ConSol CM

Administrator Manual

ConSol Software GmbH
Franziskanerstraße 38
D-81669 München

Tel. +49-89-45841-100
Fax. +49-89-45841-116
vertrieb@consol.de
www.consol.de

ConSol*CM Administrator Manual (Version 6.9, up to 6.9.4.3)

Table of Contents

1 Introduction 10
1.1 ConSol*CM 11
1.2 ConSol*CM Documents 12
1.3 This Book's Structure 13
1.4 Layout Explanations 14
1.5 Basic Principles of ConSol*CM6 15
1.5.1 System Components from a User's Point of View 15
1.5.2 Basic Technical ConSol*CM Principles and Objects 16
1.5.3 ConSol*CM from a System Administrator's Point of View 20
2 Start of the Admin Tool 21
2.1 Login 22
2.2 Troubleshooting: When the Admin Tool Does Not Start 24
2.2.1 Correct Process 24
2.2.2 Process with Errors 27
2.3 How to Use the Admin Tool 28
3 CM6 Administrator Manual 6.9.4 - --Power User Section-- 31
4 Engineer Administration 32
4.1 Introduction to Engineer Administration 33
4.2 Engineer Administration Using the Admin Tool 34
4.2.1 Create or Edit an Engineer Account 35
4.2.2 Delete an Engineer Account 38
4.2.3 Disable or Enable an Engineer Account 38
4.2.4 Tab Roles - Assign Roles to an Engineer Account 38
4.2.5 Tab View Criteria - Define Engineer-Specific View Criteria 39
4.3 Related Topics 40
5 Role Administration 41
5.1 Introduction to Role Administration 42
5.2 Role Administration Using the Admin Tool 43
5.2.1 Create a Role 44
5.2.2 Tab Engineer Functions 51
5.2.3 Delete a Role 52
5.2.4 Copy a Role 52
5.2.5 Edit a Role 52
5.3 Related Topics 53
6 View Administration 54
6.1 Introduction to View Administration 55
6.2 View Administration Using the Admin Tool 56
6.2.1 Create a View 56
6.2.2 Edit a View 61
6.2.3 Delete a View 62
6.2.4 Copy a View 62
6.3 Related Topics 63

7 Queue Administration 64
- 7.1 Introduction to Queue Administration 65
- 7.2 Queue Administration Using the Admin Tool 66
  - 7.2.1 Filter the Queue List 66
  - 7.2.2 Create a Queue 67
  - 7.2.3 Edit a Queue 70
  - 7.2.4 Delete a Queue 71
  - 7.2.5 Copy a Queue 71
  - 7.2.6 Enable or Disable a Queue 71
- 7.3 Related Topics 72

8 Customer Data Model Section 73

9 CM6 Administrator Manual 6.9.4 - The CM Customer Data Model - FlexCDM 74
- 9.1 The CM Customer Data Model: FlexCDM 75
- 9.2 Introduction to FlexCDM 76
  - 9.2.1 FlexCDM at a Glance 76
  - 9.2.2 Introduction to FlexCDM Objects 79
  - 9.2.3 Management of FlexCDM Objects Using the Admin Tool 81
- 9.3 A Short Introduction to FlexCDM-Specific Web Client Functionalities 82
  - 9.3.1 Introduction 82
  - 9.3.2 Working with the ConSol*CM Web Client with FlexCDM 83
- 9.4 Setting Up the Customer Data Model 96
  - 9.4.1 Introduction to Setting Up the Customer Data Model Based on FlexCDM 96
  - 9.4.2 Managing Contacts and Companies Using the Admin Tool 97
- 9.5 Data Object Group Field Management and GUI Design for Customer Data 106
  - 9.5.1 Introduction 106
  - 9.5.2 Defining Data Object Group Fields for Customer Data Using the Admin Tool 107
  - 9.5.3 Scripting Using Objects from the FlexCDM 113
  - 9.5.4 Changes in Scripting from Consol*CM Version 6.8 to Version 6.9 114
  - 9.5.5 New Objects in ConSol*CM 6.9 and Up 120
- 9.6 Templates for Customer Data 121
  - 9.6.1 Introduction to Using Templates for the Display of Customer Data 121
  - 9.6.2 Coding Templates 122
  - 9.6.3 Template Types 124

10 Managing Customer Groups 132
- 10.1 Basic Principle for Customer Data Models and Customer Groups 133
- 10.2 Managing Customer Groups Using the Admin Tool 134
  - 10.2.1 Customer Groups List 134
  - 10.2.2 Customer Group Details 135
  - 10.2.3 Creating a New Customer Group 135
  - 10.2.4 Editing a Customer Group 136
  - 10.2.5 Deleting a Customer Group 136
  - 10.2.6 Disabling and (Re-)Enabling a Customer Group 136
- 10.3 Assigning Access Rights for Customer Groups 137

11 Customer (Data Object) Relations 140
- 11.1 Introduction to Customer (Data Object) Relations 141
- 11.2 Management of Customer Relations Using the Admin Tool 142

11.3 Creating Customer Relations Using the Web Client 145
11.4 Scripting Using Relations 146
12 Action Framework 147
12.1 Introduction to Data Object Actions 148
12.2 Managing Data Object Actions Using the Admin Tool 150
12.2.1 Step 1: Write the Data Object Action Script 150
12.2.2 Step 2: Create Data Object Action(s) Which Use the Script 151
12.2.3 Step 3: Assign Data Object Action(s) to Customer Group(s) 152
12.3 Using Data Object Actions as an Engineer (User) 154
12.4 Examples for Data Object Action Scripts 155
12.4.1 Example 1: Simple Manual Action 155
12.4.2 Example 2: New Ticket for Contact 158
12.5 Scripts for the Action Framework: Programming Data Object Actions 159
12.5.1 Data Object Action Scripts 159
12.5.2 Data Object Condition Scripts 166
13 CM6 Administrator Manual 6.9.4 - Additional User Attributes 167
13.1 Additional User Attributes: Customer Roles, Engineer Functions , and Projects 168
13.1.1 Introduction 168
13.1.2 Tab Card Customer Roles 169
13.1.3 Tab Engineer Functions 171
13.1.4 Tab Projects 175
13.1.5 Related Topics 176
13.2 Tab Autocomplete Address 177
13.2.1 Introduction 177
13.2.2 Switch on the Autocomplete Address Feature using the Admin Tool 179
13.2.3 Import Zip/City/Address Data into the CM Database 181
13.2.4 Define the Autocomplete Strategy using the ConSol*CM Admin Tool 182
13.2.5 Edit an Autocomplete Strategy Definition 184
13.2.6 Delete an Autocomplete Strategy or Autocomplete Address Fields 185
14 Ticket Data Model and GUI Designer Section 186
15 Custom Field Administration 187
15.1 Introduction to Custom Field Administration 188
15.2 Custom Field Administration Using the Admin Tool 190
15.2.1 Tab Ticket Data 190
15.2.2 Tab Activity Form Data 197
15.2.3 Frequently Used Annotations 201
15.3 Related Topics 206
16 Managing Sorted Lists: Enum Administration 207
16.1 Introduction to Enum Administration 208
16.2 Enum Administration Using the Admin Tool 210
16.2.1 Enum Types 210
16.2.2 Enum Groups 212
16.2.3 Enum Values 213
16.2.4 Placing an Enum in the Data Model 216
16.3 Related Topics 218
17 MLA Administration 219

17.1 Introduction to MLA Administration 220
17.2 MLA Administration Using the Admin Tool 222
17.2.1 Create an MLA 222
17.2.2 Edit an MLA 225
17.2.3 Delete an MLA 225
17.2.4 Enable or Disable an MLA 226
17.3 Related Topics 227
18 Configuration of the Web Client Dashboard 228
19 CM6 Administrator Manual 6.9.4 - Page Customization 231
19.1 Page Customization 232
19.1.1 General Introduction to Page Customization 232
19.1.2 Page Customization in the Web Client 233
19.1.3 Page Customization Using Attributes 240
19.1.4 Order and Priorities of Page Customization 278
19.2 Page Customization for the Web Client Dashboard 280
19.2.1 Introduction 280
19.2.2 Definition of the Overall Dashboard Layout 282
19.2.3 Configuration of Widgets 283
19.2.4 Print Functionality in the Web Client Dashboard 296
19.2.5 3D-Rendering for Graphics (Chart Widgets) 297
19.2.6 Drilldown Functionality for Graphics (Chart Widgets) 298
20 Expert Section 300
21 Ticket Administration 301
21.1 Introduction to Ticket Administration 302
21.2 Ticket Administration Using the Admin Tool 303
21.2.1 Delete or Reopen Tickets 303
21.2.2 Switching off the Delete Functionality Using a System Property 304
21.2.3 Search Tickets 305
21.3 Related Topics 308
22 CM6 Administrator Manual 6.9.4 - Configuration 309
22.1 Configuration 310
22.1.1 Introduction to the Configuration Page 310
22.1.2 Perform Configuration Operations Using the Admin Tool 311
22.1.3 Related Topics 312
22.2 Tab General 313
22.2.1 The Use of Locales 314
22.3 Tab CM Services 315
22.4 Tab E-Mail 317
22.4.1 Introduction to E-Mails in ConSol*CM 317
22.4.2 E-Mail Configuration Using the Admin Tool 320
22.4.3 E-Mail Duplication in the ConSol*CM Web Client 331
22.4.4 Related Topics 331
22.5 Tab E-Mail Backups 332
22.5.1 Introduction 332
22.5.2 E-Mail Backups in the Admin Tool 332
22.5.3 Management of E-Mail Backups with ESB/Mule 334

22.5.4 Management of E-Mail Backups with NIMH 334
22.6 Tab Licence 335
22.6.1 General Information about Licenses in ConSol*CM 335
22.6.2 Managing the ConSol*CM License Using the Admin Tool 335
22.7 ConSol*CM ESB Services 338
22.7.1 Introduction to ESB Services 338
22.7.2 Starting and Stopping ESB Services Using the Admin Tool 341
22.8 Tab Business Calendars 342
22.8.1 Configuration of Business Calendars in the Admin Tool 343
22.9 Tab Classes of Text 348
22.9.1 Installing a New Class of Text 349
22.9.2 Edit a Class of Text 353
22.9.3 Delete a Class of Text 353
22.9.4 Setting the Default Class of Text 353
22.9.5 Working With Classes of Text in Scripts 353
22.10 Tab Ticket History 354
22.11 Search Configuration and Indexer Management (Tab Index) 356
22.11.1 Search Modes 356
22.11.2 Fields which can be Searched 358
22.11.3 Administrator Tasks Concerning the Indexer 361
22.11.4 Introduction to the ConSol*CM Indexer 361
22.11.5 Indexer and Index Management Using the Admin Tool 362
22.11.6 Indexer and Index-Relevant System Properties 365
22.11.7 CM Indexer Services 366
22.11.8 Systems with more than one Indexing Server 367
23 Deployment 368
23.1 Introduction to Deployment in the Admin Tool 369
23.2 Introduction to ConSol*CM Scenarios 370
23.3 Tab Deployment 371
23.3.1 Export 371
23.3.2 Import 374
23.3.3 Workflow Deployment (for Deployment Error Recovery Only) 375
23.4 Related Topics 376
24 The Task Execution Framework (TEF) 377
24.1 Introduction 378
24.2 Admin Tool Scripts of Type Task 380
24.2.1 Introduction to Admin Tool Scripts of Type Task 380
24.2.2 Example Admin Tool Script of Type Task 380
24.2.3 Execution of a Task Using the Admin Tool 380
24.3 Programming with Tasks 383
24.3.1 Introduction 383
24.3.2 Coding Examples 383
24.3.3 Example for the Use of a Task Script 385
25 CM6 Administrator Manual 6.9.4 - Script and Admin Tool Template Administration 387
25.1 Script and Admin Tool Template Administration 388
25.2 Introduction to Scripts in the Admin Tool 389

