mode. I.e., use LDAP for some customer groups and regular authentication for other customer groups.

password

- **Type**: contact authentication
- **Description**: Indicates that this field will be used as a password in the authentication process.
- **Values**:
  - **<string>**: Used for CM/Track.

username

- **Type**: contact authentification
- **Description**: Indicates that this field will be used as a login name in the authentication process.
- **Values**:
  - *true / false*: Used for CM/Track.

H.1.1.6 dwh (type)

dwh-no-history-field

- **Type**: dwh
- **Description**: Annotation used to indicate that field will not be historized in DWH

- **Values**:
  - *true / false*: Since version 6.10.2.0.

reportable

- **Type**: dwh
- **Description**: Indicates that the field is reportable and that it should be transferred to the DWH.
- **Values**:
  - *true / false*: Field is reportable if value is set to “true”.

H.1.1.7 indexing (type)

field indexed

- **Type**: indexing
- **Description**: Indicates that a database index will be created for this field. This is necessary if the field should be used to search for entities or to sort entries in a table.
- **Values**:
  - *transitive*: Only relevant for customers. An index is created for the field, so the customer can be searched by this field. For contacts, the tickets of the contact are also found. For companies, the tickets of the company and the contacts of the company are found as well.
  - *unit*: Only relevant for customers. An index is created for the field, so the customer can be searched by this field. For contact, only the contact itself is found. For companies, the contacts of the company are found as well. This is the default value for customers.
  - *local*: An index is created for the field, so the object (ticket, customer or resource) can be searched by this field. Only the object itself is found. This is the default value for tickets and resources.
  - *<annotation not set>*: No index is created for the field, so it it cannot be searched.

phonetic

- **Type**: indexing
- **Description**: activates the phonetic search for this field. Can only be used for data fields of type String (also long or short string).
- **Values**: *true/false*. Will automatically set to “true” when the annotation is added.

H.1.1.8 layout (type)

colspan

- **Type**: layout
- **Description**: Defines how many columns are reserved for the field in the layout.
- **Values**:
  - **<number>**: Number of columns.



This annotation only works if the annotation `position` is also set for the field.

field-group

- **Type**: layout
- **Description**: Allows grouping of fields in view mode. Annotation is ignored in edit mode.
- **Values**:
  - **<string>**: To group fields the same string value has to be set in the annotation of each field. Two or more data fields are bound when they share the same value for this annotation. The group of coupled data fields is shown only if all of them have values set.
- Removed in ConSol CM version 6.11.0.1.

fieldsize

- **Type**: layout
- **Description**: Displayed field size within the ticket layout.
- **Values**:
  - **<rows>;<cols>**: Displayed field size for textareas

    For STRING fields with `text-type` = `textarea`: rows;cols (corresponds to <textarea rows="" cols="">).
  - **<number>**: Displayed field size for strings and numbers

    For STRING fields with `text-type` other than `textarea` and NUMBER fields: n indicates the number of characters in the fields; for string fields this is the number of monospaced capital M characters.

    For ENUM data fields: Defines how many values are directly visible in the list box. Used only for layout purposes. This annotation is not relevant for ENUM autocomplete fields.



This is only a layout configuration, for validation use `maxlength` of type `validation`.

label-group

- **Type**: layout
- **Description**: Indicates a group of fields along with its descriptive label in view mode. Annotation is ignored in edit mode.
- **Values**:
  - **<string>**: Indicates a group of data fields along with its descriptive label. The annotation is used in view mode, ignored in edit mode. The group can have exactly one label (a data field of type STRING with assigned additional annotation `text-type` with value “label”). The label is shown when at least one data field from its group has a value set. All fields with the same label value are grouped and displayed under this label.
    The annotation `label-group` has to be assigned to the label, too.

label-in-view

- **Type**: layout
- **Description**: Shows data field value as a label in view mode. Annotation is ignored in edit mode.
- **Values**:
  - *true*: Remove the annotation if the label should not be visible in view mode.

order-in-result

- **Type**: layout
- **Description**: Shows field as a column at given position in the search result list.
- **Values**:
  - **<number>**: The columns are sorted in ascending order.
    Since CM version 6.0.1. Please see detailed explanation in info box in section Search Configuration, order-in-result.

position

- **Type**: layout
- **Description**: Defines the position of a field within a grid layout or defines the position of a field within a list (STRUCT).
- **Values**:
  - **<number>;<number>**: Values define row and column (row;column), numbering starts at 0;0. If no values are set, the data field will take the next free grid cell.
  - **0;<number>**: Only the column value is used, the row value is ignored.

rowspan

- **Type**: layout
- **Description**: Indicates how many rows within the layout are occupied by this field.
- **Values**:
  - **<number>**: Number of rows.

⚠ This annotation only works if the annotation `position` is also set for the field.

⚠ This annotation does not work in version 6.13.0 and higher.

show-label-in-edit

- **Type**: layout
- **Description**: Whether the data field should be displayed in edit mode with label.
- **Values**:
  - *true / false*: Since version 6.9.4

show-label-in-view

- **Type**: layout
- **Description**: Whether the data field should be displayed in view mode with label.
- **Values**:
  - *true / false*: Since version 6.9.4

show-tooltip

- **Type**: layout
- **Description**: Whether the data field should be displayed with tooltip.

- **Values**:
  - *true / false*: Since version 6.9.4

show-watermark

- **Type**: layout
- **Description**: Whether the data field should be displayed with watermark.
- **Values**:
  - *true / false*: Since version 6.9.4

ticket-list-colspan

- **Type**: layout
- **Description**: Defines how many columns are occupied by the field in the ticket list box.
- **Values**:
  - **<number>**: Number of columns.

⚠ This annotation only works if the annotation `ticket-list-position` is also set for the field.

ticket-list-position

- **Type**: layout
- **Description**: Defines the position of the field in the ticket list box.
- **Values**:
  - **<number>;<number>**: Values define row and column (row;column), numbering starts at 0;0.

ticket-list-rowspan

- **Type**: layout
- **Description**: Defines how many rows are occupied by the field in the ticket list box.
- **Values**:
  - **<number>**: Number of rows.

⚠ This annotation only works if the annotation `ticket-list-position` is also set for the field.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

H.1.1.9 performance (type)

no-history-field

- **Type**: performance
- **Description**: Indicates that a single data field should not be historicized. Overwrites the group annotation `no-history`.
- **Values**:
  - *true / false*: Annotation is active if value is set to “true”. For fields that should be stored but not be visible in history use annotation `visibility configuration`.
    In CM versions up to 6.10.2, the DWH transfer of a field history is also controlled by this annotation.
    Starting with CM version 6.10.2, use the annotation `dwh-no-history-field` for this.

H.1.1.10 phone commander (type)

dialable

- **Type**: phone commander (CM/Phone)
- **Description**: Defines a field with a phone number.
- **Values**:
  - *true*: Used with CM/Phone only. Marks a phone number as automatically dialable for outgoing calls for the CTI system.

H.1.1.11 resource (type)

resource-color

- **Type**: resource
- **Description**:
- **Values**:
  - *true / false*: Should be assigned to a color ENUM. Color of selected enum value should be applied to a resource icon's background.

H.1.1.12 restapi (type)

customer exposure

- **Type**: restapi
- **Description**: Indicates whether a data field should be available to customers using the REST API, e.g. in CM/Track.
- **Values**:
  - *full* (default): The data field is available for reading and writing.
  - *read*: The data field is available for reading only.
  - *none*: The data field is not available. This value can be helpful, for example, when an entire ticket field group has been configured with `customer exposure group` = “full|read” and dedicated single fields should not be available in CM/Track (or over the REST API in general).

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

H.1.1.13 search result (type)

contact search result column

- **Type**: search result
- **Description**: Identifies whether the field should be presented in the search result by default. **Deprecated! Do not use!** Removed in ConSol CM version 6.11.
- **Values**:
  - *true*:
    Remove the annotation if the field should not be visible by default.
    Since CM version 6.1.3.
    (Replaced by `order-in-result`! `contact search result column` is obsolete!)

H.1.1.14 ticket contact relation type (type)

contains contacts

- **Type**: ticket contact relation type
- **Description**: Only one instance of this annotation is allowed in one data model! Used only for list field definition, indicates that it can hold unit references to units annotated as contacts. Starting with ConSol CM version 6.11, the field annotated with contains contact will not produce history entries. This decreases the loading time for tickets which contain a great number of customers.
- **Values**:
  - *true/false*: Value type is boolean. Specifies whether the list is shown with the contact (“true”) or with the ticket (“false”).

H.1.1.15 ticket display (type)

enum field with ticket color

- **Type**: ticket display
- **Description**: Defines the background color of the ticket icon for ticket list and ticket.
- **Values**:
  - *true / false*: The field has to exist within Enum Administration where lists, values, and colors are defined.

H.1.1.16 validation (type)

accuracy

- **Type**: validation
- **Description**: For ticket, customer and resource fields of type DATE. To define the level of detail displayed
- **Values**:
  - *date (default)*: Show date without time.
  - *date-time*: Show date with time.
  - *only-time*: Show only time, no date.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

email

- **Type**: validation
- **Description**: Used for email addresses to validate that the format is correct, i.e., that it matches <name>@<domain>.
- **Values**:
  - *true*: May be used with STRING data fields. Remove the annotation if the format should not be validated.

format

- **Type**: validation
- **Description**: Used for validating the format of date fields.
- **Values**:
  - *<date format>*: The pattern for the date is based on `SimpleDateFormat`, e.g., dd.MM.yyyy.
    Remember to set the proper colspan when including hours/minutes in the format. See http://docs.oracle.com/javase/6/docs/api/java/text/SimpleDateFormat.html for the format reference.

> This annotation is applied to date fields both in the Web Client and in CM/Track. If it is not set, the standard date format of the browser locale is used in the Web Client, and the German standard date format (dd.MM.yyyy) is used in CM/Track.

matches

- **Type**: validation
- **Description**: Checks if input of STRING data fields matches the given RegEx.
- **Values**:
  - **<string>**: May be used with STRING data fields.

maxLength

- **Type**: validation
- **Description**: Defines the maximum length of input for STRING data fields.
- **Values**:
  - **<number>**: May be used with STRING data fields.

maxValue

- **Type**: validation
- **Description**: Defines the maximum value for number data fields.
- **Values**:
  - **<number>**: May be used with NUMBER data fields, i.e., NUMBER and FIXED-POINT NUMBER.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

minLength

- **Type**: validation
- **Description**: Defines the minimum length of input for STRING data fields.
- **Values**:
  - **<number>**: May be used with STRING data fields.

minValue

- **Type**: validation
- **Description**: Defines the minimum value for NUMBER data fields.
- **Values**:
  - **<number>**: May be used with NUMBER data fields, i.e., NUMBER and FIXED-POINT NUMBER.

required

- **Type**: validation
- **Description**: Indicates that this is a required field.
- **Values**:
  - *true / false*: Field is required if value is set to “true”. The user cannot save the ticket without having entered a value in a required field. In the Web Client, required fields are marked by a red asterisk.

H.1.1.17 visibility (type)

visibility configuration

- **Type**: visibility
- **Description**: Indicates the visibility of this field in history.
- **Values**:
  - *on every level*: Field is shown on every level of history.
  - *2nd level and 3rd level*: Field is shown only on the 2nd and the 3rd level of history.
  - *only 3rd level*: Field is shown only on the 3rd level of history.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

H.1.2 List of Group Annotations

This chapter describes the following group annotations grouped by annotation type:

group-visibility 899
dwh-no-history 899
reportable group 899
auto-open-group 899
show-contact-in-ticket-list 899
show-in-group-section 900
show-labels-in-edit 900
show-labels-in-view 900
show-tooltips 900
show-watermarks 900
no-history 900
resource-fields-group-mode 901
resource-custom-fields-group-mode 901
customer group exposure 901
unit is a contact 901

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

H.1.2.1 common (type)

group-visibility

- **Type**: common
- **Description**: Defines the default visibility of a data field group.
- **Values**:
  - *true / false*: The annotation can be overwritten at the field level.

H.1.2.2 dwh (type)

dwh-no-history

- **Type**: dwh
- **Description**: Indicates that all fields in the group will not be historicized in DWH
- **Values**:
  - *true / false*: Since version 6.10.2.0

reportable group

- **Type**: dwh
- **Description**: Indicates that all data fields belonging to this group are reportable and should be transferred to CMRF.
- **Values**:
  - *true / false*: A value has to be set. Annotation is active if value is set to “true”.

H.1.2.3 layout (type)

auto-open-group

- **Type**: layout
- **Description**: The group will be opened initially. You can enter more than one value as a comma- or semicolon-separated list.
- **Values**:
  - *ticket:create*: Group is opened initially when a new ticket is created.
  - *ticket:view*: Group is opened initially when a ticket page is opened.
  - *customer:create*: Group is opened initially when a new customer is created.
  - *customer:view*: Group is opened when the customer (contact or company) page is opened.
  - *resource:create*: Group is opened initially when a new resource is created.
  - *resource:view*: Group is opened initially when a resource page is opened.

show-contact-in-ticket-list

- **Type**: layout
- **Description**: Obsolete! Use page customization! accordionTicketList.mainCustomerDescriptionVisible={true, false}
- **Values**: obsolete

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

show-in-group-section

- **Type**: layout
- **Description**: Defines that a data field group is displayed in the Details section (as tab).
- **Values**:
  - *true / false*: Without this annotation the group is shown in the non-tabbed ticket, customer or resource section.

show-labels-in-edit

- **Type**: layout
- **Description**: Whether the data fields in this group should be displayed in edit mode with labels.
- **Values**:
  - *true / false*: Since version 6.9.4

show-labels-in-view

- **Type**: layout
- **Description**: Whether the data fields in this group should be displayed in view mode with labels.
- **Values**:
  - *true / false*: Since version 6.9.4

show-tooltips

- **Type**: layout
- **Description**: Whether the data fields in this group should be displayed with tooltips.
- **Values**:
  - *true / false*: Since version 6.9.4

show-watermarks

- **Type**: layout
- **Description**: Whether the data fields in this group should be displayed with watermarks.
- **Values**:
  - *true / false*: Since version 6.9.4

H.1.2.4 performance (type)

no-history

- **Type**: performance
- **Description**: Indicates that all data fields belonging to this group will not be historicized.
- **Values**:
  - *true / false*: Indicates that all data fields that belong to this group should not be historicized. Possible values are “true” if this annotation should be active or “false”, which is the same as removing the annotation. Use this annotation if you want to prevent storing history for all/many fields in a group. If you only want to prevent historization for a single/some field(s), use the annotation `no-history-field` at the field level.

In CM versions up to 6.10.2, the DWH transfer of a field history is also controlled by this annotation.
Starting with CM version 6.10.2, use the annotation `dwh-no-history` for this.

H.1.2.5 resource (type)

resource-fields-group-mode

- **Type**: resource
- **Description**: Controls the mode of a resource field group concerning editing via Web Client.
- **Values**:
  - *internal / external*: Possible values: internal, external. A resource field is not editable in the Web Client if value is “external”.
    Since 6.10.4.0
    Removed 6.10.5.0

resource-custom-fields-group-mode

- **Type**: resource
- **Description**: Controls the mode of a resource field group concerning editing via Web Client.
- **Values**:
  - *internal / external*: Possible values: internal, external. A resource field is not editable in the Web Client if value is “external”.
    Since 6.10.5.0

H.1.2.6 restapi (type)

customer group exposure

- **Type**: restapi
- **Description**: Indicates whether a data field group should be available to customers using the REST API, e.g. in CM/Track.
- **Values**:
  - *full* (default): The data field group is available for reading and writing.
  - *read*: The data field group is available for reading only.

H.1.2.7 ticket contact relation (type)

unit is a contact

- **Type**: ticket contact relation
- **Description**: deprecated
- **Values**:
  - *true / false*: Removed in version 6.9.0.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

H.2 System Properties

The following chapter provides detailed information about the system properties used in ConSol CM.

- Alphabetical List of System Properties
- List of System Properties by Module
- List of System Properties by Area

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

H.2.1 Alphabetical List of System Properties

This chapter describes the following properties:

access.token.signing.key 914
admin.email 914
admin.login 914
admin.tool.consumed.licences.check.interval 914
admin.tool.consumed.licences.pool.name 915
admin.tool.session.check.interval 915
archive.enabled 915
archive.uri 916
attachment.allowed.types 916
attachment.max.size 916
attachment.type.hints 917
attachment.upload.timeout 917
authentication.method 917
autocommit.cf.changes 917
autocomplete.enabled 918
automatic.booking.enabled 918
batch-commit-interval 918
big.task.minimum.size 919
cache-cluster-name 919
cache.logging.listener.enabled 919
calendar.csv.dateFormat 920
calendar.csv.separator 920
checkUserOnlineIntervalInSeconds 920
client.archive.access.token.validity.seconds 920
client.archive.enabled 921
client.archive.refresh.token.validity.seconds 921
client.archive.secret 921
client.was.access.token.validity.seconds 922
client.was.refresh.token.validity.seconds 922
client.was.secret 922
cluster.mode 922
cluster.unicast 923
cmas.dropSchemaBeforeSetup 923
cmoffice.enabled 923

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

cmoffice.oo.path.NUMBER 924
cmoffice.strict.versioning.enabled 924
cmoffice.websocket.port 924
comment.authors.disabled 924
commentRequiredForTicketCreation 925
communication.channel 925
config.data.version 925
config.import.global.transaction.enabled 926
connection.release.mode 926
contact.authentication.method 926
contact.inherit.permissions.only.to.own.customer.group 926
csrf.domain.allow.none 927
csrf.domain.white.list 927
csrf.domain.white.list 927
csrf.request.filter.enabled 928
csrf.request.filter.enabled 928
customfield.content.file.max.size 928
customizationVersion 928
dao.log.threshold.milliseconds 929
dao.log.username 929
data.directory 929
data.optimization 930
database.notification.enabled 930
database.notification.redelivery.delay.seconds 930
database.notification.redelivery.max.attempts 931
defaultAttachmentEntryClassName 931
defaultCommentClassName 931
defaultContentEntryClassName 931
defaultIncommingMailClassName 932
defaultNumberOfCustomFieldsColumns 932
defaultOutgoingMailClassName 932
delete.ticket.enabled 932
diffTrackingEnabled 933
diffTrackingEnabledForUnitAndResource 933
diff.tracking.disabled 933
disable.admin.task.auto.commit 934

domain.map.for.client.config.<DOMAIN_NAME> 934
dwh.administration.refresh.interval.seconds 934
dwh.mode 935
engineer.description.cache.enabled 935
engineer.description.mode 935
engineer.description.template.name 936
eviction.event.queue.size 936
eviction.lifeSpan 936
eviction.max.nodes 936
eviction.strategy 937
eviction.wakeup.interval 937
expert.mode 937
external.line.access.prefix 938
favoritesSizeLimit 938
fetchLock.interval 938
fetchSize.strategy 938
fetchSize.strategy.FetchSizeFixedStrategy.value 939
fetchSize.strategy.FetchSizePageBasedStrategy.limit 939
fetchSize.strategy.FetchSizeThresholdStrategy.value 939
filesystem.polling.threads.number 940
filesystem.polling.threads.shutdown.timeout.seconds 940
filesystem.polling.threads.watchdog.interval.seconds 940
filesystem.task.enabled 940
filesystem.task.interval.seconds 941
filesystem.task.polling.folder 941
filesystem.task.timeout.seconds 941
filesystem.task.transaction.timeout.seconds 941
forward.mails.to.representatives 942
globalSearchResultSizeLimit 942
heartbeat 942
helpFilePath 943
hibernate.dialect 943
hideTicketSubject 943
ignore-queues 944
index.attachment 944
index.history 944

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

index.status 944
index.task.worker.threads 945
index.version.current 945
index.version.newest 945
indexed.assets.per.thread.in.memory 946
indexed.engineers.per.thread.in.memory 946
indexed.resources.per.thread.in.memory 946
indexed.tickets.per.thread.in.memory 946
indexed.units.per.thread.in.memory 947
initialized 947
internal.line.access.prefix 947
is.cmrf.alive 948
java.naming.factory.initial 948
java.naming.factory.url.pkgs 948
java.naming.provider.url 949
jmx.full.exposure 949
jobExecutor.adminMail 949
jobExecutor.idleInterval 950
jobExecutor.idleInterval.seconds 950
jobExecutor.jobExecuteRetryNumber 950
jobExecutor.jobMaxRetries 950
jobExecutor.jobMaxRetriesReachedSubject 951
jobExecutor.lockingLimit 951
jobExecutor.lockTimeout.seconds 951
jobExecutor.mailFrom 952
jobExecutor.maxInactivityInterval.minutes 952
jobExecutor.threads 952
jobExecutor.timerRetryInterval 952
jobExecutor.timerRetryInterval.seconds 953
jobExecutor.txTimeout.seconds 953
kerberos.v5.enabled 953
kerberos.v5.username.regex 953
last.config.change 954
last.config.change.templates 954
last.ping.timestamp 954
last.success.live.timestamp 955

ldap.authentication 955
ldap.basedn 955
ldap.certificate.basedn 955
ldap.certificate.content.attribute 956
ldap.certificate.password 956
ldap.certificate.providerurl 956
ldap.certificate.searchattr 956
ldap.certificate.userdn 957
ldap.contact.name.basedn 957
ldap.contact.name.password 957
ldap.contact.name.providerurl 957
ldap.contact.name.searchattr 958
ldap.contact.name.userdn 958
ldap.initialcontextfactory 958
ldap.password 958
ldap.providerurl 959
ldap.searchattr 959
ldap.userdn 959
live.calculation.enabled 959
live.error 960
live.start 960
local.country.prefix 960
mail.attachments.validation.info.sender 961
mail.attachments.validation.info.subject 961
mail.db.archive 961
mail.encryption 961
mail.error.from.address 962
mail.error.to.address 962
mail.from 962
mail.notification.engineerChange 963
mail.notification.sender 963
mail.on.error 963
mail.redelivery.retry.count 963
mail.reply.to 964
mail.sender.address 964
mail.smtp.email 964

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

mail.smtp.email.password 965
mail.smtp.email.user 965
mail.smtp.envelopesender 965
mail.smtp.tls.enabled 965
mail.ticketname.pattern 966
mailbox.<NUMBER>.name 966
mailbox.1.connection.host 966
mailbox.1.connection.password 966
mailbox.1.connection.port 966
mailbox.1.connection.protocol 967
mailbox.1.connection.username 967
mailbox.2.connection.host 967
mailbox.2.connection.password 967
mailbox.2.connection.port 967
mailbox.2.connection.protocol 967
mailbox.2.connection.username 967
mailbox.default.connection.host 967
mailbox.default.connection.password 968
mailbox.default.connection.port 968
mailbox.default.connection.protocol 968
mailbox.default.connection.username 968
mailbox.default.session.mail.debug 969
mailbox.default.session.mail.mime.address.strict 969
mailbox.default.session.mail.<PROTOCOL>.connectiontimeout 969
mailbox.default.session.mail.<PROTOCOL>.fetchsize 970
mailbox.default.session.mail.<PROTOCOL>.partialfetch 970
mailbox.default.session.mail.<PROTOCOL>.timeout 970
mailbox.default.task.delete.read.messages 971
mailbox.default.task.enabled 971
mailbox.default.task.interval.seconds 971
mailbox.default.task.max.message.size 971
mailbox.default.task.max.messages.per.run 972
mailbox.default.task.timeout.seconds 972
mailbox.default.task.transaction.timeout.seconds 972
mailbox.polling.threads.mail.log.enabled 973
mailbox.polling.threads.number 973

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

mailbox.polling.unstorable.backup.folder 973
mailSender.executionInterval.seconds 973
mailSender.lockingLimit 974
mailSender.lockTimeout.seconds 974
mailSender.maxAttempts 974
mailSender.nodeId 975
mailSender.retryInterval.seconds 975
mailSender.threads.number 975
mailSender.txTimeout.seconds 975
mailTemplateAboveQuotedText 976
max.licences.perUser 976
maxSizePerPagemapInMegaBytes 976
metabase.secret.key 977
metabase.url 977
ml.evict.unused.after.hours 977
ml.learn.worker.threads 978
monitoring.engineer.login 978
monitoring.unit.login 978
nimh.enabled 978
notification.error.description 979
notification.error.from 979
notification.error.subject 979
notification.error.to 979
notification.finished_successfully.description 980
notification.finished_successfully.from 980
notification.finished_successfully.subject 980
notification.finished_successfully.to 980
notification.finished_unsuccessfully.description 981
notification.finished_unsuccessfully.from 981
notification.finished_unsuccessfully.subject 981
notification.finished_unsuccessfully.to 982
notification.host 982
notification.password 982
notification.port 982
notification.protocol 983
notification.tls.enabled 983

notification.username 983
notifications.enabled 983
number.of.tasks 984
outdated.lock.age 984
pagemapLockDurationInSeconds 984
password.reset.mail.from 985
policy.password.age 985
policy.password.pattern 985
policy.rotation.ratio 985
policy.track.username.case.sensitive 986
policy.username.case.sensitive 986
postActivityExecutionScriptName 986
queue.polling.threads.number 987
queue.polling.threads.shutdown.timeout.seconds 987
queue.polling.threads.watchdog.interval.seconds 987
queue.task.error.pause.seconds 988
queue.task.interval.seconds 988
queue.task.max.retries 988
queue.task.timeout.seconds 988
queue.task.transaction.timeout.seconds 989
queuesExcludedFromGS 989
recent.items.cleanup.cluster.node.id 989
recent.items.cleanup.interval.minutes 989
recent.items.max.per.engineer 990
recent.items.persistence.enabled 990
recoverable.exceptions 990
refreshTimeInCaseOfConcurrentRememberMeRequests 991
rememberMeLifetimeInMinutes 991
request.log.threshold.milliseconds 991
request.log.username 991
request.scope.transaction 992
resetCode.expirationPeriod 992
resource.replace.batchSize 992
resource.replace.timeout 993
scene 993
script.evict.unused.after.hours 993

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

script.logging.threshold.seconds 993
script.validation.interval.seconds 994
searchPageSize 994
searchPageSizeOptions 994
security.fields.customer.exposure.check.enabled 995
security.restrict.unit.access.to.own.data 995
serial.mods.tracking.enabled 995
server.instance.task.crash.period.seconds 996
server.instance.task.period.seconds 996
server.session.archive.reaper.interval 996
server.session.archive.timeout 996
server.session.heartbeat.enabled.WEB_CLIENT 997
server.session.heartbeat.mode 997
server.session.heartbeat.timeout.seconds 997
server.session.reaper.interval 998
server.session.timeout 998
serverPoolingInterval 999
show.error.stacktrace 999
skip-ticket 999
skip-ticket-history 999
skip-unit 1000
skip-unit-history 1000
skip.wfl.transfer.cleanup 1000
skip.wfl.transfer.translations.cleanup 1001
split.history 1001
start.groovy.task.enabled 1001
statistics.calendar 1001
statistics.client.group 1002
statistics.contact.role 1002
statistics.content.entry 1002
statistics.content.entry.class 1003
statistics.content.entry.history 1003
statistics.cube 1003
statistics.customer.definition 1004
statistics.engineer 1004
statistics.enum.group 1004

statistics.field.definition 1005
statistics.group.definition 1005
statistics.locale 1005
statistics.localized.property 1006
statistics.mla 1006
statistics.project 1006
statistics.queue 1007
statistics.resource 1007
statistics.resource.group 1007
statistics.resource.history 1008
statistics.resource.relation.definition 1008
statistics.resource.type 1008
statistics.ticket 1009
statistics.ticket.function 1009
statistics.ticket.history 1009
statistics.time.booking 1010
statistics.timestamp 1010
statistics.unit 1010
statistics.unit.history 1011
statistics.unit.relation.definition 1011
statistics.workflow 1011
strict.utf.bmp.enabled 1011
supportEmail 1012
synchronize.archive.timeout.minutes 1012
synchronize.master.address 1013
synchronize.master.security.token 1013
synchronize.master.security.user 1013
synchronize.master.timeout.minutes 1014
synchronize.megabits.per.second 1014
synchronize.sleep.millis 1014
task.execution.interval.seconds 1014
task.execution.node.id 1015
task.panel.refresh.interval.seconds 1015
themeOverlay 1015
ticket.delete.timeout 1016
ticket.from.incoming.message.accepted.links 1016

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

ticketListRefreshIntervalInSeconds 1016
ticketListSizeLimit 1017
tickets.delete.size 1017
time.buffer 1017
transaction.timeout.minutes 1017
ttl.days 1018
tx.read.only.mode.enabled 1018
tinymce.enabled 1018
unit.description.mode 1019
unit.replace.batchSize 1019
unit.replace.timeout 1019
unit.transfer.order 1019
unitIndexSearchResultSizeLimit 1020
unused.content.remover.cluster.node.id 1020
unused.content.remover.enabled 1020
unused.content.remover.polling.minutes 1021
unused.content.remover.ttl.minutes 1021
update.6.11.0.0.sleep 1021
update.6.11.0.0.timezone 1022
urlLogoutPath 1022
url.track 1022
url.webclient 1023
voCacheEnabled 1023
warmup.executor.enabled 1024
webSessionTimeoutInMinutes 1024
wfl.sticky.transfer.disabled 1025
wicketAjaxRequestHeaderFilterEnabled 1025
workflow.deploy.cache.eviction.disabled 1025
X-Frame-Options 1025

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

access.token.signing.key

- **Module**: cmas-auth-server
- **Description**: Determines the secret shared between the authorization server and all client applications using OAuth2, e.g. CM/Archive. The value of this property needs to match the value of *archive.oauth2.access.token.signing.key* in the configuration file of CM/Archive.
- **Type**: password
- **Restart required**: yes
- **System**: no
- **Optional**: no
- **Example value**: my_password
- **Since**: 6.11.2.0

admin.email

- **Module**: cmas-core-security
- **Description**: The email address of the ConSol CM administrator. The value which you entered during system setup is used initially.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Since**: 6.0

admin.login

- **Module**: cmas-core-security
- **Description**: The login name of the ConSol CM administrator. The value which you entered during system setup is used initially.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: admin
- **Since**: 6.0

admin.tool.consumed.licences.check.interval

- **Module**: cmas-app-admin-tool
- **Description**: Sets the interval (in seconds) to monitor the number of consumed licenses.
- **Type**: integer
- **Restart required**: no
- **System**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Optional**: yes
- **Example value**: 30 (default value)
- **Since**: 6.11.0.0

admin.tool.consumed.licences.pool.name

- **Module**: cmas-app-admin-tool
- **Description**: Sets the license pool name to monitor the number of consumed licenses.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: CONCURRENT_USERS (default value)
- **Since**: 6.11.0.0

admin.tool.session.check.interval

- **Module**: cmas-app-admin-tool
- **Description**: Configures the time interval (in seconds) in which the system checks for inactive (ended) Admin Tool sessions.
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: 30
- **Since**: 6.7.5

archive.enabled

- **Module**: cmas-archive-core-server
- **Description**: Determines if GDPR-compliant deletions or anonymizations, which delete ticket or contact data from the ConSol CM database, are applied to CM/Archive as well. By default, this property is set to “false”, meaning that no data is deleted from CM/Archive. If you set this property to “true”, GDPR-compliant deletions or anonymizations are also performed in CM/Archive. This affects both already archived tickets and newly archived tickets.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.11.2.5

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

archive.uri

- **Module**: cmas-archive-core-server
- **Description**: Sets the URL where the CM/Archive application is running.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: <server_url>:8090
- **Since**: 6.11.2.0

attachment.allowed.types

- **Module**: cmas-core-server
- **Description**: Comma-separated list of allowed file name extensions. If the property does not exist, all file extensions are allowed. If the property exists, but no value is provided, all file extension are blocked.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: txt,zip,doc
- **Since**: 6.5.0

attachment.max.size

- **Module**: cmas-core-server
- **Description**: Maximum attachment size, in MB. This property controls the size of attachments at tickets, customers, and resources. It also controls the size of incoming (not outgoing) email attachments. The value of this property needs to be aligned with the respective setting in the application server configuration (add the parameter `max-post-size` to the `http listener` in `cm6.xml`, `cm6-cmrf.xml` or `domain.xml`). In addition you might need to increase the value of `cmas-nimh`, `mailbox.default.task.max.message.size`. The value of this property is displayed in the Web Client when adding a new attachment.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 100 (default value)
- **Since**: 6.4.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

attachment.type.hints

- **Module**: cmas-core-server
- **Description**: Allows you to assign MIME types to not yet officially supported file extensions, so these file extensions can be detected correctly.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 7z:application/x-7z-compressed, zip:application/zip, wav:audio/x-wav
- **Since**: 6.11.2.0

attachment.upload.timeout

- **Module**: cmweb-server-adapter
- **Description**: Defines the transaction timeout in minutes for adding attachments to a ticket, resource or customer. Counts the time for the upload of all attachments of one transaction. When the timeout occurs, all files which have been temporarily stored on the server are deleted and no file is uploaded.
- **Type**: Integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 3
- **Since**: 6.10.5.3

authentication.method

- **Module**: cmas-core-security
- **Description**: Sets the user authentication method (internal database or LDAP authentication). Allowed values are LDAP or DATABASE.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: DATABASE
- **Since**: 6.0

autocommit.cf.changes

- **Module**: cmas-dwh-server
- **Description**: Defines whether DWH tasks which result from configurational changes on ticket fields are executed automatically without manual interaction in the Admin Tool. Can be also set in the Admin Tool in the navigation item *DWH*. The default and recommended value is “false”.

- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false (default value)
- **Since**: 6.7.0

autocomplete.enabled

- **Module**: cmas-app-admin-tool
- **Description**: Controls if the *Autocomplete address* navigation item is displayed in the Admin Tool. It is hidden, if this system property is missing or its value is “false”.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: true
- **Since**: 6.9.2.0

automatic.booking.enabled

- **Module**: cmweb-server-adapter
- **Description**: If enabled, time spent on creating comments or emails is measured and automatic time bookings are added.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: true
- **Since**: 6.9.4.2

batch-commit-interval

- **Module**: cmas-dwh-server
- **Description**: Number of objects sent in one DWH message. Larger values mean better transfer performance at the cost of higher memory usage.
  Starting with ConSol CM version 6.11, this property is only used if the package size of a DWH operation is not set. This can only happen when the command is directly addressed to the Java MBean `consol.cmas.global.dwh.synchronizationService`, e.g. using the `update()` method. When a DWH operation is started using the Admin Tool, there is always a value for the package size. If not explicitly set, the default value of 1000 is used as value for `batch.commit.interval`.
- **Default value**: 1000
- **Type**: integer

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 100
- **Since**: 6.0.0

big.task.minimum.size

- **Module**: cmas-core-index-common
- **Description**: Indicates the minimum size of index task (in parts, each part has 100 entities) to qualify this task as a big one. Big tasks have a lower priority than runtime tasks.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 15 (default value)
- **Since**: 6.8.3

cache-cluster-name

- **Module**: cmas-core-cache
- **Description**: Unique name of a cluster consisting of several CM nodes. This value is used to identify the nodes belonging to the same cluster during network communication. Please do not change this value manually unless advised by ConSol.
- **Type**: string
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: 635a6de1-629a-4129-8299-2d98633310f0
- **Since**: 6.4.0

cache.logging.listener.enabled

- **Module**: cmas-core-cache
- **Description**: Determines whether cache events are logged to `infinispan.log` for the standalone mode and `jgroups-infinispan.log` for the cluster mode.
- **Type**: boolean
- **Restart required**: yes
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.12.0.5

calendar.csv.dateFormat

- **Module**: cmas-core-server
- **Description**: Format of the date given in the CSV file containing the list of holidays.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: dd/MM/yyyy
- **Since**: 6.9.3.2

calendar.csv.separator

- **Module**: cmas-core-server
- **Description**: Separator used in the CSV file containing the list of holidays.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: ,
- **Since**: 6.9.3.2

checkUserOnlineIntervalInSeconds

- **Module**: cmweb-server-adapter
- **Description**: The interval in seconds to check which users are online (default 180sec = 3min).
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 180
- **Since**: 6.0
- **Removed in**: 6.5 / 6.11.0.1

client.archive.access.token.validity.seconds

- **Module**: cmas-auth-server
- **Description**: Determines the validity period in seconds of the access token required for using CM/Archive.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: 43200 (default value)
- **Since**: 6.11.2.0

client.archive.enabled

- **Module**: cmas-auth-server
- **Description**: Determines if CM/Archive is enabled (“true”) or disabled (“false”).
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: false
- **Since**: 6.11.2.0
- **Removed in**: 6.11.2.2

client.archive.refresh.token.validity.seconds

- **Module**: cmas-auth-server
- **Description**: Determines the validity period in seconds of the refresh token required for obtaining new access tokens for CM/Archive.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 2592000 (default value)
- **Since**: 6.11.2.0

client.archive.secret

- **Module**: cmas-auth-server
- **Description**: Determines the secret shared between the authorization server and CM/Archive. The value of the property needs to match the value of *archive.oauth2.client.secret* in the configuration file of CM/Archive.
- **Type**: password
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: my_password
- **Since**: 6.11.2.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

client.was.access.token.validity.seconds

- **Module**: cmas-auth-server
- **Description**: Determines the validity period in seconds of the access token required for using the Web Admin Suite.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 43200 (default value)
- **Since**: 6.12.0.0

client.was.refresh.token.validity.seconds

- **Module**: cmas-auth-server
- **Description**: Determines the validity period in seconds of the refresh token required for obtaining new access tokens for the Web Admin Suite.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 2592000 (default value)
- **Since**: 6.12.0.0

client.was.secret

- **Module**: cmas-auth-server
- **Description**: Determines the secret shared between the authorization server and the Web Admin Suite.
- **Type**: password
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: my_password
- **Since**: 6.12.0.0
- **Removed in**: 6.12.0.1

cluster.mode

- **Module**: cmas-core-shared
- **Description**: Specifies whether ConSol CM is running in a cluster.
- **Type**: boolean
- **Restart required**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.1.0

cluster.unicast

- **Module**: cmas-core-shared
- **Description**: Flag to activate the jgroups unicast mode for ConSol CM clusters (as opposed to the default multicast mode causing problems in some data center environments). If set to "true", remember to set the JVM start parameters: `jgroups.bind.port`, `jgroups.bind.address` and `jgroups.initial_hosts`.
- **Type**: boolean
- **Restart required**: yes
- **System**: yes
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.11.0.0
- **Removed in**: 6.12.0.0

cmas.dropSchemaBeforeSetup

- **Module**: cmas-setup-hibernate
- **Description**: Determines if an existing database schema should should be dropped during setup. This value is set on the setup page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.0

cmoffice.enabled

- **Module**: cmweb-server-adapter
- **Description**: Determines if CM/Doc is enabled.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.4.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

cmoffice.oo.path.NUMBER

- **Module**: cmweb-server-adapter
- **Description**: Possible location of the OpenOffice installation. The properties are numbered starting with 0.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: c:\Program Files (x86)\LibreOffice 3.6\program
- **Since**: 6.10.1.0
- **Removed in**: 6.12.0.0

cmoffice.strict.versioning.enabled

- **Module**: cmweb-server-adapter
- **Description**: Controls if the SAVE operation in Microsoft Word / OpenOffice documents creates a new attachment (“true”) or overwrites the existing attachment (“false”). This concerns the behavior within one session using the text editing program. If the program is stopped, the overwrite mechanism will not work anymore.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true
- **Since**: 6.10.5.4

cmoffice.websocket.port

- **Module**: cmweb-server-adapter
- **Description**: Determines the port on which the CM/Doc application should be started and connected to.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 30333
- **Since**: 6.12.0.0

comment.authors.disabled

- **Module**: cmas-restapi-core
- **Description**: Disables the display of the content author via REST API.
- **Type**: boolean

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.11.0

commentRequiredForTicketCreation

- **Module**: cmweb-server-adapter
- **Description**: Determines if the comment is a required field for ticket creation.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true (default value)
- **Since**: 6.2.0

communication.channel

- **Module**: cmas-dwh-server
- **Description**: Communication channel. Only possible value since CM version 6.11.0.0: DIRECT
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: DIRECT
- **Since**: 6.8.5.0
- **Removed in**: 6.11.0.0 (DIRECT mode is the only available mode and is set automatically)

config.data.version

- **Module**: cmas-core-server
- **Description**: The internal version number of the current system configuration. This property is maintained internally, please do not change it unless advised by ConSol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 11
- **Since**: 6.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

config.import.global.transaction.enabled

- **Module**: cmas-core-server
- **Description**: Determines whether the configuration (without localizations) should be imported within a single transaction.
- **Type**: Boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true
- **Since**: 6.11.1.0

connection.release.mode

- **Module**: cmas-setup-hibernate
- **Description**: Describes the JEE connection handling strategy for transactions. If set to “AFTER_TRANSACTION”, the connection will be cached during the transaction and released at the end. If set to “AFTER_STATEMENT”, the connection will be released to the pool after each statement execution. Please do not change the default here unless advised by ConSol.
- **Type**: string
- **Restart required**: yes
- **System**: no
- **Optional**: yes
- **Example value**: AFTER_STATEMENT (default for JEE environment)
- **Since**: 6.0

contact.authentication.method

- **Module**: cmas-core-security
- **Description**: Sets the authentication method for contacts accessing REST services, possible values are DATABASE or LDAP or LDAP,DATABASE or DATABASE,LDAP.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: LDAP
- **Since**: 6.9.3.0

contact.inherit.permissions.only.to.own.customer.group

- **Module**: cmas-core-security
- **Description**: Determines whether an authenticated contact via REST inherits all customer group permissions from the CM/Track user profile (“false”) or only has permissions to his own customer group (“true”).

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.9.2.3

csrf.domain.allow.none

- **Module**: cmas-restapi-core
- **Description**: Determines whether empty `Origin/Referer` headers are accepted. By default, the property is set to “false”, so that REST requests without `Origin/Referer` headers are blocked.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false (default value)
- **Since**: 6.11.2.2

csrf.domain.white.list

- **Module**: cmas-restapi-core
- **Description**: The list of domains (format protocol://domain:port, separated with “|”) which are allowed in `Origin/Referer` headers and will not be blocked by the CSRF filter. By default, the property is empty so that cross-site requests are blocked.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: https://example.com:80 | http://www.consol.de:8080
- **Since**: 6.11.2.2

csrf.domain.white.list

- **Module**: cmweb-server-adapter
- **Description**: The list of domains (separated with “|”) which are allowed and will not be blocked by the CSRF (cross-site request forgery) filter, e.g.: “example.com | consol.de”
- **Type**: String
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: example.com | consol.de
- **Since**: 6.10.7.0

csrf.request.filter.enabled

- **Module**: cmas-restapi-core
- **Description**: Determines whether the CSRF request filter (Origin/Referer) for the REST API is enabled.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.11.2.2

csrf.request.filter.enabled

- **Module**: cmweb-server-adapter
- **Description**: It allows to disable the CSRF request filter, which is enabled by default.
- **Type**: Boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true
- **Since**: 6.10.7.0

customfield.content.file.max.size

- **Module**: cmas-core-server
- **Description**: Determines the maximum size in MB of the images which can be added as content of rich text fields.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 10
- **Since**: 6.11.2.2

customizationVersion

- **Module**: cmweb-server-adapter
- **Description**: UID representing the latest web customization version. Used only internally, please do not change the value.
- **Type**: string
- **Restart required**: no
- **System**: yes

- **Optional**: no
- **Example value**: cd58453e-f3cc-4538-8030-d15e8796a4a7
- **Since**: 6.5.0

dao.log.threshold.milliseconds

- **Module**: cmas-core-server
- **Description**: Used to configure database operation times logging. DAO methods whose execution takes longer than the time set in this property (in milliseconds) are logged.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 500 (default value)
- **Since**: 6.11.1.0

dao.log.username

- **Module**: cmas-core-server
- **Description**: Used to configure database operation times logging. The execution of DAO methods which are related to the user name stated in this property is logged. Only one user name can be provided. The value is empty by default.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: webadmin
- **Since**: 6.11.1.0

data.directory

- **Module**: cmas-core-shared
- **Description**: Global directory for ConSol CM data (e.g., index).
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: C:\Users\user\cmas
- **Since**: 6.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

data.optimization

- **Module**: cmweb-server-adapter
- **Description**: Defines optimization to be applied on HTML response data. The following values are supported (for setting more than one value, separate values by '|'): MINIFICATION and COMPRESSION. MINIFICATION minifies HTML data by e.g. stripping whitespaces and comments. COMPRESSION applies gzip compression to HTTP response. (Note: If you are running in cluster mode and want to test different configurations in parallel, you can set different values for each cluster node by specifying the property `data.optimization.nodeId` to override the default property.)
- **Type**: string
- **Restart required**: COMPRESSION can be switched on/off without restart, MINIFICATION requires restart.
- **System**: yes
- **Optional**: yes
- **Example value**: MINIFICATION|COMPRESSION

database.notification.enabled

- **Module**: cmas-core-index-common
- **Description**: Indicates whether the database should be used as a notification channel for index updates instead of JMS. The default value is “true.”
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.8.4.7
- **Removed in**: 6.12.0.0

database.notification.redelivery.delay.seconds

- **Module**: cmas-core-index-common
- **Description**: If the database is used as a notification channel for index updates, this property indicates the delay for notification redelivery when an exception occurs.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 60
- **Since**: 6.8.4.7

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

database.notification.redelivery.max.attempts

- **Module**: cmas-core-index-common
- **Description**: If the database is used as a notification channel for index updates, this property indicates maximum redelivery attempts when an exception occurs.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 60
- **Since**: 6.8.4.7

defaultAttachmentEntryClassName

- **Module**: cmweb-server-adapter
- **Description**: The default text class used to classify an attachment if no other text class was set explicitly.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: DefaultTextElement
- **Since**: 6.9.2.0

defaultCommentClassName

- **Module**: cmas-core-server
- **Description**: Name of the default text class for comments.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: default_class
- **Since**: 6.3.0

defaultContentEntryClassName

- **Module**: cmweb-server-adapter
- **Description**: Name of the text class which is set as default for newly created content entries.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: default_class
- **Since**: 6.3.0

defaultIncommingMailClassName

- **Module**: cmas-core-server
- **Description**: Name of the default text class for incoming emails.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: default_class
- **Since**: 6.3.0

defaultNumberOfCustomFieldsColumns

- **Module**: cmweb-server-adapter
- **Description**: Determines the default number of columns for ticket fields. The value is only used if the ticket fields are not positioned via annotation.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 3
- **Since**: 6.2.0

defaultOutgoingMailClassName

- **Module**: cmas-core-server
- **Description**: Name of the default text class for outgoing emails.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: default_class
- **Since**: 6.3.0

delete.ticket.enabled

- **Module**: cmas-app-admin-tool
- **Description**: Controls if tickets can be deleted in the ticket administration of the Admin Tool. The default value is “true”.
- **Type**: boolean
- **Restart required**: no

- **System**: yes
- **Optional**: yes
- **Example value**: true
- **Since**: 6.9.4.0

diffTrackingEnabled

- **Module**: cmweb-server-adapter
- **Description**: Defines if parallel editing of a ticket by different engineers should be possible.
  "false": Previous way of handling changes when editing a ticket. If the ticket has been changed in the meantime, the current engineer will not be able to submit his changes without being forced to reload the page before submitting.
  "true": New changes handling mode. If the ticket has been changed, this will not block the submission of other changes anymore. If the part of the ticket that was changed was exactly the part that is changed by the submitting engineer, then an information message will be displayed, but the ticket change will be persisted/stored anyway.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true (default)
- **Since**: 6.10.1
- **Removed in**: 6.11.0

diffTrackingEnabledForUnitAndResource

- **Module**: cmweb-server-adapter
- **Description**: Enables the prevention of concurrent modifications on customers / resources.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 3
- **Since**: 6.11.0.0

diff.tracking.disabled

- **Module**: cmas-restapi-core
- **Description**: Fallback property for disabling difference tracking for CM/Track, which is history-based so it can be heavy.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes

- **Example value**: false (default value)
- **Since**: 6.10.5.6

disable.admin.task.auto.commit

- **Module**: cmas-core-index-common
- **Description**: Toggle for the automatic execution of administrative index update tasks. If set to "false", possibly expensive index tasks created as a result of configuration changes will be automatically executed right after creation. It is recommended to set this value to "true", so that the execution of administrative index tasks needs to be manually triggered in the Admin Tool or Web Admin Suite.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.6.1

domain.map.for.client.config.<DOMAIN_NAME>

- **Module**: cmas-restapi-core
- **Description**: Enables mapping multiple instances of CM/Track to a specific domain. DOMAIN_NAME is the name of the client configuration (if your CM/Track configuration is called "track-V2customized", the name of the property is `domain.map.for.client.config.trackV2customized`). You can provide several URLs separated by a comma.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: https://www.consol.de, https://www.consol.com
- **Since**: 6.10.7.0

dwh.administration.refresh.interval.seconds

- **Module**: cmas-app-admin-tool
- **Description**: Internal DWH property, not to be changed manually. This property is used to set the interval for refreshing the list of actions in the Admin Tool, section *Data Warehouse -> Administration -> Actions*.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: 10
- **Since**: 6.11.0.1

dwh.mode

- **Module**: cmas-dwh-server
- **Description**: Currently active mode for the DWH data transfer. Possible values are OFF, ADMIN, LIVE.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: OFF
- **Since**: 6.0.1

engineer.description.cache.enabled

- **Module**: cmas-core-server
- **Description**: Defines whether user descriptions are cached. The default value is “true”, please do not change it unless advised by ConSol.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.11.0

engineer.description.mode

- **Module**: cmas-core-server
- **Description**: Defines whether the user names in the ticket history are taken from the database or dynamically rendered using templates. The default value “DYNAMIC” is a bit more costly from the performance perspective, while “PROTOCOL” is faster but returns historical names which might be outdated. Use “PROTOCOL” if you have lots of history entries from many different users.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: PROTOCOL
- **Since**: 6.11.0

engineer.description.template.name

- **Module**: cmas-core-server
- **Description**: Defines the name of the template which is used to render engineer names for display in the Web Client. The template has to be stored in the *Templates* section of the Admin Tool / Web Admin Suite.
- **Type**: String
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: engineer description template name (default value)
- **Since**: 6.11.0

eviction.event.queue.size

- **Module**: cmas-core-cache
- **Description**: The size of the queue holding cache events. The default value is 200000. It is recommended to increase the value slightly (up to 400000) on systems with high traffic or load. [Deprecated - not used for infinispan]
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: 200000 (default value)
- **Since**: 6.4.0

eviction.lifeSpan

- **Module**: cmas-core-cache
- **Description**: Sets the interval (in milliseconds) for cache entry validity. When this time is elapsed, the entry is removed from the cache.
- **Type**: integer
- **Restart required**: yes
- **System**: no
- **Optional**: yes
- **Example value**: 86400000 (default value)
- **Since**: 6.11.1.0

eviction.max.nodes

- **Module**: cmas-core-cache
- **Description**: Sets the maximum size of internal caches. The default value is 100000. Increasing it will lead to higher memory consumption and is not recommended unless explicitly advised by ConSol.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: 100000 (default value)
- **Since**: 6.4.0

eviction.strategy

- **Module**: cmas-core-cache
- **Description**: Determines the cache eviction strategy. Please see the Infinispan documentation for details.
- **Type**: string
- **Restart required**: yes
- **System**: no
- **Optional**: yes
- **Example value**: LRU, LIRS (default value)
- **Since**: 6.11.1.0

eviction.wakeup.interval

- **Module**: cmas-core-cache
- **Description**: Sets the interval (in milliseconds) between two cache queue event processing cycles. The default value is 3000. It is recommended to decrease it (minimum is 1500) on systems with high traffic or load.
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: 3000 (default value)
- **Since**: 6.4.0

expert.mode

- **Module**: cmas-core-shared
- **Description**: Activates the expert mode in the Admin Tool to unlock extended features. For example, the system property `initialized` is only available in the expert mode.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.0

external.line.access.prefix

- **Module**: cmas-core-server
- **Description**: General prefix to dial before an area code. Set for each customer group separately.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 411
- **Since**: 6.9.3.0

favoritesSizeLimit

- **Module**: cmweb-server-adapter
- **Description**: Determines the maximum number of items in the favorites.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 10
- **Since**: 6.0

fetchLock.interval

- **Module**: cmas-workflow-jbpm
- **Description**:
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5000
- **Removed in**: 6.8.0

fetchSize.strategy

- **Module**: cmas-core-server
- **Description**: Strategy for selecting the fetch size on JDBC result sets.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: FetchSizePageBasedStrategy, FetchSizeThresholdStrategy, FetchSizeFixedStrategy
- **Since**: 6.8.4.1

fetchSize.strategy.FetchSizeFixedStrategy.value

- **Module**: cmas-core-server
- **Description**: Sets the fetch size value if the selected strategy to set the fetch size is `FetchSizeFixedStrategy`. Please do not change this value unless advised by ConSol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 150
- **Since**: 6.8.4.1

fetchSize.strategy.FetchSizePageBasedStrategy.limit

- **Module**: cmas-core-server
- **Description**: Sets the maximum fetch size value if the selected strategy to set the fetch size is `FetchSizePageBasedStrategy`. Please do not change this value unless advised by ConSol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 10000
- **Since**: 6.8.4.1

fetchSize.strategy.FetchSizeThresholdStrategy.value

- **Module**: cmas-core-server
- **Description**: Sets the fetch size threshold border values if the selected strategy to set the fetch size is `FetchSizeThresholdStrategy`. Please do not change this value unless advised by ConSol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 150,300,600,1000
- **Since**: 6.8.4.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

filesystem.polling.threads.number

- **Module**: cmas-nimh
- **Description**: Number of threads started for polling emails from the database queue. The default value is 1.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 10
- **Since**: 6.4.0

filesystem.polling.threads.shutdown.timeout.seconds

- **Module**: cmas-nimh
- **Description**: Waiting time after the shutdown signal. When the timeout is reached, the thread will be terminated.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.4.0

filesystem.polling.threads.watchdog.interval.seconds

- **Module**: cmas-nimh
- **Description**: Determines the interval in seconds to execute the watchdog which checks the activity of the threads of the file system poller.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 30 (default value)
- **Since**: 6.4.0

filesystem.task.enabled

- **Module**: cmas-nimh
- **Description**: Allows to disable the service thread related to a given poller.
- **Type**: boolean
- **Restart required**: no
- **System**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.4.0

filesystem.task.interval.seconds

- **Module**: cmas-nimh
- **Description**: Default interval (in seconds) for polling the file system.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.4.0

filesystem.task.polling.folder

- **Module**: cmas-nimh
- **Description**: Polling folder location which is scanned for emails in the format of `eml` files. The default value is the `mail` subdirectory of the ConSol CM data directory.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: c://cmas//mail
- **Since**: 6.4.0

filesystem.task.timeout.seconds

- **Module**: cmas-nimh
- **Description**: After this time of inactivity the service thread is considered damaged and automatically restarted. The default value is 120 seconds.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60
- **Since**: 6.4.0

filesystem.task.transaction.timeout.seconds

- **Module**: cmas-nimh
- **Description**: Default transaction timeout for transactions fetching emails from the file system. Should be correlated with number of messages fetched at once.

- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.4.0

forward.mails.to.representatives

- **Module**: cmweb-server-adapter
- **Description**: Determines if emails which are manually sent from the Web Client are also sent to representing engineers. The default value of the property is “false”, meaning that this kind of emails are not forwarded to the representing engineer. Set the property to “true” if you want to restore the previous behavior, i.e., all emails which are sent to the represented engineer are automatically forwarded to the representing engineer. Please take into account that this might not be desired if the same person is an engineer and a customer in the CM system.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: false (default value)
- **Since**: 6.11.1.7



This property only configures the handling of manually sent emails. The handling of automatically sent emails depends on the used Java method.

globalSearchResultSizeLimit

- **Module**: cmweb-server-adapter
- **Description**: Determines the maximum number of results in the quick search.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 10
- **Since**: 6.0

heartbeat

- **Module**: cmas-core-server
- **Description**: Timestamp that indicates if an instance of the application is connected to the database schema.
- **Type**: integer

- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1234567899
- **Since**: 6.10.5.3

helpFilePath

- **Module**: cmweb-server-adapter
- **Description**: URL for the online help. If not empty, the *Help* link is displayed in the Web Client.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: http://www.consol.de
- **Since**: 6.2.1

hibernate.dialect

- **Module**: cmas-setup-hibernate
- **Description**: The database dialect used by hibernate. Set during the initial setup depending on the database system.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: org.hibernate.dialect.MySQL5InnoDBDialect
- **Since**: 6.0

hideTicketSubject

- **Module**: cmweb-server-adapter
- **Description**: If set to “true”, the ticket subject is hidden in the Web Client.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.2.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

ignore-queues

- **Module**: cmas-dwh-server
- **Description**: A comma-separated list of queue names which are not not transferred to the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: QueueName1,QueueName2,QueueName3
- **Since**: 6.6.19
- **Removed in**: 6.8.1

index.attachment

- **Module**: cmas-core-index-common
- **Description**: Determines whether the content of attachments is indexed.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.4.3

index.history

- **Module**: cmas-core-index-common
- **Description**: Determines whether ticket and customer history is indexed.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.1.0
- **Removed in**: 6.11.0

index.status

- **Module**: cmas-core-index-common
- **Description**: Index synchronization status, possible values are GREEN, YELLOW or RED. This value is set internally by the index system, please do not change it manually unless advised by ConSol.
- **Type**: string

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: GREEN
- **Since**: 6.6.1

index.task.worker.threads

- **Module**: cmas-core-index-common
- **Description**: Sets the number of threads that will be used to execute index tasks. The default value is 1, we recommend to increase it slightly on systems with large indexer load up to a value of 4.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1 (default value)
- **Since**: 6.6.14, 6.7.3. Since 6.8.0 and exclusively in 6.6.21 also normal (live) index updates are affected by this property.

index.version.current

- **Module**: cmas-core-index-common
- **Description**: Holds information about the current index version. This value is maintained internally, please do not change it unless advised by ConSol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1 (default value)
- **Since**: 6.7.0

index.version.newest

- **Module**: cmas-core-index-common
- **Description**: Holds information about which index version is considered newest. This value is maintained internally, please do not change it unless advised by ConSol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1 (default value)
- **Since**: 6.7.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

indexed.assets.per.thread.in.memory

- **Module**: cmas-core-index-common
- **Description**: Determines how many assets should be loaded into memory at once, per thread, during indexing.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 200 (default value)
- **Since**: 6.8.0

indexed.engineers.per.thread.in.memory

- **Module**: cmas-core-index-common
- **Description**: Determines how many engineers should be loaded into memory at once, per thread, during indexing. Increasing it slightly will lead to better indexer performance at the price of higher memory consumption.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 300 (default value)
- **Since**: 6.6.14, 6.7.3

indexed.resources.per.thread.in.memory

- **Module**: cmas-core-index-common
- **Description**: Determines how many resources should be loaded into memory at once, per thread, during indexing. Increasing it slightly will lead to better indexer performance at the price of higher memory consumption.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 200 (default value)
- **Since**: 6.10.0.0

indexed.tickets.per.thread.in.memory

- **Module**: cmas-core-index-common
- **Description**: Determines how many tickets should be loaded into memory at once, per thread, during indexing. Increasing it slightly will lead to better indexer performance at the price of higher memory consumption.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 100 (default value)
- **Since**: 6.6.14, 6.7.3

indexed.units.per.thread.in.memory

- **Module**: cmas-core-index-common
- **Description**: Determines how many customers should be loaded into memory at once, per thread, during indexing. Increasing it slightly will lead to better indexer performance at the price of higher memory consumption.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 200 (default value)
- **Since**: 6.6.14, 6.7.3

initialized

- **Module**: cmas-setup-manager
- **Description**: Flag indicating if ConSol CM is initialized, i.e. the initial setup has been performed. If this value is missing or not “true”, the setup will be performed. Starting with ConSol CM version 6.11, this property is only available in expert mode.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.0

> ⚠ Be careful with using this property. When you set the value to “false”, the ConSol CM server will perform the system setup at the next start, i.e. all data of the existing system is lost, including system properties.

internal.line.access.prefix

- **Module**: cmas-core-server
- **Description**: Prefix that the company’s telephony system requires for outside lines. Set for each customer group separately.
- **Type**: integer

- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 199
- **Since**: 6.9.3.0

is.cmrf.alive

- **Module**: cmas-dwh-server
- **Description**: Defines a threshold timeframe (in seconds) for the check if CMRF is working properly. If no response from CMRF is received in the configured timeframe, the DWH operation status is updated to “error”.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1200
- **Since**: 6.7.0

java.naming.factory.initial

- **Module**: cmas-dwh-server
- **Description**: Factory class for the DWH context factory.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: org.jnp.interfaces.NamingContextFactory
- **Since**: 6.0.1
- **Removed in**: 6.11.0.0

java.naming.factory.url.pkgs

- **Module**: cmas-dwh-server
- **Description**:
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: org.jboss.naming:org.jnp.interfaces
- **Since**: 6.0.1
- **Removed in**: 6.11.0.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

java.naming.provider.url

- **Module**: cmas-dwh-server
- **Description**: URL of naming provider.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: localhost
- **Since**: 6.0.1
- **Removed in**: 6.11.0.0

jmx.full.exposure

- **Module**: cmas-core-server
- **Description**: Controls the availability of methods which can be used to retrieve or delete data from ConSol CM via JMX. The property is set to “false” by default, meaning that these methods cannot be used via JMX. If you set the property to “true”, these methods can be used via JMX to retrieve or delete data.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false (default value)
- **Since**: 6.11.2.5



It is strictly recommended to have this property set to “false” on production systems.

jobExecutor.adminMail

- **Module**: cmas-workflow-engine
- **Description**: Email address which will get notified about job execution problems (when retry counter is exceeded).
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: admin@consol.de
- **Since**: 6.8.0

jobExecutor.idleInterval

- **Module**: cmas-workflow-jbpm
- **Description**:
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 45000
- **Removed in**: 6.8.0
- **Replaced by**: jobExecutor.idleInterval.seconds

jobExecutor.idleInterval.seconds

- **Module**: cmas-workflow-engine
- **Description**: Determines how often (in seconds) the job executor thread will look for new jobs to execute.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 45 (default value up to CM version 6.10.5.2. The default value for CM versions 6.10.5.3 and up is 5)
- **Since**: 6.8.0

jobExecutor.jobExecuteRetryNumber

- **Module**: cmas-workflow-jbpm
- **Description**:
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5
- **Removed in**: 6.8.0
- **Replaced by**: jobExecutor.jobMaxRetries

jobExecutor.jobMaxRetries

- **Module**: cmas-workflow-engine
- **Description**: Controls the number of retry attempts the job executor will do before declaring a job as failed.
- **Type**: integer

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 5 (default value)
- **Since**: 6.8.0

jobExecutor.jobMaxRetriesReachedSubject

- **Module**: cmas-workflow-engine
- **Description**: The subject used in the notification email which administrators receive about failed job executions.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: Job maximum retries reached. Job was removed!!! (default value)
- **Since**: 6.8.0

jobExecutor.lockingLimit

- **Module**: cmas-workflow-engine
- **Description**: Maximum number of jobs which can locked at once (marked for execution) by the job executor thread.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 5 (default value since CM version 6.10.5.3)
- **Since**: 6.8.0

jobExecutor.lockTimeout.seconds

- **Module**: cmas-workflow-engine
- **Description**: Determines how long the job can be locked (marked for execution) by the job executor.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 360 (default value)
- **Since**: 6.8.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

jobExecutor.mailFrom

- **Module**: cmas-workflow-engine
- **Description**: Email address which will be set as From header for notifications to the administrator.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: jobexecutor@consol.de
- **Since**: 6.8.0

jobExecutor.maxInactivityInterval.minutes

- **Module**: cmas-workflow-engine
- **Description**: Maximum amount of time (in minutes) of allowed job executor inactivity (e.g. when it is blocked by long timer execution). After this time, executor threads are restarted automatically.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 30 (default value)
- **Since**: 6.9.2.0

jobExecutor.threads

- **Module**: cmas-workflow-engine
- **Description**: Sets the number of job execution threads.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 1 (default value)
- **Since**: 6.8.0

jobExecutor.timerRetryInterval

- **Module**: cmas-workflow-jbpm
- **Description**:
- **Type**: integer
- **Restart required**: no
- **System**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Optional**: no
- **Example value**: 10000
- **Removed in**: 6.8.0
- **Replaced by**: jobExecutor.timerRetryInterval.seconds

jobExecutor.timerRetryInterval.seconds

- **Module**: cmas-workflow-engine
- **Description**: Determines how long (in seconds) the job executor thread will wait before a retry after an error occurs during job execution.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 10 (default up to CM version 6.10.5.2. The default value for CM versions 6.10.5.3 and up is 30)
- **Since**: 6.8.0

jobExecutor.txTimeout.seconds

- **Module**: cmas-workflow-engine
- **Description**: Transaction timeout used for job execution.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.8.0

kerberos.v5.enabled

- **Module**: cmas-core-security
- **Description**: Indicates whether user authentication using SSO via Kerberos is enabled.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false (default value if Kerberos was not enabled during system setup)
- **Since**: 6.2.0

kerberos.v5.username.regex

- **Module**: cmas-core-security
- **Description**: Regular expression used for mapping Kerberos principals to CM user login names.

- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: (.*)@.*
- **Since**: 6.2.0

last.config.change

- **Module**: cmas-core-server
- **Description**: Random UUID created during the last configuration change. This is a value maintained internally, please do not change it unless advised by ConSol.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 2573c7b7-2bf5-47ff-b5a2-bad31951a266
- **Since**: 6.1.0, 6.2.1

last.config.change.templates

- **Module**: cmas-core-server
- **Description**: Random UUID created during the last change in templates. This is a value maintained internally, please do not change it unless advised by ConSol.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 2573c7c7-2af5-4eff-b9c2-bad31951a266
- **Since**: 6.10.5.0

last.ping.timestamp

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property shows the status of CMRF and is filled automatically by the CMRF. It contains the date of the last response on the ping from the ConSol CM server to the CMRF.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 32323214
- **Since**: 6.11.0.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

last.success.live.timestamp

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 13232762
- **Since**: 6.11.2.0

ldap.authentication

- **Module**: cmas-core-security
- **Description**: LDAP authentication method used when using LDAP authentication. Possible values are “anonymous” and “simple”.
- **Type**: string
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: simple (default value)
- **Since**: 6.0

ldap.basedn

- **Module**: cmas-core-security
- **Description**: Base DN used for looking up LDAP user accounts when using LDAP authentication.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: ou=accounts,dc=consol,dc=de
- **Since**: 6.0

ldap.certificate.basedn

- **Module**: cmas-core-server
- **Description**: Base DN for the certificate location in the LDAP tree. If not provided, the value of the property `cmas-core-security, ldap.basedn` is used.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes

- **Example value**: ou=accounts,dc=consol,dc=de
- **Since**: 6.8.4

ldap.certificate.content.attribute

- **Module**: cmas-core-server
- **Description**: Name of the LDAP attribute where the certificate data is stored in the LDAP tree.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: usercertificate (default value)
- **Since**: 6.8.4

ldap.certificate.password

- **Module**: cmas-core-server
- **Description**: Password of the LDAP certificate manager. If not set, the value of the property `cmas-core-security, ldap.password` is used.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.8.4

ldap.certificate.providerurl

- **Module**: cmas-core-server
- **Description**: URL of the LDAP certificate provider. If not set, the value of the property `cmas-core-security, ldap.providerurl` is used.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: ldap://ldap.consol.de:389
- **Since**: 6.8.4

ldap.certificate.searchattr

- **Module**: cmas-core-server
- **Description**: LDAP attribute name used to search for certificates in the LDAP tree.
- **Type**: string
- **Restart required**: no
- **System**: yes

- **Optional**: yes
- **Example value**: mail (default value)
- **Since**: 6.8.4

ldap.certificate.userdn

- **Module**: cmas-core-server
- **Description**: DN of the LDAP certificate manager. If not set, the value of the property `cmas-core-security, ldap.userdn` is used.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.8.4

ldap.contact.name.basedn

- **Module**: cmas-core-security
- **Description**: Base path to search for the contact DN by LDAP ID.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: ou=accounts,dc=consol,dc=de
- **Since**: 6.9.3.0

ldap.contact.name.password

- **Module**: cmas-core-security
- **Description**: Password to look up the contact DN by LDAP ID. If not set, the anonymous account is used.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Since**: 6.9.3.0

ldap.contact.name.providerurl

- **Module**: cmas-core-security
- **Description**: Address of the LDAP server (ldap[s]://host:port) used for contact authentication.
- **Type**: string
- **Restart required**: no
- **System**: no

- **Optional**: yes
- **Example value**: ldap://ldap.consol.de:389
- **Since**: 6.9.3.0

ldap.contact.name.searchattr

- **Module**: cmas-core-security
- **Description**: Attribute to search for the contact DN by LDAP ID (e.g. uid).
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: uid
- **Since**: 6.9.3.0

ldap.contact.name.userdn

- **Module**: cmas-core-security
- **Description**: User DN to look up contact DN by LDAP ID. If not set, the anonymous account is used.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Since**: 6.9.3.0

ldap.initialcontextfactory

- **Module**: cmas-core-security
- **Description**: Class name for the initial context factory of the LDAP implementation when using LDAP authentication. Usually this is `com.sun.jndi.ldap.LdapCtxFactory`.
- **Type**: string
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: com.sun.jndi.ldap.LdapCtxFactory
- **Since**: 6.0

ldap.password

- **Module**: cmas-core-security
- **Description**: Password for connecting to LDAP to look up users when using LDAP authentication. Only needed if lookup cannot be performed anonymously.
- **Type**: password

- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.1.2

ldap.providerurl

- **Module**: cmas-core-security
- **Description**: URL to LDAP provider, when using LDAP authentication.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: ldap://myserver.consol.de:389
- **Since**: 6.0

ldap.searchattr

- **Module**: cmas-core-security
- **Description**: Search attribute for looking up the LDAP entries associated with a CM login.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: uid
- **Since**: 6.0

ldap.userdn

- **Module**: cmas-core-security
- **Description**: LDAP user for connecting to LDAP to look up users when using LDAP authentication. Only needed if the lookup cannot be performed anonymously.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.1.2

live.calculation.enabled

- **Module**: cmas-dwh-server
- **Description**: Determines if the CMRF should perform an update for the date since the last successful processing of a live package or the last update, whatever happened later. Please do not change this property unless advised by ConSol.

- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: false
- **Since**: 6.11.2.0

live.error

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: false
- **Since**: 6.11.1.0

live.start

- **Module**: cmas-dwh-server
- **Description**: When the DWH synchronization mode is set to LIVE using the Admin Tool (navigation group *Data Warehouse*, navigation item *Administration*, *Configuration* button), this property is created and set to the current date.
  If LIVE mode is not enabled and there is no data in `cmas_dwh_ser_sync_object`, the property `live.start` is deleted.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes (automatically added in DWH “LIVE” mode)
- **Example value**: 15028802377645
- **Since**: 6.7.0

local.country.prefix

- **Module**: cmas-core-server
- **Description**: Prefix of the local country code. Set for each customer group separately.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 48
- **Since**: 6.9.3.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

mail.attachments.validation.info.sender

- **Module**: cmas-nimh-extension
- **Description**: Sets the From header for error notification emails which are sent when the validation of the attachment type fails.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: admin@mail.com
- **Since**: 6.7.5

mail.attachments.validation.info.subject

- **Module**: cmas-nimh-extension
- **Description**: Sets the subject for error notification emails which are sent when the validation of the attachment type fails.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: Email was not processed because its attachments were rejected!
- **Since**: 6.7.5

mail.db.archive

- **Module**: cmas-nimh-extension
- **Description**: If property is set to “true”, incoming emails are automatically archived in the database.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.8.5.5

mail.encryption

- **Module**: cmas-core-server
- **Description**: If the property is set to “true”, the encryption checkbox in the email editor of the Web Client is checked by default. The default value of the property is “false”.
- **Type**: boolean
- **Restart required**: no
- **System**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Optional**: no
- **Example value**: true
- **Since**: 6.8.4.0

mail.error.from.address

- **Module**: cmas-nimh-extension
- **Description**: Sets the From address for error notification emails which are sent when the email processing fails.
- **Type**: email
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Since**: 6.4.0

mail.error.to.address

- **Module**: cmas-nimh-extension
- **Description**: Sets the To address for error notification emails which are sent when the email processing fails. As a default the value of the property `admin.email` is used.
- **Type**: email
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Since**: 6.4.0

mail.from

- **Module**: cmweb-server-adapter
- **Description**: Central sender address to be used for email communication. The value set here appears as sender of emails sent from ConSol CM.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: consolcm@example.com
- **Since**: 6.1.2

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

mail.notification.engineerChange

- **Module**: cmas-core-server
- **Description**: Determines whether notification emails should be sent when the assignee of a ticket is changed.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.1.0

mail.notification.sender

- **Module**: cmas-core-server
- **Description**: From address for notification emails when the assignee of a ticket is changed. If not set, the value of the property `cmas-core-security, admin.email` is used instead.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: cm6notification@cm6installation
- **Since**: 6.6.3

mail.on.error

- **Module**: cmas-nimh-extension
- **Description**: If set to “true”, an error notification email is sent to the configured address if an email could not be processed.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default)
- **Since**: 6.4.0

mail.redelivery.retry.count

- **Module**: cmas-core-server
- **Description**: Number of redelivery attempts of an outgoing email.
- **Type**: integer
- **Restart required**: no
- **System**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Optional**: no
- **Example value**: 3
- **Since**: 6.1.0

mail.reply.to

- **Module**: cmweb-server-adapter
- **Description**: Central reply-to address to be used for email communication. The value set here controls the reply-to header of emails sent from ConSol CM.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: consolcm@example.com
- **Since**: 6.0.1

> ⚠ Please read the detailed information about ConSol CM Reply-To addresses in section Scripts of Type Email.

mail.sender.address

- **Module**: cmas-workflow-jbpm
- **Description**: From address for emails from the workflow engine.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Removed in**: 6.8.0
- **Replaced by**: jobExecutor.mailFrom

mail.smtp.email

- **Module**: cmas-core-server
- **Description**: SMTP URL for outgoing emails
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: smtp://mail.mydomain.com:25
- **Since**: 6.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

mail.smtp.email.password

- **Module**: cmas-core-server
- **Description**: SMTP password for outgoing emails. If this property is set, it takes precedence over the value from the SMTP URL.
- **Type**: password
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: password
- **Since**: 6.11.2.2

mail.smtp.email.user

- **Module**: cmas-core-server
- **Description**: SMTP user name for outgoing emails. If this property is set, it takes precedence over the value from the SMTP URL.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: myuser
- **Since**: 6.11.2.2

mail.smtp.envelopesender

- **Module**: cmas-core-server
- **Description**: Email address used as a sender in SMTP envelope. If not set, the From address of the email is used.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: mysender@mydomain.com
- **Since**: 6.5.7

mail.smtp.tls.enabled

- **Module**: cmas-core-server
- **Description**: Activates SMTP via SSL/TLS (SMTPS) for sending emails from the Web Client and scripts. The default value is “false”. If it is set to “true”, SMTPS is activated for sending emails.
- **Type**: boolean
- **Restart required**: yes
- **System**: no

- **Optional**: yes
- **Example value**: true
- **Since**: 6.11.1.6

mail.ticketname.pattern

- **Module**: cmas-nimh-extension
- **Description**: Regular expression pattern used to identify the ticket name in the subject of incoming mails.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: .*?Ticket\s+\((\S+)\).*
- **Since**: 6.4.0
- **Removed in**: 6.12.0.5

mailbox.<NUMBER>.name

- **Module**: cmas-nimh
- **Description**: Identifier (name) of the mailbox.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: myEmailAccount
- **Since**: 6.11.2.0

mailbox.1.connection.host

- **Module**: cmas-nimh
- **Description**: Host (server) for first configured mailbox. Will overwrite the default parameter `mailbox.default.connection.host`.

mailbox.1.connection.password

- **Module**: cmas-nimh
- **Description**: Password for first configured mailbox. Will overwrite the default parameter `mailbox.default.connection.password`.

mailbox.1.connection.port

- **Module**: cmas-nimh
- **Description**: Port for first configured mailbox. Will overwrite the default parameter `mailbox.default.connection.port`.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

mailbox.1.connection.protocol

- **Module**: cmas-nimh
- **Description**: Protocol (e.g., IMAP or POP3) for first configured mailbox. Will overwrite the default parameter `mailbox.default.connection.protocol`.

mailbox.1.connection.username

- **Module**: cmas-nimh
- **Description**: User name for first configured mailbox. Will overwrite the default parameter `mailbox.default.connection.username`.

mailbox.2.connection.host

- **Module**: cmas-nimh
- **Description**: Host (server) for second configured mailbox. Will overwrite the default parameter `mailbox.default.connection.host`.

mailbox.2.connection.password

- **Module**: cmas-nimh
- **Description**: Password for second configured mailbox. Will overwrite the default parameter `mailbox.default.connection.password`.

mailbox.2.connection.port

- **Module**: cmas-nimh
- **Description**: Port for second configured mailbox. Will overwrite the default parameter `mailbox.default.connection.port`.

mailbox.2.connection.protocol

- **Module**: cmas-nimh
- **Description**: Protocol (e.g., IMAP or POP3) for second configured mailbox. Will overwrite the default parameter `mailbox.default.connection.protocol`.

mailbox.2.connection.username

- **Module**: cmas-nimh
- **Description**: User name for second configured mailbox. Will overwrite the default parameter `mailbox.default.connection.username`.



For all NIMH-related mailbox properties, the following principle is used: a default property is defined (e.g. `mailbox.default.connection.port`). If no mailbox-specific value is configured, this default value will be used.

mailbox.default.connection.host

- **Module**: cmas-nimh
- **Description**: Host (server name) of a given mailbox which the poller reads emails from.
- **Type**: string

- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 10.10.1.157
- **Since**: 6.4.0

mailbox.default.connection.password

- **Module**: cmas-nimh
- **Description**: Password for given mailbox which the poller reads emails from.
- **Type**: password
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: consol
- **Since**: 6.4.0

mailbox.default.connection.port

- **Module**: cmas-nimh
- **Description**: Port of a given mailbox which the poller reads emails from.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 143
- **Since**: 6.4.0

mailbox.default.connection.protocol

- **Module**: cmas-nimh
- **Description**: The protocol used by the email poller, e.g. IMAP or POP3.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: imap
- **Since**: 6.4.0

mailbox.default.connection.username

- **Module**: cmas-nimh
- **Description**: User name for a given mailbox which the poller reads emails from.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: username
- **Since**: 6.4.0

mailbox.default.session.mail.debug

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. Allows for more detailed JavaMail session debugging.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true
- **Since**: 6.4.0

mailbox.default.session.mail.mime.address.strict

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. If this property is set to "false", the parsing of the email headers is less strict. Therefore, more emails with non-RFC-compliant headers will be imported into the system, instead of being stored in the email backups in the Admin Tool / Web Admin Suite.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.4.0

mailbox.default.session.mail.<PROTOCOL>.connectiontimeout

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. Determines the connection timeout in milliseconds.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes

- **Example value**: 3000
- **Since**: 6.4.0 (IMAP and POP3) / 6.11.2.0 (IMAPS and POP3S)

mailbox.default.session.mail.<PROTOCOL>.fetchsize

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. Determines the size of partial fetch in bytes for the indicated protocol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 1048576 (default value)
- **Since**: 6.9.4.0 (IMAP and POP3) / 6.11.2.0 (IMAPS and POP3S)

mailbox.default.session.mail.<PROTOCOL>.partialfetch

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. Determines whether the protocol's partial fetch capability should be used.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.9.4.0 (IMAP and POP3) / 6.11.2.0 (IMAPS and POP3S)

mailbox.default.session.mail.<PROTOCOL>.timeout

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. Determines the I/O timeout in milliseconds.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 120 for IMAP(S) and 3000 for POP3(S)
- **Since**: 6.4.0 (IMAP and POP3), 6.11.2.0 (IMAPS and POP3S)

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

mailbox.default.task.delete.read.messages

- **Module**: cmas-nimh
- **Description**: Defines whether messages should be removed from the mailbox after processing. For the IMAP protocol messages are marked as SEEN by default. For the POP3 protocol, this property is ignored and the emails are always deleted from the email server after they were processed successfully.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.4.0, changed in 6.12.1.0

mailbox.default.task.enabled

- **Module**: cmas-nimh
- **Description**: With this property, the service thread related to the given poller can be disabled.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.4.0

mailbox.default.task.interval.seconds

- **Module**: cmas-nimh
- **Description**: Default interval (in seconds) for polling mailboxes.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.4.0

mailbox.default.task.max.message.size

- **Module**: cmas-nimh
- **Description**: Maximum size of email messages (i.e., email plus attachment). Emails exceeding the size limit will not be automatically processed by NIMH but will be stored in the database (table `cmas_nimh_archived_mail`) and will therefore appear in the email backups in the Admin Tool / Web Admin Suite. From there they can be resent, downloaded to the file system, or deleted. For those operations the message size is not relevant.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 10485760 (default value, 10 MB)
- **Since**: 6.4.0

mailbox.default.task.max.messages.per.run

- **Module**: cmas-nimh
- **Description**: Number of messages fetched at once from the mailbox. Must be correlated with the transaction timeout. The default value is 20.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60
- **Since**: 6.4.0

mailbox.default.task.timeout.seconds

- **Module**: cmas-nimh
- **Description**: After this time of inactivity the service thread is considered damaged and automatically restarted. The default value is 120 seconds.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60
- **Since**: 6.4.0

mailbox.default.task.transaction.timeout.seconds

- **Module**: cmas-nimh
- **Description**: Default transaction timeout (in seconds) for email fetching transactions. Should be correlated with number of messages fetched at once.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.4.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

mailbox.polling.threads.mail.log.enabled

- **Module**: cmas-nimh
- **Description**: Enables email logging which is especially crucial in cluster environments (used as semaphore there).
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.9.4.1

mailbox.polling.threads.number

- **Module**: cmas-nimh
- **Description**: Number of threads for accessing mailboxes.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 1 (default value)
- **Since**: 6.4.0

mailbox.polling.unstorable.backup.folder

- **Module**: cmas-nimh
- **Description**: Folder on the file system which is used to store emails whose processing failed and which could not be stored in the database either. The value of the property should be the path to the data directory followed by `/mail/unstorable`.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: c:/cmas/mail/unstorable
- **Since**: 6.11.2.7

mailSender.executionInterval.seconds

- **Module**: cmas-core-server
- **Description**: Determines the period in seconds before the system checks if there are new emails to be sent.
- **Type**: integer
- **Restart required**: no
- **System**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Optional**: yes
- **Example value**: 10
- **Since**: 6.12.0.0

mailSender.lockingLimit

- **Module**: cmas-core-server
- **Description**: Determines the number of emails fetched from the database for sending in a single transaction.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 1
- **Since**: 6.12.0.0

mailSender.lockTimeout.seconds

- **Module**: cmas-core-server
- **Description**: Determines the period in seconds before a locked email is released for another lock.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 199
- **Since**: 6.12.0.0

mailSender.maxAttempts

- **Module**: cmas-core-server
- **Description**: Determines the maximum number of attempts, including the first attempt, before an email is removed.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 3
- **Since**: 6.12.0.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

mailSender.nodeId

- **Module**: cmas-core-server
- **Description**: Determines the ID of the node on which the thread for sending emails is started (only in clustered environments). If the value is “all” or no value is provided, the threads are started on every node.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: all
- **Since**: 6.12.0.0

mailSender.retryInterval.seconds

- **Module**: cmas-core-server
- **Description**: Determines the period in seconds before another attempt to send an email is made, if the previous attempt failed.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60
- **Since**: 6.12.0.0

mailSender.threads.number

- **Module**: cmas-core-server
- **Description**: Determines the number of threads for sending emails.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 1
- **Since**: 6.12.0.0

mailSender.txTimeout.seconds

- **Module**: cmas-core-server
- **Description**: Determines the transaction timeout for sending emails in seconds.
- **Type**: integer
- **Restart required**: no
- **System**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Optional**: yes
- **Example value**: 60
- **Since**: 6.12.0.0

mailTemplateAboveQuotedText

- **Module**: cmweb-server-adapter
- **Description**: Determines the behavior of the email template in the email editor when another email is quoted, i.e. forwarded or replied to. Often used to place the signature correctly.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.2.4

max.licences.perUser

- **Module**: cmas-core-server
- **Description**: Sets the maximum number of licenses which a single user can use (e.g., logging in from different browsers). By default this value is not restricted.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 10
- **Since**: 6.8.4.5

maxSizePerPagemapInMegaBytes

- **Module**: cmweb-server-adapter
- **Description**: The parameter defines the size (in MB) of the file which is created by the Wicket framework per user session. i.e. for each engineer which is currently logged in. The file is used to save pages during the running session. When the defined size limit has been reached and new entries are added, the oldest entries are removed. An engineer who works with an “old” page is redirected to the start page in the Web Client, when the page is removed from the file. This parameter can be increased if engineers work with a great number of open tabs and complain about being redirected to the start page. In large systems, you could use e.g. a value of 45 or 50. Since this is the size of the file which is saved on disk, the maximum value depends on the available disk space, however, a value which is too large is not recommended either.
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: no

- **Example value**: 15
- **Since**: 6.3.5

metabase.secret.key

- **Module**: cmas-analytics
- **Description**: Determines the string used to generate the private key used to validate Metabase JWT messages. Needs to match the setting *String used by the JWT signing key* in the Metabase configuration.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 50e2230c98344f183c996e6a3954c1a1e43ed100cc36f4ea093800aa2d8a1075
- **Since**: 6.13.0.0

metabase.url

- **Module**: cmas-analytics
- **Description**: Determines the URL where the Metabase application runs. Needs to match the setting *Site URL* in the Metabase configuration.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: http://localhost:3000 (default value)
- **Since**: 6.13.0.0

ml.evict.unused.after.hours

- **Module**: cmas-core-server
- **Description**: The number of hours after which a machine learning model is evicted from cache if it is not used. The ConSol CM server checks every hour for machine learning models to evict.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 24 (default value)
- **Since**: 6.13.0.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

ml.learn.worker.threads

- **Module**: cmas-core-server
- **Description**: The number of threads that a machine learning algorithm can use while training its machine learning model.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1 (default value)
- **Since**: 6.13.0.0

monitoring.engineer.login

- **Module**: cmas-core-server
- **Description**: Login for the special user (engineer) used for monitoring.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: nagios
- **Since**: 6.9.3.0

monitoring.unit.login

- **Module**: cmas-core-server
- **Description**: Login for the special user (contact) used for monitoring.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: nagios
- **Since**: 6.9.3.0

nimh.enabled

- **Module**: cmas-core-server
- **Description**: Enables the NIMH service which controls email polling and processing in ConSol CM. Must be suffixed with NODEID in cluster environments e.g. `nimh.enabled.NODEID` = “true”.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: false
- **Since**: 6.9.4.0

notification.error.description

- **Module**: cmas-dwh-server
- **Description**: Defines the text for error emails from the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: Error occurred
- **Since**: 6.0.1

notification.error.from

- **Module**: cmas-dwh-server
- **Description**: Defines the From address for error emails from the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: myuser@consol.de
- **Since**: 6.0.1

notification.error.subject

- **Module**: cmas-dwh-server
- **Description**: Defines the email subject for error emails from the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: Error occurred
- **Since**: 6.0.1

notification.error.to

- **Module**: cmas-dwh-server
- **Description**: Defines the To address for error emails from the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Optional**: no
- **Example value**: myuser@consol.de
- **Since**: 6.0.1

notification.finished_successfully.description

- **Module**: cmas-dwh-server
- **Description**: Defines the text for emails from the DWH when a transfer finishes successfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: Transfer finished successfully
- **Since**: 6.0.1

notification.finished_successfully.from

- **Module**: cmas-dwh-server
- **Description**: Defines the From address for emails from the DWH when a transfer finishes successfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: myuser@consol.de
- **Since**: 6.0.1

notification.finished_successfully.subject

- **Module**: cmas-dwh-server
- **Description**: Defines the subject for emails from the DWH when a transfer finishes successfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: Transfer finished successfully
- **Since**: 6.0.1

notification.finished_successfully.to

- **Module**: cmas-dwh-server
- **Description**: Defines the To address for emails from the DWH when a transfer finishes successfully.
- **Type**: string

- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Since**: 6.0.1

notification.finished_unsuccessfully.description

- **Module**: cmas-dwh-server
- **Description**: Defines the text for emails from the DWH when a transfer finishes unsuccessfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: Transfer finished unsuccessfully
- **Since**: 6.0.1

notification.finished_unsuccessfully.from

- **Module**: cmas-dwh-server
- **Description**: Defines the From address for emails from the DWH when a transfer finishes unsuccessfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: myuser@consol.de
- **Since**: 6.0.1

notification.finished_unsuccessfully.subject

- **Module**: cmas-dwh-server
- **Description**: Defines the subject for emails from the DWH when a transfer finishes unsuccessfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: Transfer finished unsuccessfully
- **Since**: 6.0.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

notification.finished_unsuccessfully.to

- **Module**: cmas-dwh-server
- **Description**: Defines the To address for emails from the DWH when a transfer finishes unsuccessfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Since**: 6.0.1

notification.host

- **Module**: cmas-dwh-server
- **Description**: Email (SMTP) server hostname for sending DWH emails.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: myserver.consol.de
- **Since**: 6.0.1

notification.password

- **Module**: cmas-dwh-server
- **Description**: SMTP password for sending DWH emails (optional).
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.0.1

notification.port

- **Module**: cmas-dwh-server
- **Description**: SMTP port for sending DWH emails.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 25
- **Since**: 6.0.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

notification.protocol

- **Module**: cmas-dwh-server
- **Description**: The protocol used for sending emails from the DWH. Possible values are smtp or pop3.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: pop3

notification.tls.enabled

- **Module**: cmas-dwh-server
- **Description**: Enables SMTP via SSL/TLS (SMTPS) for sending notification emails from the DWH. The default value is “false”. If it is set to “true”, SMTPS is enabled for sending notifications from the DWH.
- **Type**: string
- **Restart required**: yes
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.11.1.6

notification.username

- **Module**: cmas-dwh-server
- **Description**: (SMTP) user name for sending DWH emails.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: myuser
- **Since**: 6.0.1

notifications.enabled

- **Module**: cmas-web-notifications
- **Description**: Determines whether the notifications feature is enabled (“true”) or disabled (“false”). The default value is “true”. If you use the notifications feature on a ConSol CM system behind a proxy, some special proxy settings are needed. Please see section *Operating ConSol CM Behind a Proxy (e.g., Apache HTTPD)* of the *ConSol CM Setup Manual* for further information.
- **Type**: boolean

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: true
- **Since**: 6.12.0.0

number.of.tasks

- **Module**: cmas-core-server
- **Description**: Number of threads to use by the Task Execution Framework (TEF).
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 1
- **Since**: 6.9.4.0

outdated.lock.age

- **Module**: cmas-workflow-jbpm
- **Description**:
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 60000
- **Removed in**: 6.8.0
- **Replaced by**: cmas-workflow-engine, jobExecutor.lockTimeout.seconds

pagemapLockDurationInSeconds

- **Module**: cmweb-server-adapter
- **Description**: Number of seconds to pass before a pagemap is considered to be locked for too long.
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: yes
- **Example value**: 60
- **Since**: 6.7.3

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

password.reset.mail.from

- **Module**: cmas-core-security
- **Description**: Defines the from address (sender) of emails for resetting the password. If not set, the value of the `admin.email` property is used instead.
- **Type**: String
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: mypwreset@consol.de
- **Since**: 6.11.0.1

policy.password.age

- **Module**: cmas-core-security
- **Description**: Defines the maximum validity period (in days) of passwords before the user needs to change the password. Example “183” (6 months), default value: “5500” (= 15 years, i.e., no password change enforced).
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 5500 (15 years, default value)
- **Since**: 6.10.1.0

policy.password.pattern

- **Module**: cmas-core-security
- **Description**: Defines a regular expression pattern which new passwords have to match. The default value is ^.{3,}$ (at least 3 characters).
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: ^.{3,}$ (default value)
- **Since**: 6.10.1.0

policy.rotation.ratio

- **Module**: cmas-core-security
- **Description**: Defines how often passwords may repeat. For example, setting the value to X means that the new password cannot be present among the user’s X previous passwords.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

> Since 6.11.2.8, this property is also taken into account for CM/Track. Nevertheless, any value greater than 1 is interpreted as 1, so that it is only possible to compare the new password to the current one.

> You can set the property to 0 to disable password rotation checks in CM/Track.

- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 1 (default value)
- **Since**: 6.10.1.0, changed in 6.11.2.8

policy.track.username.case.sensitive

- **Module**: cmas-core-security
- **Description**: Defines whether customer user names in CM/Track case-sensitive.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.11.0.0

policy.username.case.sensitive

- **Module**: cmas-core-security
- **Description**: Defines whether user names are case-sensitive.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.10.1.0

postActivityExecutionScriptName

- **Module**: cmweb-server-adapter
- **Description**: Defines the name for the script which should be executed after every workflow activity, see section PostActivityExecutionScript. If no script should be executed, leave the value empty.
- **Type**: string

- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: postActivityExecutionHandler
- **Since**: 6.2.0

queue.polling.threads.number

- **Module**: cmas-nimh
- **Description**: Number of threads started for polling the email queue in the database.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 1 (default value)
- **Since**: 6.4.0

queue.polling.threads.shutdown.timeout.seconds

- **Module**: cmas-nimh
- **Description**: Waiting time after the shutdown signal for the email poller. When the timeout is reached, the thread will be terminated.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.4.0

queue.polling.threads.watchdog.interval.seconds

- **Module**: cmas-nimh
- **Description**: Determines the interval in seconds to execute the watchdog which checks the activity of the threads of the email queue poller.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 30 (default value)
- **Since**: 6.4.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

queue.task.error.pause.seconds

- **Module**: cmas-nimh
- **Description**: Maximum number of seconds, the queue poller waits after infrastructure (e.g. database) error.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 180 (default value)
- **Since**: 6.4.0

queue.task.interval.seconds

- **Module**: cmas-nimh
- **Description**: Determines the interval in seconds for the main email queue polling thread.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 15 (default value)
- **Since**: 6.4.0

queue.task.max.retries

- **Module**: cmas-nimh
- **Description**: Maximum number of email processing retries after an exception. When reached, the email is moved to the email archive. This email can be rescheduled again using the NIMH API or the Admin Tool / Web Admin Suite.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 10
- **Since**: 6.4.0

queue.task.timeout.seconds

- **Module**: cmas-nimh
- **Description**: After this time of inactivity (in seconds), the service thread is considered damaged and automatically restarted.
- **Type**: integer
- **Restart required**: no
- **System**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Optional**: yes
- **Example value**: 600 (default value)
- **Since**: 6.4.0

queue.task.transaction.timeout.seconds

- **Module**: cmas-nimh
- **Description**: Transaction timeout (in seconds) for email processing in the pipe.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.4.0

queuesExcludedFromGS

- **Module**: cmweb-server-adapter
- **Description**: Comma-separated list of queue names which are excluded from the quick search.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: Helpdesk,FAQ
- **Since**: 6.0

recent.items.cleanup.cluster.node.id

- **Module**: cmas-core-server
- **Description**: Value of a `-Dcmas.clusternode.id` designating the node which will clean up recent items.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 1 (assuming the cluster node started with `-Dcmas.clusternode.id=1` parameter)
- **Since**: 6.11.0.1

recent.items.cleanup.interval.minutes

- **Module**: cmas-core-server
- **Description**: Determines the time interval (in minutes) in which recent items should be checked for removal.

- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 60
- **Since**: 6.11.0.1

recent.items.max.per.engineer

- **Module**: cmas-core-server
- **Description**: Maximum number of preserved recent items per engineer while cleaning up (older recent items will be deleted).
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 500
- **Since**: 6.11.0.1

recent.items.persistence.enabled

- **Module**: cmas-core-server
- **Description**: Enables persistence of recent items. If the property is set to “false”, it prevents storing new recent items. If a recently visited widget (`recentlyVisitedWidget`) is used on a dashboard, this property needs to be set to “true”. Otherwise, you can set it to “false” to save system resources.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.11.1.0

recoverable.exceptions

- **Module**: cmas-dwh-server
- **Description**: Comma-separated list of exception definitions: CLASS[+][:REGEX]. The exceptions included in the list do not stop CM from sending to the CMRF process, but force it to try again. If optional '+' after CLASS is present, classes which extend CLASS are matched.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: java.sql.SQLRecoverableException,java.lang.RuntimeException+:.*T.1\,2T.*
- **Since**: 6.8.4.6

refreshTimeInCaseOfConcurrentRememberMeRequests

- **Module**: cmweb-server-adapter
- **Description**: Sets the refresh time (in seconds) after which the page is reloaded in case of concurrent remember me requests. This feature prevents one user from occupying many licenses.
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: yes
- **Example value**: 5
- **Since**: 6.8.2

rememberMeLifetimeInMinutes

- **Module**: cmweb-server-adapter
- **Description**: Lifetime for *remember me* in minutes.
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: 1440
- **Since**: 6.0

request.log.threshold.milliseconds

- **Module**: cmas-restapi-core
- **Description**: Time threshold in milliseconds that needs to be exceeded to generate an entry in the file `operationtimes-rest.log`.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5000 (default value)
- **Since**: 6.13.0.2

request.log.username

- **Module**: cmas-restapi-core
- **Description**: Name of the user whose requests are logged in the file `operationtimes-rest.log`. Only one name is supported. If the property is empty (default value), the requests of all users are logged.

- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: admin
- **Since**: 6.13.0.2

request.scope.transaction

- **Module**: cmweb-server-adapter
- **Description**: It allows to disable request scope transactions. By default one transaction is used per request. Setting this property to “false” will cause one transaction per service method invocation.
- **Type**: boolean
- **Restart required**: yes
- **System**: yes
- **Optional**: yes
- **Example value**: true
- **Since**: 6.8.1

resetCode.expirationPeriod

- **Module**: cmas-core-security
- **Description**: Defines the expiration period for the link when resetting the password in CM/Track.
- **Type**: Integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 86400000 (default value, 24 hours)
- **Since**: 6.10.1

resource.replace.batchSize

- **Module**: cmas-core-server
- **Description**: Defines the number of objects to be processed in a single resource replacement batch.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5
- **Since**: 6.10.0.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

resource.replace.timeout

- **Module**: cmas-core-server
- **Description**: Determines the transaction timeout (in seconds) of a resource replacement action step.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 120
- **Since**: 6.10.0.0

scene

- **Module**: cmas-setup-scene
- **Description**: Name of the scene file which was imported during the setup (can be empty).
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 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/
- **Since**: 6.0

script.evict.unused.after.hours

- **Module**: cmas-core-server
- **Description**: Determines the number of hours for which unused scripts remain in the cache. After this time, the compiled class of the script is removed. The ConSol CM server checks for scripts to evict every hour.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 24 (default)
- **Since**: 6.11.1.14

script.logging.threshold.seconds

- **Module**: cmas-core-server
- **Description**: Determines the time (in seconds) after which a warning is written to the log file if it is exceeded during script execution.
- **Type**: integer

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 10 (default)
- **Since**: 6.10.1.0

script.validation.interval.seconds

- **Module**: cmas-app-admin-tool
- **Description**: Sets the interval in seconds between two code checks in the script editor of the Admin Tool and Process Designer.
- **Type**: Integer
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 1 (default value)
- **Since**: 6.11.0.1

searchPageSize

- **Module**: cmweb-server-adapter
- **Description**: The global default page size for search results in the Web Client. Only used for new users who have not yet set preferences.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 20
- **Since**: 6.0

searchPageSizeOptions

- **Module**: cmweb-server-adapter
- **Description**: Paging options for the page size of search results in the Web Client.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 10|20|30|40|50|75|100
- **Since**: 6.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

security.fields.customer.exposure.check.enabled

- **Module**: cmas-restapi-core
- **Description**: Enables `customer exposure` annotation checks for ticket fields.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.10.5.4

security.restrict.unit.access.to.own.data

- **Module**: cmas-restapi-core
- **Description**: If set to “true”, an additional check is performed when a user logs in as a customer using the REST API, e.g. CM/Track. When requesting customer data, only the company of the user or other contacts of the user’s company are returned. If set to “false”, no additional security check is performed and the former security rules apply.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true (default value)
- **Since**: 6.9.2.14

serial.mods.tracking.enabled

- **Module**: cmas-core-server
- **Description**: Low level technical flag deciding whether serial diff tracking for entities is enabled. If enabled, there will be no `StackOverflow` error in case a dependency between two entities (for example engineer and ticket) causes an infinite loop first and then as a result, the stack overflow. The property must be added to the configuration manually. It will not be added to a system configuration during setup or update.

  > ⚠ Please do not enable this property unless advised by ConSol. It is a low level technical flag with intricate consequences for the system behavior.

- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.10.7.0, 6.11.0.5

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

server.instance.task.crash.period.seconds

- **Module**: cmas-core-server
- **Description**: Determines the period (in seconds) for detecting live and crashed ConSol CM server instances.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 15 (default value)
- **Since**: 6.11.1.6

server.instance.task.period.seconds

- **Module**: cmas-core-server
- **Description**: Determines the interval (in seconds) between repeated task executions. The task will be executed repeatedly with the given number of seconds between each execution.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 5 (default value)
- **Since**: 6.11.1.6

server.session.archive.reaper.interval

- **Module**: cmas-core-server
- **Description**: Determines the interval (in seconds) when the reaper for archived server sessions is executed (refers to the database table `cmas_user_session`).
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 60
- **Since**: 6.7.1

server.session.archive.timeout

- **Module**: cmas-core-server
- **Description**: Server sessions archive validity timeout (in days). After this time session info is removed from the database (refers to the database table `cmas_user_session`).
- **Type**: integer
- **Restart required**: no
- **System**: yes

- **Optional**: no
- **Example value**: 3 or 7
- **Since**: 6.7.1

server.session.heartbeat.enabled.WEB_CLIENT

- **Module**: cmas-core-server
- **Description**: Determines if the session heartbeat is enabled for the Web Client. If this property is set to “true”, a heartbeat is sent to the ConSol CM server every 20 seconds and with every page reload. If the server does not receive a heartbeat during the configured interval, the session is considered orphaned and ended according to the configured mode.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.11.2.9

server.session.heartbeat.mode

- **Module**: cmas-core-server
- **Description**: Determines when orphaned sessions are ended if the session heartbeat is enabled. There are two possible values:
  - **LAZY**: Orphaned sessions are ended when the same user authenticates again or when there are no more licenses available (default value).
  - **EAGER**: Orphaned sessions are ended immediately.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: LAZY (default value)
- **Since**: 6.11.2.9

server.session.heartbeat.timeout.seconds

- **Module**: cmas-core-server
- **Description**: Determines the period after which a session is considered orphaned if the session heartbeat is enabled and no heartbeat is received. The default value is 300, meaning that orphaned sessions are ended after 5 minutes.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: 300 (default value)
- **Since**: 6.11.2.9

server.session.reaper.interval

- **Module**: cmas-core-server
- **Description**: Determines the interval (in seconds) when the reaper for inactive (ended) server sessions is executed, refers to the database table `cmas_user_session`.
- **Type**: integer
- **Restart required**: only Session Service
- **System**: yes
- **Optional**: no
- **Example value**: 10800 (3 hours)
- **Since**: 6.6.1, 6.7.1

server.session.timeout

- **Module**: cmas-core-server
- **Description**: Server session timeout (in seconds) for connected clients (database table `cmas_user_session`). Each client can overwrite this timeout with custom value using its ID (ADMIN_TOOL, WEB_CLIENT, WORKFLOW_EDITOR, TRACK, ETL, REST) appended to property name, e.g., `server.session.timeout.ADMIN_TOOL`.
  Please see also the page customization attributes *updateTimeServerSessionActivityEnabled* and *updateTimeServerSessionActivity*, both of type *cmApplicationCustomization*.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1800
- **Since**: 6.6.1, 6.7.1

Detailed explanation for the Admin Tool:

- server.session.timeout.ADMIN_TOOL
  Defines the time interval how long the server considers a session valid while there is no activity from the Admin Tool holding the session. The Admin Tool is not aware of this value, it only suffers having an invalid session, if the last activity has been longer in the past.
- admin.tool.session.check.interval
  Defines the time between two checks done by the Admin Tool, if the server still considers its session valid.

For example, if admin.tool.session.check.interval = 60, the Admin Tool queries the server every minute if its session is still active/valid. In case server.session.timeout.ADMIN_TOOL = 600 the Admin Tool will get the response that the session is now invalid after ten minutes of inactivity.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

serverPoolingInterval

- **Module**: cmweb-server-adapter
- **Description**: Defines the time in seconds for the pooling server to invalidate caches on the web layer.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5
- **Since**: 6.1.0

show.error.stacktrace

- **Module**: cmweb-server-adapter
- **Description**: If this property is set to “true”, the full stracktrace can be shown in the red error message which is displayed in the Web Client if a user action caused an exception.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.12.1.0

skip-ticket

- **Module**: cmas-dwh-server
- **Description**: Tickets are not transferred during transfer/update.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.6.19
- **Removed in**: 6.8.1

skip-ticket-history

- **Module**: cmas-dwh-server
- **Description**: History of ticket is not transferred during transfer/update.
- **Type**: boolean
- **Restart required**: no
- **System**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Optional**: no
- **Example value**: false
- **Since**: 6.6.19
- **Removed in**: 6.8.1

skip-unit

- **Module**: cmas-dwh-server
- **Description**: Units are not transferred during transfer/update.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.6.19
- **Removed in**: 6.8.1

skip-unit-history

- **Module**: cmas-dwh-server
- **Description**: History of unit is not transferred during transfer/update.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.6.19
- **Removed in**: 6.8.1

skip.wfl.transfer.cleanup

- **Module**: cmas-core-server
- **Description**: Enables skipping the cleanup process after the deployment of a new workflow version. The cleanup checks whether elements which were deleted in the workflow model need to be preserved and - if not - removes them physically. Changing the default (“false”) to “true” might help if problems with deleted elements during the deployment of a new workflow version occur.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.9.4.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

skip.wfl.transfer.translations.cleanup

- **Module**: cmas-core-server
- **Description**: Enables skipping the cleanup of localized properties of removed workflow elements.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.10.5.5

split.history

- **Module**: cmas-dwh-server
- **Description**: Changes the SQL that fetches the history for the tickets during DWH transfer so that the history is not fetched for all tickets at once but only for one ticket per SQL.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: false
- **Since**: 6.8.0
- **Removed in**: 6.8.4.0

start.groovy.task.enabled

- **Module**: cmas-app-admin-tool
- **Description**: Determines if it is possible to execute tasks in the Admin Tool, section *Services -> Task Execution*. Set this system property to “true” to enable task execution.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true
- **Since**: 6.9.4.0

statistics.calendar

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.client.group

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.contact.role

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.content.entry

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: 0
- **Since**: 6.11.0.1

statistics.content.entry.class

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.content.entry.history

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.cube

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.13.0.2

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

statistics.customer.definition

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.engineer

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.enum.group

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

statistics.field.definition

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.group.definition

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.locale

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

statistics.localized.property

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.mla

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.project

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

statistics.queue

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.resource

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.resource.group

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

statistics.resource.history

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.resource.relation.definition

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.resource.type

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

statistics.ticket

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.ticket.function

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.ticket.history

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

statistics.time.booking

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.timestamp

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.unit

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

statistics.unit.history

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.unit.relation.definition

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.workflow

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

strict.utf.bmp.enabled

- **Module**: cmas-core-server
- **Description**: ConSol CM installations which use a MySQL database with a 2-byte encoding / collation setting cannot handle incoming emails which contain 4-byte UTF8 characters.
  If you set this property to “true”, all 4-byte UTF8 characters are filtered out from email subjects

before any database interaction. Since 6.11.2.7, it also filters out 4-byte UTF8 characters from other header fields, and since 6.11.2.8 UTF8 characters are filtered out from the names of email attachments.
By default, the property value is “true” for MySQL databases, and “false” for any other database where the underlying problem should not occur. Do not change it for a MySQL database, unless you are sure that the database supports 4-byte characters.

- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.10.6.0, changed in 6.11.2.7 and 6.11.2.8

supportEmail

- **Module**: cmweb-server-adapter
- **Description**:
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.0
- **Removed in**: 6.11.0.1

synchronize.archive.timeout.minutes

- **Module**: cmas-core-index-common
- **Description**: Determines the time period in minutes during which index updates are archived. The default value of this property is “60” for clustered environments and “0” for non-clustered environments. If the master node is switched, the new master can recover missing data in case it had not been fully synchronized with the previous master indexing server at the moment of switch.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 60
- **Since**: 6.12.0.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

synchronize.master.address

- **Module**: cmas-core-index-common
- **Description**: Value of `-Dcmas.http.host.port` specifying how to connect to the indexing master server. Since 6.12.0.0 it is possible to switch the node acting as master node by entering the address of the new master node. The index updates are archived for the time configured in the property `cmas-core-index-common`, `synchronize.archive.timeout.minutes`, so the new master can recover missing data in case it had not been fully synchronized with the previous master indexing server at the moment of switch.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 127.0.0.1:80
- **Since**: 6.6.0

synchronize.master.security.token

- **Module**: cmas-core-index-common
- **Description**: The password for accessing the index snapshot via URL, e.g., for index synchronization or for backups. This value is negotiated between nodes internally, please do not change it manually unless advised by ConSol.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: token
- **Since**: 6.6.0

synchronize.master.security.user

- **Module**: cmas-core-index-common
- **Description**: The user name for accessing the index snapshot via URL, e.g., for index synchronization or for backups. This value is negotiated between nodes internally, please do not change it manually unless advised by ConSol.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: user
- **Since**: 6.6.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

synchronize.master.timeout.minutes

- **Module**: cmas-core-index-common
- **Description**: Determines how much time (in minutes) the master server may continually fail until a new master gets elected. 0 means that master server will never change (failover mechanism is disabled).
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5 (default value)
- **Since**: 6.6.0

synchronize.megabits.per.second

- **Module**: cmas-core-index-common
- **Description**: Determines how much bandwidth (in megabits) the master server may consume when transferring index changes to all slave servers. Do not use all available bandwidth to transfer index changes between hosts, as doing so will most probably partition the cluster due to some subsystems being unable to communicate.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 85 (default value)
- **Since**: 6.6.0

synchronize.sleep.millis

- **Module**: cmas-core-index-common
- **Description**: Determines how often each slave server polls the master server for index changes.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1000 (default value)
- **Since**: 6.6.0

task.execution.interval.seconds

- **Module**: cmas-core-server
- **Description**: Time in seconds between the end of a completed task and the start of the next task in the Task Execution Framework.
- **Type**: Integer

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 5
- **Since**: 6.9.4.0

task.execution.node.id

- **Module**: cmas-core-server
- **Description**: Only relevant in clustered environments. The ID of the node where scripts of the TEF (Task Execution Framework) will be executed. This applies to both scripts called from the workflow and scripts called manually using the Admin Tool / Web Admin Suite, which can be started from any node.
- **Type**: Integer
- **Restart required**: yes
- **System**: no
- **Optional**: yes
- **Example value**: 2
- **Since**: 6.11.0.1

task.panel.refresh.interval.seconds

- **Module**: cmas-app-admin-tool
- **Description**: Sets the time in seconds after which the task list in the Admin Tool is refreshed.
- **Type**: Integer
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 10
- **Since**: 6.10.5.3 (not added automatically during update from versions prior to 6.10.5.3!)

themeOverlay

- **Module**: cmweb-server-adapter
- **Description**: Defines the user interface theme (CSS file) for the Web Client layout. Set this property to the name of your custom skin if you are using one.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: consolINT
- **Since**: 6.0, functionality changed in 6.11.2.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

ticket.delete.timeout

- **Module**: cmas-core-server
- **Description**: Transaction timeout (in seconds) for deleting tickets in a bulk.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 60
- **Since**: 6.1.3

ticket.from.incoming.message.accepted.links

- **Module**: cmas-core-server
- **Description**: List of domains to which links in incoming emails and links in comments added via REST API are clickable in the ticket history. Regular expressions can be used to specify the allowed URLs. It is possible to add several URLs by using a whitespace as delimiter. The URL must start with one of the allowed protocols (http, https, ftp, ftps, file, mailto). All other links are removed, i.e., the link is displayed in the ticket history as text but it cannot be clicked. If the property is left empty, all links are removed. The regular expression `.+` can be used to allow all domains.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: https://.*\.consol\.de (allows links to “https://<any>.consol.de”)
- **Since**: 6.11.1.7



Please note that whitelisting domains might make ConSol CM vulnerable to cross-site scripting and other attacks. Choose the domains you whitelist carefully!

ticketListRefreshIntervalInSeconds

- **Module**: cmweb-server-adapter
- **Description**: Refresh interval for the ticket list (in seconds).
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 180
- **Since**: 6.0

ticketListSizeLimit

- **Module**: cmweb-server-adapter
- **Description**: Maximum number of tickets in the ticket list.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 100
- **Since**: 6.0

tickets.delete.size

- **Module**: cmas-core-server
- **Description**: Defines the number of tickets deleted in a single transaction.
- **Type**: integer
- **Restart required**: only Session Service
- **System**: yes
- **Optional**: no
- **Example value**: 10 (default value)
- **Since**: 6.8.1

time.buffer

- **Module**: cmas-dwh-server
- **Description**: Number of minutes to extend the date of the start of the live mode.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 5
- **Since**: 6.8.1.11

transaction.timeout.minutes

- **Module**: cmas-core-server
- **Description**: Sets the transaction timeout (in minutes) for the task execution service, i.e., one run of a task must finish before this timeout is reached. The changes are visible only for new tasks, the execution of which started after the configuration change.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: 10*60 (10 hours - default value)
- **Since**: 6.10

ttl.days

- **Module**: cmas-web-notifications
- **Description**: Determines the time (in days) after which unpinned notifications are removed.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 15
- **Since**: 6.12.0.0

tx.read.only.mode.enabled

- **Module**: cmweb-server-adapter
- **Description**: Enables read-only transactions for faster page loading. This transactional behavior was introduced in 6.11.0, and this property acts as a safety guard to restore the old behaviors. Do not change this value unless you are facing transaction problems and advised by ConSol.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.11

tinymce.enabled

- **Module**: cmweb-server-adapter
- **Description**: Determines whether Froala (“false”) or TinyMCE (“true”) is used as an editor for the Web Client.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.12.1.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

unit.description.mode

- **Module**: cmas-core-server
- **Description**: Defines whether contact descriptions in the ticket history are taken from the database or dynamically rendered using templates. The value, “DYNAMIC”, is a bit more costly from the performance perspective, while “PROTOCOL” is faster but returns historical names which might be outdated. Use “PROTOCOL” if you have lots of history entries from many different units. This is also the default value in CM versions 6.11.1.1 and up.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: PROTOCOL
- **Since**: 6.11.0

unit.replace.batchSize

- **Module**: cmas-core-server
- **Description**: Defines the number of objects to be processed in a single customer replacement batch.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5
- **Since**: 6.8.2

unit.replace.timeout

- **Module**: cmas-core-server
- **Description**: Determines the transaction timeout (in seconds) of a customer replacement action step.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 120
- **Since**: 6.8.2

unit.transfer.order

- **Module**: cmas-dwh-server
- **Description**: Defines in which order customer field groups should be transferred to the DWH.
- **Type**: string

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: company;customer
- **Since**: 6.6.19
- **Removed in**: 6.8.1

unitIndexSearchResultSizeLimit

- **Module**: cmweb-server-adapter
- **Description**: Defines the maximum number of hints for all customer autocomplete or suggestion components (e.g. when adding customer to a ticket). Components customizations (maxHints, maxSuggestions) may change that value.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5
- **Since**: 6.0

unused.content.remover.cluster.node.id

- **Module**: cmas-core-server
- **Description**: Value of a `cmas.clusternode.id` designating which node will remove unused ticket attachments and customer content entries.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 1 (assuming cluster node started with the parameter `-Dcmas.clusternode.id=1`)
- **Since**: 6.9.0.0

unused.content.remover.enabled

- **Module**: cmas-core-server
- **Description**: Specifies whether a removal of unused ticket attachments and customer content entries should take place.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: true
- **Since**: 6.9.0.0

unused.content.remover.polling.minutes

- **Module**: cmas-core-server
- **Description**: Determines the time interval (in minutes) for checking unused ticket attachments and customer content entries for removal.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 15
- **Since**: 6.9.0.0

unused.content.remover.ttl.minutes

- **Module**: cmas-core-server
- **Description**: Minimum interval (in minutes) while removed attachments and comments are preserved before finally deleting them from the database.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1440
- **Since**: 6.9.0.0

update.6.11.0.0.sleep

- **Module**: cmas-setup-hibernate
- **Description**: Helper property for the update preparation scripts introduced in context of CM database refactoring in version 6.11. This is an optional setting allowing a delay (in milliseconds) after each loop iteration of the preparation scripts. Setting the delay should lower the database load, for example during working hours. This property may be removed after the update preparation tasks finish.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.0, for use in 6.10.5.x

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

update.6.11.0.0.timezone

- **Module**: cmas-setup-hibernate
- **Description**: Helper property for the ticket history migration (the new way of counting history groups). Since 6.11.0.0 the groups are constant (2h time span), but before 6.11.0.0 groups were not constant and depended on the customer's time zone. Migration scripts use an old algorithm to calculate groups and therefore need information about the time zone. The property should be set to the timezone which is most commonly used by the customers. If the property is not set, the default server time zone is used (`TimeZone.getDefault()`). The property should be set before updating to 6.11.0.0 and will be removed automatically after migration. The list of accepted timezones can be found for example here: http://joda-time.sourceforge.net/timezones.html.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: Europe/Berlin
- **Since**: 6.11.0.0, for use before updating to this version

urlLogoutPath

- **Module**: cmweb-server-adapter
- **Description**: URL which is opened when the user logs out. There are three configuration possibilities:
  - Empty value: The user is redirected to the login page.
  - <CM URL>/cm-client/logout: The user is redirected to the logout page, which includes a link to log in again.
  - Any URL: The user is redirected to the given URL outside ConSol CM.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: https://mycmserver/cm-client/logout
- **Since**: 6.3.1, behavior changed in 6.11.2.0

url.track

- **Module**: cmas-core-server
- **Description**: Determines the URL of CM/Track to be used by the class `LinkService` to generate links to tickets in CM/Track. The following example shows how the use the convenience method to generate a link to a ticket (the ticket object is passed as a parameter).

```
url = linkTo.track(ticket);
```

- **Type**: string

- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: http://localhost:8080/track
- **Since**: 6.13.0.0

url.webclient

- **Module**: cmas-core-server
- **Description**: Determines the URL of the Web Client to be used by the class `LinkService` to generate links to tickets, customers or resources in the Web Client. The following example shows how the use the convenience method to generate a link to a customer (the customer object is passed as a parameter).

```
url = linkTo.webclient(unit);
```

- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: http://localhost:8080/cm-client
- **Since**: 6.13.0.0

voCacheEnabled

- **Module**: cmweb-server-adapter
- **Description**: This property enables additional caching for the Web Client, voCaching, of complete objects, thus improving performance.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: true
- **Since**: 6.11.1.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Notes:

- Since ConSol CM version 6.11.1.1, the default value is “true” for non-clustered environments. The value is set to “true” automatically during the setup or update of ConSol CM 6.11.1.1.
- When voCaching is enabled and lazy loading is used for folding ticket history entries, once the engineer unfolded an entry, he cannot fold it again by reloading the page or opening the ticket from the workspace.
- When using the dynamic mode for displaying engineer and customer names in the ticket history (as configured in the system properties cmas-core-server, engineer.description.mode and cmas-core-server, unit.description.mode), the new version of the engineer and/or customer name is only displayed after the ticket has been changed.
- This system property is ignored for clustered environments (environments with cmas-core-shared, cluster.mode set to “true”). In clustered environments, voCaching is always disabled to avoid problems that changes made to objects on one node are not visible on the other nodes.

warmup.executor.enabled

- **Module**: cmas-core-server
- **Description**: Specifies whether the server should asynchronously warm up during startup (e.g., fill some of the internal caches) for better performance.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.9.4.2

webSessionTimeoutInMinutes

- **Module**: cmweb-server-adapter
- **Description**: Session timeout in minutes.
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: 180
- **Removed in**: 6.7.1
- **Replaced by**: cmas-core-server, server.session.timeout

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

wfl.sticky.transfer.disabled

- **Module**: cmas-core-server
- **Description**: Enables using preserved original names of workflow elements.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false
- **Since**: 6.10.1.0

wicketAjaxRequestHeaderFilterEnabled

- **Module**: cmweb-server-adapter
- **Description**: This enables the filter for Wicket AJAX requests, coming from stale pages with Wicket 1.4 scripting (CM < 6.8.0), after an update to CM > 6.8.0.
- **Type**: boolean
- **Restart required**: yes
- **System**: yes
- **Optional**: yes
- **Example value**: false
- **Since**: 6.8.1

workflow.deploy.cache.eviction.disabled

- **Module**: cmas-core-server
- **Description**: Determines if the infinispan cache should be cleared after deployment (“true”) or not (“false”). If the property is set to “true”, all mappings are removed from the cache.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.11.1.4

X-Frame-Options

- **Module**: cmweb-server-http-headers
- **Description**: Example property to illustrate the configuration of HTTP headers. In this case the delivered HTTP header contains the field *X-Frame-Options* with the value “SAMEORIGIN”.

  Each property in the module `cmweb-server-http-headers` represents one header field. The property name/key identifies the response header field and the value of the property is the field value sent in this header.

> ⚠ Please be aware that additional HTTP response headers must be correctly defined with the exact spelling as officially specified! Please note also that the correct interpretation and application of these headers is fully in the realm and responsibility of the client browser which requested the page!

- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: SAMEORIGIN
- **Since**: 6.10.8

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

H.2.2 List of System Properties by Area

This chapter lists the system properties which are relevant for the following areas.

- CMRF & DWH Configuration
- Indexer and Search Configuration
- LDAP Configuration
- Email Configuration
- Activity Interval Configuration
- Administrator Email Addresses
- HTTP Header Configuration

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

H.2.2.1 CMRF & DWH Configuration

autocommit.cf.changes

- **Module**: cmas-dwh-server
- **Description**: Defines whether DWH tasks which result from configurational changes on ticket fields are executed automatically without manual interaction in the Admin Tool. Can be also set in the Admin Tool in the navigation item *DWH*. The default and recommended value is “false”.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false (default value)
- **Since**: 6.7.0

batch-commit-interval

- **Module**: cmas-dwh-server
- **Description**: Number of objects sent in one DWH message. Larger values mean better transfer performance at the cost of higher memory usage.
  Starting with ConSol CM version 6.11, this property is only used if the package size of a DWH operation is not set. This can only happen when the command is directly addressed to the Java MBean `consol.cmas.global.dwh.synchronizationService`, e.g. using the `update()` method. When a DWH operation is started using the Admin Tool, there is always a value for the package size. If not explicitly set, the default value of 1000 is used as value for `batch.commit.interval`.
- **Default value**: 1000
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 100
- **Since**: 6.0.0

communication.channel

- **Module**: cmas-dwh-server
- **Description**: Communication channel. Only possible value since CM version 6.11.0.0: DIRECT
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: DIRECT

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Since**: 6.8.5.0
- **Removed in**: 6.11.0.0 (DIRECT mode is the only available mode and is set automatically)

dwh.mode

- **Module**: cmas-dwh-server
- **Description**: Currently active mode for the DWH data transfer. Possible values are OFF, ADMIN, LIVE.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: OFF
- **Since**: 6.0.1

ignore-queues

- **Module**: cmas-dwh-server
- **Description**: A comma-separated list of queue names which are not not transferred to the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: QueueName1,QueueName2,QueueName3
- **Since**: 6.6.19
- **Removed in**: 6.8.1

is.cmrf.alive

- **Module**: cmas-dwh-server
- **Description**: Defines a threshold timeframe (in seconds) for the check if CMRF is working properly. If no response from CMRF is received in the configured timeframe, the DWH operation status is updated to “error”.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1200
- **Since**: 6.7.0

java.naming.factory.initial

- **Module**: cmas-dwh-server
- **Description**: Factory class for the DWH context factory.

- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: org.jnp.interfaces.NamingContextFactory
- **Since**: 6.0.1
- **Removed in**: 6.11.0.0

java.naming.factory.url.pkgs

- **Module**: cmas-dwh-server
- **Description**:
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: org.jboss.naming:org.jnp.interfaces
- **Since**: 6.0.1
- **Removed in**: 6.11.0.0

java.naming.provider.url

- **Module**: cmas-dwh-server
- **Description**: URL of naming provider.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: localhost
- **Since**: 6.0.1
- **Removed in**: 6.11.0.0

notification.error.description

- **Module**: cmas-dwh-server
- **Description**: Defines the text for error emails from the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: Error occurred
- **Since**: 6.0.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

notification.error.from

- **Module**: cmas-dwh-server
- **Description**: Defines the From address for error emails from the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: myuser@consol.de
- **Since**: 6.0.1

notification.error.subject

- **Module**: cmas-dwh-server
- **Description**: Defines the email subject for error emails from the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: Error occurred
- **Since**: 6.0.1

notification.error.to

- **Module**: cmas-dwh-server
- **Description**: Defines the To address for error emails from the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Since**: 6.0.1

notification.finished_successfully.description

- **Module**: cmas-dwh-server
- **Description**: Defines the text for emails from the DWH when a transfer finishes successfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: Transfer finished successfully
- **Since**: 6.0.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

notification.finished_successfully.from

- **Module**: cmas-dwh-server
- **Description**: Defines the From address for emails from the DWH when a transfer finishes successfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: myuser@consol.de
- **Since**: 6.0.1

notification.finished_successfully.subject

- **Module**: cmas-dwh-server
- **Description**: Defines the subject for emails from the DWH when a transfer finishes successfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: Transfer finished successfully
- **Since**: 6.0.1

notification.finished_successfully.to

- **Module**: cmas-dwh-server
- **Description**: Defines the To address for emails from the DWH when a transfer finishes successfully.
- **Type**: string
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Since**: 6.0.1

notification.finished_unsuccessfully.description

- **Module**: cmas-dwh-server
- **Description**: Defines the text for emails from the DWH when a transfer finishes unsuccessfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: Transfer finished unsuccessfully
- **Since**: 6.0.1

notification.finished_unsuccessfully.from

- **Module**: cmas-dwh-server
- **Description**: Defines the From address for emails from the DWH when a transfer finishes unsuccessfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: myuser@consol.de
- **Since**: 6.0.1

notification.finished_unsuccessfully.subject

- **Module**: cmas-dwh-server
- **Description**: Defines the subject for emails from the DWH when a transfer finishes unsuccessfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: Transfer finished unsuccessfully
- **Since**: 6.0.1

notification.finished_unsuccessfully.to

- **Module**: cmas-dwh-server
- **Description**: Defines the To address for emails from the DWH when a transfer finishes unsuccessfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Since**: 6.0.1

notification.host

- **Module**: cmas-dwh-server
- **Description**: Email (SMTP) server hostname for sending DWH emails.
- **Type**: string
- **Restart required**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **System**: yes
- **Optional**: yes
- **Example value**: myserver.consol.de
- **Since**: 6.0.1

notification.password

- **Module**: cmas-dwh-server
- **Description**: SMTP password for sending DWH emails (optional).
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.0.1

notification.port

- **Module**: cmas-dwh-server
- **Description**: SMTP port for sending DWH emails.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 25
- **Since**: 6.0.1

notification.protocol

- **Module**: cmas-dwh-server
- **Description**: The protocol used for sending emails from the DWH. Possible values are smtp or pop3.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: pop3

notification.tls.enabled

- **Module**: cmas-dwh-server
- **Description**: Enables SMTP via SSL/TLS (SMTPS) for sending notification emails from the DWH. The default value is “false”. If it is set to “true”, SMTPS is enabled for sending notifications from the DWH.
- **Type**: string
- **Restart required**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.11.1.6

notification.username

- **Module**: cmas-dwh-server
- **Description**: (SMTP) user name for sending DWH emails.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: myuser
- **Since**: 6.0.1

recoverable.exceptions

- **Module**: cmas-dwh-server
- **Description**: Comma-separated list of exception definitions: CLASS[+][:REGEX]. The exceptions included in the list do not stop CM from sending to the CMRF process, but force it to try again. If optional '+' after CLASS is present, classes which extend CLASS are matched.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: java.sql.SQLRecoverableException,java.lang.RuntimeException+:.*T.1\,2T.*
- **Since**: 6.8.4.6

skip-ticket

- **Module**: cmas-dwh-server
- **Description**: Tickets are not transferred during transfer/update.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.6.19
- **Removed in**: 6.8.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

skip-ticket-history

- **Module**: cmas-dwh-server
- **Description**: History of ticket is not transferred during transfer/update.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.6.19
- **Removed in**: 6.8.1

skip-unit

- **Module**: cmas-dwh-server
- **Description**: Units are not transferred during transfer/update.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.6.19
- **Removed in**: 6.8.1

skip-unit-history

- **Module**: cmas-dwh-server
- **Description**: History of unit is not transferred during transfer/update.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.6.19
- **Removed in**: 6.8.1

split.history

- **Module**: cmas-dwh-server
- **Description**: Changes the SQL that fetches the history for the tickets during DWH transfer so that the history is not fetched for all tickets at once but only for one ticket per SQL.
- **Type**: boolean
- **Restart required**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **System**: yes
- **Optional**: yes
- **Example value**: false
- **Since**: 6.8.0
- **Removed in**: 6.8.4.0

unit.transfer.order

- **Module**: cmas-dwh-server
- **Description**: Defines in which order customer field groups should be transferred to the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: company;customer
- **Since**: 6.6.19
- **Removed in**: 6.8.1

H.2.2.2 Indexer and Search Configuration

Indexer

big.task.minimum.size

- **Module**: cmas-core-index-common
- **Description**: Indicates the minimum size of index task (in parts, each part has 100 entities) to qualify this task as a big one. Big tasks have a lower priority than runtime tasks.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 15 (default value)
- **Since**: 6.8.3

database.notification.redelivery.delay.seconds

- **Module**: cmas-core-index-common
- **Description**: If the database is used as a notification channel for index updates, this property indicates the delay for notification redelivery when an exception occurs.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: 60
- **Since**: 6.8.4.7

database.notification.redelivery.max.attempts

- **Module**: cmas-core-index-common
- **Description**: If the database is used as a notification channel for index updates, this property indicates maximum redelivery attempts when an exception occurs.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 60
- **Since**: 6.8.4.7

disable.admin.task.auto.commit

- **Module**: cmas-core-index-common
- **Description**: Toggle for the automatic execution of administrative index update tasks. If set to “false”, possibly expensive index tasks created as a result of configuration changes will be automatically executed right after creation. It is recommended to set this value to “true”, so that the execution of administrative index tasks needs to be manually triggered in the Admin Tool or Web Admin Suite.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.6.1

index.attachment

- **Module**: cmas-core-index-common
- **Description**: Determines whether the content of attachments is indexed.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.4.3

index.history

- **Module**: cmas-core-index-common
- **Description**: Determines whether ticket and customer history is indexed.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.1.0
- **Removed in**: 6.11.0

index.status

- **Module**: cmas-core-index-common
- **Description**: Index synchronization status, possible values are GREEN, YELLOW or RED. This value is set internally by the index system, please do not change it manually unless advised by ConSol.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: GREEN
- **Since**: 6.6.1

index.task.worker.threads

- **Module**: cmas-core-index-common
- **Description**: Sets the number of threads that will be used to execute index tasks. The default value is 1, we recommend to increase it slightly on systems with large indexer load up to a value of 4.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1 (default value)
- **Since**: 6.6.14, 6.7.3. Since 6.8.0 and exclusively in 6.6.21 also normal (live) index updates are affected by this property.

index.version.current

- **Module**: cmas-core-index-common
- **Description**: Holds information about the current index version. This value is maintained internally, please do not change it unless advised by ConSol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: 1 (default value)
- **Since**: 6.7.0

index.version.newest

- **Module**: cmas-core-index-common
- **Description**: Holds information about which index version is considered newest. This value is maintained internally, please do not change it unless advised by ConSol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1 (default value)
- **Since**: 6.7.0

indexed.assets.per.thread.in.memory

- **Module**: cmas-core-index-common
- **Description**: Determines how many assets should be loaded into memory at once, per thread, during indexing.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 200 (default value)
- **Since**: 6.8.0

indexed.engineers.per.thread.in.memory

- **Module**: cmas-core-index-common
- **Description**: Determines how many engineers should be loaded into memory at once, per thread, during indexing. Increasing it slightly will lead to better indexer performance at the price of higher memory consumption.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 300 (default value)
- **Since**: 6.6.14, 6.7.3

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

indexed.resources.per.thread.in.memory

- **Module**: cmas-core-index-common
- **Description**: Determines how many resources should be loaded into memory at once, per thread, during indexing. Increasing it slightly will lead to better indexer performance at the price of higher memory consumption.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 200 (default value)
- **Since**: 6.10.0.0

indexed.tickets.per.thread.in.memory

- **Module**: cmas-core-index-common
- **Description**: Determines how many tickets should be loaded into memory at once, per thread, during indexing. Increasing it slightly will lead to better indexer performance at the price of higher memory consumption.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 100 (default value)
- **Since**: 6.6.14, 6.7.3

indexed.units.per.thread.in.memory

- **Module**: cmas-core-index-common
- **Description**: Determines how many customers should be loaded into memory at once, per thread, during indexing. Increasing it slightly will lead to better indexer performance at the price of higher memory consumption.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 200 (default value)
- **Since**: 6.6.14, 6.7.3

synchronize.master.address

- **Module**: cmas-core-index-common
- **Description**: Value of `-Dcmas.http.host.port` specifying how to connect to the indexing master server. Since 6.12.0.0 it is possible to switch the node acting as master node by entering the address of the new master node. The index updates are archived for the time configured in the

property `cmas-core-index-common`, `synchronize.archive.timeout.minutes`, so the new master can recover missing data in case it had not been fully synchronized with the previous master indexing server at the moment of switch.

- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 127.0.0.1:80
- **Since**: 6.6.0

synchronize.master.security.token

- **Module**: cmas-core-index-common
- **Description**: The password for accessing the index snapshot via URL, e.g., for index synchronization or for backups. This value is negotiated between nodes internally, please do not change it manually unless advised by ConSol.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: token
- **Since**: 6.6.0

synchronize.master.security.user

- **Module**: cmas-core-index-common
- **Description**: The user name for accessing the index snapshot via URL, e.g., for index synchronization or for backups. This value is negotiated between nodes internally, please do not change it manually unless advised by ConSol.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: user
- **Since**: 6.6.0

synchronize.master.timeout.minutes

- **Module**: cmas-core-index-common
- **Description**: Determines how much time (in minutes) the master server may continually fail until a new master gets elected. 0 means that master server will never change (failover mechanism is disabled).
- **Type**: integer
- **Restart required**: no

- **System**: yes
- **Optional**: no
- **Example value**: 5 (default value)
- **Since**: 6.6.0

synchronize.megabits.per.second

- **Module**: cmas-core-index-common
- **Description**: Determines how much bandwidth (in megabits) the master server may consume when transferring index changes to all slave servers. Do not use all available bandwidth to transfer index changes between hosts, as doing so will most probably partition the cluster due to some subsystems being unable to communicate.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 85 (default value)
- **Since**: 6.6.0

synchronize.sleep.millis

- **Module**: cmas-core-index-common
- **Description**: Determines how often each slave server polls the master server for index changes.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1000 (default value)
- **Since**: 6.6.0

Search Results

globalSearchResultSizeLimit

- **Module**: cmweb-server-adapter
- **Description**: Determines the maximum number of results in the quick search.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 10
- **Since**: 6.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

searchPageSize

- **Module**: cmweb-server-adapter
- **Description**: The global default page size for search results in the Web Client. Only used for new users who have not yet set preferences.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 20
- **Since**: 6.0

searchPageSizeOptions

- **Module**: cmweb-server-adapter
- **Description**: Paging options for the page size of search results in the Web Client.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 10|20|30|40|50|75|100
- **Since**: 6.0

unitIndexSearchResultSizeLimit

- **Module**: cmweb-server-adapter
- **Description**: Defines the maximum number of hints for all customer autocomplete or suggestion components (e.g. when adding customer to a ticket). Components customizations (maxHints, maxSuggestions) may change that value.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5
- **Since**: 6.0

H.2.2.3 LDAP Configuration

LDAP Configuration (if LDAP is Used as Authentication Mode in the CM Web Client)

LDAP parameters apply only if the authentication mode for the CM Web Client has been set to “LDAP”:

authentication.method

- **Module**: cmas-core-security
- **Description**: Sets the user authentication method (internal database or LDAP authentication). Allowed values are LDAP or DATABASE.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: DATABASE
- **Since**: 6.0

ldap.authentication

- **Module**: cmas-core-security
- **Description**: LDAP authentication method used when using LDAP authentication. Possible values are “anonymous” and “simple”.
- **Type**: string
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: simple (default value)
- **Since**: 6.0

ldap.basedn

- **Module**: cmas-core-security
- **Description**: Base DN used for looking up LDAP user accounts when using LDAP authentication.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: ou=accounts,dc=consol,dc=de
- **Since**: 6.0

ldap.initialcontextfactory

- **Module**: cmas-core-security
- **Description**: Class name for the initial context factory of the LDAP implementation when using LDAP authentication. Usually this is `com.sun.jndi.ldap.LdapCtxFactory`.
- **Type**: string
- **Restart required**: yes
- **System**: yes
- **Optional**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: com.sun.jndi.ldap.LdapCtxFactory
- **Since**: 6.0

ldap.password

- **Module**: cmas-core-security
- **Description**: Password for connecting to LDAP to look up users when using LDAP authentication. Only needed if lookup cannot be performed anonymously.
- **Type**: password
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.1.2

ldap.providerurl

- **Module**: cmas-core-security
- **Description**: URL to LDAP provider, when using LDAP authentication.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: ldap://myserver.consol.de:389
- **Since**: 6.0

ldap.searchattr

- **Module**: cmas-core-security
- **Description**: Search attribute for looking up the LDAP entries associated with a CM login.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: uid
- **Since**: 6.0

ldap.userdn

- **Module**: cmas-core-security
- **Description**: LDAP user for connecting to LDAP to look up users when using LDAP authentication. Only needed if the lookup cannot be performed anonymously.
- **Type**: string
- **Restart required**: no
- **System**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Optional**: yes
- **Since**: 6.1.2

LDAP Configuration (if LDAP is Used as Authentication Mode in CM/Track)

LDAP parameters apply only if the authentication mode for CM/Track has been set to “LDAP”:

contact.authentication.method

- **Module**: cmas-core-security
- **Description**: Sets the authentication method for contacts accessing REST services, possible values are DATABASE or LDAP or LDAP,DATABASE or DATABASE,LDAP.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: LDAP
- **Since**: 6.9.3.0

ldap.contact.name.basedn

- **Module**: cmas-core-security
- **Description**: Base path to search for the contact DN by LDAP ID.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: ou=accounts,dc=consol,dc=de
- **Since**: 6.9.3.0

ldap.contact.name.password

- **Module**: cmas-core-security
- **Description**: Password to look up the contact DN by LDAP ID. If not set, the anonymous account is used.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Since**: 6.9.3.0

ldap.contact.name.providerurl

- **Module**: cmas-core-security
- **Description**: Address of the LDAP server (ldap[s]://host:port) used for contact authentication.
- **Type**: string

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: ldap://ldap.consol.de:389
- **Since**: 6.9.3.0

ldap.contact.name.searchattr

- **Module**: cmas-core-security
- **Description**: Attribute to search for the contact DN by LDAP ID (e.g. uid).
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: uid
- **Since**: 6.9.3.0

ldap.contact.name.userdn

- **Module**: cmas-core-security
- **Description**: User DN to look up contact DN by LDAP ID. If not set, the anonymous account is used.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Since**: 6.9.3.0

ldap.initialcontextfactory

- **Module**: cmas-core-security
- **Description**: Class name for the initial context factory of the LDAP implementation when using LDAP authentication. Usually this is `com.sun.jndi.ldap.LdapCtxFactory`.
- **Type**: string
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: com.sun.jndi.ldap.LdapCtxFactory
- **Since**: 6.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

H.2.2.4 Email Configuration

Outgoing Email

mail.smtp.email

- **Module**: cmas-core-server
- **Description**: SMTP URL for outgoing emails
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: smtp://mail.mydomain.com:25
- **Since**: 6.0

mail.smtp.envelopesender

- **Module**: cmas-core-server
- **Description**: Email address used as a sender in SMTP envelope. If not set, the From address of the email is used.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: mysender@mydomain.com
- **Since**: 6.5.7

mail.from

- **Module**: cmweb-server-adapter
- **Description**: Central sender address to be used for email communication. The value set here appears as sender of emails sent from ConSol CM.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: consolcm@example.com
- **Since**: 6.1.2

mail.reply.to

- **Module**: cmweb-server-adapter
- **Description**: Central reply-to address to be used for email communication. The value set here controls the reply-to header of emails sent from ConSol CM.
- **Type**: string

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: consolcm@example.com
- **Since**: 6.0.1

> ⚠ Please read the detailed information about ConSol CM Reply-To addresses in section Scripts of Type Email.

mailTemplateAboveQuotedText

- **Module**: cmweb-server-adapter
- **Description**: Determines the behavior of the email template in the email editor when another email is quoted, i.e. forwarded or replied to. Often used to place the signature correctly.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.2.4

mail.sender.address

- **Module**: cmas-workflow-jbpm
- **Description**: From address for emails from the workflow engine.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Removed in**: 6.8.0
- **Replaced by**: jobExecutor.mailFrom

mail.smtp.tls.enabled

- **Module**: cmas-core-server
- **Description**: Activates SMTP via SSL/TLS (SMTPS) for sending emails from the Web Client and scripts. The default value is “false”. If it is set to “true”, SMTPS is activated for sending emails.
- **Type**: boolean
- **Restart required**: yes
- **System**: no
- **Optional**: yes

- **Example value**: true
- **Since**: 6.11.1.6

Incoming Email

nimh.enabled

- **Module**: cmas-core-server
- **Description**: Enables the NIMH service which controls email polling and processing in ConSol CM. Must be suffixed with NODEID in cluster environments e.g. `nimh.enabled.NODEID` = “true”.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false
- **Since**: 6.9.4.0

filesystem.polling.threads.number

- **Module**: cmas-nimh
- **Description**: Number of threads started for polling emails from the database queue. The default value is 1.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 10
- **Since**: 6.4.0

filesystem.polling.threads.shutdown.timeout.seconds

- **Module**: cmas-nimh
- **Description**: Waiting time after the shutdown signal. When the timeout is reached, the thread will be terminated.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.4.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

filesystem.polling.threads.watchdog.interval.seconds

- **Module**: cmas-nimh
- **Description**: Determines the interval in seconds to execute the watchdog which checks the activity of the threads of the file system poller.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 30 (default value)
- **Since**: 6.4.0

filesystem.task.enabled

- **Module**: cmas-nimh
- **Description**: Allows to disable the service thread related to a given poller.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.4.0

filesystem.task.interval.seconds

- **Module**: cmas-nimh
- **Description**: Default interval (in seconds) for polling the file system.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.4.0

filesystem.task.polling.folder

- **Module**: cmas-nimh
- **Description**: Polling folder location which is scanned for emails in the format of `eml` files. The default value is the `mail` subdirectory of the ConSol CM data directory.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: c://cmas//mail
- **Since**: 6.4.0

filesystem.task.timeout.seconds

- **Module**: cmas-nimh
- **Description**: After this time of inactivity the service thread is considered damaged and automatically restarted. The default value is 120 seconds.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60
- **Since**: 6.4.0

filesystem.task.transaction.timeout.seconds

- **Module**: cmas-nimh
- **Description**: Default transaction timeout for transactions fetching emails from the file system. Should be correlated with number of messages fetched at once.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.4.0

mailbox.<number>.name

- **Module**: cmas-nimh
- **Description**: Identifier (name) of the mailbox.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: myEmailAccount
- **Since**: 6.11.2.0

mailbox.default.connection.host

- **Module**: cmas-nimh
- **Description**: Host (server name) of a given mailbox which the poller reads emails from.
- **Type**: string
- **Restart required**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **System**: no
- **Optional**: yes
- **Example value**: 10.10.1.157
- **Since**: 6.4.0

mailbox.default.connection.password

- **Module**: cmas-nimh
- **Description**: Password for given mailbox which the poller reads emails from.
- **Type**: password
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: consol
- **Since**: 6.4.0

mailbox.default.connection.port

- **Module**: cmas-nimh
- **Description**: Port of a given mailbox which the poller reads emails from.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 143
- **Since**: 6.4.0

mailbox.default.connection.protocol

- **Module**: cmas-nimh
- **Description**: The protocol used by the email poller, e.g. IMAP or POP3.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: imap
- **Since**: 6.4.0

mailbox.default.connection.username

- **Module**: cmas-nimh
- **Description**: User name for a given mailbox which the poller reads emails from.
- **Type**: string
- **Restart required**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **System**: no
- **Optional**: yes
- **Example value**: username
- **Since**: 6.4.0

mailbox.default.session.mail.debug

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. Allows for more detailed JavaMail session debugging.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true
- **Since**: 6.4.0

mailbox.default.session.mail.mime.address.strict

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. If this property is set to “false”, the parsing of the email headers is less strict. Therefore, more emails with non-RFC-compliant headers will be imported into the system, instead of being stored in the email backups in the Admin Tool / Web Admin Suite.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.4.0

mailbox.default.session.mail.<PROTOCOL>.connectiontimeout

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. Determines the connection timeout in milliseconds.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 3000
- **Since**: 6.4.0 (IMAP and POP3) / 6.11.2.0 (IMAPS and POP3S)

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

mailbox.default.session.mail.<PROTOCOL>.fetchsize

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. Determines the size of partial fetch in bytes for the indicated protocol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 1048576 (default value)
- **Since**: 6.9.4.0 (IMAP and POP3) / 6.11.2.0 (IMAPS and POP3S)

mailbox.default.session.mail.<PROTOCOL>.partialfetch

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. Determines whether the protocol's partial fetch capability should be used.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.9.4.0 (IMAP and POP3) / 6.11.2.0 (IMAPS and POP3S)

mailbox.default.session.mail.<PROTOCOL>.timeout

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. Determines the I/O timeout in milliseconds.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 120 for IMAP(S) and 3000 for POP3(S)
- **Since**: 6.4.0 (IMAP and POP3), 6.11.2.0 (IMAPS and POP3S)

mailbox.default.task.delete.read.messages

- **Module**: cmas-nimh
- **Description**: Defines whether messages should be removed from the mailbox after processing. For the IMAP protocol messages are marked as SEEN by default. For the POP3 protocol, this property is ignored and the emails are always deleted from the email server after they were processed successfully.
- **Type**: boolean
- **Restart required**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.4.0, changed in 6.12.1.0

mailbox.default.task.enabled

- **Module**: cmas-nimh
- **Description**: With this property, the service thread related to the given poller can be disabled.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.4.0

mailbox.default.task.interval.seconds

- **Module**: cmas-nimh
- **Description**: Default interval (in seconds) for polling mailboxes.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.4.0

mailbox.default.task.max.message.size

- **Module**: cmas-nimh
- **Description**: Maximum size of email messages (i.e., email plus attachment). Emails exceeding the size limit will not be automatically processed by NIMH but will be stored in the database (table `cmas_nimh_archived_mail`) and will therefore appear in the email backups in the Admin Tool / Web Admin Suite. From there they can be resent, downloaded to the file system, or deleted. For those operations the message size is not relevant.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 10485760 (default value, 10 MB)
- **Since**: 6.4.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

mailbox.default.task.max.messages.per.run

- **Module**: cmas-nimh
- **Description**: Number of messages fetched at once from the mailbox. Must be correlated with the transaction timeout. The default value is 20.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60
- **Since**: 6.4.0

mailbox.default.task.timeout.seconds

- **Module**: cmas-nimh
- **Description**: After this time of inactivity the service thread is considered damaged and automatically restarted. The default value is 120 seconds.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60
- **Since**: 6.4.0

mailbox.default.task.transaction.timeout.seconds

- **Module**: cmas-nimh
- **Description**: Default transaction timeout (in seconds) for email fetching transactions. Should be correlated with number of messages fetched at once.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.4.0

mailbox.polling.threads.mail.log.enabled

- **Module**: cmas-nimh
- **Description**: Enables email logging which is especially crucial in cluster environments (used as semaphore there).
- **Type**: boolean
- **Restart required**: no
- **System**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.9.4.1

mailbox.polling.threads.number

- **Module**: cmas-nimh
- **Description**: Number of threads for accessing mailboxes.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 1 (default value)
- **Since**: 6.4.0

mailbox.polling.unstorable.backup.folder

- **Module**: cmas-nimh
- **Description**: Folder on the file system which is used to store emails whose processing failed and which could not be stored in the database either. The value of the property should be the path to the data directory followed by `/mail/unstorable`.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: c:/cmas/mail/unstorable
- **Since**: 6.11.2.7

queue.polling.threads.number

- **Module**: cmas-nimh
- **Description**: Number of threads started for polling the email queue in the database.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 1 (default value)
- **Since**: 6.4.0

queue.polling.threads.shutdown.timeout.seconds

- **Module**: cmas-nimh
- **Description**: Waiting time after the shutdown signal for the email poller. When the timeout is reached, the thread will be terminated.
- **Type**: integer

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.4.0

queue.polling.threads.watchdog.interval.seconds

- **Module**: cmas-nimh
- **Description**: Determines the interval in seconds to execute the watchdog which checks the activity of the threads of the email queue poller.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 30 (default value)
- **Since**: 6.4.0

queue.task.error.pause.seconds

- **Module**: cmas-nimh
- **Description**: Maximum number of seconds, the queue poller waits after infrastructure (e.g. database) error.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 180 (default value)
- **Since**: 6.4.0

queue.task.interval.seconds

- **Module**: cmas-nimh
- **Description**: Determines the interval in seconds for the main email queue polling thread.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 15 (default value)
- **Since**: 6.4.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

queue.task.max.retries

- **Module**: cmas-nimh
- **Description**: Maximum number of email processing retries after an exception. When reached, the email is moved to the email archive. This email can be rescheduled again using the NIMH API or the Admin Tool / Web Admin Suite.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 10
- **Since**: 6.4.0

queue.task.timeout.seconds

- **Module**: cmas-nimh
- **Description**: After this time of inactivity (in seconds), the service thread is considered damaged and automatically restarted.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 600 (default value)
- **Since**: 6.4.0

queue.task.transaction.timeout.seconds

- **Module**: cmas-nimh
- **Description**: Transaction timeout (in seconds) for email processing in the pipe.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.4.0

mail.attachments.validation.info.sender

- **Module**: cmas-nimh-extension
- **Description**: Sets the From header for error notification emails which are sent when the validation of the attachment type fails.
- **Type**: string
- **Restart required**: no
- **System**: yes

- **Optional**: no
- **Example value**: admin@mail.com
- **Since**: 6.7.5

mail.attachments.validation.info.subject

- **Module**: cmas-nimh-extension
- **Description**: Sets the subject for error notification emails which are sent when the validation of the attachment type fails.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: Email was not processed because its attachments were rejected!
- **Since**: 6.7.5

mail.db.archive

- **Module**: cmas-nimh-extension
- **Description**: If property is set to “true”, incoming emails are automatically archived in the database.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.8.5.5

mail.error.from.address

- **Module**: cmas-nimh-extension
- **Description**: Sets the From address for error notification emails which are sent when the email processing fails.
- **Type**: email
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Since**: 6.4.0

mail.error.to.address

- **Module**: cmas-nimh-extension
- **Description**: Sets the To address for error notification emails which are sent when the email processing fails. As a default the value of the property `admin.email` is used.