25.2.1 The Source Code Editor 392
25.2.2 Script Types 393
25.3 Introduction to Templates in the Admin Tool 413
25.3.1 The Admin Tool Template Editor 413
25.3.2 Working with Admin Tool Templates 414
26 CM6 Administrator Manual 6.9.4 - Working with Text Templates 419
26.1 Working with Text Templates 420
26.2 The ConSol*CM Template Manager 421
26.2.1 Introduction to the Work with E-Mail and Ticket Text Templates 421
26.2.2 Introduction to the Template Manager 425
26.2.3 Work with the Template Manager 426
26.2.4 Migrating Templates from CM Version 6.8 and Less to CM Version 6.9 and Up 449
26.2.5 Page Customization for E-Mail Template Functionalities 449
26.3 CM.Doc 450
26.3.1 Introduction to CM.Doc 450
26.3.2 Requirements for Using CM.Doc 451
26.3.3 Availability of CM.Doc 451
26.3.4 Configuring the ConSol*CM System for CM.Doc 451
26.3.5 Creating an Engineer Role with Permissions for the Doc Template Manager 452
26.3.6 Creating MS Word Templates and Making Them Available 454
26.3.7 Using MS Word Templates from within the Web Client 464
27 Time Booking Using ConSol*CM 467
27.1 General Introduction to Time Booking Using ConSol*CM 468
27.2 Manual Time Bookings 469
27.2.1 Configuration of Manual Time Booking Using the Admin Tool 469
27.2.2 Manual Time Booking from a User's Point of View (Web Client) 470
27.2.3 Manual Time Bookings on the Engineer Profile Page 473
27.3 Automatic Time Bookings (Available in CM Version 6.9.4.2 and Up) 474
27.3.1 Introduction to Automatic Time Booking 474
27.3.2 Configuration of Automatic Time Booking 474
27.3.3 Automatic Time Bookings on the Engineer Profile Page 475
27.4 DWH Reports 477
27.5 Page Customization for Time Booking 478
28 CM6 Administrator Manual 6.9.4 - Authentication Methods for Engineers in the CM Web Client 479
28.1 Authentication Methods for Engineers in the Web Client 480
28.2 Introduction to ConSol*CM LDAP Authentication 481
28.2.1 Configuring the System to Enable LDAP Authentication 482
28.2.2 Managing Engineer Accounts for LDAP Authentication 484
28.2.3 Using LDAPS (LDAP over SSL) 485
28.3 Single Sign-On with ConSol*CM Using Kerberos 487
28.3.1 Configuration of Kerberos Single Sign-On 487
28.3.2 Setting Up the System 488
28.3.3 Using the System 493
29 CTI with ConSol*CM: CM.Phone 495
29.1 Introduction to CM.Phone 496
29.1.1 Incoming Calls 496

29.1.2 Outgoing Calls 497
29.2 CM.Phone Set-Up 498
29.2.1 System Requirements 498
29.2.2 Components Required for CM.Phone Set-Up 498
29.2.3 Installing CM.Phone on the Application Server 498
29.2.4 Installing CM.Phone on Each Windows Client 501
29.3 Configuration of CM.Phone in the Admin Tool 504
29.3.1 Set the Annotations for the Data Object Group Fields Which Contain Phone Numbers 504
29.3.2 Configure the Admin Tool Templates for Customer Data for Each Customer Group 505
29.3.3 Configure the Phone Number Format for Each Customer Group 507
29.3.4 Set the System Properties 509
29.3.5 Change the Prefix for Outgoing Calls 510
29.4 Troubleshooting 511
29.4.1 Logging 511
29.4.2 Registration as phone: protocol handler 511
30 Data Warehouse (DWH) Management 513
30.1 Introduction 514
30.1.1 Data Warehouse 514
30.1.2 ConSol*CM Data Warehouse and ConSol*CM Reporting Framework 514
30.2 DWH Management Using the Admin Tool 516
30.2.1 DWH Administration Overview 516
30.2.2 Basic DWH Configuration 517
30.2.3 Initialization of the DWH 520
30.2.4 First DWH Synchronization 521
30.2.5 DWH Synchronization During System Operation 521
30.2.6 DWH Tasks 522
30.2.7 DWH Troubleshooting and Repair 522
30.3 DWH-Related System Properties 524
31 CM6 Administrator Manual 6.9.4 - The Customer Portal - CM.Track 525
31.1 The Customer Portal: CM.Track 526
31.2 System Access for CM.Track Users (Customers) 528
31.2.1 Precondition 528
31.2.2 CM.Track Technical Background 528
31.2.3 General Principle of System Access via CM.Track 528
31.2.4 Defining the User Profiles/Access Permissions for CM.Track 529
31.2.5 Defining the Data Object Group Fields for CM.Track Login and Password 530
31.2.6 Granting Access to CM.Track for Customers 532
31.2.7 Customer Login to the System 532
31.2.8 Extended Customer Permissions to See Company Tickets 533
31.3 CM.Track: Authentication Modes for the Portal 534
31.3.1 Introduction to Authentication Modes in CM.Track 534
31.3.2 Definition of the CM.Track Authentication Mode 534
31.3.3 DATABASE Authentication Mode 535
31.3.4 LDAP Authentication Mode 535
31.3.5 Mixed Authentication Mode 537

31.3.6 Logging of LDAP Login Attempts in CM.Track 538
31.3.7 Using LDAPs for Authentication 538
31.4 FAQs in CM.Track 539
31.4.1 Introduction to FAQs in CM.Track 539
31.4.2 Configuring the ConSol*CM System to Allow FAQ Search in CM.Track 539
31.4.3 FAQ Search in CM.Track from a Customer's Point of View 542
31.4.4 More Complex Solutions for Managing FAQs 543
32 System Overview 545
32.1 System Architecture 546
32.1.1 Introduction to ConSol*CM System Architecture 546
32.1.2 Basic System Architecture 546
32.1.3 Components for E-Mail Interactions 547
32.1.4 System Architecture with Reporting Infrastructure 547
32.2 Short Overview of the File Structure 552
32.2.1 ConSol*CM Data Directory 552
32.2.2 JBoss 5 Application Server File Structure 553
32.2.3 JBoss 7 Application Server File Structure 554
32.2.4 Oracle WebLogic Application Server File Structure 555
32.2.5 Log Files 556
33 Appendix A - List of Annotations (up to Version 6.9.4) 559
33.1 Alphabetical List of Field Annotations 560
33.2 Alphabetical List of Group Annotations 574
34 Appendix B - Glossary 578
35 Appendix C - System Properties 585
35.1 System Properties Ordered by Module 586
35.2 System Properties Ordered by Property Name 646
36 Appendix D (Important System Properties, Ordered by Area of Application) 706
36.1 CMRF & DWH Configuration 707
36.2 Indexer & Search Configuration 715
36.2.1 Indexer 715
36.2.2 Search Results 720
36.3 LDAP Configuration 722
36.3.1 LDAP Configuration (If LDAP Is Used as Authentication Mode in the CM Web Client) 722
36.3.2 LDAP Configuration (If LDAP Is Used as Authentication Mode in CM.Track) 724
36.4 E-Mail Configuration 727
36.4.1 Outgoing E-Mail 727
36.4.2 Incoming E-mail 729
36.4.3 E-Mail Encryption (Outgoing and Incoming) 748
36.5 Activity Interval Configuration 751
37 Appendix E - Trademarks 753
38 Index 754

1 Introduction

- ConSol*CM
- ConSol*CM Documents
- This Book's Structure
- Layout Explanations
- Basic Principles of ConSol*CM6
    - System Components from a User’s Point of View
    - Basic Technical ConSol*CM Principles and Objects
        - The Ticket
        - The Workflow
        - The Queue
        - The Customer
        - The Engineer
        - ConSol*CM Dogma
    - ConSol*CM from a System Administrator's Point of View

1.1 ConSol*CM

ConSol*CM is a **customer centric business process management software**. Using ConSol*CM you can control and steer business processes with a strong focus on human communication and interaction, e.g. user help desk, customer service processes, marketing and sales or ordering processes. Basically, every process that is in operation in a company can be modeled and brought to life with ConSol*CM6.

When you read this manual, your company is presumably using ConSol*CM6 as a process management tool and it is your job to administer the system. The book will help you get a quick overview of the most important components of ConSol*CM and will also provide a deeper and more detailed introduction to all aspects of the CM administration.

1.2 ConSol*CM Documents

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.
- **Process Designer Manual -** a guideline for workflow developers about the graphical user interface of the Process Designer and how to program workflow scripts.
- **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.
- **Set-Up Manual** - a technical description for CM set-up in different IT environments. For expert CM administrators.
- **User Manual** - an introduction to the ConSol*CM Web Client for end users.
- **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.

For you as a CM administrator the book you are reading, the **Administrator Manual**, provides the required information to cover all fields of CM administrator work, see the following section for more information about the sections of the book. Please make sure that all system requirements have been taken into consideration.

1.3 This Book's Structure

First, some basic principles of the ConSol*CM6 application will be explained to provide the theoretical background you need to become a CM administrator.

The *Overview* section explains how to get access to the system.

The following four sections explain the features and functionalities of the main administration application, the ConSol*CM Admin Tool. You can decide which section(s) you need:

1. **Power User Section**
   In this section (see Power User Section), the user, role, view, and queue administration are explained, i.e. the basic operations you need in everyday work life. As a team manager you might want to learn more about those features, without necessarily *going deeper*.
2. **Customer Data Model Section**
   Here (see Customer Data Model Section) you get to know the principle of the ConSol*CM customer data model, *FlexCDM*, and you learn how to manage all components which are related to it.
3. **Ticket Data Model and GUI Designer Section**
   Here (see Ticket Data Model and GUI Designer Section), you learn how to design the data models that form the basis for ticket data management and how to display those data on the GUI. Furthermore, the Web Client Dashboard is explained.
4. **Expert Section**
   Here (see Expert Section), the system parameter management is explained and the scripts and templates that steer the system *behind the scenes*. Furthermore, the chapters in this section provide information about the system management parameters concerning the operating system, like log file management or indexer files.

In the appendix, you find lists of all important terms that are used in the book (glossary), of all annotations (important for the GUI design), and properties (important for the CM system management). Please see also the trademarks page.

1.4 Layout Explanations

In order to emphasize and/or mark a section, icons are used.

Information:

This is an additional information.

Attention:

This is an important note. Be careful here!

Warning:

This is a warning!

Tip:

This is a recommendation from our everyday consulting life.

1.5 Basic Principles of ConSol*CM6

1.5.1 System Components from a User's Point of View

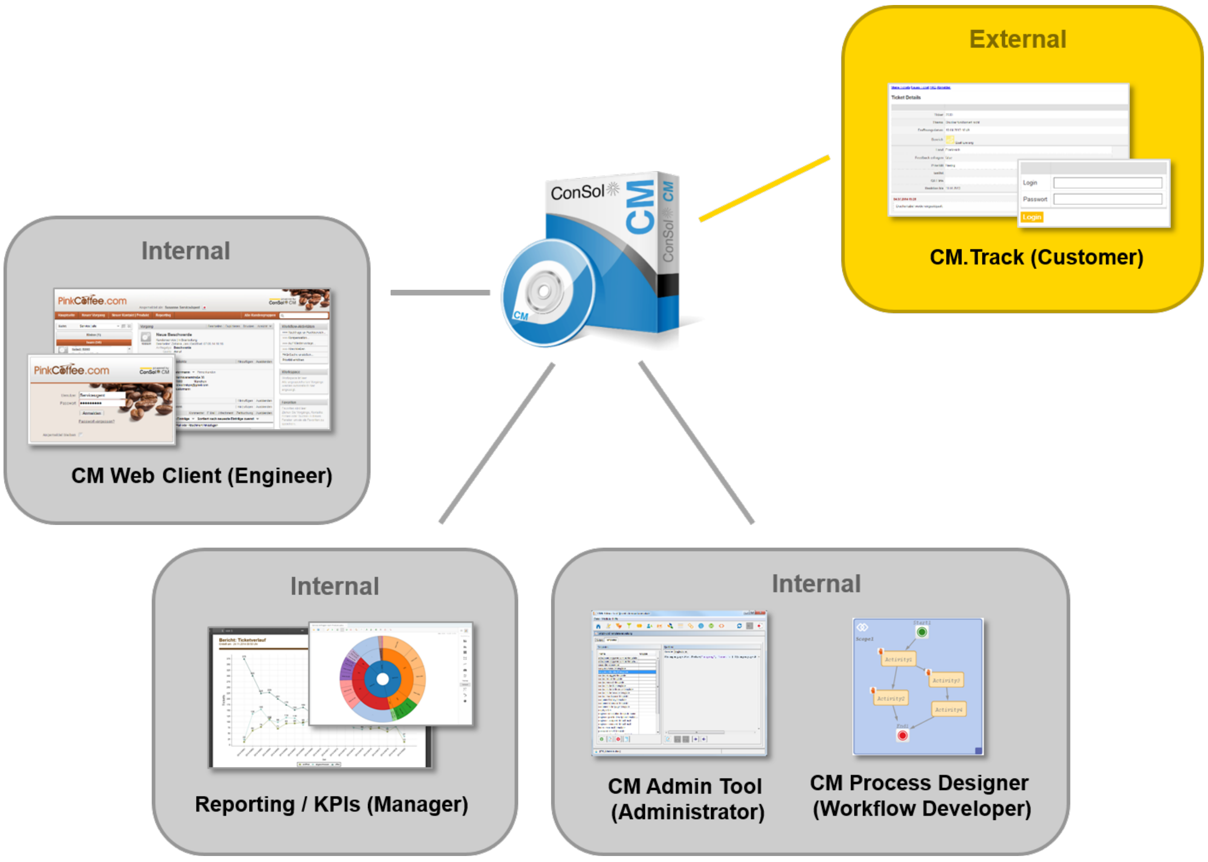

External
CM.Track (Customer)
Internal
CM Web Client (Engineer)
Internal
Reporting / KPIs (Manager)
Internal
CM Admin Tool (Administrator)
CM Process Designer (Workflow Developer)

Fig. 1: ConSol*CM System Components

- **Web Client**
  The primary access to the system for engineers
- **Portal**
  CM.Track, the primary access to the system for (internal or external) customers
- **Admin Tool**
  For all system configuration tasks
- **Process Designer**
  For the workflow design and implementation

The default scope of delivery also includes a data warehouse (DWH) that allows reporting about the data of your tickets.

Furthermore, ConSol*CM is not a stand-alone application but can be easily integrated into your company's IT infrastructure, e.g. using Web Services and/or an Enterprise Service Bus (ESB).

For a detailed explanation of the system components seen from a more technical point of view, please refer to the system administrator's section (System Overview).

1.5.2 Basic Technical ConSol*CM Principles and Objects

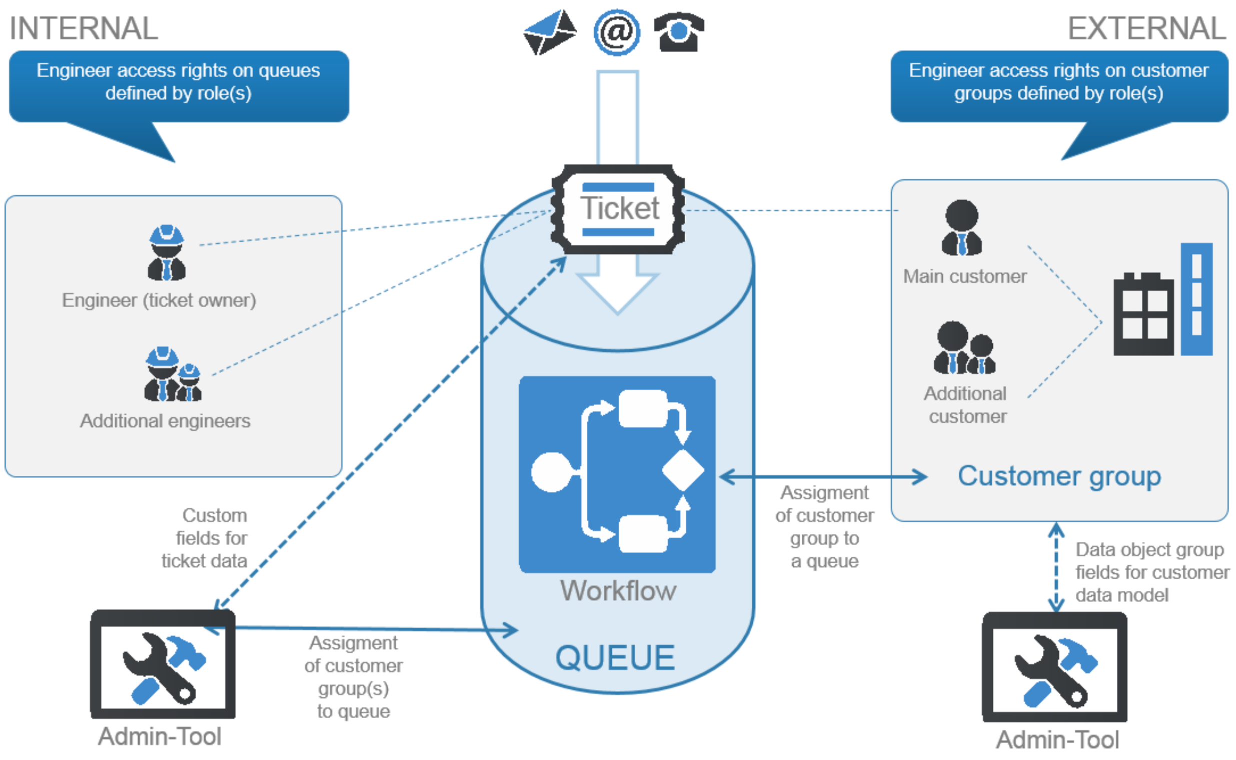

INTERNAL
Engineer access rights on queues defined by role(s)
Engineer (ticket owner)
Additional engineers
Custom fields for ticket data
Admin-Tool
Assigment of customer group(s) to queue
Ticket
Workflow
QUEUE
EXTERNAL
Engineer access rights on customer groups defined by role(s)
Main customer
Additional customer
Customer group
Assigment of customer group to a queue
Data object group fields for customer data model
Admin-Tool

Fig. 2: ConSol*CM Basic Principles

The Ticket

ConSol*CM can manage incidents, service cases, and/or other requests of internal and/or external customers. Every request is managed as a ticket that is created in the system, passes through the desired process, and is then (hopefully) solved and closed.

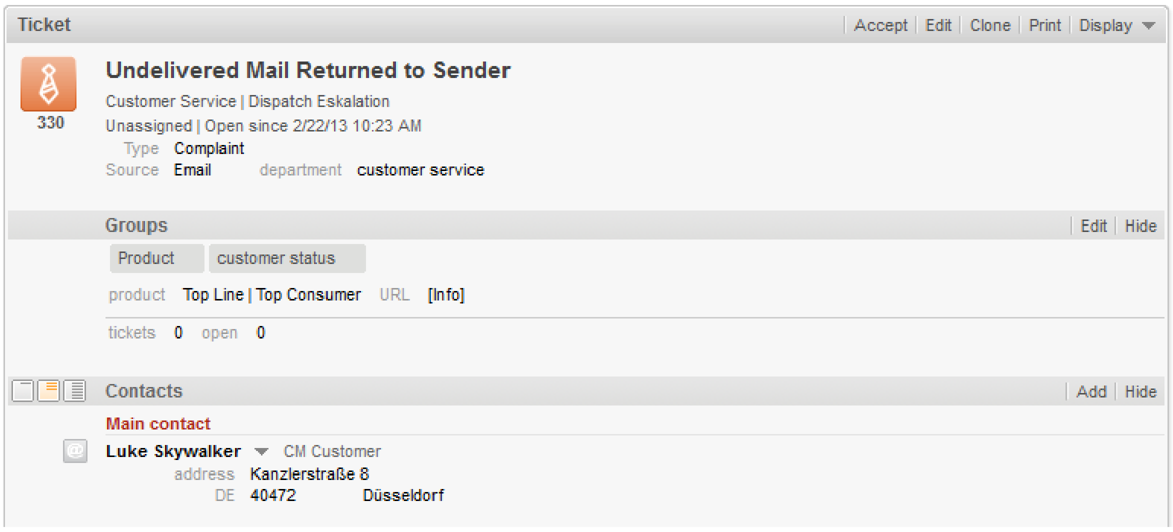
Closed tickets are not *lost*, but they represent a powerful archive and knowledge base. The user can search the tickets using the system search. Furthermore, the system can be configured to provide FAQ (frequently asked questions) functionalities.

Every ticket has an ID that is used internally and cannot be seen by the user.

Every ticket has a name (often called *ticket number*) that is displayed on the GUI to mark a ticket for the user.

The ticket icon in the GUI 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.

However, for every process, a different color-defining value can be used. While the user help desk uses the priority, the marketing and sales department can use the probability for the conclusion of a contract, and a multi-brand service team can use the brand.



Undelivered Mail Returned to Sender
Customer Service | Dispatch Eskalation
Unassigned | Open since 2/22/13 10:23 AM
Type Complaint
Source Email department customer service

product Top Line | Top Consumer URL [Info]
tickets 0 open 0

Main contact
Luke Skywalker CM Customer
address Kanzlerstraße 8
DE 40472 Düsseldorf

Fig. 3: ConSol*CM Web Client - Ticket

The Workflow

Every process that should be managed using ConSol*CM is modeled as a workflow. During its life cycle a ticket runs through several steps, e.g it is opened as a new ticket, has to be processed by several experts and can then be closed. During the process there might also be a pause, usually called *resubmission*. All those steps are modeled as steps in a workflow.

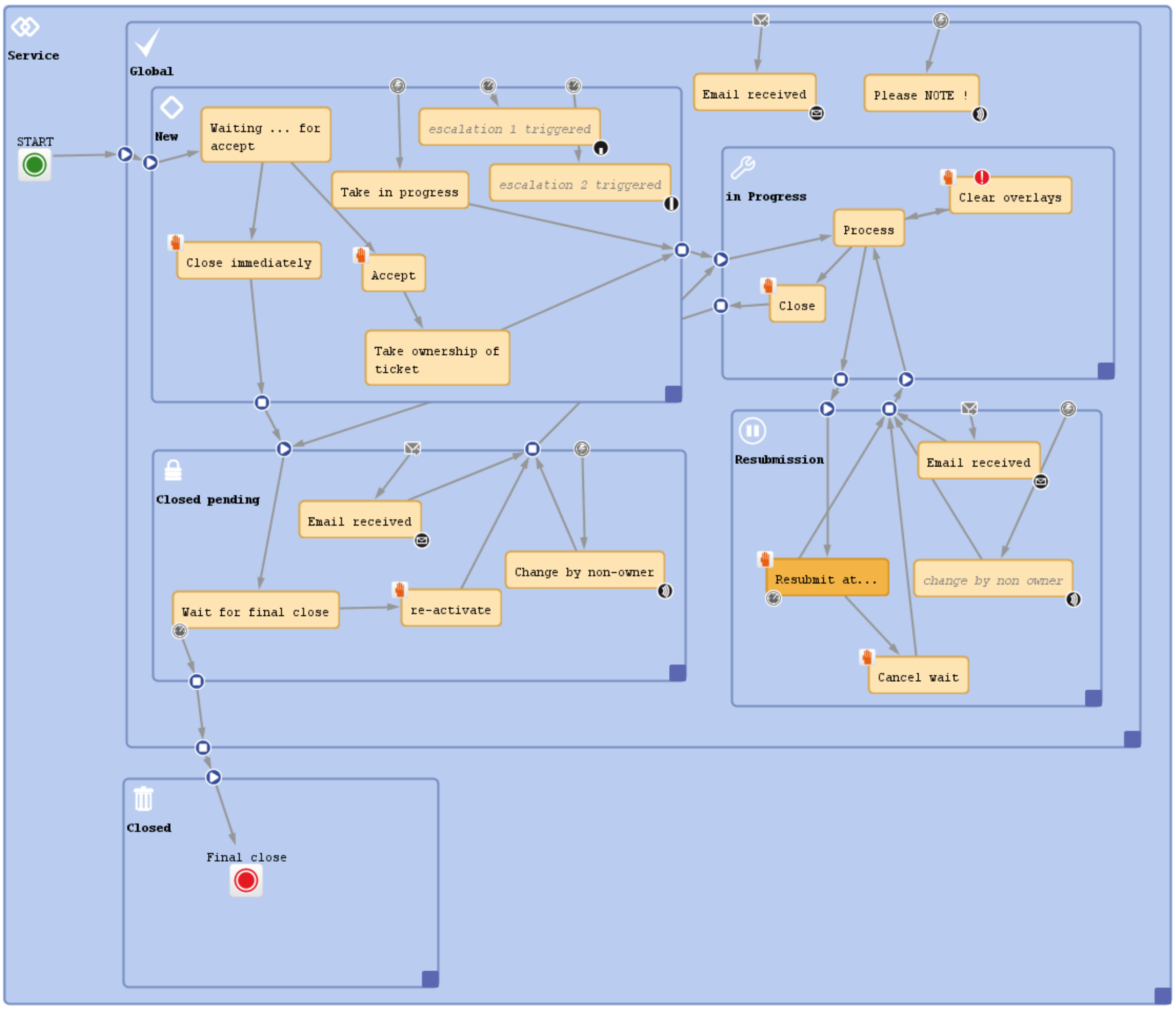


Fig. 4: ConSol*CM Process Designer - Workflow

A business process is modeled in ConSol*CM using the *Process Designer*, an application which is an integral element of a standard ConSol*CM installation. 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.

For example, a ticket starts in an entry pool, is directed to the 1st level team who pass it on to the 2nd level network team. Or a sales ticket starts as a customer request, becomes a lead which gets more *serious* and becomes an opportunity. Once the customer has signed the contract, an order ticket is created which generates so-called child tickets for the internal tasks up to billing. When all child tickets are closed, the parent ticket can be closed as well.