- **Type**: email
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Since**: 6.4.0

mail.on.error

- **Module**: cmas-nimh-extension
- **Description**: If set to “true”, an error notification email is sent to the configured address if an email could not be processed.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default)
- **Since**: 6.4.0

mail.ticketname.pattern

- **Module**: cmas-nimh-extension
- **Description**: Regular expression pattern used to identify the ticket name in the subject of incoming mails.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: .*?Ticket\s+\((\S+)\).*
- **Since**: 6.4.0
- **Removed in**: 6.12.0.5

Attachments for Incoming Emails

attachment.allowed.types

- **Module**: cmas-core-server
- **Description**: Comma-separated list of allowed file name extensions. If the property does not exist, all file extensions are allowed. If the property exists, but no value is provided, all file extension are blocked.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes

- **Example value**: txt,zip,doc
- **Since**: 6.5.0

attachment.max.size

- **Module**: cmas-core-server
- **Description**: Maximum attachment size, in MB. This property controls the size of attachments at tickets, customers, and resources. It also controls the size of incoming (not outgoing) email attachments. The value of this property needs to be aligned with the respective setting in the application server configuration (add the parameter `max-post-size` to the `http listener` in `cm6.xml`, `cm6-cmrf.xml` or `domain.xml`). In addition you might need to increase the value of `cmas-nimh`, `mailbox.default.task.max.message.size`. The value of this property is displayed in the Web Client when adding a new attachment.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 100 (default value)
- **Since**: 6.4.0

Email Encryption (Outgoing and Incoming)

These settings only apply if email encryption is active (true).

mail.encryption

- **Module**: cmas-core-server
- **Description**: If the property is set to “true”, the encryption checkbox in the email editor of the Web Client is checked by default. The default value of the property is “false”.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.8.4.0

In case certificates are stored in an LDAP directory, the following settings have to be made:

ldap.certificate.basedn

- **Module**: cmas-core-server
- **Description**: Base DN for the certificate location in the LDAP tree. If not provided, the value of the property `cmas-core-security, ldap.basedn` is used.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: ou=accounts,dc=consol,dc=de
- **Since**: 6.8.4

ldap.certificate.content.attribute

- **Module**: cmas-core-server
- **Description**: Name of the LDAP attribute where the certificate data is stored in the LDAP tree.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: usercertificate (default value)
- **Since**: 6.8.4

ldap.certificate.password

- **Module**: cmas-core-server
- **Description**: Password of the LDAP certificate manager. If not set, the value of the property `cmas-core-security, ldap.password` is used.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.8.4

ldap.certificate.providerurl

- **Module**: cmas-core-server
- **Description**: URL of the LDAP certificate provider. If not set, the value of the property `cmas-core-security, ldap.providerurl` is used.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: ldap://ldap.consol.de:389
- **Since**: 6.8.4

ldap.certificate.searchattr

- **Module**: cmas-core-server
- **Description**: LDAP attribute name used to search for certificates in the LDAP tree.
- **Type**: string
- **Restart required**: no
- **System**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Optional**: yes
- **Example value**: mail (default value)
- **Since**: 6.8.4

ldap.certificate.userdn

- **Module**: cmas-core-server
- **Description**: DN of the LDAP certificate manager. If not set, the value of the property `cmas-core-security, ldap.userdn` is used.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.8.4

H.2.2.5 Activity Interval Configuration

admin.tool.session.check.interval

- **Module**: cmas-app-admin-tool
- **Description**: Configures the time interval (in seconds) in which the system checks for inactive (ended) Admin Tool sessions.
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: 30
- **Since**: 6.7.5

server.session.timeout

- **Module**: cmas-core-server
- **Description**: Server session timeout (in seconds) for connected clients (database table `cmas_user_session`). Each client can overwrite this timeout with custom value using its ID (ADMIN_TOOL, WEB_CLIENT, WORKFLOW_EDITOR, TRACK, ETL, REST) appended to property name, e.g., `server.session.timeout.ADMIN_TOOL`.
  Please see also the page customization attributes *updateTimeServerSessionActivityEnabled* and *updateTimeServerSessionActivity*, both of type *cmApplicationCustomization*.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1800
- **Since**: 6.6.1, 6.7.1

Detailed explanation for the Admin Tool:

- server.session.timeout.ADMIN_TOOL
  Defines the time interval how long the server considers a session valid while there is no activity from the Admin Tool holding the session. The Admin Tool is not aware of this value, it only suffers having an invalid session, if the last activity has been longer in the past.
- admin.tool.session.check.interval
  Defines the time between two checks done by the Admin Tool, if the server still considers its session valid.

For example, if admin.tool.session.check.interval = 60, the Admin Tool queries the server every minute if its session is still active/valid. In case server.session.timeout.ADMIN_TOOL = 600 the Admin Tool will get the response that the session is now invalid after ten minutes of inactivity.

H.2.2.6 Administrator Email Addresses

ConSol CM can use different administrator email addresses, depending on the subsystem. Please see Administrator and Notification Email Addresses for detailed explanations concerning admin email addresses. If no specific admin email addresses are configured, the global admin email address (that you have defined during system set-up) is used.

H.2.2.7 HTTP Header Configuration

It is possible to configure the HTTP response header returned with the web page in the Web Client. This allows, for example, to define security-related response headers according to the requirements dictated by policy or environment. These properties are managed in the module `cmweb-server-http-headers`. The name of the property is the field of the response header and the value of the property is the field value sent in the header.

> Both name and value must match the exact spelling of the HTTP specification for the desired response header field. The correct interpretation and handling of the header lies in the responsibility of the client browser.

Example Property: X-Frame-Options

- **Module**: cmweb-server-http-headers
- **Description**: Example property to illustrate the configuration of HTTP headers. In this case the delivered HTTP header contains the field *X-Frame-Options* with the value “SAMEORIGIN”.

  Each property in the module `cmweb-server-http-headers` represents one header field. The property name/key identifies the response header field and the value of the property is the field value sent in this header.

  > Please be aware that additional HTTP response headers must be correctly defined with the exact spelling as officially specified! Please note also that the correct interpretation and application of these headers is fully in the realm and responsibility of the client browser which requested the page!

- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes

- **Example value**: SAMEORIGIN
- **Since**: 6.10.8

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

H.2.3 List of System Properties by Module

This chapter lists the system properties included in the following modules.

- cmas-analytics (module)
- cmas-app-admin-tool (module)
- cmas-archive-core-server (module)
- cmas-auth-server (module)
- cmas-core-cache (module)
- cmas-core-index-common (module)
- cmas-core-security (module)
- cmas-core-server (module)
- cmas-core-shared (module)
- cmas-dwh-server (module)
- cmas-nimh (module)
- cmas-nimh-extension (module)
- cmas-restapi-core (module)
- cmas-restapi-http-headers (module)
- cmas-setup-hibernate (module)
- cmas-setup-manager (module)
- cmas-setup-scene (module)
- cmas-web-notifications (module)
- cmas-workflow-engine (module)
- cmas-workflow-jbpm (module)
- cmweb-server-http-headers (module)
- cmweb-server-adapter (module)

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

H.2.3.1 cmas-analytics (module)

metabase.secret.key

- **Module**: cmas-analytics
- **Description**: Determines the string used to generate the private key used to validate Metabase JWT messages. Needs to match the setting *String used by the JWT signing key* in the Metabase configuration.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 50e2230c98344f183c996e6a3954c1a1e43ed100cc36f4ea093800aa2d8a1075
- **Since**: 6.13.0.0

metabase.url

- **Module**: cmas-analytics
- **Description**: Determines the URL where the Metabase application runs. Needs to match the setting *Site URL* in the Metabase configuration.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: http://localhost:3000 (default value)
- **Since**: 6.13.0.0

H.2.3.2 cmas-app-admin-tool (module)

admin.tool.consumed.licences.check.interval

- **Module**: cmas-app-admin-tool
- **Description**: Sets the interval (in seconds) to monitor the number of consumed licenses.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 30 (default value)
- **Since**: 6.11.0.0

admin.tool.consumed.licences.pool.name

- **Module**: cmas-app-admin-tool
- **Description**: Sets the license pool name to monitor the number of consumed licenses.
- **Type**: string

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: CONCURRENT_USERS (default value)
- **Since**: 6.11.0.0

admin.tool.session.check.interval

- **Module**: cmas-app-admin-tool
- **Description**: Configures the time interval (in seconds) in which the system checks for inactive (ended) Admin Tool sessions.
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: 30
- **Since**: 6.7.5

autocomplete.enabled

- **Module**: cmas-app-admin-tool
- **Description**: Controls if the *Autocomplete address* navigation item is displayed in the Admin Tool. It is hidden, if this system property is missing or its value is “false”.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: true
- **Since**: 6.9.2.0

delete.ticket.enabled

- **Module**: cmas-app-admin-tool
- **Description**: Controls if tickets can be deleted in the ticket administration of the Admin Tool. The default value is “true”.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: true
- **Since**: 6.9.4.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

dwh.administration.refresh.interval.seconds

- **Module**: cmas-app-admin-tool
- **Description**: Internal DWH property, not to be changed manually. This property is used to set the interval for refreshing the list of actions in the Admin Tool, section *Data Warehouse -> Administration -> Actions*.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 10
- **Since**: 6.11.0.1

script.validation.interval.seconds

- **Module**: cmas-app-admin-tool
- **Description**: Sets the interval in seconds between two code checks in the script editor of the Admin Tool and Process Designer.
- **Type**: Integer
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 1 (default value)
- **Since**: 6.11.0.1

start.groovy.task.enabled

- **Module**: cmas-app-admin-tool
- **Description**: Determines if it is possible to execute tasks in the Admin Tool, section *Services -> Task Execution*. Set this system property to “true” to enable task execution.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true
- **Since**: 6.9.4.0

task.panel.refresh.interval.seconds

- **Module**: cmas-app-admin-tool
- **Description**: Sets the time in seconds after which the task list in the Admin Tool is refreshed.
- **Type**: Integer
- **Restart required**: no
- **System**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Optional**: no
- **Example value**: 10
- **Since**: 6.10.5.3 (not added automatically during update from versions prior to 6.10.5.3!)

H.2.3.3 cmas-archive-core-server (module)

archive.enabled

- **Module**: cmas-archive-core-server
- **Description**: Determines if GDPR-compliant deletions or anonymizations, which delete ticket or contact data from the ConSol CM database, are applied to CM/Archive as well. By default, this property is set to “false”, meaning that no data is deleted from CM/Archive. If you set this property to “true”, GDPR-compliant deletions or anonymizations are also performed in CM/Archive. This affects both already archived tickets and newly archived tickets.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.11.2.5

archive.uri

- **Module**: cmas-archive-core-server
- **Description**: Sets the URL where the CM/Archive application is running.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: <server_url>:8090
- **Since**: 6.11.2.0

H.2.3.4 cmas-auth-server (module)

access.token.signing.key

- **Module**: cmas-auth-server
- **Description**: Determines the secret shared between the authorization server and all client applications using OAuth2, e.g. CM/Archive. The value of this property needs to match the value of *archive.oauth2.access.token.signing.key* in the configuration file of CM/Archive.
- **Type**: password
- **Restart required**: yes
- **System**: no
- **Optional**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: my_password
- **Since**: 6.11.2.0

client.archive.access.token.validity.seconds

- **Module**: cmas-auth-server
- **Description**: Determines the validity period in seconds of the access token required for using CM/Archive.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 43200 (default value)
- **Since**: 6.11.2.0

client.archive.enabled

- **Module**: cmas-auth-server
- **Description**: Determines if CM/Archive is enabled (“true”) or disabled (“false”).
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: false
- **Since**: 6.11.2.0
- **Removed in**: 6.11.2.2

client.archive.refresh.token.validity.seconds

- **Module**: cmas-auth-server
- **Description**: Determines the validity period in seconds of the refresh token required for obtaining new access tokens for CM/Archive.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 2592000 (default value)
- **Since**: 6.11.2.0

client.archive.secret

- **Module**: cmas-auth-server
- **Description**: Determines the secret shared between the authorization server and CM/Archive. The value of the property needs to match the value of *archive.oauth2.client.secret* in the configuration file of CM/Archive.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Type**: password
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: my_password
- **Since**: 6.11.2.0

client.was.access.token.validity.seconds

- **Module**: cmas-auth-server
- **Description**: Determines the validity period in seconds of the access token required for using the Web Admin Suite.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 43200 (default value)
- **Since**: 6.12.0.0

client.was.refresh.token.validity.seconds

- **Module**: cmas-auth-server
- **Description**: Determines the validity period in seconds of the refresh token required for obtaining new access tokens for the Web Admin Suite.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 2592000 (default value)
- **Since**: 6.12.0.0

client.was.secret

- **Module**: cmas-auth-server
- **Description**: Determines the secret shared between the authorization server and the Web Admin Suite.
- **Type**: password
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: my_password
- **Since**: 6.12.0.0
- **Removed in**: 6.12.0.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

H.2.3.5 cmas-core-cache (module)

cache-cluster-name

- **Module**: cmas-core-cache
- **Description**: Unique name of a cluster consisting of several CM nodes. This value is used to identify the nodes belonging to the same cluster during network communication. Please do not change this value manually unless advised by ConSol.
- **Type**: string
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: 635a6de1-629a-4129-8299-2d98633310f0
- **Since**: 6.4.0

cache.logging.listener.enabled

- **Module**: cmas-core-cache
- **Description**: Determines whether cache events are logged to `infinispan.log` for the standalone mode and `jgroups-infinispan.log` for the cluster mode.
- **Type**: boolean
- **Restart required**: yes
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.12.0.5

eviction.event.queue.size

- **Module**: cmas-core-cache
- **Description**: The size of the queue holding cache events. The default value is 200000. It is recommended to increase the value slightly (up to 400000) on systems with high traffic or load. [Deprecated - not used for infinispan]
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: 200000 (default value)
- **Since**: 6.4.0

eviction.lifeSpan

- **Module**: cmas-core-cache
- **Description**: Sets the interval (in milliseconds) for cache entry validity. When this time is elapsed, the entry is removed from the cache.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Type**: integer
- **Restart required**: yes
- **System**: no
- **Optional**: yes
- **Example value**: 86400000 (default value)
- **Since**: 6.11.1.0

eviction.max.nodes

- **Module**: cmas-core-cache
- **Description**: Sets the maximum size of internal caches. The default value is 100000. Increasing it will lead to higher memory consumption and is not recommended unless explicitly advised by ConSol.
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: 100000 (default value)
- **Since**: 6.4.0

eviction.strategy

- **Module**: cmas-core-cache
- **Description**: Determines the cache eviction strategy. Please see the Infinispan documentation for details.
- **Type**: string
- **Restart required**: yes
- **System**: no
- **Optional**: yes
- **Example value**: LRU, LIRS (default value)
- **Since**: 6.11.1.0

eviction.wakeup.interval

- **Module**: cmas-core-cache
- **Description**: Sets the interval (in milliseconds) between two cache queue event processing cycles. The default value is 3000. It is recommended to decrease it (minimum is 1500) on systems with high traffic or load.
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: no

- **Example value**: 3000 (default value)
- **Since**: 6.4.0

H.2.3.6 cmas-core-index-common (module)

big.task.minimum.size

- **Module**: cmas-core-index-common
- **Description**: Indicates the minimum size of index task (in parts, each part has 100 entities) to qualify this task as a big one. Big tasks have a lower priority than runtime tasks.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 15 (default value)
- **Since**: 6.8.3

database.notification.enabled

- **Module**: cmas-core-index-common
- **Description**: Indicates whether the database should be used as a notification channel for index updates instead of JMS. The default value is “true.”
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.8.4.7
- **Removed in**: 6.12.0.0

database.notification.redelivery.delay.seconds

- **Module**: cmas-core-index-common
- **Description**: If the database is used as a notification channel for index updates, this property indicates the delay for notification redelivery when an exception occurs.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 60
- **Since**: 6.8.4.7

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

database.notification.redelivery.max.attempts

- **Module**: cmas-core-index-common
- **Description**: If the database is used as a notification channel for index updates, this property indicates maximum redelivery attempts when an exception occurs.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 60
- **Since**: 6.8.4.7

disable.admin.task.auto.commit

- **Module**: cmas-core-index-common
- **Description**: Toggle for the automatic execution of administrative index update tasks. If set to “false”, possibly expensive index tasks created as a result of configuration changes will be automatically executed right after creation. It is recommended to set this value to “true”, so that the execution of administrative index tasks needs to be manually triggered in the Admin Tool or Web Admin Suite.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.6.1

index.attachment

- **Module**: cmas-core-index-common
- **Description**: Determines whether the content of attachments is indexed.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.4.3

index.history

- **Module**: cmas-core-index-common
- **Description**: Determines whether ticket and customer history is indexed.
- **Type**: boolean
- **Restart required**: no
- **System**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Optional**: no
- **Example value**: false
- **Since**: 6.1.0
- **Removed in**: 6.11.0

index.status

- **Module**: cmas-core-index-common
- **Description**: Index synchronization status, possible values are GREEN, YELLOW or RED. This value is set internally by the index system, please do not change it manually unless advised by ConSol.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: GREEN
- **Since**: 6.6.1

index.task.worker.threads

- **Module**: cmas-core-index-common
- **Description**: Sets the number of threads that will be used to execute index tasks. The default value is 1, we recommend to increase it slightly on systems with large indexer load up to a value of 4.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1 (default value)
- **Since**: 6.6.14, 6.7.3. Since 6.8.0 and exclusively in 6.6.21 also normal (live) index updates are affected by this property.

index.version.current

- **Module**: cmas-core-index-common
- **Description**: Holds information about the current index version. This value is maintained internally, please do not change it unless advised by ConSol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1 (default value)
- **Since**: 6.7.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

index.version.newest

- **Module**: cmas-core-index-common
- **Description**: Holds information about which index version is considered newest. This value is maintained internally, please do not change it unless advised by ConSol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1 (default value)
- **Since**: 6.7.0

indexed.assets.per.thread.in.memory

- **Module**: cmas-core-index-common
- **Description**: Determines how many assets should be loaded into memory at once, per thread, during indexing.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 200 (default value)
- **Since**: 6.8.0

indexed.engineers.per.thread.in.memory

- **Module**: cmas-core-index-common
- **Description**: Determines how many engineers should be loaded into memory at once, per thread, during indexing. Increasing it slightly will lead to better indexer performance at the price of higher memory consumption.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 300 (default value)
- **Since**: 6.6.14, 6.7.3

indexed.resources.per.thread.in.memory

- **Module**: cmas-core-index-common
- **Description**: Determines how many resources should be loaded into memory at once, per thread, during indexing. Increasing it slightly will lead to better indexer performance at the price of higher memory consumption.
- **Type**: integer
- **Restart required**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **System**: yes
- **Optional**: no
- **Example value**: 200 (default value)
- **Since**: 6.10.0.0

indexed.tickets.per.thread.in.memory

- **Module**: cmas-core-index-common
- **Description**: Determines how many tickets should be loaded into memory at once, per thread, during indexing. Increasing it slightly will lead to better indexer performance at the price of higher memory consumption.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 100 (default value)
- **Since**: 6.6.14, 6.7.3

indexed.units.per.thread.in.memory

- **Module**: cmas-core-index-common
- **Description**: Determines how many customers should be loaded into memory at once, per thread, during indexing. Increasing it slightly will lead to better indexer performance at the price of higher memory consumption.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 200 (default value)
- **Since**: 6.6.14, 6.7.3

synchronize.archive.timeout.minutes

- **Module**: cmas-core-index-common
- **Description**: Determines the time period in minutes during which index updates are archived. The default value of this property is “60” for clustered environments and “0” for non-clustered environments. If the master node is switched, the new master can recover missing data in case it had not been fully synchronized with the previous master indexing server at the moment of switch.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no

- **Example value**: 60
- **Since**: 6.12.0.0

synchronize.master.address

- **Module**: cmas-core-index-common
- **Description**: Value of `-Dcmas.http.host.port` specifying how to connect to the indexing master server. Since 6.12.0.0 it is possible to switch the node acting as master node by entering the address of the new master node. The index updates are archived for the time configured in the property `cmas-core-index-common`, `synchronize.archive.timeout.minutes`, so the new master can recover missing data in case it had not been fully synchronized with the previous master indexing server at the moment of switch.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 127.0.0.1:80
- **Since**: 6.6.0

synchronize.master.security.token

- **Module**: cmas-core-index-common
- **Description**: The password for accessing the index snapshot via URL, e.g., for index synchronization or for backups. This value is negotiated between nodes internally, please do not change it manually unless advised by ConSol.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: token
- **Since**: 6.6.0

synchronize.master.security.user

- **Module**: cmas-core-index-common
- **Description**: The user name for accessing the index snapshot via URL, e.g., for index synchronization or for backups. This value is negotiated between nodes internally, please do not change it manually unless advised by ConSol.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: user
- **Since**: 6.6.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

synchronize.master.timeout.minutes

- **Module**: cmas-core-index-common
- **Description**: Determines how much time (in minutes) the master server may continually fail until a new master gets elected. 0 means that master server will never change (failover mechanism is disabled).
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5 (default value)
- **Since**: 6.6.0

synchronize.megabits.per.second

- **Module**: cmas-core-index-common
- **Description**: Determines how much bandwidth (in megabits) the master server may consume when transferring index changes to all slave servers. Do not use all available bandwidth to transfer index changes between hosts, as doing so will most probably partition the cluster due to some subsystems being unable to communicate.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 85 (default value)
- **Since**: 6.6.0

synchronize.sleep.millis

- **Module**: cmas-core-index-common
- **Description**: Determines how often each slave server polls the master server for index changes.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1000 (default value)
- **Since**: 6.6.0

H.2.3.7 cmas-core-security (module)

admin.email

- **Module**: cmas-core-security
- **Description**: The email address of the ConSol CM administrator. The value which you entered during system setup is used initially.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Since**: 6.0

admin.login

- **Module**: cmas-core-security
- **Description**: The login name of the ConSol CM administrator. The value which you entered during system setup is used initially.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: admin
- **Since**: 6.0

authentication.method

- **Module**: cmas-core-security
- **Description**: Sets the user authentication method (internal database or LDAP authentication). Allowed values are LDAP or DATABASE.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: DATABASE
- **Since**: 6.0

contact.authentication.method

- **Module**: cmas-core-security
- **Description**: Sets the authentication method for contacts accessing REST services, possible values are DATABASE or LDAP or LDAP,DATABASE or DATABASE,LDAP.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: LDAP
- **Since**: 6.9.3.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

contact.inherit.permissions.only.to.own.customer.group

- **Module**: cmas-core-security
- **Description**: Determines whether an authenticated contact via REST inherits all customer group permissions from the CM/Track user profile (“false”) or only has permissions to his own customer group (“true”).
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.9.2.3

kerberos.v5.enabled

- **Module**: cmas-core-security
- **Description**: Indicates whether user authentication using SSO via Kerberos is enabled.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false (default value if Kerberos was not enabled during system setup)
- **Since**: 6.2.0

kerberos.v5.username.regex

- **Module**: cmas-core-security
- **Description**: Regular expression used for mapping Kerberos principals to CM user login names.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: (.*)@.*
- **Since**: 6.2.0

ldap.authentication

- **Module**: cmas-core-security
- **Description**: LDAP authentication method used when using LDAP authentication. Possible values are “anonymous” and “simple”.
- **Type**: string
- **Restart required**: yes
- **System**: yes
- **Optional**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: simple (default value)
- **Since**: 6.0

ldap.basedn

- **Module**: cmas-core-security
- **Description**: Base DN used for looking up LDAP user accounts when using LDAP authentication.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: ou=accounts,dc=consol,dc=de
- **Since**: 6.0

ldap.contact.name.basedn

- **Module**: cmas-core-security
- **Description**: Base path to search for the contact DN by LDAP ID.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: ou=accounts,dc=consol,dc=de
- **Since**: 6.9.3.0

ldap.contact.name.password

- **Module**: cmas-core-security
- **Description**: Password to look up the contact DN by LDAP ID. If not set, the anonymous account is used.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Since**: 6.9.3.0

ldap.contact.name.providerurl

- **Module**: cmas-core-security
- **Description**: Address of the LDAP server (ldap[s]://host:port) used for contact authentication.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: ldap://ldap.consol.de:389
- **Since**: 6.9.3.0

ldap.contact.name.searchattr

- **Module**: cmas-core-security
- **Description**: Attribute to search for the contact DN by LDAP ID (e.g. uid).
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: uid
- **Since**: 6.9.3.0

ldap.contact.name.userdn

- **Module**: cmas-core-security
- **Description**: User DN to look up contact DN by LDAP ID. If not set, the anonymous account is used.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Since**: 6.9.3.0

ldap.initialcontextfactory

- **Module**: cmas-core-security
- **Description**: Class name for the initial context factory of the LDAP implementation when using LDAP authentication. Usually this is `com.sun.jndi.ldap.LdapCtxFactory`.
- **Type**: string
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: com.sun.jndi.ldap.LdapCtxFactory
- **Since**: 6.0

ldap.password

- **Module**: cmas-core-security
- **Description**: Password for connecting to LDAP to look up users when using LDAP authentication. Only needed if lookup cannot be performed anonymously.
- **Type**: password
- **Restart required**: no
- **System**: yes

- **Optional**: yes
- **Since**: 6.1.2

ldap.providerurl

- **Module**: cmas-core-security
- **Description**: URL to LDAP provider, when using LDAP authentication.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: ldap://myserver.consol.de:389
- **Since**: 6.0

ldap.searchattr

- **Module**: cmas-core-security
- **Description**: Search attribute for looking up the LDAP entries associated with a CM login.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: uid
- **Since**: 6.0

ldap.userdn

- **Module**: cmas-core-security
- **Description**: LDAP user for connecting to LDAP to look up users when using LDAP authentication. Only needed if the lookup cannot be performed anonymously.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.1.2

password.reset.mail.from

- **Module**: cmas-core-security
- **Description**: Defines the from address (sender) of emails for resetting the password. If not set, the value of the `admin.email` property is used instead.
- **Type**: String
- **Restart required**: no
- **System**: no

- **Optional**: no
- **Example value**: mypwreset@consol.de
- **Since**: 6.11.0.1

policy.password.age

- **Module**: cmas-core-security
- **Description**: Defines the maximum validity period (in days) of passwords before the user needs to change the password. Example “183” (6 months), default value: “5500” (= 15 years, i.e., no password change enforced).
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 5500 (15 years, default value)
- **Since**: 6.10.1.0

policy.password.pattern

- **Module**: cmas-core-security
- **Description**: Defines a regular expression pattern which new passwords have to match. The default value is ^.{3,}$ (at least 3 characters).
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: ^.{3,}$ (default value)
- **Since**: 6.10.1.0

policy.rotation.ratio

- **Module**: cmas-core-security
- **Description**: Defines how often passwords may repeat. For example, setting the value to X means that the new password cannot be present among the user’s X previous passwords.

  Since 6.11.2.8, this property is also taken into account for CM/Track. Nevertheless, any value greater than 1 is interpreted as 1, so that it is only possible to compare the new password to the current one.

  You can set the property to 0 to disable password rotation checks in CM/Track.