The *intelligence* of the process, like escalations, reminders, e-mails that are sent automatically, 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 to process modeling using the ConSol*CM Process Designer.

The Queue

The queue is **the** core component of the ConSol*CM administration. It comprises tickets from the same domain and makes sure that all tickets of this domain are treated in the same way. Every queue has exactly one workflow which cannot be changed. All data fields required in a process are assigned to tickets of the process by queue management.

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.

Access permissions are also managed using the queue as basic entity.

The Customer

The customer is the person (contact) or company who has the question or service request. This person or company is the main customer for the ticket. This represents the *external* side of the CM system.

With *FlexCDM*, the Flexible Customer Data Model, ConSol*CM provides a data model which can define customer and contact 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, e-mail address, address) and complex, two-level models which contain contact data (e.g. name, phone number, e-mail 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 see section The CM Customer Data Model: FlexCDM for a detailed explanation of all components of the *FlexCDM*.

The Engineer

The engineer is the *worker* who has a login to the Web Client and who has to manage the tasks defined in the tickets. This represents the *internal* side of the company or service team.

The engineer’s access permissions are managed using roles, i.e. the roles are defined using the Admin Tool and engineers are assigned the role(s) they need.

Often, a ticket does not have an engineer when it just has been opened and it is assigned to an engineer in one of the first steps of the process. This engineer is responsible for the ticket, he/she will get the escalation e-mails and will see the ticket in his/her view *My Tickets* (*personal to-do list*).

There can be additional engineers in certain so-called *engineer roles* for a ticket, who also have tasks to do for the case. For example, a ticket has a regular engineer and an additional engineer in the engineer role *supervisor*. In this way, in the run of the process, the ticket can be automatically assigned to the supervisor.

ConSol*CM Dogma

In ConSol*CM6, there is a main *dogma*:

> A ticket always has a **main customer**. This can be a contact or a company.
>
> A ticket can have no or one **engineer** who has to work on the ticket.

1.5.3 ConSol*CM from a System Administrator's Point of View

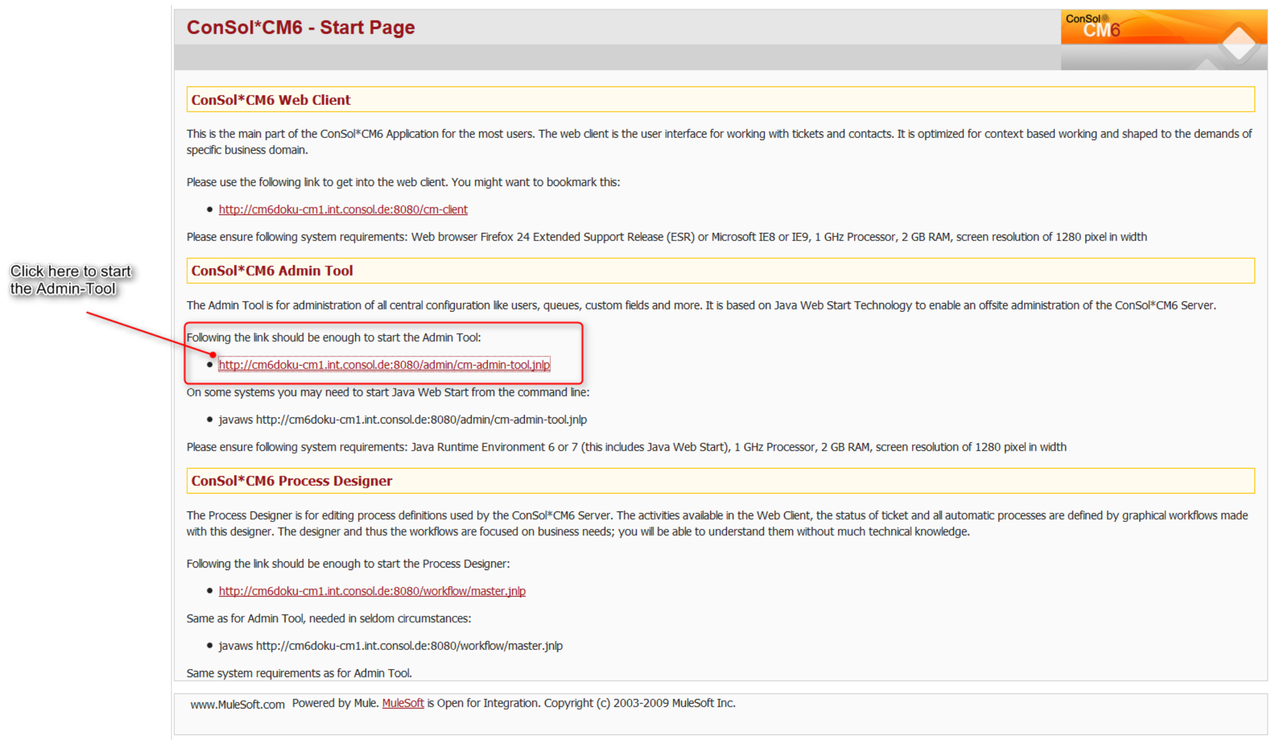
ConSol*CM is a Java EE application which runs in a standard application server. The data is stored in a relational database. ConSol*CM connects to an e-mail server to retrieve incoming e-mails and sends e-mails using an SMTP server. Please refer to the *ConSol*CM Operations Manual* for a detailed explanation of all aspects concerning running ConSol*CM in an IT environment. A first introduction is provided in section System Overview in this manual. If you need to know the supported application servers and relational database systems, please ask for the current *System Requirements* and/or the *Release Notes* of the latest ConSol*CM version.

2 Start of the Admin Tool

- Login
- Troubleshooting: When the Admin Tool Does Not Start
  - Correct Process
  - Process with Errors
- How to Use the Admin Tool

2.1 Login

Most of the ConSol*CM system is administrated using a Java Web Start application called *Admin Tool* which is provided on the main web page of the CM application server system. 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. Java Web Start is part of each standard JRE.



ConSol*CM6 - Start Page

ConSol*CM6 Web Client

This is the main part of the ConSol*CM6 Application for the most users. The web client is the user interface for working with tickets and contacts. It is optimized for context based working and shaped to the demands of specific business domain.

Please use the following link to get into the web client. You might want to bookmark this:

- http://cm6doku-cm1.int.consol.de:8080/cm-client

Please ensure following system requirements: Web browser Firefox 24 Extended Support Release (ESR) or Microsoft IE8 or IE9, 1 GHz Processor, 2 GB RAM, screen resolution of 1280 pixel in width

ConSol*CM6 Admin Tool

The Admin Tool is for administration of all central configuration like users, queues, custom fields and more. It is based on Java Web Start Technology to enable an offsite administration of the ConSol*CM6 Server.

Following the link should be enough to start the Admin Tool:

- http://cm6doku-cm1.int.consol.de:8080/admin/cm-admin-tool.jnlp

On some systems you may need to start Java Web Start from the command line:

- javaws http://cm6doku-cm1.int.consol.de:8080/admin/cm-admin-tool.jnlp

Please ensure following system requirements: Java Runtime Environment 6 or 7 (this includes Java Web Start), 1 GHz Processor, 2 GB RAM, screen resolution of 1280 pixel in width

ConSol*CM6 Process Designer

The Process Designer is for editing process definitions used by the ConSol*CM6 Server. The activities available in the Web Client, the status of ticket and all automatic processes are defined by graphical workflows made with this designer. The designer and thus the workflows are focused on business needs; you will be able to understand them without much technical knowledge.

Following the link should be enough to start the Process Designer:

- http://cm6doku-cm1.int.consol.de:8080/workflow/master.jnlp

Same as for Admin Tool, needed in seldom circumstances:

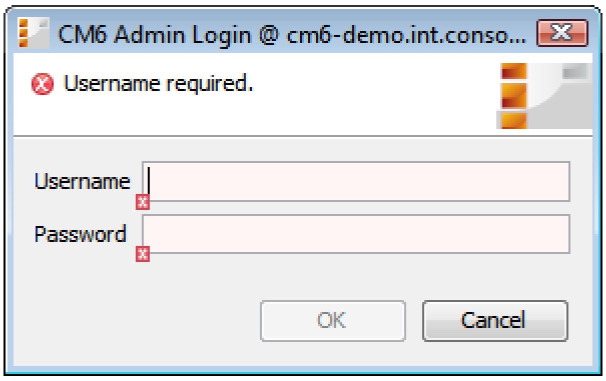
- javaws http://cm6doku-cm1.int.consol.de:8080/workflow/master.jnlp

Same system requirements as for Admin Tool.

www.MuleSoft.com Powered by Mule. MuleSoft is Open for Integration. Copyright (c) 2003-2009 MuleSoft Inc.

Fig. 1: ConSol*CM Start Page

After clicking on *Admin Tool.jnlp*, the *jnlp* file is downloaded, the Admin Tool is started, and the login window is displayed (for details see the *Troubleshooting* section):



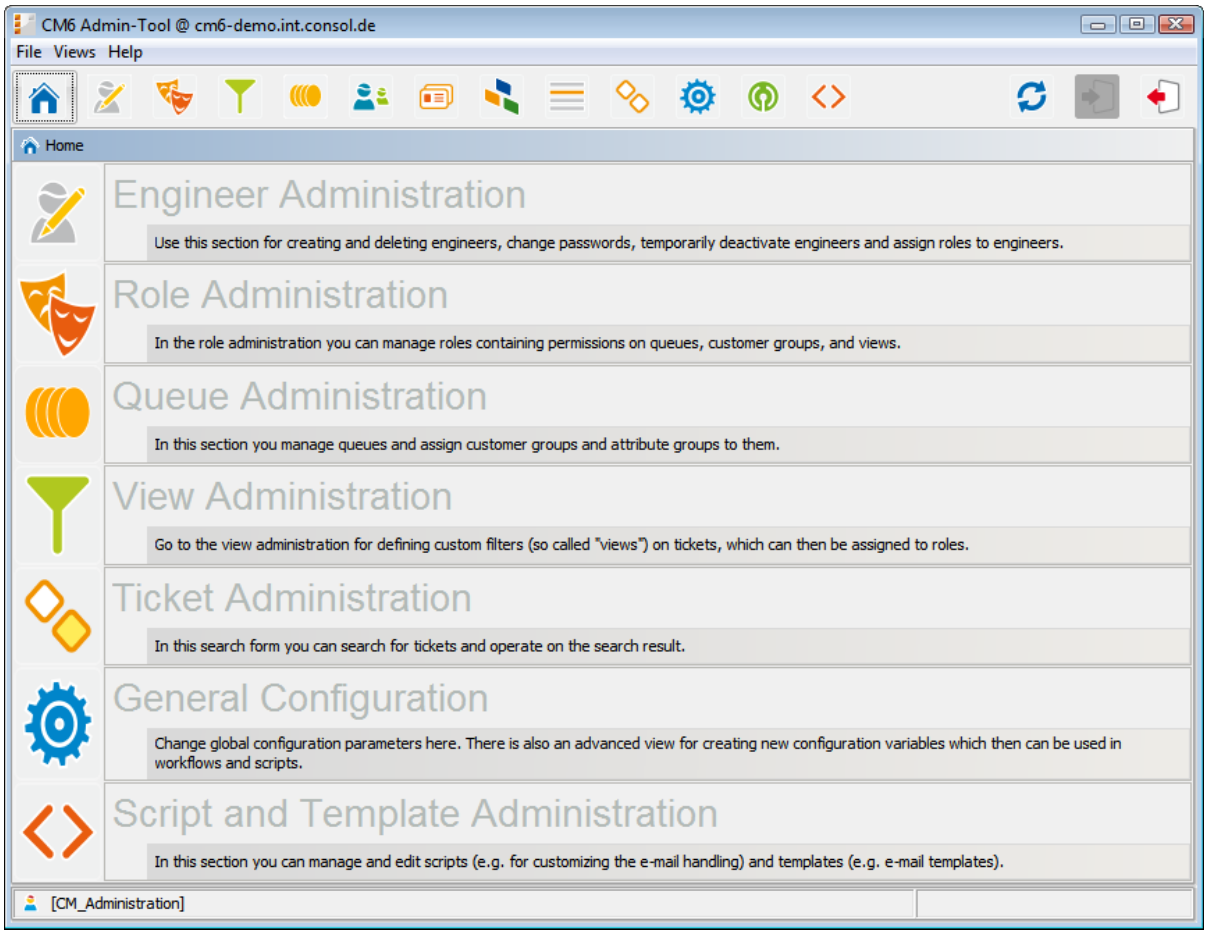
CM6 Admin Login @ cm6-demo.int.conso...