- **Type**: integer
- **Restart required**: no
- **System**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Optional**: yes
- **Example value**: 1 (default value)
- **Since**: 6.10.1.0, changed in 6.11.2.8

policy.username.case.sensitive

- **Module**: cmas-core-security
- **Description**: Defines whether user names are case-sensitive.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.10.1.0

policy.track.username.case.sensitive

- **Module**: cmas-core-security
- **Description**: Defines whether customer user names in CM/Track case-sensitive.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.11.0.0

resetCode.expirationPeriod

- **Module**: cmas-core-security
- **Description**: Defines the expiration period for the link when resetting the password in CM/Track.
- **Type**: Integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 86400000 (default value, 24 hours)
- **Since**: 6.10.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

H.2.3.8 cmas-core-server (module)

attachment.allowed.types

- **Module**: cmas-core-server
- **Description**: Comma-separated list of allowed file name extensions. If the property does not exist, all file extensions are allowed. If the property exists, but no value is provided, all file extension are blocked.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: txt,zip,doc
- **Since**: 6.5.0

attachment.max.size

- **Module**: cmas-core-server
- **Description**: Maximum attachment size, in MB. This property controls the size of attachments at tickets, customers, and resources. It also controls the size of incoming (not outgoing) email attachments. The value of this property needs to be aligned with the respective setting in the application server configuration (add the parameter `max-post-size` to the `http listener` in `cm6.xml`, `cm6-cmrf.xml` or `domain.xml`). In addition you might need to increase the value of `cmas-nimh`, `mailbox.default.task.max.message.size`. The value of this property is displayed in the Web Client when adding a new attachment.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 100 (default value)
- **Since**: 6.4.0

attachment.type.hints

- **Module**: cmas-core-server
- **Description**: Allows you to assign MIME types to not yet officially supported file extensions, so these file extensions can be detected correctly.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 7z:application/x-7z-compressed, zip:application/zip, wav:audio/x-wav
- **Since**: 6.11.2.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

calendar.csv.dateFormat

- **Module**: cmas-core-server
- **Description**: Format of the date given in the CSV file containing the list of holidays.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: dd/MM/yyyy
- **Since**: 6.9.3.2

calendar.csv.separator

- **Module**: cmas-core-server
- **Description**: Separator used in the CSV file containing the list of holidays.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: ,
- **Since**: 6.9.3.2

config.data.version

- **Module**: cmas-core-server
- **Description**: The internal version number of the current system configuration. This property is maintained internally, please do not change it unless advised by ConSol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 11
- **Since**: 6.0

config.import.global.transaction.enabled

- **Module**: cmas-core-server
- **Description**: Determines whether the configuration (without localizations) should be imported within a single transaction.
- **Type**: Boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: true
- **Since**: 6.11.1.0

customfield.content.file.max.size

- **Module**: cmas-core-server
- **Description**: Determines the maximum size in MB of the images which can be added as content of rich text fields.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 10
- **Since**: 6.11.2.2

dao.log.threshold.milliseconds

- **Module**: cmas-core-server
- **Description**: Used to configure database operation times logging. DAO methods whose execution takes longer than the time set in this property (in milliseconds) are logged.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 500 (default value)
- **Since**: 6.11.1.0

dao.log.username

- **Module**: cmas-core-server
- **Description**: Used to configure database operation times logging. The execution of DAO methods which are related to the user name stated in this property is logged. Only one user name can be provided. The value is empty by default.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: webadmin
- **Since**: 6.11.1.0

defaultCommentClassName

- **Module**: cmas-core-server
- **Description**: Name of the default text class for comments.
- **Type**: string

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: default_class
- **Since**: 6.3.0

defaultIncommingMailClassName

- **Module**: cmas-core-server
- **Description**: Name of the default text class for incoming emails.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: default_class
- **Since**: 6.3.0

defaultOutgoingMailClassName

- **Module**: cmas-core-server
- **Description**: Name of the default text class for outgoing emails.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: default_class
- **Since**: 6.3.0

engineer.description.cache.enabled

- **Module**: cmas-core-server
- **Description**: Defines whether user descriptions are cached. The default value is “true”, please do not change it unless advised by ConSol.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.11.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

engineer.description.mode

- **Module**: cmas-core-server
- **Description**: Defines whether the user names in the ticket history are taken from the database or dynamically rendered using templates. The default value “DYNAMIC” is a bit more costly from the performance perspective, while “PROTOCOL” is faster but returns historical names which might be outdated. Use “PROTOCOL” if you have lots of history entries from many different users.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: PROTOCOL
- **Since**: 6.11.0

engineer.description.template.name

- **Module**: cmas-core-server
- **Description**: Defines the name of the template which is used to render engineer names for display in the Web Client. The template has to be stored in the *Templates* section of the Admin Tool / Web Admin Suite.
- **Type**: String
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: engineer description template name (default value)
- **Since**: 6.11.0

external.line.access.prefix

- **Module**: cmas-core-server
- **Description**: General prefix to dial before an area code. Set for each customer group separately.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 411
- **Since**: 6.9.3.0

fetchSize.strategy

- **Module**: cmas-core-server
- **Description**: Strategy for selecting the fetch size on JDBC result sets.
- **Type**: string
- **Restart required**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **System**: yes
- **Optional**: yes
- **Example value**: FetchSizePageBasedStrategy, FetchSizeThresholdStrategy, FetchSizeFixedStrategy
- **Since**: 6.8.4.1

fetchSize.strategy.FetchSizeFixedStrategy.value

- **Module**: cmas-core-server
- **Description**: Sets the fetch size value if the selected strategy to set the fetch size is `FetchSizeFixedStrategy`. Please do not change this value unless advised by ConSol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 150
- **Since**: 6.8.4.1

fetchSize.strategy.FetchSizePageBasedStrategy.limit

- **Module**: cmas-core-server
- **Description**: Sets the maximum fetch size value if the selected strategy to set the fetch size is `FetchSizePageBasedStrategy`. Please do not change this value unless advised by ConSol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 10000
- **Since**: 6.8.4.1

fetchSize.strategy.FetchSizeThresholdStrategy.value

- **Module**: cmas-core-server
- **Description**: Sets the fetch size threshold border values if the selected strategy to set the fetch size is `FetchSizeThresholdStrategy`. Please do not change this value unless advised by ConSol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 150,300,600,1000
- **Since**: 6.8.4.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

heartbeat

- **Module**: cmas-core-server
- **Description**: Timestamp that indicates if an instance of the application is connected to the database schema.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1234567899
- **Since**: 6.10.5.3

internal.line.access.prefix

- **Module**: cmas-core-server
- **Description**: Prefix that the company's telephony system requires for outside lines. Set for each customer group separately.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 199
- **Since**: 6.9.3.0

jmx.full.exposure

- **Module**: cmas-core-server
- **Description**: Controls the availability of methods which can be used to retrieve or delete data from ConSol CM via JMX. The property is set to "false" by default, meaning that these methods cannot be used via JMX. If you set the property to "true", these methods can be used via JMX to retrieve or delete data.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false (default value)
- **Since**: 6.11.2.5



It is strictly recommended to have this property set to "false" on production systems.

last.config.change

- **Module**: cmas-core-server
- **Description**: Random UUID created during the last configuration change. This is a value maintained internally, please do not change it unless advised by ConSol.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 2573c7b7-2bf5-47ff-b5a2-bad31951a266
- **Since**: 6.1.0, 6.2.1

last.config.change.templates

- **Module**: cmas-core-server
- **Description**: Random UUID created during the last change in templates. This is a value maintained internally, please do not change it unless advised by ConSol.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 2573c7c7-2af5-4eff-b9c2-bad31951a266
- **Since**: 6.10.5.0

ldap.certificate.basedn

- **Module**: cmas-core-server
- **Description**: Base DN for the certificate location in the LDAP tree. If not provided, the value of the property `cmas-core-security, ldap.basedn` is used.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: ou=accounts,dc=consol,dc=de
- **Since**: 6.8.4

ldap.certificate.content.attribute

- **Module**: cmas-core-server
- **Description**: Name of the LDAP attribute where the certificate data is stored in the LDAP tree.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes

- **Example value**: usercertificate (default value)
- **Since**: 6.8.4

ldap.certificate.password

- **Module**: cmas-core-server
- **Description**: Password of the LDAP certificate manager. If not set, the value of the property `cmas-core-security, ldap.password` is used.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.8.4

ldap.certificate.providerurl

- **Module**: cmas-core-server
- **Description**: URL of the LDAP certificate provider. If not set, the value of the property `cmas-core-security, ldap.providerurl` is used.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: ldap://ldap.consol.de:389
- **Since**: 6.8.4

ldap.certificate.searchattr

- **Module**: cmas-core-server
- **Description**: LDAP attribute name used to search for certificates in the LDAP tree.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: mail (default value)
- **Since**: 6.8.4

ldap.certificate.userdn

- **Module**: cmas-core-server
- **Description**: DN of the LDAP certificate manager. If not set, the value of the property `cmas-core-security, ldap.userdn` is used.
- **Type**: string
- **Restart required**: no
- **System**: yes

- **Optional**: yes
- **Since**: 6.8.4

local.country.prefix

- **Module**: cmas-core-server
- **Description**: Prefix of the local country code. Set for each customer group separately.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 48
- **Since**: 6.9.3.0

mail.encryption

- **Module**: cmas-core-server
- **Description**: If the property is set to “true”, the encryption checkbox in the email editor of the Web Client is checked by default. The default value of the property is “false”.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.8.4.0

mail.notification.engineerChange

- **Module**: cmas-core-server
- **Description**: Determines whether notification emails should be sent when the assignee of a ticket is changed.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.1.0

mail.notification.sender

- **Module**: cmas-core-server
- **Description**: From address for notification emails when the assignee of a ticket is changed. If not set, the value of the property `cmas-core-security, admin.email` is used instead.
- **Type**: string
- **Restart required**: no

- **System**: yes
- **Optional**: yes
- **Example value**: cm6notification@cm6installation
- **Since**: 6.6.3

mail.redelivery.retry.count

- **Module**: cmas-core-server
- **Description**: Number of redelivery attempts of an outgoing email.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 3
- **Since**: 6.1.0

mail.smtp.email

- **Module**: cmas-core-server
- **Description**: SMTP URL for outgoing emails
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: smtp://mail.mydomain.com:25
- **Since**: 6.0

mail.smtp.email.password

- **Module**: cmas-core-server
- **Description**: SMTP password for outgoing emails. If this property is set, it takes precedence over the value from the SMTP URL.
- **Type**: password
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: password
- **Since**: 6.11.2.2

mail.smtp.email.user

- **Module**: cmas-core-server
- **Description**: SMTP user name for outgoing emails. If this property is set, it takes precedence over the value from the SMTP URL.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: myuser
- **Since**: 6.11.2.2

mail.smtp.envelopesender

- **Module**: cmas-core-server
- **Description**: Email address used as a sender in SMTP envelope. If not set, the From address of the email is used.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: mysender@mydomain.com
- **Since**: 6.5.7

mail.smtp.tls.enabled

- **Module**: cmas-core-server
- **Description**: Activates SMTP via SSL/TLS (SMTPS) for sending emails from the Web Client and scripts. The default value is “false”. If it is set to “true”, SMTPS is activated for sending emails.
- **Type**: boolean
- **Restart required**: yes
- **System**: no
- **Optional**: yes
- **Example value**: true
- **Since**: 6.11.1.6

mailSender.executionInterval.seconds

- **Module**: cmas-core-server
- **Description**: Determines the period in seconds before the system checks if there are new emails to be sent.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 10
- **Since**: 6.12.0.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

mailSender.lockingLimit

- **Module**: cmas-core-server
- **Description**: Determines the number of emails fetched from the database for sending in a single transaction.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 1
- **Since**: 6.12.0.0

mailSender.lockTimeout.seconds

- **Module**: cmas-core-server
- **Description**: Determines the period in seconds before a locked email is released for another lock.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 199
- **Since**: 6.12.0.0

mailSender.maxAttempts

- **Module**: cmas-core-server
- **Description**: Determines the maximum number of attempts, including the first attempt, before an email is removed.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 3
- **Since**: 6.12.0.0

mailSender.nodeId

- **Module**: cmas-core-server
- **Description**: Determines the ID of the node on which the thread for sending emails is started (only in clustered environments). If the value is “all” or no value is provided, the threads are started on every node.
- **Type**: string
- **Restart required**: no
- **System**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Optional**: yes
- **Example value**: all
- **Since**: 6.12.0.0

mailSender.retryInterval.seconds

- **Module**: cmas-core-server
- **Description**: Determines the period in seconds before another attempt to send an email is made, if the previous attempt failed.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60
- **Since**: 6.12.0.0

mailSender.threads.number

- **Module**: cmas-core-server
- **Description**: Determines the number of threads for sending emails.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 1
- **Since**: 6.12.0.0

mailSender.txTimeout.seconds

- **Module**: cmas-core-server
- **Description**: Determines the transaction timeout for sending emails in seconds.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60
- **Since**: 6.12.0.0

max.licences.perUser

- **Module**: cmas-core-server
- **Description**: Sets the maximum number of licenses which a single user can use (e.g., logging in from different browsers). By default this value is not restricted.
- **Type**: integer

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 10
- **Since**: 6.8.4.5

ml.evict.unused.after.hours

- **Module**: cmas-core-server
- **Description**: The number of hours after which a machine learning model is evicted from cache if it is not used. The ConSol CM server checks every hour for machine learning models to evict.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 24 (default value)
- **Since**: 6.13.0.0

ml.learn.worker.threads

- **Module**: cmas-core-server
- **Description**: The number of threads that a machine learning algorithm can use while training its machine learning model.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1 (default value)
- **Since**: 6.13.0.0

monitoring.engineer.login

- **Module**: cmas-core-server
- **Description**: Login for the special user (engineer) used for monitoring.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: nagios
- **Since**: 6.9.3.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

monitoring.unit.login

- **Module**: cmas-core-server
- **Description**: Login for the special user (contact) used for monitoring.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: nagios
- **Since**: 6.9.3.0

nimh.enabled

- **Module**: cmas-core-server
- **Description**: Enables the NIMH service which controls email polling and processing in ConSol CM. Must be suffixed with NODEID in cluster environments e.g. `nimh.enabled.NODEID` = “true”.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false
- **Since**: 6.9.4.0

number.of.tasks

- **Module**: cmas-core-server
- **Description**: Number of threads to use by the Task Execution Framework (TEF).
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 1
- **Since**: 6.9.4.0

recent.items.cleanup.cluster.node.id

- **Module**: cmas-core-server
- **Description**: Value of a `-Dcmas.clusternode.id` designating the node which will clean up recent items.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes

- **Example value**: 1 (assuming the cluster node started with `-Dcmas.clusternode.id=1` parameter)
- **Since**: 6.11.0.1

recent.items.cleanup.interval.minutes

- **Module**: cmas-core-server
- **Description**: Determines the time interval (in minutes) in which recent items should be checked for removal.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 60
- **Since**: 6.11.0.1

recent.items.max.per.engineer

- **Module**: cmas-core-server
- **Description**: Maximum number of preserved recent items per engineer while cleaning up (older recent items will be deleted).
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 500
- **Since**: 6.11.0.1

recent.items.persistence.enabled

- **Module**: cmas-core-server
- **Description**: Enables persistence of recent items. If the property is set to “false”, it prevents storing new recent items. If a recently visited widget (`recentlyVisitedWidget`) is used on a dashboard, this property needs to be set to “true”. Otherwise, you can set it to “false” to save system resources.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.11.1.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

resource.replace.batchSize

- **Module**: cmas-core-server
- **Description**: Defines the number of objects to be processed in a single resource replacement batch.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5
- **Since**: 6.10.0.0

resource.replace.timeout

- **Module**: cmas-core-server
- **Description**: Determines the transaction timeout (in seconds) of a resource replacement action step.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 120
- **Since**: 6.10.0.0

script.evict.unused.after.hours

- **Module**: cmas-core-server
- **Description**: Determines the number of hours for which unused scripts remain in the cache. After this time, the compiled class of the script is removed. The ConSol CM server checks for scripts to evict every hour.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 24 (default)
- **Since**: 6.11.1.14

script.logging.threshold.seconds

- **Module**: cmas-core-server
- **Description**: Determines the time (in seconds) after which a warning is written to the log file if it is exceeded during script execution.
- **Type**: integer
- **Restart required**: no
- **System**: no

- **Optional**: yes
- **Example value**: 10 (default)
- **Since**: 6.10.1.0

serial.mods.tracking.enabled

- **Module**: cmas-core-server
- **Description**: Low level technical flag deciding whether serial diff tracking for entities is enabled. If enabled, there will be no `StackOverflow` error in case a dependency between two entities (for example engineer and ticket) causes an infinite loop first and then as a result, the stack overflow. The property must be added to the configuration manually. It will not be added to a system configuration during setup or update.

  > ⚠ Please do not enable this property unless advised by ConSol. It is a low level technical flag with intricate consequences for the system behavior.

- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.10.7.0, 6.11.0.5

server.instance.task.crash.period.seconds

- **Module**: cmas-core-server
- **Description**: Determines the period (in seconds) for detecting live and crashed ConSol CM server instances.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 15 (default value)
- **Since**: 6.11.1.6

server.instance.task.period.seconds

- **Module**: cmas-core-server
- **Description**: Determines the interval (in seconds) between repeated task executions. The task will be executed repeatedly with the given number of seconds between each execution.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: 5 (default value)
- **Since**: 6.11.1.6

server.session.archive.reaper.interval

- **Module**: cmas-core-server
- **Description**: Determines the interval (in seconds) when the reaper for archived server sessions is executed (refers to the database table `cmas_user_session`).
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 60
- **Since**: 6.7.1

server.session.archive.timeout

- **Module**: cmas-core-server
- **Description**: Server sessions archive validity timeout (in days). After this time session info is removed from the database (refers to the database table `cmas_user_session`).
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 3 or 7
- **Since**: 6.7.1

server.session.heartbeat.enabled.WEB_CLIENT

- **Module**: cmas-core-server
- **Description**: Determines if the session heartbeat is enabled for the Web Client. If this property is set to “true”, a heartbeat is sent to the ConSol CM server every 20 seconds and with every page reload. If the server does not receive a heartbeat during the configured interval, the session is considered orphaned and ended according to the configured mode.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.11.2.9

server.session.heartbeat.mode

- **Module**: cmas-core-server
- **Description**: Determines when orphaned sessions are ended if the session heartbeat is enabled. There are two possible values:
  - **LAZY**: Orphaned sessions are ended when the same user authenticates again or when there are no more licenses available (default value).
  - **EAGER**: Orphaned sessions are ended immediately.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: LAZY (default value)
- **Since**: 6.11.2.9

server.session.heartbeat.timeout.seconds

- **Module**: cmas-core-server
- **Description**: Determines the period after which a session is considered orphaned if the session heartbeat is enabled and no heartbeat is received. The default value is 300, meaning that orphaned sessions are ended after 5 minutes.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 300 (default value)
- **Since**: 6.11.2.9

server.session.reaper.interval

- **Module**: cmas-core-server
- **Description**: Determines the interval (in seconds) when the reaper for inactive (ended) server sessions is executed, refers to the database table `cmas_user_session`.
- **Type**: integer
- **Restart required**: only Session Service
- **System**: yes
- **Optional**: no
- **Example value**: 10800 (3 hours)
- **Since**: 6.6.1, 6.7.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

server.session.timeout

- **Module**: cmas-core-server
- **Description**: Server session timeout (in seconds) for connected clients (database table `cmas_user_session`). Each client can overwrite this timeout with custom value using its ID (ADMIN_TOOL, WEB_CLIENT, WORKFLOW_EDITOR, TRACK, ETL, REST) appended to property name, e.g., `server.session.timeout.ADMIN_TOOL`.
  Please see also the page customization attributes *updateTimeServerSessionActivityEnabled* and *updateTimeServerSessionActivity*, both of type *cmApplicationCustomization*.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1800
- **Since**: 6.6.1, 6.7.1

Detailed explanation for the Admin Tool:

- server.session.timeout.ADMIN_TOOL
  Defines the time interval how long the server considers a session valid while there is no activity from the Admin Tool holding the session. The Admin Tool is not aware of this value, it only suffers having an invalid session, if the last activity has been longer in the past.
- admin.tool.session.check.interval
  Defines the time between two checks done by the Admin Tool, if the server still considers its session valid.

For example, if admin.tool.session.check.interval = 60, the Admin Tool queries the server every minute if its session is still active/valid. In case server.session.timeout.ADMIN_TOOL = 600 the Admin Tool will get the response that the session is now invalid after ten minutes of inactivity.

skip.wfl.transfer.cleanup

- **Module**: cmas-core-server
- **Description**: Enables skipping the cleanup process after the deployment of a new workflow version. The cleanup checks whether elements which were deleted in the workflow model need to be preserved and - if not - removes them physically. Changing the default (“false”) to “true” might help if problems with deleted elements during the deployment of a new workflow version occur.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.9.4.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

skip.wfl.transfer.translations.cleanup

- **Module**: cmas-core-server
- **Description**: Enables skipping the cleanup of localized properties of removed workflow elements.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.10.5.5

strict.utf.bmp.enabled

- **Module**: cmas-core-server
- **Description**: ConSol CM installations which use a MySQL database with a 2-byte encoding / collation setting cannot handle incoming emails which contain 4-byte UTF8 characters.
  If you set this property to “true”, all 4-byte UTF8 characters are filtered out from email subjects before any database interaction. Since 6.11.2.7, it also filters out 4-byte UTF8 characters from other header fields, and since 6.11.2.8 UTF8 characters are filtered out from the names of email attachments.
  By default, the property value is “true” for MySQL databases, and “false” for any other database where the underlying problem should not occur. Do not change it for a MySQL database, unless you are sure that the database supports 4-byte characters.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.10.6.0, changed in 6.11.2.7 and 6.11.2.8

task.execution.interval.seconds

- **Module**: cmas-core-server
- **Description**: Time in seconds between the end of a completed task and the start of the next task in the Task Execution Framework.
- **Type**: Integer
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 5
- **Since**: 6.9.4.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

task.execution.node.id

- **Module**: cmas-core-server
- **Description**: Only relevant in clustered environments. The ID of the node where scripts of the TEF (Task Execution Framework) will be executed. This applies to both scripts called from the workflow and scripts called manually using the Admin Tool / Web Admin Suite, which can be started from any node.
- **Type**: Integer
- **Restart required**: yes
- **System**: no
- **Optional**: yes
- **Example value**: 2
- **Since**: 6.11.0.1

tickets.delete.size

- **Module**: cmas-core-server
- **Description**: Defines the number of tickets deleted in a single transaction.
- **Type**: integer
- **Restart required**: only Session Service
- **System**: yes
- **Optional**: no
- **Example value**: 10 (default value)
- **Since**: 6.8.1

ticket.delete.timeout

- **Module**: cmas-core-server
- **Description**: Transaction timeout (in seconds) for deleting tickets in a bulk.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 60
- **Since**: 6.1.3

ticket.from.incoming.message.accepted.links

- **Module**: cmas-core-server
- **Description**: List of domains to which links in incoming emails and links in comments added via REST API are clickable in the ticket history. Regular expressions can be used to specify the allowed URLs. It is possible to add several URLs by using a whitespace as delimiter. The URL must start with one of the allowed protocols (http, https, ftp, ftps, file, mailto). All other

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

links are removed, i.e., the link is displayed in the ticket history as text but it cannot be clicked. If the property is left empty, all links are removed. The regular expression `.+` can be used to allow all domains.

- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: https://.*\.consol\.de (allows links to “https://<any>.consol.de”)
- **Since**: 6.11.1.7



Please note that whitelisting domains might make ConSol CM vulnerable to cross-site scripting and other attacks. Choose the domains you whitelist carefully!

transaction.timeout.minutes

- **Module**: cmas-core-server
- **Description**: Sets the transaction timeout (in minutes) for the task execution service, i.e., one run of a task must finish before this timeout is reached. The changes are visible only for new tasks, the execution of which started after the configuration change.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 10*60 (10 hours - default value)
- **Since**: 6.10

unit.description.mode

- **Module**: cmas-core-server
- **Description**: Defines whether contact descriptions in the ticket history are taken from the database or dynamically rendered using templates. The value, “DYNAMIC”, is a bit more costly from the performance perspective, while “PROTOCOL” is faster but returns historical names which might be outdated. Use “PROTOCOL” if you have lots of history entries from many different units. This is also the default value in CM versions 6.11.1.1 and up.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: PROTOCOL
- **Since**: 6.11.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

unit.replace.batchSize

- **Module**: cmas-core-server
- **Description**: Defines the number of objects to be processed in a single customer replacement batch.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5
- **Since**: 6.8.2

unit.replace.timeout

- **Module**: cmas-core-server
- **Description**: Determines the transaction timeout (in seconds) of a customer replacement action step.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 120
- **Since**: 6.8.2

unused.content.remover.cluster.node.id

- **Module**: cmas-core-server
- **Description**: Value of a `cmas.clusternode.id` designating which node will remove unused ticket attachments and customer content entries.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 1 (assuming cluster node started with the parameter `-Dcmas.clusternode.id=1`)
- **Since**: 6.9.0.0

unused.content.remover.enabled

- **Module**: cmas-core-server
- **Description**: Specifies whether a removal of unused ticket attachments and customer content entries should take place.
- **Type**: boolean
- **Restart required**: no
- **System**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Optional**: no
- **Example value**: true
- **Since**: 6.9.0.0

unused.content.remover.polling.minutes

- **Module**: cmas-core-server
- **Description**: Determines the time interval (in minutes) for checking unused ticket attachments and customer content entries for removal.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 15
- **Since**: 6.9.0.0

unused.content.remover.ttl.minutes

- **Module**: cmas-core-server
- **Description**: Minimum interval (in minutes) while removed attachments and comments are preserved before finally deleting them from the database.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1440
- **Since**: 6.9.0.0

url.track

- **Module**: cmas-core-server
- **Description**: Determines the URL of CM/Track to be used by the class `LinkService` to generate links to tickets in CM/Track. The following example shows how the use the convenience method to generate a link to a ticket (the ticket object is passed as a parameter).

  ```
  url = linkTo.track(ticket);
  ```

- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: http://localhost:8080/track
- **Since**: 6.13.0.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

url.webclient

- **Module**: cmas-core-server
- **Description**: Determines the URL of the Web Client to be used by the class `LinkService` to generate links to tickets, customers or resources in the Web Client. The following example shows how the use the convenience method to generate a link to a customer (the customer object is passed as a parameter).

```
url = linkTo.webclient(unit);
```

- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: http://localhost:8080/cm-client
- **Since**: 6.13.0.0

warmup.executor.enabled

- **Module**: cmas-core-server
- **Description**: Specifies whether the server should asynchronously warm up during startup (e.g., fill some of the internal caches) for better performance.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.9.4.2

wfl.sticky.transfer.disabled

- **Module**: cmas-core-server
- **Description**: Enables using preserved original names of workflow elements.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false
- **Since**: 6.10.1.0

workflow.deploy.cache.eviction.disabled

- **Module**: cmas-core-server
- **Description**: Determines if the infinispan cache should be cleared after deployment (“true”) or not (“false”). If the property is set to “true”, all mappings are removed from the cache.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.11.1.4

H.2.3.9 cmas-core-shared (module)

cluster.mode

- **Module**: cmas-core-shared
- **Description**: Specifies whether ConSol CM is running in a cluster.
- **Type**: boolean
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.1.0

cluster.unicast

- **Module**: cmas-core-shared
- **Description**: Flag to activate the jgroups unicast mode for ConSol CM clusters (as opposed to the default multicast mode causing problems in some data center environments). If set to “true”, remember to set the JVM start parameters: `jgroups.bind.port`, `jgroups.bind.address` and `jgroups.initial_hosts`.
- **Type**: boolean
- **Restart required**: yes
- **System**: yes
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.11.0.0
- **Removed in**: 6.12.0.0

data.directory

- **Module**: cmas-core-shared
- **Description**: Global directory for ConSol CM data (e.g., index).
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: C:\Users\user\cmas
- **Since**: 6.0

expert.mode

- **Module**: cmas-core-shared
- **Description**: Activates the expert mode in the Admin Tool to unlock extended features. For example, the system property `initialized` is only available in the expert mode.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.0

H.2.3.10 cmas-dwh-server (module)

autocommit.cf.changes

- **Module**: cmas-dwh-server
- **Description**: Defines whether DWH tasks which result from configurational changes on ticket fields are executed automatically without manual interaction in the Admin Tool. Can be also set in the Admin Tool in the navigation item *DWH*. The default and recommended value is “false”.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false (default value)
- **Since**: 6.7.0

batch-commit-interval

- **Module**: cmas-dwh-server
- **Description**: Number of objects sent in one DWH message. Larger values mean better transfer performance at the cost of higher memory usage.
  Starting with ConSol CM version 6.11, this property is only used if the package size of a DWH operation is not set. This can only happen when the command is directly addressed to the Java MBean `consol.cmas.global.dwh.synchronizationService`, e.g. using the `update()` method. When a DWH operation is started using the Admin Tool, there is always a value for the package size. If not explicitly set, the default value of 1000 is used as value for `batch.commit.interval`.
- **Default value**: 1000
- **Type**: integer
- **Restart required**: no
- **System**: yes

- **Optional**: yes
- **Example value**: 100
- **Since**: 6.0.0

communication.channel

- **Module**: cmas-dwh-server
- **Description**: Communication channel. Only possible value since CM version 6.11.0.0: DIRECT
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: DIRECT
- **Since**: 6.8.5.0
- **Removed in**: 6.11.0.0 (DIRECT mode is the only available mode and is set automatically)

dwh.mode

- **Module**: cmas-dwh-server
- **Description**: Currently active mode for the DWH data transfer. Possible values are OFF, ADMIN, LIVE.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: OFF
- **Since**: 6.0.1

ignore-queues

- **Module**: cmas-dwh-server
- **Description**: A comma-separated list of queue names which are not not transferred to the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: QueueName1,QueueName2,QueueName3
- **Since**: 6.6.19
- **Removed in**: 6.8.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

is.cmrf.alive

- **Module**: cmas-dwh-server
- **Description**: Defines a threshold timeframe (in seconds) for the check if CMRF is working properly. If no response from CMRF is received in the configured timeframe, the DWH operation status is updated to “error”.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1200
- **Since**: 6.7.0

java.naming.factory.initial

- **Module**: cmas-dwh-server
- **Description**: Factory class for the DWH context factory.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: org.jnp.interfaces.NamingContextFactory
- **Since**: 6.0.1
- **Removed in**: 6.11.0.0

java.naming.factory.url.pkgs

- **Module**: cmas-dwh-server
- **Description**:
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: org.jboss.naming:org.jnp.interfaces
- **Since**: 6.0.1
- **Removed in**: 6.11.0.0

java.naming.provider.url

- **Module**: cmas-dwh-server
- **Description**: URL of naming provider.
- **Type**: string
- **Restart required**: no
- **System**: yes

- **Optional**: no
- **Example value**: localhost
- **Since**: 6.0.1
- **Removed in**: 6.11.0.0

last.ping.timestamp

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property shows the status of CMRF and is filled automatically by the CMRF. It contains the date of the last response on the ping from the ConSol CM server to the CMRF.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 32323214
- **Since**: 6.11.0.1

last.success.live.timestamp

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 13232762
- **Since**: 6.11.2.0

live.calculation.enabled

- **Module**: cmas-dwh-server
- **Description**: Determines if the CMRF should perform an update for the date since the last successful processing of a live package or the last update, whatever happened later. Please do not change this property unless advised by ConSol.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: false
- **Since**: 6.11.2.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

live.error

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: false
- **Since**: 6.11.1.0

live.start

- **Module**: cmas-dwh-server
- **Description**: When the DWH synchronization mode is set to LIVE using the Admin Tool (navigation group *Data Warehouse*, navigation item *Administration*, *Configuration* button), this property is created and set to the current date.
  If LIVE mode is not enabled and there is no data in `cmas_dwh_ser_sync_object`, the property `live.start` is deleted.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes (automatically added in DWH “LIVE” mode)
- **Example value**: 15028802377645
- **Since**: 6.7.0

notification.error.description

- **Module**: cmas-dwh-server
- **Description**: Defines the text for error emails from the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: Error occurred
- **Since**: 6.0.1

notification.error.from

- **Module**: cmas-dwh-server
- **Description**: Defines the From address for error emails from the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Optional**: yes
- **Example value**: myuser@consol.de
- **Since**: 6.0.1

notification.error.subject

- **Module**: cmas-dwh-server
- **Description**: Defines the email subject for error emails from the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: Error occurred
- **Since**: 6.0.1

notification.error.to

- **Module**: cmas-dwh-server
- **Description**: Defines the To address for error emails from the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Since**: 6.0.1

notification.finished_successfully.description

- **Module**: cmas-dwh-server
- **Description**: Defines the text for emails from the DWH when a transfer finishes successfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: Transfer finished successfully
- **Since**: 6.0.1

notification.finished_successfully.from

- **Module**: cmas-dwh-server
- **Description**: Defines the From address for emails from the DWH when a transfer finishes successfully.
- **Type**: string
- **Restart required**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **System**: yes
- **Optional**: yes
- **Example value**: myuser@consol.de
- **Since**: 6.0.1

notification.finished_successfully.subject

- **Module**: cmas-dwh-server
- **Description**: Defines the subject for emails from the DWH when a transfer finishes successfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: Transfer finished successfully
- **Since**: 6.0.1

notification.finished_successfully.to

- **Module**: cmas-dwh-server
- **Description**: Defines the To address for emails from the DWH when a transfer finishes successfully.
- **Type**: string
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Since**: 6.0.1

notification.finished_unsuccessfully.description

- **Module**: cmas-dwh-server
- **Description**: Defines the text for emails from the DWH when a transfer finishes unsuccessfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: Transfer finished unsuccessfully
- **Since**: 6.0.1

notification.finished_unsuccessfully.from

- **Module**: cmas-dwh-server
- **Description**: Defines the From address for emails from the DWH when a transfer finishes unsuccessfully.

- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: myuser@consol.de
- **Since**: 6.0.1

notification.finished_unsuccessfully.subject

- **Module**: cmas-dwh-server
- **Description**: Defines the subject for emails from the DWH when a transfer finishes unsuccessfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: Transfer finished unsuccessfully
- **Since**: 6.0.1

notification.finished_unsuccessfully.to

- **Module**: cmas-dwh-server
- **Description**: Defines the To address for emails from the DWH when a transfer finishes unsuccessfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Since**: 6.0.1

notification.host

- **Module**: cmas-dwh-server
- **Description**: Email (SMTP) server hostname for sending DWH emails.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: myserver.consol.de
- **Since**: 6.0.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

notification.password

- **Module**: cmas-dwh-server
- **Description**: SMTP password for sending DWH emails (optional).
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.0.1

notification.port

- **Module**: cmas-dwh-server
- **Description**: SMTP port for sending DWH emails.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 25
- **Since**: 6.0.1

notification.protocol

- **Module**: cmas-dwh-server
- **Description**: The protocol used for sending emails from the DWH. Possible values are smtp or pop3.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: pop3

notification.tls.enabled

- **Module**: cmas-dwh-server
- **Description**: Enables SMTP via SSL/TLS (SMTPS) for sending notification emails from the DWH. The default value is “false”. If it is set to “true”, SMTPS is enabled for sending notifications from the DWH.
- **Type**: string
- **Restart required**: yes
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.11.1.6

notification.username

- **Module**: cmas-dwh-server
- **Description**: (SMTP) user name for sending DWH emails.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: myuser
- **Since**: 6.0.1

recoverable.exceptions

- **Module**: cmas-dwh-server
- **Description**: Comma-separated list of exception definitions: CLASS[+][:REGEX]. The exceptions included in the list do not stop CM from sending to the CMRF process, but force it to try again. If optional '+' after CLASS is present, classes which extend CLASS are matched.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: java.sql.SQLRecoverableException,java.lang.RuntimeException+:.*T.1\,2T.*
- **Since**: 6.8.4.6

skip-ticket

- **Module**: cmas-dwh-server
- **Description**: Tickets are not transferred during transfer/update.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.6.19
- **Removed in**: 6.8.1

skip-ticket-history

- **Module**: cmas-dwh-server
- **Description**: History of ticket is not transferred during transfer/update.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: false
- **Since**: 6.6.19
- **Removed in**: 6.8.1

skip-unit

- **Module**: cmas-dwh-server
- **Description**: Units are not transferred during transfer/update.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.6.19
- **Removed in**: 6.8.1

skip-unit-history

- **Module**: cmas-dwh-server
- **Description**: History of unit is not transferred during transfer/update.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.6.19
- **Removed in**: 6.8.1

split.history

- **Module**: cmas-dwh-server
- **Description**: Changes the SQL that fetches the history for the tickets during DWH transfer so that the history is not fetched for all tickets at once but only for one ticket per SQL.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: false
- **Since**: 6.8.0
- **Removed in**: 6.8.4.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

statistics.calendar

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.client.group

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.contact.role

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.content.entry

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.content.entry.class

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.content.entry.history

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.cube

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes

- **Example value**: 0
- **Since**: 6.13.0.2

statistics.customer.definition

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.engineer

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.enum.group

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

statistics.field.definition

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.group.definition

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.locale

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.localized.property

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.mla

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.project

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.queue

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: 0
- **Since**: 6.11.0.1

statistics.resource

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.resource.group

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.resource.history

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

statistics.resource.relation.definition

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.resource.type

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.ticket

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.ticket.function

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.ticket.history

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.time.booking

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.timestamp

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: 0
- **Since**: 6.11.0.1

statistics.unit

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.unit.history

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.unit.relation.definition

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

statistics.workflow

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Admin Tool, section *Data Warehouse -> Monitor -> CMRF Statistics*.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

time.buffer

- **Module**: cmas-dwh-server
- **Description**: Number of minutes to extend the date of the start of the live mode.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 5
- **Since**: 6.8.1.11

unit.transfer.order

- **Module**: cmas-dwh-server
- **Description**: Defines in which order customer field groups should be transferred to the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: company;customer
- **Since**: 6.6.19
- **Removed in**: 6.8.1

H.2.3.11 cmas-nimh (module)

filesystem.polling.threads.number

- **Module**: cmas-nimh
- **Description**: Number of threads started for polling emails from the database queue. The default value is 1.
- **Type**: integer
- **Restart required**: no

- **System**: no
- **Optional**: yes
- **Example value**: 10
- **Since**: 6.4.0

filesystem.polling.threads.shutdown.timeout.seconds

- **Module**: cmas-nimh
- **Description**: Waiting time after the shutdown signal. When the timeout is reached, the thread will be terminated.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.4.0

filesystem.polling.threads.watchdog.interval.seconds

- **Module**: cmas-nimh
- **Description**: Determines the interval in seconds to execute the watchdog which checks the activity of the threads of the file system poller.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 30 (default value)
- **Since**: 6.4.0

filesystem.task.enabled

- **Module**: cmas-nimh
- **Description**: Allows to disable the service thread related to a given poller.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.4.0

filesystem.task.interval.seconds

- **Module**: cmas-nimh
- **Description**: Default interval (in seconds) for polling the file system.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.4.0

filesystem.task.polling.folder

- **Module**: cmas-nimh
- **Description**: Polling folder location which is scanned for emails in the format of `eml` files. The default value is the `mail` subdirectory of the ConSol CM data directory.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: c://cmas//mail
- **Since**: 6.4.0

filesystem.task.timeout.seconds

- **Module**: cmas-nimh
- **Description**: After this time of inactivity the service thread is considered damaged and automatically restarted. The default value is 120 seconds.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60
- **Since**: 6.4.0

filesystem.task.transaction.timeout.seconds

- **Module**: cmas-nimh
- **Description**: Default transaction timeout for transactions fetching emails from the file system. Should be correlated with number of messages fetched at once.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.4.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

mailbox.<NUMBER>.name

- **Module**: cmas-nimh
- **Description**: Identifier (name) of the mailbox.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: myEmailAccount
- **Since**: 6.11.2.0

mailbox.1.connection.host

- **Module**: cmas-nimh
- **Description**: Host (server) for first configured mailbox. Will overwrite the default parameter `mailbox.default.connection.host`.

mailbox.1.connection.password

- **Module**: cmas-nimh
- **Description**: Password for first configured mailbox. Will overwrite the default parameter `mailbox.default.connection.password`.

mailbox.1.connection.port

- **Module**: cmas-nimh
- **Description**: Port for first configured mailbox. Will overwrite the default parameter `mailbox.default.connection.port`.

mailbox.1.connection.protocol

- **Module**: cmas-nimh
- **Description**: Protocol (e.g., IMAP or POP3) for first configured mailbox. Will overwrite the default parameter `mailbox.default.connection.protocol`.

mailbox.1.connection.username

- **Module**: cmas-nimh
- **Description**: User name for first configured mailbox. Will overwrite the default parameter `mailbox.default.connection.username`.

mailbox.2.connection.host

- **Module**: cmas-nimh
- **Description**: Host (server) for second configured mailbox. Will overwrite the default parameter `mailbox.default.connection.host`.

mailbox.2.connection.password

- **Module**: cmas-nimh
- **Description**: Password for second configured mailbox. Will overwrite the default parameter `mailbox.default.connection.password`.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

mailbox.2.connection.port

- **Module**: cmas-nimh
- **Description**: Port for second configured mailbox. Will overwrite the default parameter `mailbox.default.connection.port`.

mailbox.2.connection.protocol

- **Module**: cmas-nimh
- **Description**: Protocol (e.g., IMAP or POP3) for second configured mailbox. Will overwrite the default parameter `mailbox.default.connection.protocol`.

mailbox.2.connection.username

- **Module**: cmas-nimh
- **Description**: User name for second configured mailbox. Will overwrite the default parameter `mailbox.default.connection.username`.

For all NIMH-related mailbox properties, the following principle is used: a default property is defined (e.g. `mailbox.default.connection.port`). If no mailbox-specific value is configured, this default value will be used.

mailbox.default.connection.host

- **Module**: cmas-nimh
- **Description**: Host (server name) of a given mailbox which the poller reads emails from.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 10.10.1.157
- **Since**: 6.4.0

mailbox.default.connection.password

- **Module**: cmas-nimh
- **Description**: Password for given mailbox which the poller reads emails from.
- **Type**: password
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: consol
- **Since**: 6.4.0

mailbox.default.connection.port

- **Module**: cmas-nimh
- **Description**: Port of a given mailbox which the poller reads emails from.

- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 143
- **Since**: 6.4.0

mailbox.default.connection.protocol

- **Module**: cmas-nimh
- **Description**: The protocol used by the email poller, e.g. IMAP or POP3.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: imap
- **Since**: 6.4.0

mailbox.default.connection.username

- **Module**: cmas-nimh
- **Description**: User name for a given mailbox which the poller reads emails from.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: username
- **Since**: 6.4.0

mailbox.default.session.mail.debug

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. Allows for more detailed JavaMail session debugging.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true
- **Since**: 6.4.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

mailbox.default.session.mail.mime.address.strict

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. If this property is set to "false", the parsing of the email headers is less strict. Therefore, more emails with non-RFC-compliant headers will be imported into the system, instead of being stored in the email backups in the Admin Tool / Web Admin Suite.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.4.0

mailbox.default.session.mail.<PROTOCOL>.connectiontimeout

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. Determines the connection timeout in milliseconds.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 3000
- **Since**: 6.4.0 (IMAP and POP3) / 6.11.2.0 (IMAPS and POP3S)

mailbox.default.session.mail.<PROTOCOL>.fetchsize

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. Determines the size of partial fetch in bytes for the indicated protocol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 1048576 (default value)
- **Since**: 6.9.4.0 (IMAP and POP3) / 6.11.2.0 (IMAPS and POP3S)

mailbox.default.session.mail.<PROTOCOL>.partialfetch

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. Determines whether the protocol's partial fetch capability should be used.
- **Type**: boolean
- **Restart required**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.9.4.0 (IMAP and POP3) / 6.11.2.0 (IMAPS and POP3S)

mailbox.default.session.mail.<PROTOCOL>.timeout

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. Determines the I/O timeout in milliseconds.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 120 for IMAP(S) and 3000 for POP3(S)
- **Since**: 6.4.0 (IMAP and POP3), 6.11.2.0 (IMAPS and POP3S)

mailbox.default.task.delete.read.messages

- **Module**: cmas-nimh
- **Description**: Defines whether messages should be removed from the mailbox after processing. For the IMAP protocol messages are marked as SEEN by default. For the POP3 protocol, this property is ignored and the emails are always deleted from the email server after they were processed successfully.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.4.0, changed in 6.12.1.0

mailbox.default.task.enabled

- **Module**: cmas-nimh
- **Description**: With this property, the service thread related to the given poller can be disabled.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.4.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

mailbox.default.task.interval.seconds

- **Module**: cmas-nimh
- **Description**: Default interval (in seconds) for polling mailboxes.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.4.0

mailbox.default.task.max.message.size

- **Module**: cmas-nimh
- **Description**: Maximum size of email messages (i.e., email plus attachment). Emails exceeding the size limit will not be automatically processed by NIMH but will be stored in the database (table `cmas_nimh_archived_mail`) and will therefore appear in the email backups in the Admin Tool / Web Admin Suite. From there they can be resent, downloaded to the file system, or deleted. For those operations the message size is not relevant.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 10485760 (default value, 10 MB)
- **Since**: 6.4.0

mailbox.default.task.max.messages.per.run

- **Module**: cmas-nimh
- **Description**: Number of messages fetched at once from the mailbox. Must be correlated with the transaction timeout. The default value is 20.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60
- **Since**: 6.4.0

mailbox.default.task.timeout.seconds

- **Module**: cmas-nimh
- **Description**: After this time of inactivity the service thread is considered damaged and automatically restarted. The default value is 120 seconds.
- **Type**: integer
- **Restart required**: no

- **System**: no
- **Optional**: yes
- **Example value**: 60
- **Since**: 6.4.0

mailbox.default.task.transaction.timeout.seconds

- **Module**: cmas-nimh
- **Description**: Default transaction timeout (in seconds) for email fetching transactions. Should be correlated with number of messages fetched at once.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.4.0

mailbox.polling.threads.mail.log.enabled

- **Module**: cmas-nimh
- **Description**: Enables email logging which is especially crucial in cluster environments (used as semaphore there).
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.9.4.1

mailbox.polling.threads.number

- **Module**: cmas-nimh
- **Description**: Number of threads for accessing mailboxes.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 1 (default value)
- **Since**: 6.4.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

mailbox.polling.unstorable.backup.folder

- **Module**: cmas-nimh
- **Description**: Folder on the file system which is used to store emails whose processing failed and which could not be stored in the database either. The value of the property should be the path to the data directory followed by `/mail/unstorable`.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: c:/cmas/mail/unstorable
- **Since**: 6.11.2.7

queue.polling.threads.number

- **Module**: cmas-nimh
- **Description**: Number of threads started for polling the email queue in the database.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 1 (default value)
- **Since**: 6.4.0

queue.polling.threads.shutdown.timeout.seconds

- **Module**: cmas-nimh
- **Description**: Waiting time after the shutdown signal for the email poller. When the timeout is reached, the thread will be terminated.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.4.0

queue.polling.threads.watchdog.interval.seconds

- **Module**: cmas-nimh
- **Description**: Determines the interval in seconds to execute the watchdog which checks the activity of the threads of the email queue poller.
- **Type**: integer
- **Restart required**: no
- **System**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Optional**: yes
- **Example value**: 30 (default value)
- **Since**: 6.4.0

queue.task.error.pause.seconds

- **Module**: cmas-nimh
- **Description**: Maximum number of seconds, the queue poller waits after infrastructure (e.g. database) error.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 180 (default value)
- **Since**: 6.4.0

queue.task.interval.seconds

- **Module**: cmas-nimh
- **Description**: Determines the interval in seconds for the main email queue polling thread.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 15 (default value)
- **Since**: 6.4.0

queue.task.max.retries

- **Module**: cmas-nimh
- **Description**: Maximum number of email processing retries after an exception. When reached, the email is moved to the email archive. This email can be rescheduled again using the NIMH API or the Admin Tool / Web Admin Suite.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 10
- **Since**: 6.4.0

queue.task.timeout.seconds

- **Module**: cmas-nimh
- **Description**: After this time of inactivity (in seconds), the service thread is considered damaged and automatically restarted.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 600 (default value)
- **Since**: 6.4.0

queue.task.transaction.timeout.seconds

- **Module**: cmas-nimh
- **Description**: Transaction timeout (in seconds) for email processing in the pipe.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.4.0

H.2.3.12 cmas-nimh-extension (module)

mail.attachments.validation.info.sender

- **Module**: cmas-nimh-extension
- **Description**: Sets the From header for error notification emails which are sent when the validation of the attachment type fails.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: admin@mail.com
- **Since**: 6.7.5

mail.attachments.validation.info.subject

- **Module**: cmas-nimh-extension
- **Description**: Sets the subject for error notification emails which are sent when the validation of the attachment type fails.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: Email was not processed because its attachments were rejected!
- **Since**: 6.7.5

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

mail.db.archive

- **Module**: cmas-nimh-extension
- **Description**: If property is set to “true”, incoming emails are automatically archived in the database.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.8.5.5

mail.error.from.address

- **Module**: cmas-nimh-extension
- **Description**: Sets the From address for error notification emails which are sent when the email processing fails.
- **Type**: email
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Since**: 6.4.0

mail.error.to.address

- **Module**: cmas-nimh-extension
- **Description**: Sets the To address for error notification emails which are sent when the email processing fails. As a default the value of the property `admin.email` is used.
- **Type**: email
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Since**: 6.4.0

mail.on.error

- **Module**: cmas-nimh-extension
- **Description**: If set to “true”, an error notification email is sent to the configured address if an email could not be processed.
- **Type**: boolean
- **Restart required**: no
- **System**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Optional**: yes
- **Example value**: true (default)
- **Since**: 6.4.0

mail.ticketname.pattern

- **Module**: cmas-nimh-extension
- **Description**: Regular expression pattern used to identify the ticket name in the subject of incoming mails.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: .*?Ticket\s+\((\S+)\).*
- **Since**: 6.4.0
- **Removed in**: 6.12.0.5

H.2.3.13 cmas-restapi-core (module)

comment.authors.disabled

- **Module**: cmas-restapi-core
- **Description**: Disables the display of the content author via REST API.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.11.0

csrf.domain.allow.none

- **Module**: cmas-restapi-core
- **Description**: Determines whether empty `Origin/Referer` headers are accepted. By default, the property is set to “false”, so that REST requests without `Origin/Referer` headers are blocked.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false (default value)
- **Since**: 6.11.2.2

csrf.domain.white.list

- **Module**: cmas-restapi-core
- **Description**: The list of domains (format protocol://domain:port, separated with "|") which are allowed in `Origin/Referer` headers and will not be blocked by the CSRF filter. By default, the property is empty so that cross-site requests are blocked.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: https://example.com:80 | http://www.consol.de:8080
- **Since**: 6.11.2.2

csrf.request.filter.enabled

- **Module**: cmas-restapi-core
- **Description**: Determines whether the CSRF request filter (Origin/Referer) for the REST API is enabled.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.11.2.2

diff.tracking.disabled

- **Module**: cmas-restapi-core
- **Description**: Fallback property for disabling difference tracking for CM/Track, which is history-based so it can be heavy.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.10.5.6

domain.map.for.client.config.<DOMAIN_NAME>

- **Module**: cmas-restapi-core
- **Description**: Enables mapping multiple instances of CM/Track to a specific domain. DOMAIN_NAME is the name of the client configuration (if your CM/Track configuration is called "track-V2customized", the name of the property is `domain.map.for.client.config.trackV2customized`). You can provide several URLs separated by a comma.

- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: https://www.consol.de, https://www.consol.com
- **Since**: 6.10.7.0

request.log.threshold.milliseconds

- **Module**: cmas-restapi-core
- **Description**: Time threshold in milliseconds that needs to be exceeded to generate an entry in the file `operationtimes-rest.log`.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5000 (default value)
- **Since**: 6.13.0.2

request.log.username

- **Module**: cmas-restapi-core
- **Description**: Name of the user whose requests are logged in the file `operationtimes-rest.log`. Only one name is supported. If the property is empty (default value), the requests of all users are logged.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: admin
- **Since**: 6.13.0.2

security.fields.customer.exposure.check.enabled

- **Module**: cmas-restapi-core
- **Description**: Enables `customer exposure` annotation checks for ticket fields.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.10.5.4

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

security.restrict.unit.access.to.own.data

- **Module**: cmas-restapi-core
- **Description**: If set to “true”, an additional check is performed when a user logs in as a customer using the REST API, e.g. CM/Track. When requesting customer data, only the company of the user or other contacts of the user’s company are returned. If set to “false”, no additional security check is performed and the former security rules apply.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true (default value)
- **Since**: 6.9.2.14

H.2.3.14 cmas-restapi-http-headers (module)

This module contains properties which represent header fields which should be sent with REST API requests and responses.

Each property in the module `cmas-restapi-http-headers` represents one header field. The property name/key identifies the HTTP header field and the value of the property is the field value sent in this header.

Please be aware that additional HTTP response headers must be correctly defined with the exact spelling as officially specified. Please note also that the correct interpretation and application of these headers is fully in the realm and responsibility of the HTTP client.

For example, if you add the property `Access-Control-Allow-Origin` with the value “http://www.example.com” to the `cmas-restapi-http-headers` module, REST API requests from *http://www.example.com* can be allowed in your domain even if the general policy is to allow only requests from the same domain.

As there are no default values for the headers in this module, the module is not displayed when creating the first property in it. The module name needs to be typed in manually in this case.

H.2.3.15 cmas-setup-hibernate (module)

cmas.dropSchemaBeforeSetup

- **Module**: cmas-setup-hibernate
- **Description**: Determines if an existing database schema should should be dropped during setup. This value is set on the setup page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

connection.release.mode

- **Module**: cmas-setup-hibernate
- **Description**: Describes the JEE connection handling strategy for transactions. If set to “AFTER_TRANSACTION”, the connection will be cached during the transaction and released at the end. If set to “AFTER_STATEMENT”, the connection will be released to the pool after each statement execution. Please do not change the default here unless advised by ConSol.
- **Type**: string
- **Restart required**: yes
- **System**: no
- **Optional**: yes
- **Example value**: AFTER_STATEMENT (default for JEE environment)
- **Since**: 6.0

hibernate.dialect

- **Module**: cmas-setup-hibernate
- **Description**: The database dialect used by hibernate. Set during the initial setup depending on the database system.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: org.hibernate.dialect.MySQL5InnoDBDialect
- **Since**: 6.0

update.6.11.0.0.sleep

- **Module**: cmas-setup-hibernate
- **Description**: Helper property for the update preparation scripts introduced in context of CM database refactoring in version 6.11. This is an optional setting allowing a delay (in milliseconds) after each loop iteration of the preparation scripts. Setting the delay should lower the database load, for example during working hours. This property may be removed after the update preparation tasks finish.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.0, for use in 6.10.5.x

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

update.6.11.0.0.timezone

- **Module**: cmas-setup-hibernate
- **Description**: Helper property for the ticket history migration (the new way of counting history groups). Since 6.11.0.0 the groups are constant (2h time span), but before 6.11.0.0 groups were not constant and depended on the customer's time zone. Migration scripts use an old algorithm to calculate groups and therefore need information about the time zone. The property should be set to the timezone which is most commonly used by the customers. If the property is not set, the default server time zone is used (`TimeZone.getDefault()`). The property should be set before updating to 6.11.0.0 and will be removed automatically after migration. The list of accepted timezones can be found for example here: http://joda-time.sourceforge.net/timezones.html.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: Europe/Berlin
- **Since**: 6.11.0.0, for use before updating to this version

H.2.3.16 cmas-setup-manager (module)

initialized

- **Module**: cmas-setup-manager
- **Description**: Flag indicating if ConSol CM is initialized, i.e. the initial setup has been performed. If this value is missing or not "true", the setup will be performed. Starting with ConSol CM version 6.11, this property is only available in expert mode.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.0

> ⚠ Be careful with using this property. When you set the value to "false", the ConSol CM server will perform the system setup at the next start, i.e. all data of the existing system is lost, including system properties.

H.2.3.17 cmas-setup-scene (module)

scene

- **Module**: cmas-setup-scene
- **Description**: Name of the scene file which was imported during the setup (can be empty).
- **Type**: string
- **Restart required**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **System**: yes
- **Optional**: no
- **Example value**: 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/
- **Since**: 6.0

H.2.3.18 cmas-web-notifications (module)

notifications.enabled

- **Module**: cmas-web-notifications
- **Description**: Determines whether the notifications feature is enabled (“true”) or disabled (“false”). The default value is “true”. If you use the notifications feature on a ConSol CM system behind a proxy, some special proxy settings are needed. Please see section *Operating ConSol CM Behind a Proxy (e.g., Apache HTTPD)* of the *ConSol CM Setup Manual* for further information.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: true
- **Since**: 6.12.0.0

ttl.days

- **Module**: cmas-web-notifications
- **Description**: Determines the time (in days) after which unpinned notifications are removed.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 15
- **Since**: 6.12.0.0

H.2.3.19 cmas-workflow-engine (module)

jobExecutor.adminMail

- **Module**: cmas-workflow-engine
- **Description**: Email address which will get notified about job execution problems (when retry counter is exceeded).
- **Type**: string
- **Restart required**: no
- **System**: yes

- **Optional**: yes
- **Example value**: admin@consol.de
- **Since**: 6.8.0

jobExecutor.idleInterval.seconds

- **Module**: cmas-workflow-engine
- **Description**: Determines how often (in seconds) the job executor thread will look for new jobs to execute.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 45 (default value up to CM version 6.10.5.2. The default value for CM versions 6.10.5.3 and up is 5)
- **Since**: 6.8.0

jobExecutor.jobMaxRetries

- **Module**: cmas-workflow-engine
- **Description**: Controls the number of retry attempts the job executor will do before declaring a job as failed.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 5 (default value)
- **Since**: 6.8.0

jobExecutor.jobMaxRetriesReachedSubject

- **Module**: cmas-workflow-engine
- **Description**: The subject used in the notification email which administrators receive about failed job executions.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: Job maximum retries reached. Job was removed!!! (default value)
- **Since**: 6.8.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

jobExecutor.lockingLimit

- **Module**: cmas-workflow-engine
- **Description**: Maximum number of jobs which can locked at once (marked for execution) by the job executor thread.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 5 (default value since CM version 6.10.5.3)
- **Since**: 6.8.0

jobExecutor.lockTimeout.seconds

- **Module**: cmas-workflow-engine
- **Description**: Determines how long the job can be locked (marked for execution) by the job executor.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 360 (default value)
- **Since**: 6.8.0

jobExecutor.mailFrom

- **Module**: cmas-workflow-engine
- **Description**: Email address which will be set as From header for notifications to the administrator.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: jobexecutor@consol.de
- **Since**: 6.8.0

jobExecutor.maxInactivityInterval.minutes

- **Module**: cmas-workflow-engine
- **Description**: Maximum amount of time (in minutes) of allowed job executor inactivity (e.g. when it is blocked by long timer execution). After this time, executor threads are restarted automatically.
- **Type**: integer
- **Restart required**: no
- **System**: yes

- **Optional**: yes
- **Example value**: 30 (default value)
- **Since**: 6.9.2.0

jobExecutor.threads

- **Module**: cmas-workflow-engine
- **Description**: Sets the number of job execution threads.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 1 (default value)
- **Since**: 6.8.0

jobExecutor.timerRetryInterval.seconds

- **Module**: cmas-workflow-engine
- **Description**: Determines how long (in seconds) the job executor thread will wait before a retry after an error occurs during job execution.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 10 (default up to CM version 6.10.5.2. The default value for CM versions 6.10.5.3 and up is 30)
- **Since**: 6.8.0

jobExecutor.txTimeout.seconds

- **Module**: cmas-workflow-engine
- **Description**: Transaction timeout used for job execution.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.8.0

H.2.3.20 cmas-workflow-jbpm (module)

fetchLock.interval

- **Module**: cmas-workflow-jbpm
- **Description**:

- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5000
- **Removed in**: 6.8.0

jobExecutor.idleInterval

- **Module**: cmas-workflow-jbpm
- **Description**:
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 45000
- **Removed in**: 6.8.0
- **Replaced by**: jobExecutor.idleInterval.seconds

jobExecutor.jobExecuteRetryNumber

- **Module**: cmas-workflow-jbpm
- **Description**:
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5
- **Removed in**: 6.8.0
- **Replaced by**: jobExecutor.jobMaxRetries

jobExecutor.timerRetryInterval

- **Module**: cmas-workflow-jbpm
- **Description**:
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 10000
- **Removed in**: 6.8.0
- **Replaced by**: jobExecutor.timerRetryInterval.seconds

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

mail.sender.address

- **Module**: cmas-workflow-jbpm
- **Description**: From address for emails from the workflow engine.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Removed in**: 6.8.0
- **Replaced by**: jobExecutor.mailFrom

outdated.lock.age

- **Module**: cmas-workflow-jbpm
- **Description**:
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 60000
- **Removed in**: 6.8.0
- **Replaced by**: cmas-workflow-engine, jobExecutor.lockTimeout.seconds

H.2.3.21 cmweb-server-http-headers (module)

X-Frame-Options

- **Module**: cmweb-server-http-headers
- **Description**: Example property to illustrate the configuration of HTTP headers. In this case the delivered HTTP header contains the field *X-Frame-Options* with the value “SAMEORIGIN”.

  Each property in the module `cmweb-server-http-headers` represents one header field. The property name/key identifies the response header field and the value of the property is the field value sent in this header.

  > ⚠ Please be aware that additional HTTP response headers must be correctly defined with the exact spelling as officially specified! Please note also that the correct interpretation and application of these headers is fully in the realm and responsibility of the client browser which requested the page!

- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes

- **Example value**: SAMEORIGIN
- **Since**: 6.10.8

H.2.3.22 cmweb-server-adapter (module)

attachment.upload.timeout

- **Module**: cmweb-server-adapter
- **Description**: Defines the transaction timeout in minutes for adding attachments to a ticket, resource or customer. Counts the time for the upload of all attachments of one transaction. When the timeout occurs, all files which have been temporarily stored on the server are deleted and no file is uploaded.
- **Type**: Integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 3
- **Since**: 6.10.5.3

automatic.booking.enabled

- **Module**: cmweb-server-adapter
- **Description**: If enabled, time spent on creating comments or emails is measured and automatic time bookings are added.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: true
- **Since**: 6.9.4.2

checkUserOnlineIntervalInSeconds

- **Module**: cmweb-server-adapter
- **Description**: The interval in seconds to check which users are online (default 180sec = 3min).
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 180
- **Since**: 6.0
- **Removed in**: 6.5 / 6.11.0.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

cmoffice.enabled

- **Module**: cmweb-server-adapter
- **Description**: Determines if CM/Doc is enabled.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.4.0

cmoffice.oo.path.NUMBER

- **Module**: cmweb-server-adapter
- **Description**: Possible location of the OpenOffice installation. The properties are numbered starting with 0.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: c:\Program Files (x86)\LibreOffice 3.6\program
- **Since**: 6.10.1.0
- **Removed in**: 6.12.0.0

cmoffice.strict.versioning.enabled

- **Module**: cmweb-server-adapter
- **Description**: Controls if the SAVE operation in Microsoft Word / OpenOffice documents creates a new attachment (“true”) or overwrites the existing attachment (“false”). This concerns the behavior within one session using the text editing program. If the program is stopped, the overwrite mechanism will not work anymore.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true
- **Since**: 6.10.5.4

cmoffice.websocket.port

- **Module**: cmweb-server-adapter
- **Description**: Determines the port on which the CM/Doc application should be started and connected to.
- **Type**: integer
- **Restart required**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **System**: no
- **Optional**: yes
- **Example value**: 30333
- **Since**: 6.12.0.0

commentRequiredForTicketCreation

- **Module**: cmweb-server-adapter
- **Description**: Determines if the comment is a required field for ticket creation.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true (default value)
- **Since**: 6.2.0

csrf.domain.white.list

- **Module**: cmweb-server-adapter
- **Description**: The list of domains (separated with "|") which are allowed and will not be blocked by the CSRF (cross-site request forgery) filter, e.g.: "example.com | consol.de"
- **Type**: String
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: example.com | consol.de
- **Since**: 6.10.7.0

csrf.request.filter.enabled

- **Module**: cmweb-server-adapter
- **Description**: It allows to disable the CSRF request filter, which is enabled by default.
- **Type**: Boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true
- **Since**: 6.10.7.0

customizationVersion

- **Module**: cmweb-server-adapter
- **Description**: UID representing the latest web customization version. Used only internally, please do not change the value.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: cd58453e-f3cc-4538-8030-d15e8796a4a7
- **Since**: 6.5.0

data.optimization

- **Module**: cmweb-server-adapter
- **Description**: Defines optimization to be applied on HTML response data. The following values are supported (for setting more than one value, separate values by '|'): MINIFICATION and COMPRESSION. MINIFICATION minifies HTML data by e.g. stripping whitespaces and comments. COMPRESSION applies gzip compression to HTTP response. (Note: If you are running in cluster mode and want to test different configurations in parallel, you can set different values for each cluster node by specifying the property `data.optimization.nodeId` to override the default property.)
- **Type**: string
- **Restart required**: COMPRESSION can be switched on/off without restart, MINIFICATION requires restart.
- **System**: yes
- **Optional**: yes
- **Example value**: MINIFICATION|COMPRESSION

defaultAttachmentEntryClassName

- **Module**: cmweb-server-adapter
- **Description**: The default text class used to classify an attachment if no other text class was set explicitly.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: DefaultTextElement
- **Since**: 6.9.2.0

defaultContentEntryClassName

- **Module**: cmweb-server-adapter
- **Description**: Name of the text class which is set as default for newly created content entries.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no

- **Example value**: default_class
- **Since**: 6.3.0

defaultNumberOfCustomFieldsColumns

- **Module**: cmweb-server-adapter
- **Description**: Determines the default number of columns for ticket fields. The value is only used if the ticket fields are not positioned via annotation.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 3
- **Since**: 6.2.0

diffTrackingEnabled

- **Module**: cmweb-server-adapter
- **Description**: Defines if parallel editing of a ticket by different engineers should be possible.
  "false": Previous way of handling changes when editing a ticket. If the ticket has been changed in the meantime, the current engineer will not be able to submit his changes without being forced to reload the page before submitting.
  "true": New changes handling mode. If the ticket has been changed, this will not block the submission of other changes anymore. If the part of the ticket that was changed was exactly the part that is changed by the submitting engineer, then an information message will be displayed, but the ticket change will be persisted/stored anyway.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true (default)
- **Since**: 6.10.1
- **Removed in**: 6.11.0

diffTrackingEnabledForUnitAndResource

- **Module**: cmweb-server-adapter
- **Description**: Enables the prevention of concurrent modifications on customers / resources.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 3
- **Since**: 6.11.0.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

favoritesSizeLimit

- **Module**: cmweb-server-adapter
- **Description**: Determines the maximum number of items in the favorites.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 10
- **Since**: 6.0

forward.mails.to.representatives

- **Module**: cmweb-server-adapter
- **Description**: Determines if emails which are manually sent from the Web Client are also sent to representing engineers. The default value of the property is “false”, meaning that this kind of emails are not forwarded to the representing engineer. Set the property to “true” if you want to restore the previous behavior, i.e., all emails which are sent to the represented engineer are automatically forwarded to the representing engineer. Please take into account that this might not be desired if the same person is an engineer and a customer in the CM system.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: false (default value)
- **Since**: 6.11.1.7



This property only configures the handling of manually sent emails. The handling of automatically sent emails depends on the used Java method.

globalSearchResultSizeLimit

- **Module**: cmweb-server-adapter
- **Description**: Determines the maximum number of results in the quick search.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 10
- **Since**: 6.0

helpFilePath

- **Module**: cmweb-server-adapter
- **Description**: URL for the online help. If not empty, the *Help* link is displayed in the Web Client.

- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: http://www.consol.de
- **Since**: 6.2.1

hideTicketSubject

- **Module**: cmweb-server-adapter
- **Description**: If set to “true”, the ticket subject is hidden in the Web Client.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.2.1

mail.from

- **Module**: cmweb-server-adapter
- **Description**: Central sender address to be used for email communication. The value set here appears as sender of emails sent from ConSol CM.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: consolcm@example.com
- **Since**: 6.1.2

mail.reply.to

- **Module**: cmweb-server-adapter
- **Description**: Central reply-to address to be used for email communication. The value set here controls the reply-to header of emails sent from ConSol CM.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: consolcm@example.com
- **Since**: 6.0.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

> ⚠ Please read the detailed information about ConSol CM Reply-To addresses in section Scripts of Type Email.

mailTemplateAboveQuotedText

- **Module**: cmweb-server-adapter
- **Description**: Determines the behavior of the email template in the email editor when another email is quoted, i.e. forwarded or replied to. Often used to place the signature correctly.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.2.4

maxSizePerPagemapInMegaBytes

- **Module**: cmweb-server-adapter
- **Description**: The parameter defines the size (in MB) of the file which is created by the Wicket framework per user session. i.e. for each engineer which is currently logged in. The file is used to save pages during the running session. When the defined size limit has been reached and new entries are added, the oldest entries are removed. An engineer who works with an “old” page is redirected to the start page in the Web Client, when the page is removed from the file. This parameter can be increased if engineers work with a great number of open tabs and complain about being redirected to the start page. In large systems, you could use e.g. a value of 45 or 50. Since this is the size of the file which is saved on disk, the maximum value depends on the available disk space, however, a value which is too large is not recommended either.
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: 15
- **Since**: 6.3.5

pagemapLockDurationInSeconds

- **Module**: cmweb-server-adapter
- **Description**: Number of seconds to pass before a pagemap is considered to be locked for too long.
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: yes

- **Example value**: 60
- **Since**: 6.7.3

postActivityExecutionScriptName

- **Module**: cmweb-server-adapter
- **Description**: Defines the name for the script which should be executed after every workflow activity, see section PostActivityExecutionScript. If no script should be executed, leave the value empty.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: postActivityExecutionHandler
- **Since**: 6.2.0

queuesExcludedFromGS

- **Module**: cmweb-server-adapter
- **Description**: Comma-separated list of queue names which are excluded from the quick search.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: Helpdesk,FAQ
- **Since**: 6.0

refreshTimeInCaseOfConcurrentRememberMeRequests

- **Module**: cmweb-server-adapter
- **Description**: Sets the refresh time (in seconds) after which the page is reloaded in case of concurrent remember me requests. This feature prevents one user from occupying many licenses.
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: yes
- **Example value**: 5
- **Since**: 6.8.2

rememberMeLifetimeInMinutes

- **Module**: cmweb-server-adapter
- **Description**: Lifetime for *remember me* in minutes.
- **Type**: integer
- **Restart required**: yes

- **System**: yes
- **Optional**: no
- **Example value**: 1440
- **Since**: 6.0

request.scope.transaction

- **Module**: cmweb-server-adapter
- **Description**: It allows to disable request scope transactions. By default one transaction is used per request. Setting this property to “false” will cause one transaction per service method invocation.
- **Type**: boolean
- **Restart required**: yes
- **System**: yes
- **Optional**: yes
- **Example value**: true
- **Since**: 6.8.1

searchPageSize

- **Module**: cmweb-server-adapter
- **Description**: The global default page size for search results in the Web Client. Only used for new users who have not yet set preferences.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 20
- **Since**: 6.0

searchPageSizeOptions

- **Module**: cmweb-server-adapter
- **Description**: Paging options for the page size of search results in the Web Client.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 10|20|30|40|50|75|100
- **Since**: 6.0

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

serverPoolingInterval

- **Module**: cmweb-server-adapter
- **Description**: Defines the time in seconds for the pooling server to invalidate caches on the web layer.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5
- **Since**: 6.1.0

show.error.stacktrace

- **Module**: cmweb-server-adapter
- **Description**: If this property is set to “true”, the full stracktrace can be shown in the red error message which is displayed in the Web Client if a user action caused an exception.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.12.1.0

supportEmail

- **Module**: cmweb-server-adapter
- **Description**:
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.0
- **Removed in**: 6.11.0.1

themeOverlay

- **Module**: cmweb-server-adapter
- **Description**: Defines the user interface theme (CSS file) for the Web Client layout. Set this property to the name of your custom skin if you are using one.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: consolINT
- **Since**: 6.0, functionality changed in 6.11.2.0

ticketListRefreshIntervalInSeconds

- **Module**: cmweb-server-adapter
- **Description**: Refresh interval for the ticket list (in seconds).
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 180
- **Since**: 6.0

ticketListSizeLimit

- **Module**: cmweb-server-adapter
- **Description**: Maximum number of tickets in the ticket list.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 100
- **Since**: 6.0

tx.read.only.mode.enabled

- **Module**: cmweb-server-adapter
- **Description**: Enables read-only transactions for faster page loading. This transactional behavior was introduced in 6.11.0, and this property acts as a safety guard to restore the old behaviors. Do not change this value unless you are facing transaction problems and advised by ConSol.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.11

tinymce.enabled

- **Module**: cmweb-server-adapter
- **Description**: Determines whether Froala (“false”) or TinyMCE (“true”) is used as an editor for the Web Client.
- **Type**: boolean
- **Restart required**: no

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.12.1.0

unitIndexSearchResultSizeLimit

- **Module**: cmweb-server-adapter
- **Description**: Defines the maximum number of hints for all customer autocomplete or suggestion components (e.g. when adding customer to a ticket). Components customizations (maxHints, maxSuggestions) may change that value.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5
- **Since**: 6.0

urlLogoutPath

- **Module**: cmweb-server-adapter
- **Description**: URL which is opened when the user logs out. There are three configuration possibilities:
  - Empty value: The user is redirected to the login page.
  - <CM URL>/cm-client/logout: The user is redirected to the logout page, which includes a link to log in again.
  - Any URL: The user is redirected to the given URL outside ConSol CM.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: https://mycmserver/cm-client/logout
- **Since**: 6.3.1, behavior changed in 6.11.2.0

voCacheEnabled

- **Module**: cmweb-server-adapter
- **Description**: This property enables additional caching for the Web Client, voCaching, of complete objects, thus improving performance.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- **Example value**: true
- **Since**: 6.11.1.0

Notes:

- Since ConSol CM version 6.11.1.1, the default value is “true” for non-clustered environments. The value is set to “true” automatically during the setup or update of ConSol CM 6.11.1.1.
- When voCaching is enabled and lazy loading is used for folding ticket history entries, once the engineer unfolded an entry, he cannot fold it again by reloading the page or opening the ticket from the workspace.
- When using the dynamic mode for displaying engineer and customer names in the ticket history (as configured in the system properties cmas-core-server, engineer.description.mode and cmas-core-server, unit.description.mode), the new version of the engineer and/or customer name is only displayed after the ticket has been changed.
- This system property is ignored for clustered environments (environments with cmas-core-shared, cluster.mode set to “true”). In clustered environments, voCaching is always disabled to avoid problems that changes made to objects on one node are not visible on the other nodes.

webSessionTimeoutInMinutes

- **Module**: cmweb-server-adapter
- **Description**: Session timeout in minutes.
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: 180
- **Removed in**: 6.7.1
- **Replaced by**: cmas-core-server, server.session.timeout

wicketAjaxRequestHeaderFilterEnabled

- **Module**: cmweb-server-adapter
- **Description**: This enables the filter for Wicket AJAX requests, coming from stale pages with Wicket 1.4 scripting (CM < 6.8.0), after an update to CM > 6.8.0.
- **Type**: boolean
- **Restart required**: yes
- **System**: yes
- **Optional**: yes
- **Example value**: false
- **Since**: 6.8.1

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

H.3 Administrator and Notification Email Addresses

This chapter discusses the following:

H.3.1 Introduction 1181
H.3.2 Default Configuration 1181
H.3.3 Subsystem-Specific Notification Email Addresses 1183

H.3.1 Introduction

In ConSol CM, several administrator email addresses (or notification email addresses respectively) can be configured. Here, an overview of all those addresses is provided.

H.3.2 Default Configuration

When you set-up a ConSol CM system, you have to enter one global admin email address.

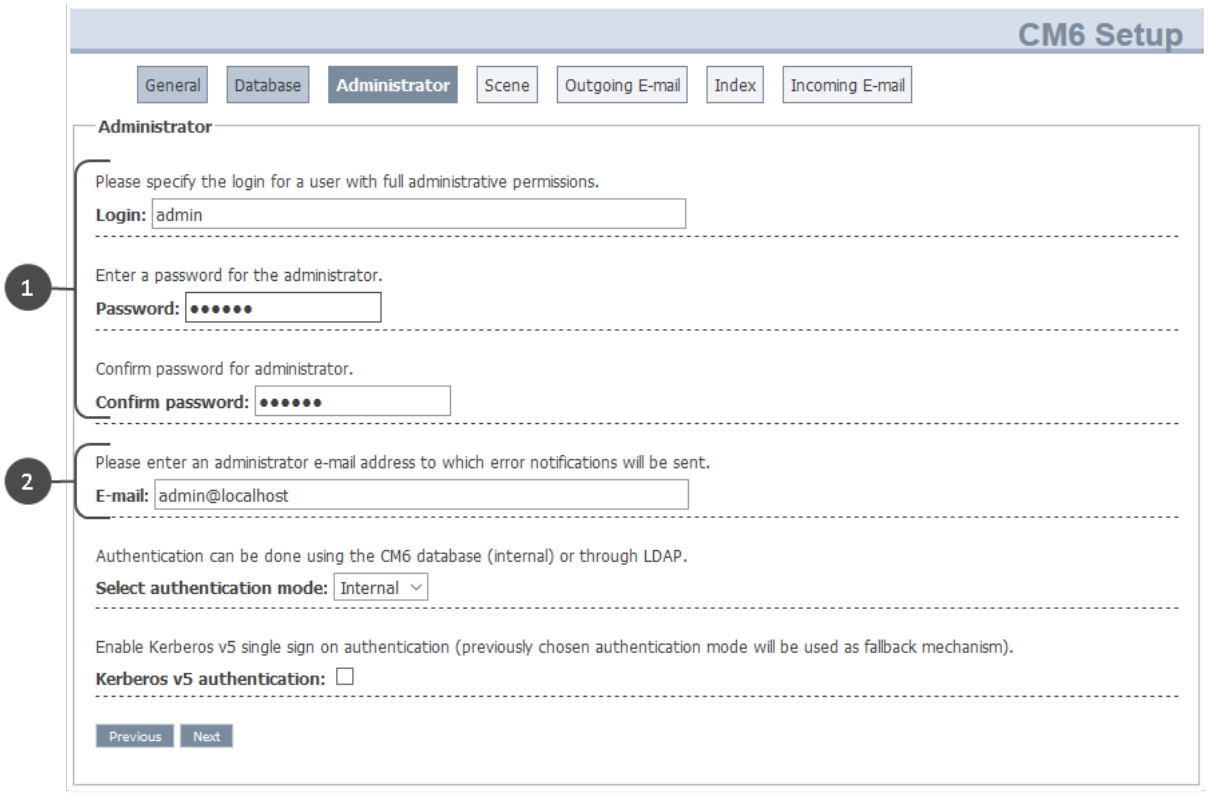


Figure 508: *Admin email address configuration during system set-up*

- Login for the CM admin user (1)
- Global email address of the CM administrator (2)
  This global email address (system property cmas-core-security, admin.email) will be used for all notifications and will be entered automatically for all subsystem-specific email addresses. This means that this admin email address will initially be set automatically for all system properties which contain admin or notification email addresses. The properties can then be changed using the Admin Tool GUI to configure the specific subsystem, e.g., you can configure a specific notification address for DWH operations in the *DWH Configuration* section of the Admin Tool.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

By configuring different admin/notification email addresses for different subsystems, you can spread responsibilities according to responsibilities and roles within your company. For some notifications, even the From address, text, and the subject can be configured.