Username required.

Fig. 2: ConSol*CM Admin Tool - Login Window

Enter your login data to get access to the Admin Tool functions. A first user name and password are assigned 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 appears:



Engineer Administration

Use this section for creating and deleting engineers, change passwords, temporarily deactivate engineers and assign roles to engineers.

Role Administration

In the role administration you can manage roles containing permissions on queues, customer groups, and views.

Queue Administration

In this section you manage queues and assign customer groups and attribute groups to them.

View Administration

Go to the view administration for defining custom filters (so called "views") on tickets, which can then be assigned to roles.

Ticket Administration

In this search form you can search for tickets and operate on the search result.

General Configuration

Change global configuration parameters here. There is also an advanced view for creating new configuration variables which then can be used in workflows and scripts.

Script and Template Administration

In this section you can manage and edit scripts (e.g. for customizing the e-mail handling) and templates (e.g. e-mail templates).

[CM_Administration]

Fig. 3: ConSol*CM Admin Tool - Start Page

2.2 Troubleshooting: When the Admin Tool Does Not Start

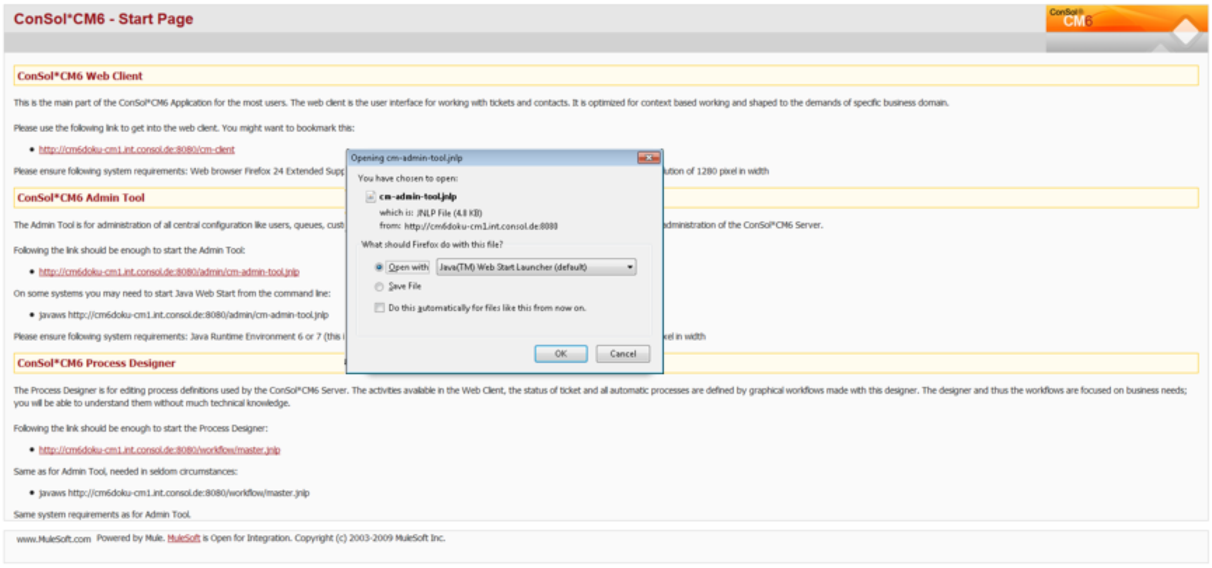
2.2.1 Correct Process

To be able to find the problems you should know the correct process.

In case everything is set-up correctly, after clicking on the Admin Tool hyper link, the following steps are performed:

1. In a pop-up window, you are prompted for the decision if you would like to open the *jnlp* file - *Java (TM) Web Start Launcher* should be offered as default application for that - or if you want to store a local copy of the *jnlp* file.
   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. In case the Java Web Console is activated, the console is opened and you can follow the download.
5. The Admin Tool GUI is displayed with the login window in the foreground.

Step1



ConSol*CM6 - Start Page

ConSol*CM6 Web Client

This is the main part of the ConSol*CM6 Application for the most users. The web client is the user interface for working with tickets and contacts. It is optimized for context based working and shaped to the demands of specific business domain.

Please use the following link to get into the web client. You might want to bookmark this:

- http://cm6doku-cm1.int.consol.de:8080/cm-client

Please ensure following system requirements: Web browser Firefox 24 Extended Supp … ution of 1280 pixel in width

ConSol*CM6 Admin Tool

The Admin Tool is for administration of all central configuration like users, queues, cust … dministration of the ConSol*CM6 Server.

Following the link should be enough to start the Admin Tool:

- http://cm6doku-cm1.int.consol.de:8080/admin/cm-admin-tool.jnlp

On some systems you may need to start Java Web Start from the command line:

- javaws http://cm6doku-cm1.int.consol.de:8080/admin/cm-admin-tool.jnlp

Please ensure following system requirements: Java Runtime Environment 6 or 7 (this i … xel in width

Opening cm-admin-tool.jnlp

You have chosen to open:

cm-admin-tool.jnlp

which is: JNLP File (4.8 KB)

from: http://cm6doku-cm1.int.consol.de:8080

What should Firefox do with this file?

[✓ Open with] Java(TM) Web Start Launcher (default)

Save File

☐ Do this automatically for files like this from now on.

ConSol*CM6 Process Designer

The Process Designer is for editing process definitions used by the ConSol*CM6 Server. The activities available in the Web Client, the status of ticket and all automatic processes are defined by graphical workflows made with this designer. The designer and thus the workflows are focused on business needs; you will be able to understand them without much technical knowledge.

Following the link should be enough to start the Process Designer:

- http://cm6doku-cm1.int.consol.de:8080/workflow/master.jnlp

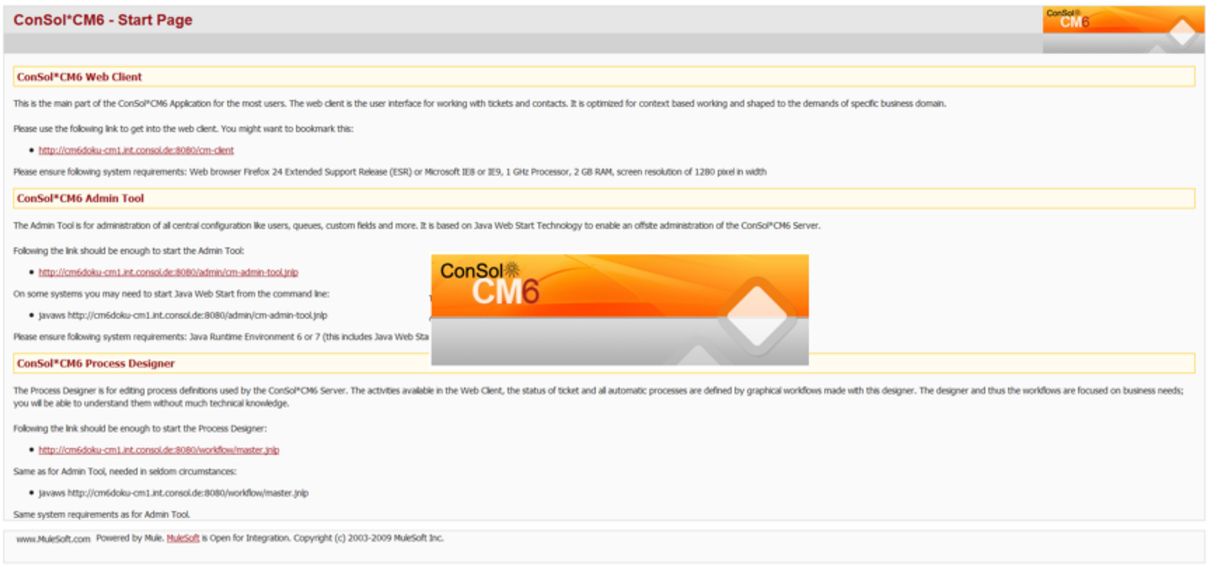
Same as for Admin Tool, needed in seldom circumstances:

- javaws http://cm6doku-cm1.int.consol.de:8080/workflow/master.jnlp

Same system requirements as for Admin Tool.

www.MuleSoft.com Powered by Mule. MuleSoft is Open for Integration. Copyright (c) 2003-2009 MuleSoft Inc.

Step2



ConSol*CM6 - Start Page

ConSol*CM6 Web Client

This is the main part of the ConSol*CM6 Application for the most users. The web client is the user interface for working with tickets and contacts. It is optimized for context based working and shaped to the demands of specific business domain.

Please use the following link to get into the web client. You might want to bookmark this:

- http://cm6doku-cm1.int.consol.de:8080/cm-client

Please ensure following system requirements: Web browser Firefox 24 Extended Support Release (ESR) or Microsoft IE8 or IE9, 1 GHz Processor, 2 GB RAM, screen resolution of 1280 pixel in width

ConSol*CM6 Admin Tool

The Admin Tool is for administration of all central configuration like users, queues, custom fields and more. It is based on Java Web Start Technology to enable an offsite administration of the ConSol*CM6 Server.

Following the link should be enough to start the Admin Tool:

- http://cm6doku-cm1.int.consol.de:8080/admin/cm-admin-tool.jnlp

On some systems you may need to start Java Web Start from the command line:

- javaws http://cm6doku-cm1.int.consol.de:8080/admin/cm-admin-tool.jnlp

Please ensure following system requirements: Java Runtime Environment 6 or 7 (this includes Java Web Sta

ConSol*CM6 Process Designer

The Process Designer is for editing process definitions used by the ConSol*CM6 Server. The activities available in the Web Client, the status of ticket and all automatic processes are defined by graphical workflows made with this designer. The designer and thus the workflows are focused on business needs; you will be able to understand them without much technical knowledge.

Following the link should be enough to start the Process Designer:

- http://cm6doku-cm1.int.consol.de:8080/workflow/master.jnlp

Same as for Admin Tool, needed in seldom circumstances:

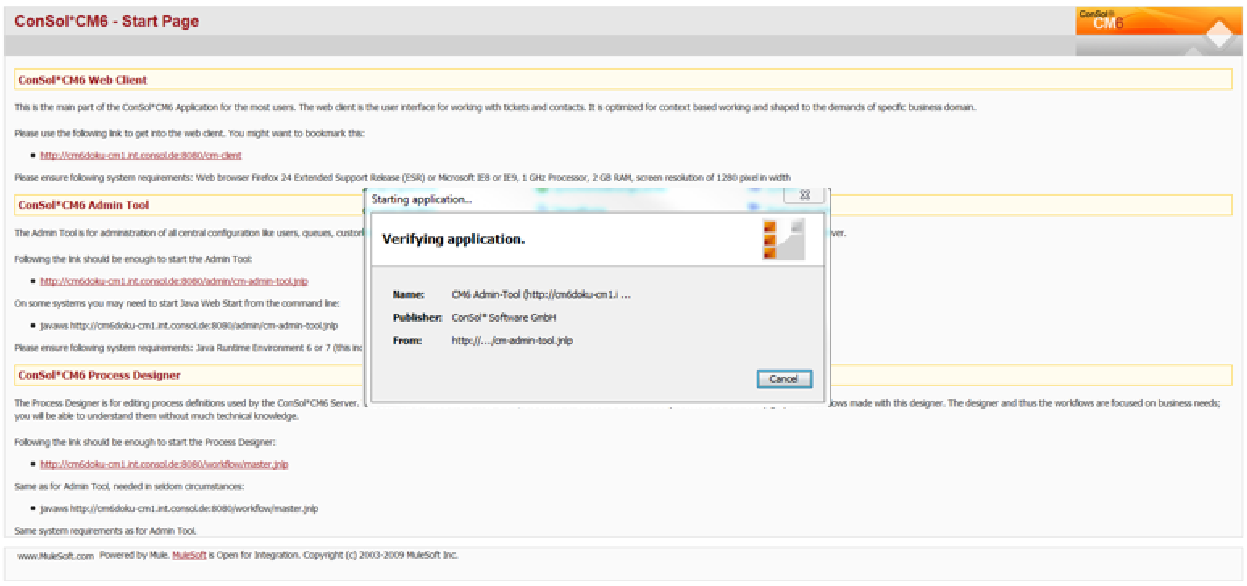
- javaws http://cm6doku-cm1.int.consol.de:8080/workflow/master.jnlp

Same system requirements as for Admin Tool.

www.MuleSoft.com Powered by Mule. MuleSoft is Open for Integration. Copyright (c) 2003-2009 MuleSoft Inc.

Fig. 4: ConSol*CM Admin Tool - Start: Steps 1 and 2

Step3



ConSol*CM6 - Start Page

ConSol*CM6 Web Client

This is the main part of the ConSol*CM6 Application for the most users. The web client is the user interface for working with tickets and contacts. It is optimized for context based working and shaped to the demands of specific business domain.

Please use the following link to get into the web client. You might want to bookmark this:

- http://cm6doku-cm1.int.consol.de:8080/cm-client

Please ensure following system requirements: Web browser Firefox 24 Extended Support Release (ESR) or Microsoft IE8 or IE9, 1 GHz Processor, 2 GB RAM, screen resolution of 1280 pixel in width

ConSol*CM6 Admin Tool

The Admin Tool is for administration of all central configuration like users, queues, custo ... ver.

Following the link should be enough to start the Admin Tool:

- http://cm6doku-cm1.int.consol.de:8080/admin/cm-admin-tool.jnlp

On some systems you may need to start Java Web Start from the command line:

- javaws http://cm6doku-cm1.int.consol.de:8080/admin/cm-admin-tool.jnlp

Please ensure following system requirements: Java Runtime Environment 6 or 7 (this inc

ConSol*CM6 Process Designer

The Process Designer is for editing process definitions used by the ConSol*CM6 Server. ... ows made with this designer. The designer and thus the workflows are focused on business needs; you will be able to understand them without much technical knowledge.

Following the link should be enough to start the Process Designer:

- http://cm6doku-cm1.int.consol.de:8080/workflow/master.jnlp

Same as for Admin Tool, needed in seldom circumstances:

- javaws http://cm6doku-cm1.int.consol.de:8080/workflow/master.jnlp

Same system requirements as for Admin Tool.

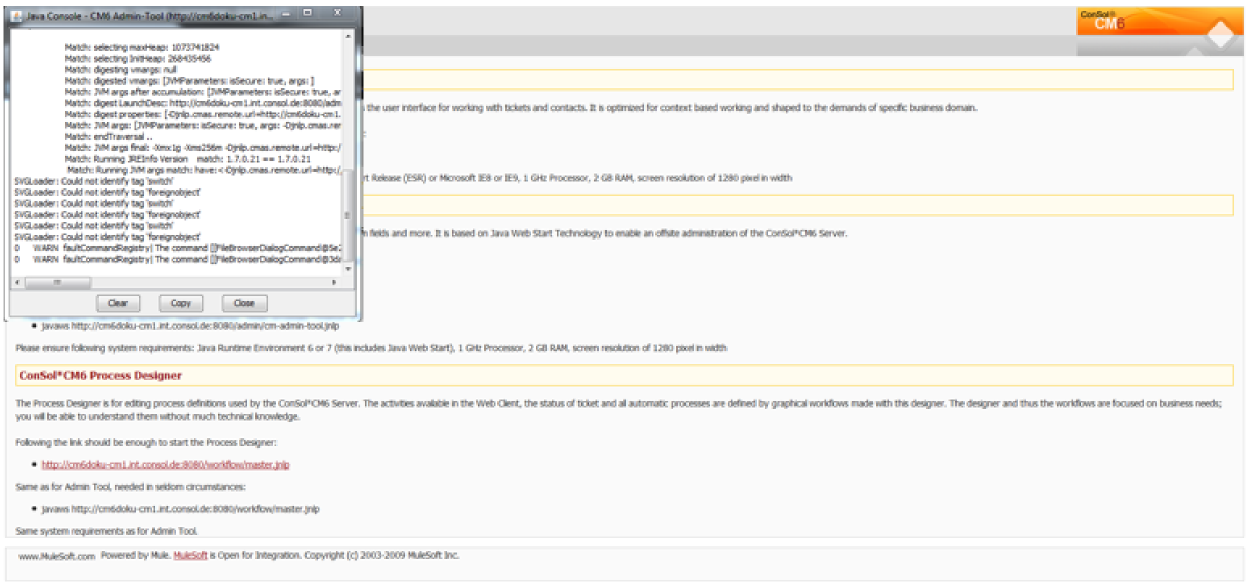
www.MuleSoft.com Powered by Mule. MuleSoft is Open for Integration. Copyright (c) 2003-2009 MuleSoft Inc.

Starting application...

Verifying application.

Name:	CM6 Admin-Tool (http://cm6doku-cm1.i ...
Publisher:	ConSol* Software GmbH
From:	http://.../cm-admin-tool.jnlp

Step 4 (only if Java console is activated)



Java Console - CM6 Admin-Tool (http://cm6doku-cm1.in...

```
            Match: selecting maxHeap: 1073741824
            Match: selecting InitHeap: 268435456
            Match: digesting vmargs: null
            Match: digested vmargs: [JVMParameters: isSecure: true, args: ]
            Match: JVM args after accumulation: [JVMParameters: isSecure: true, ar
            Match: digest LaunchDesc: http://cm6doku-cm1.int.consol.de:8080/adm
            Match: digest properties: [-Djnlp.cmas.remote.url=http://cm6doku-cm1.
            Match: JVM args: [JVMParameters: isSecure: true, args: -Djnlp.cmas.rer
            Match: endTraversal ..
            Match: JVM args final: -Xmx1g -Xms256m -Djnlp.cmas.remote.url=http:/
            Match: Running JREInfo Version    match: 1.7.0.21 == 1.7.0.21
             Match: Running JVM args match: have:<-Djnlp.cmas.remote.url=http:/
SVGLoader: Could not identify tag 'switch'
SVGLoader: Could not identify tag 'foreignobject'
SVGLoader: Could not identify tag 'switch'
SVGLoader: Could not identify tag 'foreignobject'
SVGLoader: Could not identify tag 'switch'
SVGLoader: Could not identify tag 'foreignobject'
0    WARN  faultCommandRegistry| The command []FileBrowserDialogCommand@5e2
0    WARN  faultCommandRegistry| The command []FileBrowserDialogCommand@3de
```

the user interface for working with tickets and contacts. It is optimized for context based working and shaped to the demands of specific business domain.

t Release (ESR) or Microsoft IE8 or IE9, 1 GHz Processor, 2 GB RAM, screen resolution of 1280 pixel in width

n fields and more. It is based on Java Web Start Technology to enable an offsite administration of the ConSol*CM6 Server.

- javaws http://cm6doku-cm1.int.consol.de:8080/admin/cm-admin-tool.jnlp

Please ensure following system requirements: Java Runtime Environment 6 or 7 (this includes Java Web Start), 1 GHz Processor, 2 GB RAM, screen resolution of 1280 pixel in width

ConSol*CM6 Process Designer

The Process Designer is for editing process definitions used by the ConSol*CM6 Server. The activities available in the Web Client, the status of ticket and all automatic processes are defined by graphical workflows made with this designer. The designer and thus the workflows are focused on business needs; you will be able to understand them without much technical knowledge.

Following the link should be enough to start the Process Designer:

- http://cm6doku-cm1.int.consol.de:8080/workflow/master.jnlp

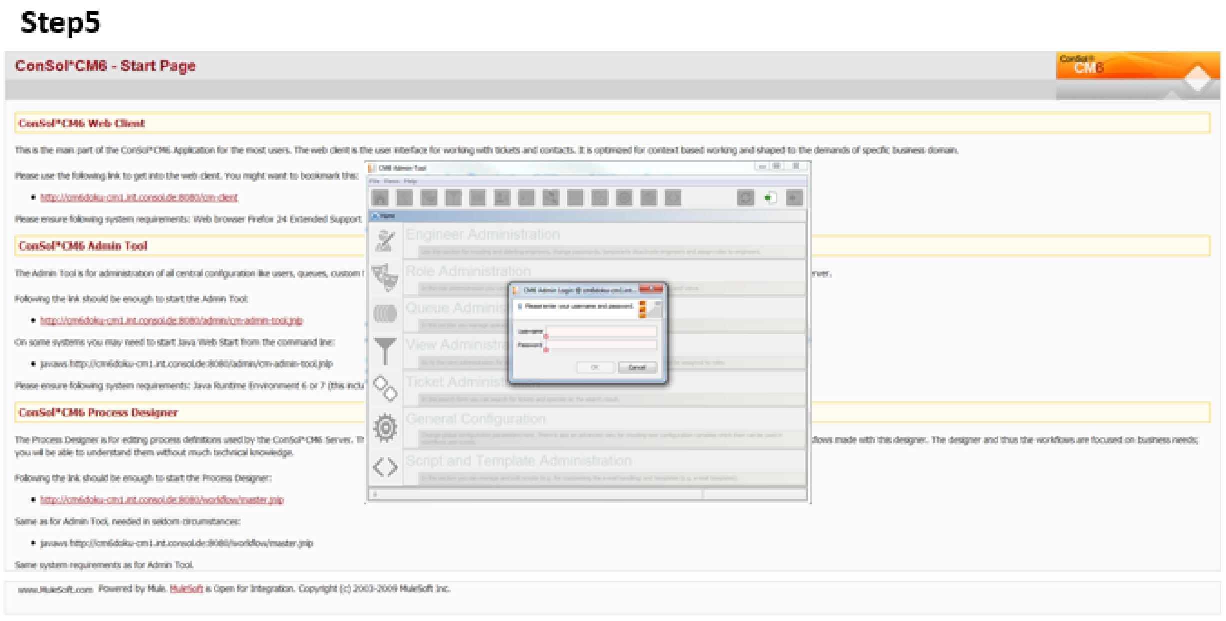
Same as for Admin Tool, needed in seldom circumstances:

- javaws http://cm6doku-cm1.int.consol.de:8080/workflow/master.jnlp

Same system requirements as for Admin Tool.

www.MuleSoft.com Powered by Mule. MuleSoft is Open for Integration. Copyright (c) 2003-2009 MuleSoft Inc.

Fig. 5: ConSol*CM Admin Tool - Start: Steps 3 and 4


Step5

ConSol*CM6 - Start Page

ConSol*CM6 Web Client

This is the main part of the ConSol*CM6 Application for the most users. The web client is the user interface for working with tickets and contacts. It is optimized for context based working and shaped to the demands of specific business domain.

Please use the following link to get into the web client. You might want to bookmark this:

- http://cm6doku-cm1.int.consol.de:8080/cm-client

Please ensure following system requirements: Web browser Firefox 24 Extended Support

ConSol*CM6 Admin Tool

The Admin Tool is for administration of all central configuration like users, queues, custom … rver.

Following the link should be enough to start the Admin Tool:

- http://cm6doku-cm1.int.consol.de:8080/admin/cm-admin-tool.jnlp

On some systems you may need to start Java Web Start from the command line:

- javaws http://cm6doku-cm1.int.consol.de:8080/admin/cm-admin-tool.jnlp

Please ensure following system requirements: Java Runtime Environment 6 or 7 (this inclu

ConSol*CM6 Process Designer

The Process Designer is for editing process definitions used by the ConSol*CM6 Server. Th … flows made with this designer. The designer and thus the workflows are focused on business needs; you will be able to understand them without much technical knowledge.

Following the link should be enough to start the Process Designer:

- http://cm6doku-cm1.int.consol.de:8080/workflow/master.jnlp

Same as for Admin Tool, needed in seldom circumstances:

- javaws http://cm6doku-cm1.int.consol.de:8080/workflow/master.jnlp

Same system requirements as for Admin Tool.

www.MuleSoft.com Powered by Mule. MuleSoft is Open for Integration. Copyright (c) 2003-2009 MuleSoft Inc.

Fig. 6: ConSol*CM Admin Tool - Start: Step 5

2.2.2 Process with Errors

In case the Admin Tool cannot be started, check the following settings:

1. **Problems with step 1:**
   a. Is Java in the correct version installed on your machine?
   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, you enter your login data. If a connection error occurs then, check the proxy settings.

2.3 How to Use the Admin Tool

You can reach the Admin Tool functions via the icon bar or via the drop-down menu under *Views* above the bar.