H.3.2.1 Special Cases Where the Admin Email Address Is Used

Queue-Specific Emails When an Engineer Has Been (De-)Assigned

The emails which are sent when ticket assignment templates are set for a queue (see section *Queue Administration*), use the `admin.email` address as

- `From` address
- `ReplyTo` address

The queue-specific email scripts are not used for those automatic emails, thus it is not possible to change the `From` or `ReplyTo` address for (un-)assignment emails using this script!

> Please note: If the CM system property cmas-core-server, mail.notification.sender is set, the value of this property will be used as `From` address and as `ReplyTo` address for (un-)assignment emails.

H.3.3 Subsystem-Specific Notification Email Addresses

H.3.3.1 DWH (Data Warehouse) - Specific Notification Email Addresses and Email Configurations

System Properties

- **Error**
  - cmas-dwh-server, notification.error.to
    An email will be sent to this address when a DWH operation has failed. If the property is not set, no email will be sent.
  - cmas-dwh-server, notification.error.from
    An email will be sent with this From address when a DWH operation has failed.
  - cmas-dwh-server, notification.error.subject
    Subject for error emails from the DWH
  - cmas-dwh-server, notification.error.description
    Text for error emails from the DWH.
- **Successful**
  - cmas-dwh-server, notification.finished_successfully.to
    An email will be sent to this address when a DWH transfer has been completed successfully, e.g., when the transfer has been completed without errors. If the property is not set, no email will be sent.
  - cmas-dwh-server, notification.finished_successfully.from
    From address for emails from the DWH when a transfer finishes successfully.
  - cmas-dwh-server, notification.finished_successfully.subject
    Subject for emails from the DWH when a transfer finishes successfully.
  - cmas-dwh-server, notification.finished_successfully.description
    Text for emails from the DWH when a transfer finishes successfully.
- **Unsuccessful**
  - cmas-dwh-server, notification.finished_unsuccessfully.to
    An email will be sent to this address when a DWH transfer has been completed, but not successfully, e.g., when the transfer has been completed with errors. If the property is not set, no email will be sent.
  - cmas-dwh-server, notification.finished_unsuccessfully.from
    From address for emails from the DWH when a transfer finishes unsuccessfully.
  - cmas-dwh-server, notification.finished_unsuccessfully.subject
    Subject for emails from the DWH when a transfer finishes unsuccessfully.
  - cmas-dwh-server, notification.finished_unsuccessfully.description
    Text for emails from the DWH when a transfer finishes unsuccessfully.
  - cmas-dwh-server, notification.error.description
    The text for error emails from the DWH

For an overview of all system properties which can be set for DWH notifications, please refer to section CMRF & DWH Configuration of the List of System Properties by Area chapter. All properties which are relevant in this context start with *notification*.

Graphical Configuration

You reach this screen by opening the navigation item *Administration*, navigation group *Data Warehouse*, and clicking the tool icon.

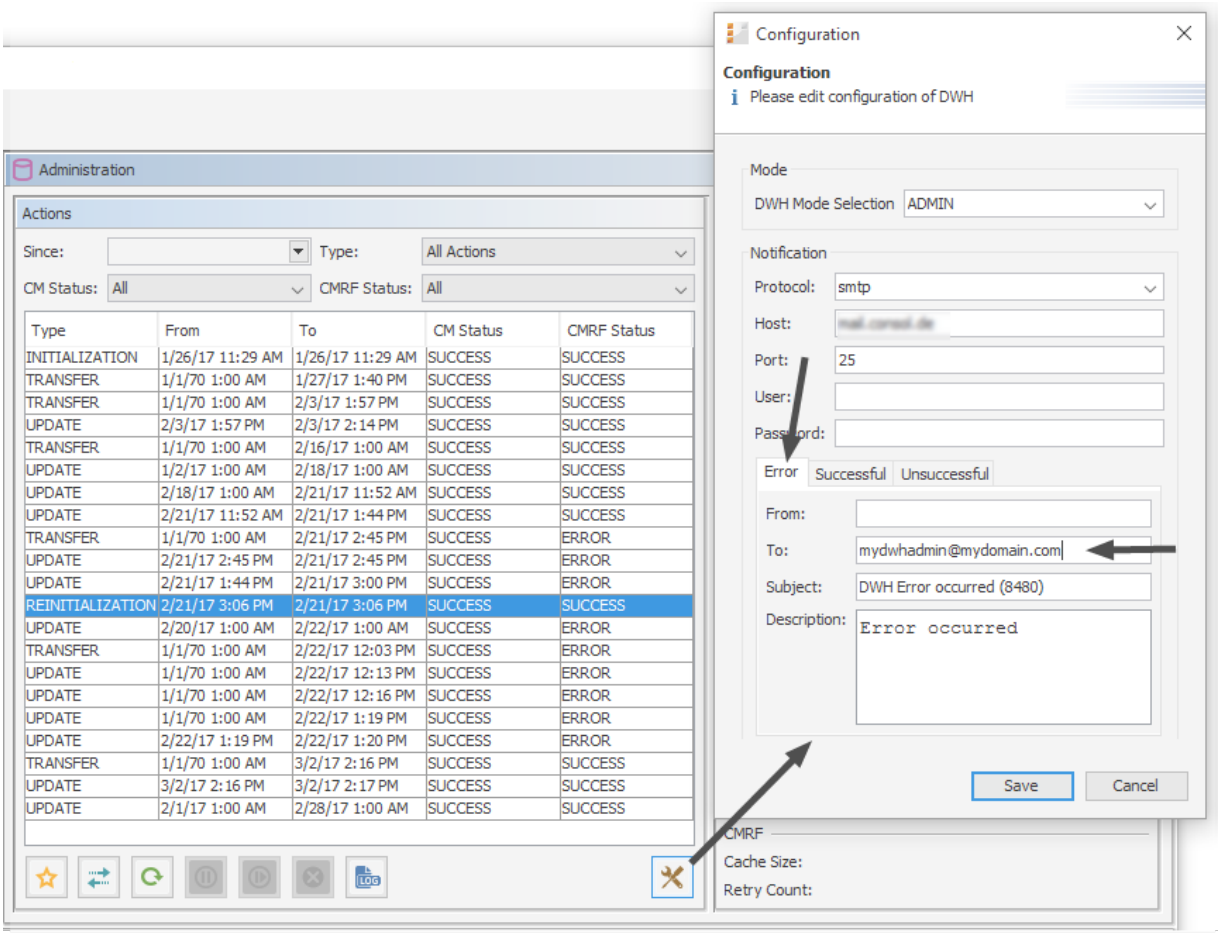


Figure 509: *ConSol CM Admin Tool - Data Warehouse, Administration: Notification address for DWH errors*

The email addresses are checked when you click *Save*. If an email address is not valid, a message is displayed. No mails will be sent to this address.

H.3.3.2 Email-Specific Notification Email Addresses

- cmas-nimh-extension, mail.error.to.address
  An error email is sent to the address in case an email message could not be processed. Starting with CM version 6.10.5.1, an email to this address is also sent when a mail timeout occurs.
  If the value for this property is not set, no email will be sent and an exception will be written into the log file.
- cmas-nimh-extension, mail.attachments.validation.info.sender
  Sets the From header of attachments type *error notification email*.

Please note that the sending is controlled by the system property cmas-nimh-extension, mail.on.error. Only if this property is set to “true”, an error email is sent to the above configured address in case an email message could not be processed.

Graphical Configuration

The value which is entered here (*Error email address*) in the graphical user interface is set as general email notification address.

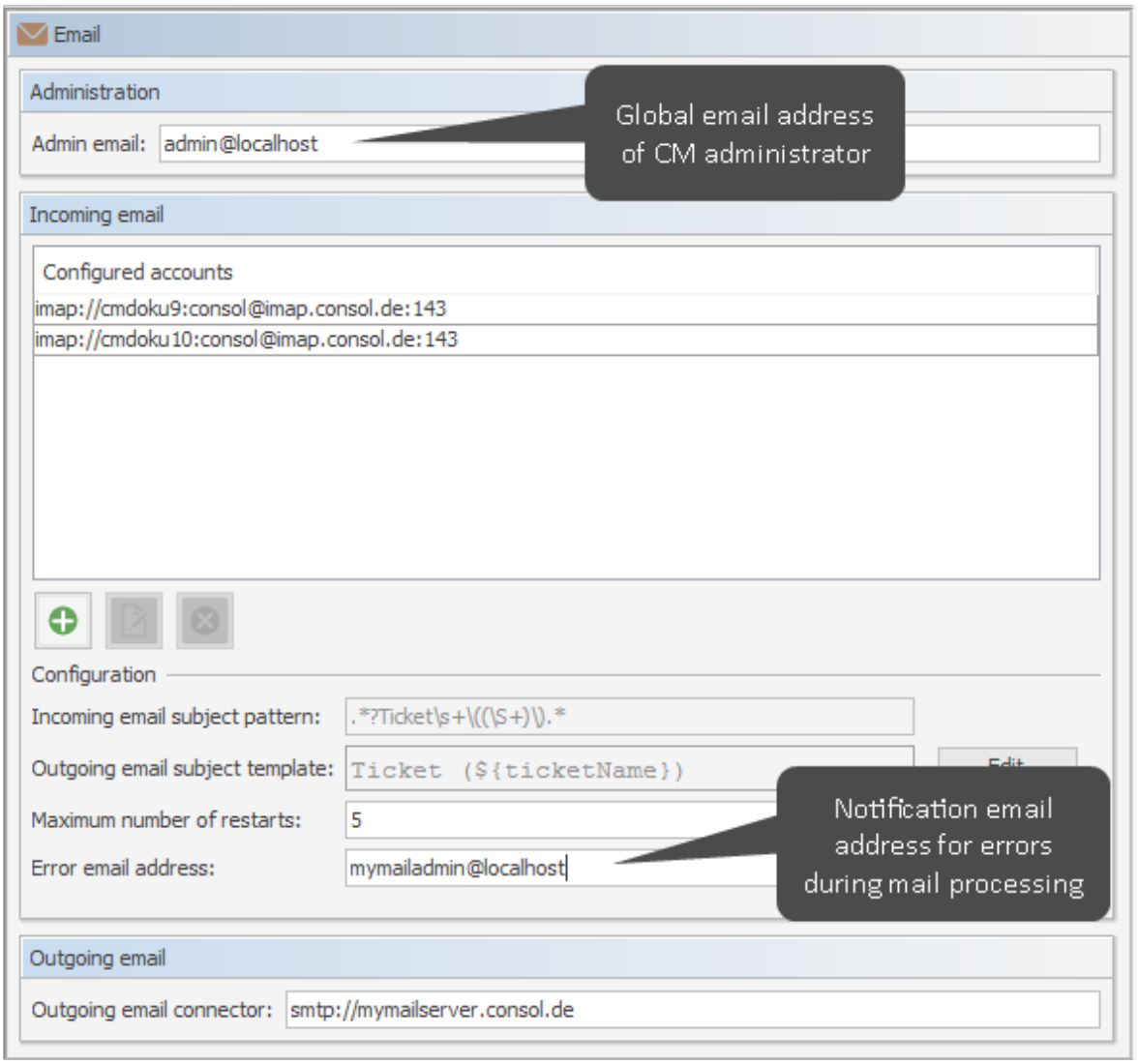


Figure 510: *ConSol CM Admin Tool - Email, Email: Configuration of global admin email address and of notification email address for email processing problems*

Do not mix up the two email addresses which can be configured on the tab of the navigation item *Email*:

- **Admin email**
  is the global administrator email (which has been set during system set-up).
- **Error email address**
  is the address of an email administrator or another person who should receive the error messages if the processing of emails (incoming or outgoing) does not work correctly.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

H.3.3.3 Workflow Engine - Specific Notification Email Addresses

- cmas-workflow-engine, jobExecutor.adminMail
  Email address which will get notified about job execution problems (when retry counter is exceeded). If the property is not set, no email will be sent.
- cmas-workflow-engine, jobExecutor.mailFrom
  Email address which will be set as `From` header during admin notifications.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

H.4 Default Java Imports

In order to have all required classes and methods available in ConSol CM scripts, you might have to import Java/Groovy classes and packages. This is relevant for the Admin Tool and for the Process Designer. However, starting with ConSol CM version 6.11.1, you will only have to import classes and packages which are not used in most standard cases. All packages which contain classes which are used rather frequently are imported implicitly by the Admin Tool and by the Process Designer. In this way, the code validation during writing the scripts as well as the execution of the code during runtime is based on this implicit imports which makes coding CM scripts rather comfortable.

The following classes are implicitly imported in the Admin Tool:

- 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.*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

- 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.*

The following classes are implicitly imported in the Process Designer:

- 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.*

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

H.5 List of Code Examples

In this section, you can find a list of the code examples from this manual.

Code example 1: Example script for ticket list configuration (viewSpecificTicketListConfig.groovy) ...155
Code example 2: Example to get the name of the company's contact ...202
Code example 3: Example 1: Search for the tickets of a contact or of a company ...203
Code example 4: Search for the tickets of the contact who is member of a certain company ...203
Code example 5: Search for contacts of a certain company ...203
Code example 6: Example of a customer template with if statement (has to be written in one line!) ...207
Code example 7: Example of a customer template with empty default value (has to be written in one line!) ...208
Code example 8: Setting a number format, so only one decimal place is displayed ...208
Code example 9: Script which opens the create ticket page with the customer as the main customer, CM version 6.11 and up ...237
Code example 10: Customer action script, CM version 6.11 and up ...245
Code example 11: Script which opens the create ticket page with the customer as the main customer, CM version 6.11 and up ...245
Code example 12: Set a value in customer data and update the unit ...246
Code example 13: Company script which fills some customer data, CM version 6.11 and up ...247
Code example 14: Script which opens the create ticket page with the customer as the main customer, CM version 6.11 and up ...248
Code example 15: Script which opens the company page, CM version 6.11 and up ...250
Code example 16: Customer condition script which checks if reseller relation is present ...251
Code example 17: Open a ticket page in view mode, CM version 6.11 and up ...252
Code example 18: Script which opens a certain web site (URL), CM version 6.11 and up ...253
Code example 19: Customer condition script ...253
Code example 20: Unit Update script where changes are monitored and printed out to server.log ...255
Code example 21: Log output from the script above ...256
Code example 22: Example calendar integration script ...290
Code example 23: Admin Tool script, example of a calendarEventHandlerScript ...298
Code example 24: Script of activity "New IT ticket (Accept ticket)", engineer workload is checked ...317
Code example 25: Script to set the default font for the rich text editor depending on the customer group of the ticket's main contact ...361
Code example 26: JSon object for customer data format in box preview ...391
Code example 27: Example value for customer data preview box layout ...391
Code example 28: Simple graphScript ...399
Code example 29: Simple graphScript with simple node layout definition ...400
Code example 30: Script for customRelationGraph , showGraphicalRelationsOfResource2.groovy ...404
Code example 31: Widget visibility depending on the user's roles ...419

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Code example 32: Dashboard layout depending on the user's roles 419
Code example 33: Chart object 426
Code example 34: Labels object 426
Code example 35: Script for the default chart widget ticketsInViewDataWidget.groovy 428
Code example 36: Script for a sales funnel chart with demo data 430
Code example 37: Script for a KPI widget which counts the cases which have been opened during the last week 433
Code example 38: Script for a table widget 441
Code example 39: Example config.json file 452
Code example 40: welcomePage section of config.json file 455
Code example 41: Excerpt from localization_en.json showing the custom labels created for the queue description 455
Code example 42: newsConfiguration section of config.json file 456
Code example 43: Excerpt from config.json to disable the functionality to change the password 456
Code example 44: Search action script for tickets 507
Code example 45: Search action script for resources 508
Code example 46: Search action script for customers 512
Code example 47: Search condition script for units 512
Code example 48: Positive feedback 518
Code example 49: Negative feedback 518
Code example 50: Admin Tool Script of Type Task 524
Code example 51: Creating a task descriptor 527
Code example 52: Cancelling a task 528
Code example 53: Repeating a task 528
Code example 54: Scheduling a task 528
Code example 55: Repeating a task after an error occurred 529
Code example 56: Workflow activity script for task execution 530
Code example 57: Open the page to create a case 538
Code example 58: Open the ticket page in view mode 538
Code example 59: Open a ticket page with an ACF 539
Code example 60: Company action script ("unit" is the current company) to open the page for creating a new contact 539
Code example 61: Open a URL 539
Code example 62: Relation action script to send an email when a resource-company relation has been created or deleted 543
Code example 63: Control Form condition script which checks if fields are filled 551
Code example 64: Script of a text autocomplete field for a resource field 583
Code example 65: Script which writes values of a text autocomplete field to the server.log file 584

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Code example 66: Script of a text autocomplete field which can be displayed in CM/Track 585
Code example 67: Clone script to reset ticket field for desired deadline 587
Code example 68: Default values script with different values according to the client 588
Code example 69: Email script 592
Code example 70: Script to display a graph with static data based on the Highcharts library 596
Code example 71: Pseudocode to create news items from an RSS channel 598
Code example 72: Script to retrieve news from ticket entries 598
Code example 73: Script for the default chart widget ticketsInViewDataWidget.groovy 601
Code example 74: Script for a sales funnel chart with demo data 602
Code example 75: Script for a KPI widget which counts the cases which have been opened during the last week 603
Code example 76: Script for a table widget 604
Code example 77: Script for a HTML widget displaying weather data 605
Code example 78: Customer template 612
Code example 79: Example of a customer template with if statement (has to be written in one line!) 613
Code example 80: Example of a customer template with empty default value (has to be written in one line!) 614
Code example 81: Setting a number format, so only one decimal place is displayed 614
Code example 82: Content of an assignment template 614
Code example 83: Template to reset the password for the Web Client 615
Code example 84: Admin Tool script, called from a workflow: all time intervals booked on the ticket are printed into the ticket history as comment 678
Code example 85: JBoss 7 696
Code example 86: cm6-kerberos.properties (Windows) 697
Code example 87: cm6-kerberos.properties (Linux) 697
Code example 88: krb5.ini 697
Code example 89: Example of downgrading the session to a user's privileges 713
Code example 90: Example script which implements a webhook service 715
Code example 91: Example of a customer template with if statement (has to be written in one line!) 777
Code example 92: Example of a customer template with empty default value (has to be written in one line!) 777
Code example 93: Setting a number format, so only one decimal place is displayed 777
Code example 94: Resource action script for PC_Desktops to create a new Service Desk ticket for responsible PC contact 791
Code example 95: Resource action script for PC_Desktops to create a new Service Desk ticket for responsible PC contact 796
Code example 96: Resource action script which opens a ticket and uses an ACF 803

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Code example 97: Resource Update script where changes are monitored and printed out to server.log 807
Code example 98: Log output from the script above 808
Code example 99: Task script for archiving a single ticket 866

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

H.6 Trademarks

- The Apache Commons Codec ™ library is a trademark of the Apache Software Foundation. See Apache Commons Codec web page.
- Apache OpenOffice™ – Apache and the Apache feather logos are trademarks of The Apache Software Foundation. OpenOffice.org and the seagull logo are registered trademarks of The Apache Software Foundation. See Apache OpenOffice Trademarks web page.
- Google Maps™ – Google Maps is a trademark of Google Inc. See Google trademark web page for details.
- HAProxy – HAProxy is copyright of Willy Tarreau. See HAProxy website.
- Microsoft® – Microsoft and Windows are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. See Microsoft trademark web page.
- Microsoft® Active Directory® – Microsoft and Microsoft Active Directory are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. See Microsoft trademark web page.
- Microsoft® Exchange Server – Microsoft and Microsoft Exchange Server are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. See Microsoft trademark web page.
- Microsoft® Office – Microsoft and Microsoft Office are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. See Microsoft trademark web page.
- Microsoft® SQL Server® – Microsoft and Microsoft SQL Server are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. See Microsoft trademark web page.
- Microsoft® Windows® operating system – Microsoft and Windows are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. See Microsoft trademark web page.
- Microsoft® Word® – Microsoft and Microsoft Word are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. See Microsoft trademark web page.
- MongoDB® – Mongo, MongoDB, and the MongoDB leaf logo are registered trademarks of MongoDB, Inc.
- NGiNX – NGiNX is copyright of Igor Sysoev and Nginx, Inc. See NGiNX license page.
- OpenStreetMap - OpenStreetMap® is open data, licensed under the Open Data Commons Open Database License (ODbL) by the OpenStreetMap Foundation (OSMF). See OpenStreetMap Copyright and License page.
- Oracle® – Oracle is a registered trademark of Oracle Corporation and/or its affiliates. See Oracle trademarks web page.
- Oracle® WebLogic – Oracle is a registered trademark of Oracle Corporation and/or its affiliates. See Oracle trademarks web page.
- Pentaho® – Pentaho and the Pentaho logo are registered trademarks of Pentaho Inc. See Pentaho trademark web page.
- Vis.js – Vis.js is copyright of Almende B.V. See Vis.js license page.

- Wicket™ - Apache Wicket and Wicket, Apache, the Apache feather logo, and the Apache Wicket project logo are trademarks of The Apache Software Foundation. See, for example, the hint at the bottom of the Wicket home page

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Glossary

A

ACF

ACF is the abbreviation of Activity Control Form. ACFs can be used in workflow activities to force the engineer to fill out certain fields before proceeding.

ACIM

Activity item - entry in the history section of a ticket (e.g., comment, email, attachment, time booking entry).

AD

Microsoft Active Directory - an LDAP-based directory service for Microsoft Windows domain networks.

additional customer

Additional customers are customers (companies or contacts) who are interested in the ticket. They are optional and usually have a role indicating the reason why they were added.

additional engineer

Additional engineers are users who have a specific purpose, which depends on your business process. Usually, they have to carry out certain tasks within the process.

Admin Tool

ConSol CM component, Java application to configure and manage a ConSol CM system.

AJP

Apache JServ Protocol, see, for example https://en.wikipedia.org/wiki/Apache_JServ_Protocol

B

BI

Business Intelligence - methods, technologies, and architectures to transform data into useful information for business purposes.

C

CFEL

Custom Field Expression Language - Java classes and methods of the ConSol CM API to access data in ticket fields, customer fields and resource fields.

CIDR

Classless Inter-Domain Routing, notation for IP address ranges

CM/Archive

CM/Archive is a ConSol CM add-on which allows archiving tickets, so that old tickets can be deleted from the ConSol CM database.

CM/Doc

A standard module of ConSol CM which enables the engineer via ConSol CM Web Client to work with Microsoft Word or OpenOffice documents prefilled with ConSol CM ticket or customer parameters.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

CM/Phone

The ConSol CM module which provides CTI for CM.

CM/Resource Pool

CM/Resource Pool is an optional add-on which allows to store different kinds of objects as resources in ConSol CM.

CM/Track

CM/Track is the portal of ConSol CM. Customers can access their tickets through CM/Track.

CMDB

ConSol CM database - the working database of the CM system.

CMRF

ConSol CM Reporting Framework - a JEE application which synchronizes data between the ConSol CM database and the DWH.

company

The company is the upper hierarchical level of a two-level customer model. A company can have several contacts.

contact

The contact is the lower hierarchical level of a two-level customer model. A contact can only belong to one company.

CRM

Customer Relationship Management. Approach to manage a company's customers, e.g., to collect data from different sources and integrate the data to generate information which allows, e.g., to optimize the services for the customers.

CTI

Computer Telephony Integration - a denomination for any technology that facilitates interaction between a telephone and a computer.

customer

The customer represents the external side of a ticket. It designates the person or object that gave the reason for creating a ticket. A customer can either be a company or a contact.

customer action

Part of the Action Framework. An action which is performed for a customer object, i.e., a contact or company object.

customer data model

The customer data model is the definition of the customers. It determines the available data fields and possible relations.

customer field

A field where data for customers (contacts or companies) can be stored. Similar to ticket fields for ticket data. Previously called Data Object Group Field.

customer field group

A group of fields where data for customers (contacts or companies) can be stored. Similar to ticket field group for ticket data. Previously called Data Object Group.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

customer group

The customer group determines which customer data model is used for its customers and which actions are available.

customer object

A customer (a contact or a company). Formerly called Data Object. The term Unit is used in the programming context.

D

Dashboard

A type of report which integrates data from different sources providing an overall perspective of a certain topic. Often times graphical representation is used.

DWH

Data Warehouse - A database used for reporting and data analysis. In a standard ConSol CM distribution, a DWH is included and only has to be installed and configured.

E

engineer

Engineers are the users who work on the tickets in the Web Client

ERP system

Enterprise Resource Planning - often used for this type of enterprise management software.

ESB

Enterprise Service Bus - a software architecture used for communication between mutually interacting software applications in a service-oriented architecture (SOA).

ETL

Extract Transform Load - extracts data from one source (a database or other source), transforms it, and loads it into a database, e.g., a data warehouse.

F

FlexCDM

Flexible Customer Data Model - the customer data model introduced in ConSol CM in version 6.9. For each customer group, a specific customer data model can be defined.

G

GUI

Graphical User Interface

H

history

The history contains all changes which were carried out for the ticket, customer, or resource.

HMAC

Hash-based Message Authentication Code, message authentication function using hashes

I

IMAP

Internet Message Access Protocol - Internet standard protocol to access email on a remote email server. Can be used as plain IMAP or as secure IMAP

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

(IMAPs). In the latter case, proper certificates are required.

J

Java EE

Java Enterprise Edition

JMS

Java Message Service - Java EE component used to send messages between JMS clients.

JRE

Java Runtime Environment. Provides a Java Virtual Machine for Clients.

K

Kerberos

A network authentication protocol based on (Kerberos) tickets which requires a special infrastructure.

KPI

Key Performance Indicator - parameter used for performance measurement for companies, projects, etc.

L

LDAP

LDAP is the abbreviation of Lightweight Directory Access Protocol. It is a protocol used to manage login information for several applications.

LDAPS

LDAP over SSL

M

mailbox

Destination to which email messages are delivered. Mailboxes are managed on an email server. ConSol CM can access one or more mailboxes to retrieve emails.

main customer

The main customer is the customer who gave the reason for creating the ticket. The main customer is mandatory for a ticket.

Mule

An open source Java-based Enterprise Service Bus (ESB).

N

NIMH

New Incoming Mail Handler - module for retrieving incoming emails.

P

PCDS

Page Customization Definition Section

Pentaho

PentahoTM is a business intelligence (BI) suite which is available in open source and as enterprise editions.

permission

Permissions determine which objects the users can see in the Web Client and which actions they are allowed to perform. Permissions are always granted via roles, i.e., they are not assigned to a single user but to a group of users sharing a common role. Usually these users

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

belong to the same team and/or have similar functions in the company.

POP

Post Office Protocol - Internet standard protocol to retrieve emails from a remote server via TCP/IP. Can be used as plain POP or as secure POP (POPs). In the latter case, proper certificates are required.

portal

CM/Track - provides customer access to ConSol CM.

Process Designer

ConSol CM component used to design, develop, and deploy workflows.

Q

queue

The queue contains thematically related tickets which should be handled in the same way and follow the same business process (workflow). Permissions and other parameters are also defined based on queues.

R

RDBMS

Relational Database Management System - e.g. Oracle ®, MS SQL Server ®, MySQL.

relation

Relations are connections between different data objects in ConSol CM. This can be a relation between two objects of the same type, e.g., between tickets, customers, and resources, or a relation between objects of different types, e.g., between a ticket and a resource or a customer and a resource.

representation

Representations are temporary replacements. The representing user receives a copy of all emails which are sent from the CM system to the represented user's email address. In addition, his ticket list has an option to display the tickets of the represented user.

resource

Resources are objects managed in CM/Resource Pool.

resource action

Part of the Action Framework. An action performed for a resource object.

resource field

A field where resource data can be stored.

resource field group

A group of fields where data for resources can be stored. Similar to ticket field group for ticket data.

resource type

The resource type is the definition of the resources. It determines the available data fields and possible relations and actions.

REST

Representational State Transfer - conventions for transferring data over HTTP connections.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

role

Roles are assigned to users. They define the users' access permissions and views.

S

script

Program written for a specific run-time environment that can interpret and automate the execution of tasks. In ConSol CM, scripts are stored in the Admin Tool and are stored as scripts for activities in workflows.

search action

Part of the Action Framework. An action performed for the result set of a search.

SMTP

Simple Message Transfer Protocol - standard protocol for sending emails.

T

TAPI

Telephony Application Programming Interface - a Microsoft Windows API which provides computer/telephony integration and enables PCs running Microsoft Windows to use telephone services.

TEF

Task Execution Framework - a ConSol CM module which can execute tasks asynchronously.

template

Templates contain predefined and pre-formatted text. They can be used for comments, emails, and documents.

ticket

The ticket is the request of the customer which the engineer works on. It is the object which runs through the business process defined by the workflow.

ticket field

A field where ticket data can be stored. Previously called Custom Field

ticket field group

A group of ticket fields where ticket data can be stored. Previously called Custom Field Group.

ticket list

The ticket list is located to the left of the main working area of the Web Client. It shows tickets which are relevant for the current user.

time booking

Time bookings allow the engineers to register the time they worked on a ticket or project.

U

Unit

Java class which represents a customer object. i.e. a contact is an object of class Unit and a company is also an object of class Unit.

user

Users are the people who work with the ConSol CM system. Users who work on tickets in the Web Client are also called engineers.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

V

view

Views limit the tickets which are shown in the ticket list in the ConSol CM Web Client to those tickets matching specific criteria (scopes from one or more workflows). Views are assigned to roles.

W

Web Admin Suite

ConSol CM component, web application to configure and manage the ConSol CM system. Will replace the Admin Tool.

Web Client

The Web Client is the primary access to the system for the users. They use the Web Client to work on tickets.

Wicket

Apache Wicket is an open source, component oriented, serverside, Java web application framework. See https://wicket.apache.org/ for details information.

workflow

The workflow is the implementation of the business process managed in ConSol CM. It contains a series of steps which are carried out by the engineers.

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

Index

A

ACF 101
  layout 103
acim 346
Action Framework
  script 534
Activity Form 101
activity item 346
address autocomplete 257
annotation 882
  data field 196
  resource field 770
  string field 95
  ticket field 99, 105
  ticket field group 91
authentication
  Kerberos 685
Autocomplete Search 495

B

boolean 93, 194, 765
business calendar 283

C

calendar
  business calendar 283
  holidays 285
  import holidays 285
  Microsoft Exchange 287
  queue 274
  timezone 284
calendar script 289
calendar widget 420
chart widget 424
class of text 299
  availability 303
  CM/Track 305
  color 302
  icon 305
  visibility 303
CM service 486
CM/Archive 863
  permissions 61
CM/Doc 652
  merge fields 659
  permission 654
CM/Phone 870
CM/Resource Pool 731
  label 308
CM/Track 812
  class of text 305
  credentials 702, 816, 827
  FAQ 836
  login 830
  permission 823, 831
  permissions 61
  system access 822
CM/Track user 52
company page 168
contact
  anonymize 175
contact data reference 95, 196, 767
contact page 170
Control Forms 545
CSV export 520
CTI 870

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

customer
customer object 182
deactivate 174
delete 175
template 206
customer action 234
action script 239, 242, 246
Admin Tool 236
condition script 239, 253
Web Client 172
customer action script 246
customer condition script 253
customer data model 158
setup 180
customer field 193
customer field group 192
customer group 214
permissions 218
Quick Search 179
customer group filter 165
customer page
ticket filter 176
customer relation 226
Admin Tool 228
Web Client 173
customer role 222
customer template 206, 611
customization 307

D

dashboard
calendar 420
chart 424
HTML content 431
news 434
table 439
data type 93, 194, 765
boolean 93, 194, 765
contact data reference 95, 196, 767
date 93, 194, 766
enum 93, 194, 766
fixed point number 94, 195, 767
list 93, 194, 766
long string 95, 195, 767
MLA field 95, 196, 767
number 94, 195, 767
short string 95, 196, 767
string 94-95, 195-196, 767
struct 94, 194, 766
Data Warehouse 466
date 93, 194, 766
date format
acim 347-350
Dependent Enum
ticket field group 90
Detailed Search 491
export 520
results 492
DWH 466
admin mode 473
internationalization 484
live mode 473
synchronization 475
task 477
ticket history 138
transfer mode 481

E

email 554, 722
automatic 555
backup 565
encryption 568

incoming 559
manual 554
NIMH 561, 567
outgoing 560
TO-address 353
email backup 565
encryption
client certificate 570
server certificate 570
engineer 49
label 308
roles 52
view criteria 53
engineer account 50
engineer function 80
permissions 83
enum 93, 111, 194, 766
group 113, 115
type 113
value 113, 116
enum parameter 116
export 619, 621
configuration 622
runtime data 621
single ticket 621
external interfaces 708

F

FAQ 836
file structure 724
fixed point number 94, 195, 767
Flexible Customer Data Model 158

G

generic widget 431
GUI 31

H

HTML content widget 431

I

icon 38
import 619, 623
indexing 497

K

Kerberos 685
Kerberos Principal Name 51

L

label 308
language 269
LDAP 723
LDAP ID 51
LDAPS 684, 706, 820
license 627
list 93, 194, 766
log file 725
long string 95, 195, 767

M

main customer 171
message
label 314
Microsoft Exchange calendar 287
script 289
MLA 120
MLA field 95, 196, 767
Multi Level Attribute 120

N

news widget 434
NIMH 561, 590

number 94, 195, 767

P

page customization 322
- scope 328, 334
- subscope 329
- type 328

password policy 681
password reset 682
- Web Client 615

permissions
- administrator 59
- customer group 62
- template 60
- workflow 60

PostActivityExecutionScript 606
project 280

Q

queue 271
- calendar 274
- FAQ 274
- script 276

Quick Search 489
- customer group 179

R

relations
- customers 226
- graphical representation 319
- resources 780

reporting 467
resource
- Detailed Search 750
- permissions 809
- Quick Search 750
- template 776

resource action 789
- Admin Tool 790
- Web Client 747

resource category 754
resource dashboard 810
resource field 764
resource field group 761
resource mode 758
resource model 733
- setup 753

Resource Pool Dashboard 810
resource relation 780
- Admin Tool 782
- Web Client 744

resource template 611, 776
resource type 756
resource type page 742
role 54
- administrator 59
- engineer functions 68
- global permissions 59
- queue permissions 56
- resource permissions 66
- views 67

S

scenario 619
script 574
- clone 586
- default values 587
- Dependent Enum 589
- email 590
- feedback 518
- PostActivityExecutionScript 606
- queue 276
- task 524

workflow 606
search action 505
customer 511
resources 508
ticket 506
search result
export 520
service
CM 486
short string 95, 196, 767
skinning 307
sorted list 111
SSO 685
string 94, 195, 767
struct 94, 194, 766
system architecture 716
system properties 632, 902

T

table widget 439
Task Execution Framework 522
task script 524
TEF 522
template 608
customer 611
password reset 615
resource 611
ticket assignment 614
Template Manager 640
text template 640
ticket
delete 462
label 308
reopen 462
ticket data
layout 100
ticket field 87, 92
annotations 105
ticket field group 86, 89
Dependent Enum 90
ticket history 127
display mode 135
DWH 138
visibility 129, 135-136
ticket icon
color 117
ticket list 345
assign link 345
customer 345
loading 345
time booking 667
activate 354
automatic 674
engineer profile 671
manual 668

V

view 72
dynamic criterion 76
queue filter 74
scope filter 74
static criterion 75
visibility 129, 136

W

webhooks 709
widget
calendar 420
chart 424
generic 431
HTML content 431
news 434
table 439

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH

workflow 273

CM/ is a registered trademark of ConSol Consulting & Solutions Software GmbH