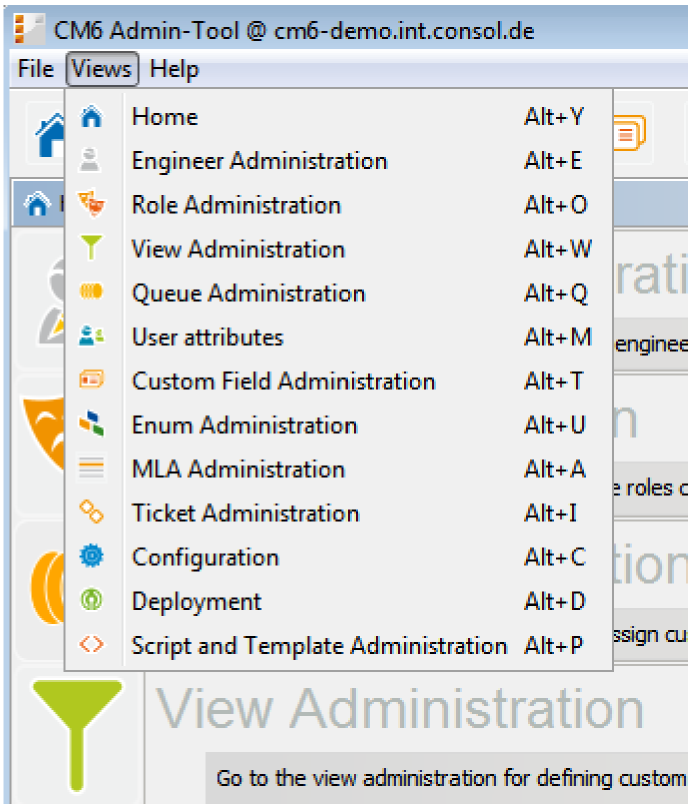
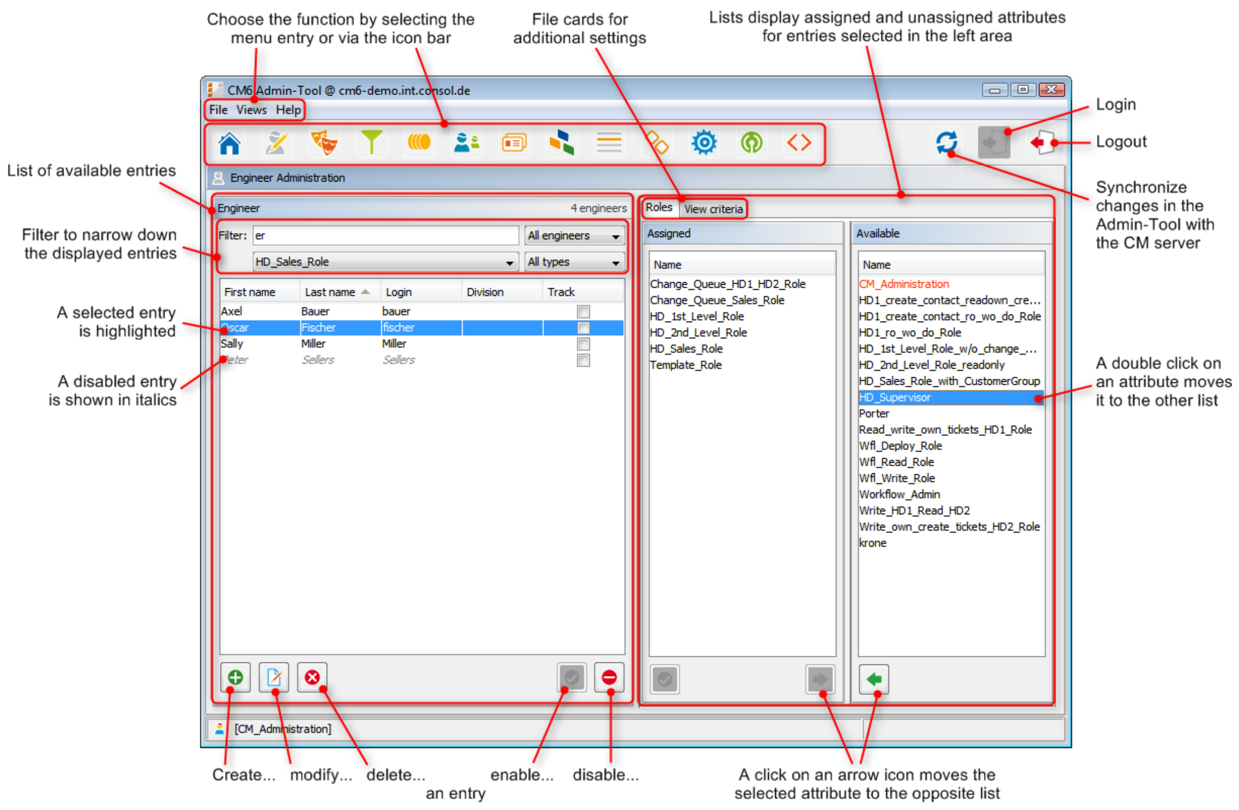

CM6 Admin-Tool @ cm6-demo.int.consol.de
File
Views
Help
Home Alt+Y
Engineer Administration Alt+E
Role Administration Alt+O
View Administration Alt+W
Queue Administration Alt+Q
User attributes Alt+M
Custom Field Administration Alt+T
Enum Administration Alt+U
MLA Administration Alt+A
Ticket Administration Alt+I
Configuration Alt+C
Deployment Alt+D
Script and Template Administration Alt+P
View Administration
Go to the view administration for defining custom

Fig. 7: ConSol*CM Admin Tool - Views Menu

The function pages are very similar in structure. The following picture shows the operational concept using the engineer administration as an example:


Choose the function by selecting the menu entry or via the icon bar
File cards for additional settings
Lists display assigned and unassigned attributes for entries selected in the left area
Login
Logout
Synchronize changes in the Admin-Tool with the CM server
List of available entries
Filter to narrow down the displayed entries
A selected entry is highlighted
A disabled entry is shown in italics
A double click on an attribute moves it to the other list
Create...
modify...
delete...
enable...
disable...
an entry
A click on an arrow icon moves the selected attribute to the opposite list
CM6 Admin-Tool @ cm6-demo.int.consol.de
File
Views
Help
Engineer Administration
Engineer
4 engineers
Filter:
er
All engineers
HD_Sales_Role
All types
Roles
View criteria
Assigned
Available
[CM_Administration]

First name	Last name	Login	Division	Track
Axel	Bauer	bauer		☐
Oscar	Fischer	fischer		☐
Sally	Miller	Miller		☐
Peter	*Sellers*	*Sellers*		☐

Name
Change_Queue_HD1_HD2_Role
Change_Queue_Sales_Role
HD_1st_Level_Role
HD_2nd_Level_Role
HD_Sales_Role
Template_Role

Name
CM_Administration
HD1_create_contact_readown_cre...
HD1_create_contact_ro_wo_do_Role
HD1_ro_wo_do_Role
HD_1st_Level_Role_w/o_change_...
HD_2nd_Level_Role_readonly
HD_Sales_Role_with_CustomerGroup
HD_Supervisor
Porter
Read_write_own_tickets_HD1_Role
Wfl_Deploy_Role
Wfl_Read_Role
Wfl_Write_Role
Workflow_Admin
Write_HD1_Read_HD2
Write_own_create_tickets_HD2_Role
krone

Fig. 8: ConSol*CM - Handling of the Admin Tool

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. You can move them from a list of *available attributes* to a list of *assigned attributes* either via double click or via click on the icon ⬅ (example: *available roles* and *assigned roles*). Attributes can also be assigned via check or list boxes (not displayed here).

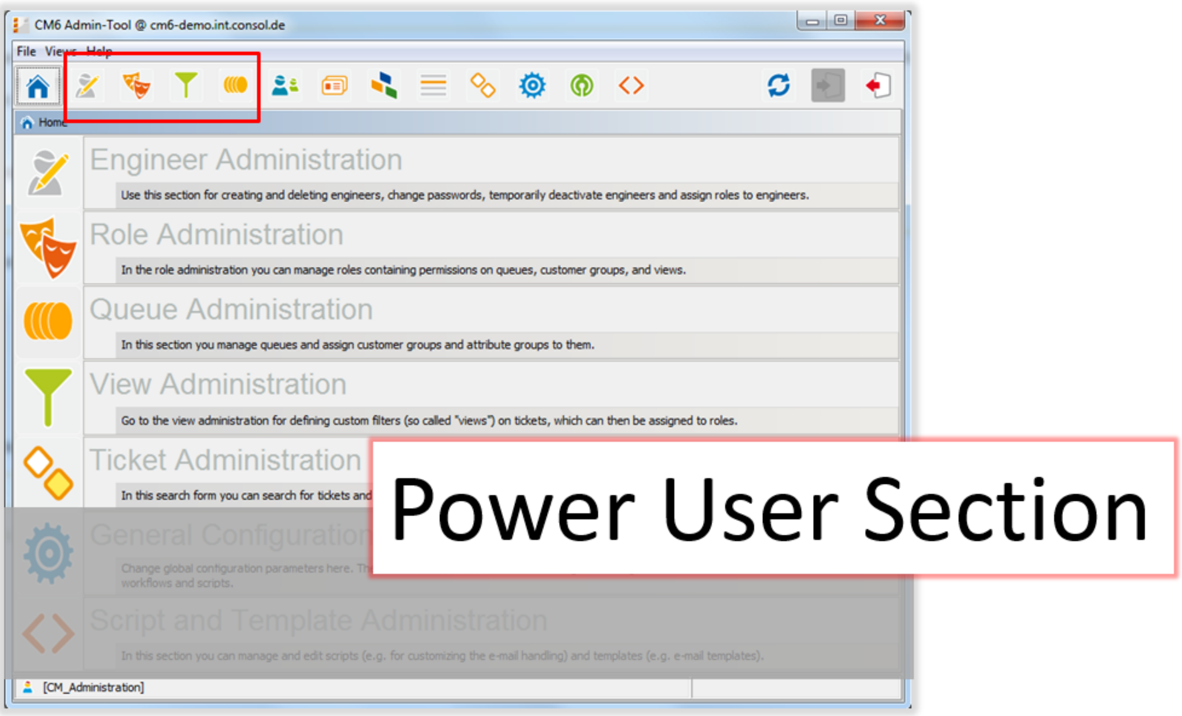
There are a couple of options to help you find the entries you want to edit more quickly:

- **Filters**
  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 will be adopted automatically, only the matching entries are displayed (with *er* in the example above).
  - **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 icons ▲ or ▼ show the sort order.

Usually all changes you perform in the Admin Tool are submitted immediately without the need to synchronize the data. However, if changes in another module have been performed and the Admin Tool has to use the new data, it is required to synchronize the data. You can achieve this by clicking on the *Synchronize* button in the icon bar.

One example for this is the deployment of a new workflow using the Process Designer. Before you can assign the new workflow 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. Then this new workflow can be used for further operations like assigning it to a new queue.

3 CM6 Administrator Manual 6.9.4 - --Power User Section--



Engineer Administration

Use this section for creating and deleting engineers, change passwords, temporarily deactivate engineers and assign roles to engineers.

Role Administration

In the role administration you can manage roles containing permissions on queues, customer groups, and views.

Queue Administration

In this section you manage queues and assign customer groups and attribute groups to them.

View Administration

Go to the view administration for defining custom filters (so called "views") on tickets, which can then be assigned to roles.

Ticket Administration

In this search form you can search for tickets and

General Configuration

Change global configuration parameters here. Th
workflows and scripts.

Script and Template Administration

In this section you can manage and edit scripts (e.g. for customizing the e-mail handling) and templates (e.g. e-mail templates).

This section provides some background knowledge about basic ConSol*CM components. This might be interesting for you if you are a CM key user like a team leader and it is absolutely indispensable if you are (or want to become) a CM administrator.

Please find here the following sections:

- Engineer Administration
- Role Administration
- View Administration
- Queue Administration

4 Engineer Administration

- Introduction to Engineer Administration
- Engineer Administration Using the Admin Tool
    - Create or Edit an Engineer Account
    - Delete an Engineer Account
    - Disable or Enable an Engineer Account
    - Tab Roles - Assign Roles to an Engineer Account
        - Set Roles as Main Roles
    - Tab View Criteria - Define Engineer-Specific View Criteria
- Related Topics

4.1 Introduction to Engineer Administration

An *engineer account* is the basic access object which allows the engineer or the administrator to access the Web Client, the Admin Tool, or the 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.

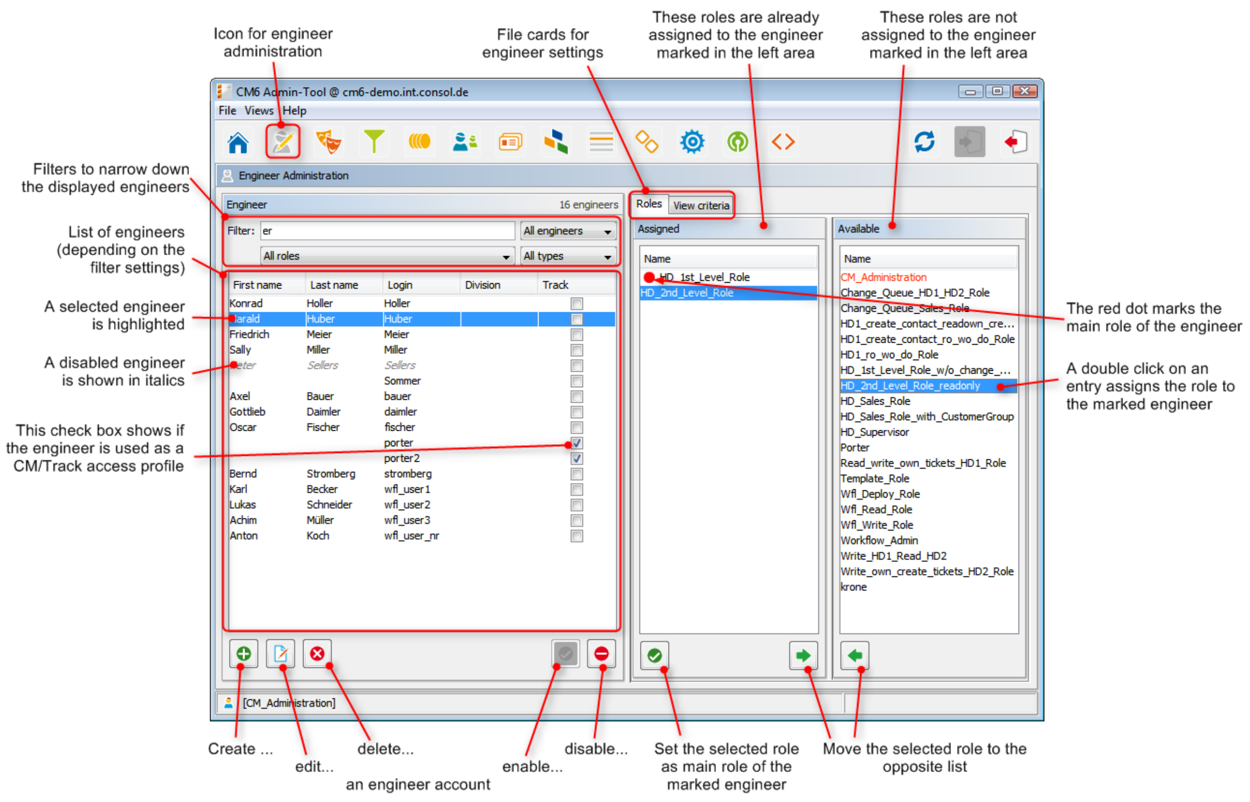
Newly created engineer accounts do not have any permissions. These permissions have to be assigned through one or multiple *role(s)* displayed in the tab *Roles*. If you have not created any roles yet, you will see only the administrator role (see tab *Roles*).

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

Information:

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

4.2 Engineer Administration Using the Admin Tool


Icon for engineer administration
File cards for engineer settings
These roles are already assigned to the engineer marked in the left area
These roles are not assigned to the engineer marked in the left area
Filters to narrow down the displayed engineers
List of engineers (depending on the filter settings)
A selected engineer is highlighted
A disabled engineer is shown in italics
This check box shows if the engineer is used as a CM/Track access profile
The red dot marks the main role of the engineer
A double click on an entry assigns the role to the marked engineer
Create ...
edit...
delete...
an engineer account
enable...
disable...
Set the selected role as main role of the marked engineer
Move the selected role to the opposite list
CM6 Admin-Tool @ cm6-demo.int.consol.de
File
Views
Help
Engineer Administration
Engineer
16 engineers
Filter:
er
All engineers
All roles
All types
Roles
View criteria
Assigned
Available
[CM_Administration]

First name	Last name	Login	Division	Track
Konrad	Holler	Holler		☐
Harald	Huber	Huber		☐
Friedrich	Meier	Meier		☐
Sally	Miller	Miller		☐
Peter	*Sellers*	*Sellers*		☐
		Sommer		☐
Axel	Bauer	bauer		☐
Gottlieb	Daimler	daimler		☐
Oscar	Fischer	fischer		☐
		porter		☑
		porter2		☑
Bernd	Stromberg	stromberg		☐
Karl	Becker	wfl_user1		☐
Lukas	Schneider	wfl_user2		☐
Achim	Müller	wfl_user3		☐
Anton	Koch	wfl_user_nr		☐

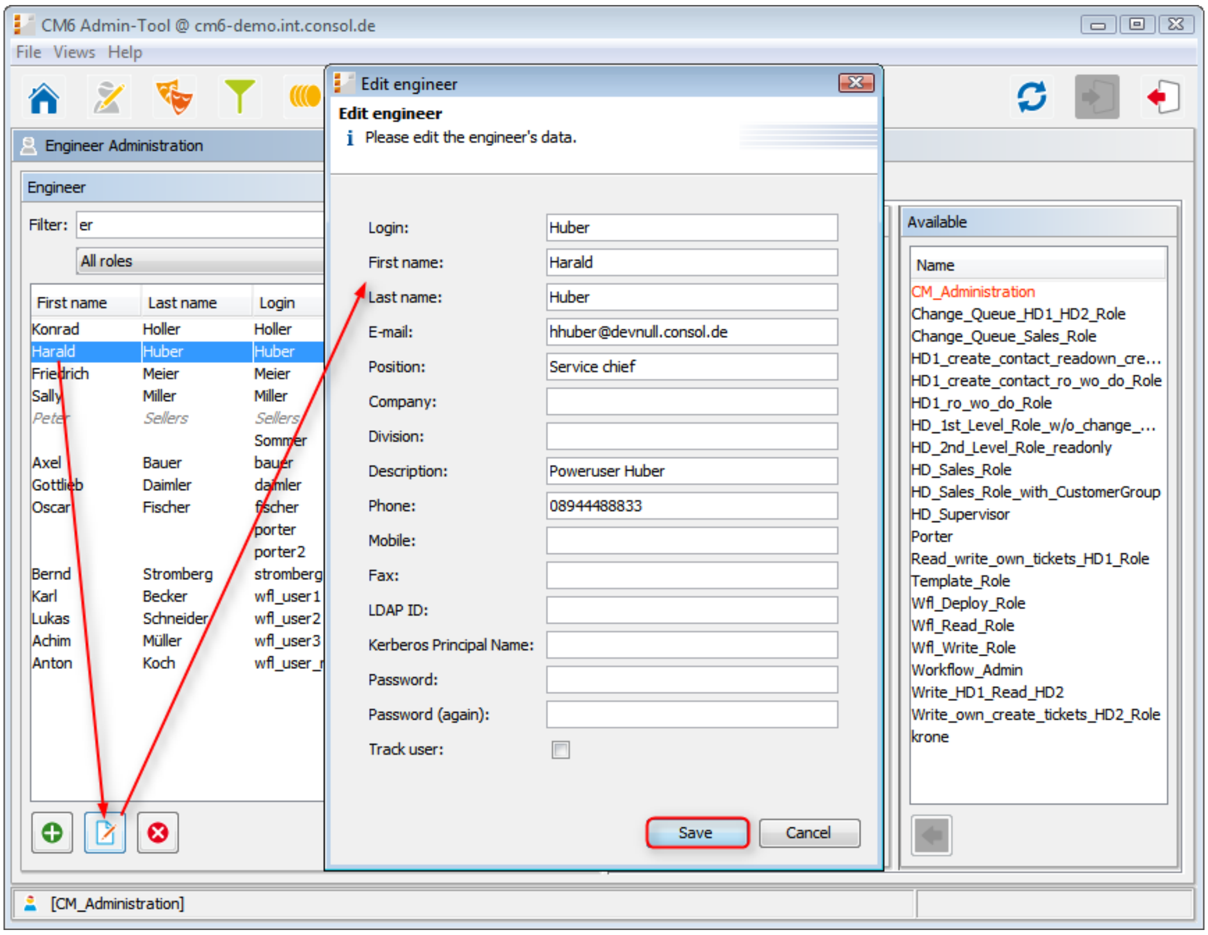
Name
● HD_1st_Level_Role
HD_2nd_Level_Role

Name
CM_Administration
Change_Queue_HD1_HD2_Role
Change_Queue_Sales_Role
HD1_create_contact_readown_cre...
HD1_create_contact_ro_wo_do_Role
HD1_ro_wo_do_Role
HD_1st_Level_Role_w/o_change_...
HD_2nd_Level_Role_readonly
HD_Sales_Role
HD_Sales_Role_with_CustomerGroup
HD_Supervisor
Porter
Read_write_own_tickets_HD1_Role
Template_Role
Wfl_Deploy_Role
Wfl_Read_Role
Wfl_Write_Role
Workflow_Admin
Write_HD1_Read_HD2
Write_own_create_tickets_HD2_Role
krone

Fig. 1: ConSol*CM Admin Tool - Engineer Administration

4.2.1 Create or Edit an Engineer Account

To create an engineer account click on the icon below the account list. Or click on if you want to edit the settings of an existing account. The same pop-up window appears:



Filter: er

All roles

First name	Last name	Login
Konrad	Holler	Holler
Harald	Huber	Huber
Friedrich	Meier	Meier
Sally	Miller	Miller
Peter	Sellers	Sellers
		Sommer
Axel	Bauer	bauer
Gottlieb	Daimler	daimler
Oscar	Fischer	fischer
		porter
		porter2
Bernd	Stromberg	stromberg
Karl	Becker	wfl_user1
Lukas	Schneider	wfl_user2
Achim	Müller	wfl_user3
Anton	Koch	wfl_user

Login:	Huber
First name:	Harald
Last name:	Huber
E-mail:	hhuber@devnull.consol.de
Position:	Service chief
Company:	
Division:	
Description:	Poweruser Huber
Phone:	08944488833
Mobile:	
Fax:	
LDAP ID:	
Kerberos Principal Name:	
Password:	
Password (again):	
Track user:	☐

Name
CM_Administration
Change_Queue_HD1_HD2_Role
Change_Queue_Sales_Role
HD1_create_contact_readown_cre...
HD1_create_contact_ro_wo_do_Role
HD1_ro_wo_do_Role
HD_1st_Level_Role_w/o_change_...
HD_2nd_Level_Role_readonly
HD_Sales_Role
HD_Sales_Role_with_CustomerGroup
HD_Supervisor
Porter
Read_write_own_tickets_HD1_Role
Template_Role
Wfl_Deploy_Role
Wfl_Read_Role
Wfl_Write_Role
Workflow_Admin
Write_HD1_Read_HD2
Write_own_create_tickets_HD2_Role
krone

Fig. 2: ConSol*CM Admin Tool - Edit an Engineer Account

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.

- **E-mail:**
  Mandatory. The engineer's e-mail address. Please use only international alphabetic and numeric characters, hyphens, underscores, periods, and the @ sign. The entry of multiple e-mail 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.
- **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.

Attention:

Several fields which contain engineer data (like *Company*, *Division*, or *Phone*) are optional fields. However, when you work with text templates which contain engineer data fields (see section The ConSol*CM Template Manager) the e-mails or comments will not be formed correctly when the data is missing. For example, the field *ticket-engineer, phone* cannot be filled-in in the template when 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.

  > **Information:**
  >
  > 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)!

- **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.
- **Password:**
  Mandatory. The engineer's password. The entry of a password is mandatory. Please use only international alphabetic and numeric characters, and punctuation marks, do **not** use any special characters as e.g. umlauts. The password entered will be shown as a string of asterisks.

  > **Information:**
  >
  > This field will only appear if the engineer authenticates against the Web Client via the CM6 database, i.e. when LDAP or Kerberos authentication is used, the field is not visible.

- **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. The repetition of the password is mandatory.

  > **Information:**
  >
  > This field will only appear if the engineer authenticates against the Web Client via the CM6 database, i.e. when LDAP or Kerberos authentication is used, the field is not visible.

- **Track user**
  This check box 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) will be shown in the Web Client when creating or modifying a customer. So, by ticking this check box, 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 on *Save* afterwards to store your entries and to close the window.

4.2.2 Delete an Engineer Account

To delete an engineer account, select the account in the list and click on . 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. For all history entries in tickets and in customer pages which were performed by the deleted engineer, his or her name will still be displayed.

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

4.2.3 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 login anymore and other engineers cannot assign any tickets to their accounts.

To disable an engineer account, select the account and click on . 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. Just click on at the bottom of the page, if you want to enable the account again.

4.2.4 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 on 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 Role Administration).

Information:

When engineers log in to the system, they will have access permissions from all roles that have been assigned to them. So all permissions are added! There is no way of explicitly denying access to objects in ConSol*CM, you always grant access! The sum of all granted permissions defines the final access permissions for the engineer.

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 on below the list. Afterwards the main role is marked with a red dot. Now the views of the main role will always appear at the top of the view list in the Web Client for this engineer account.

4.2.5 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 View Administration).

Information:

This tab will show view criteria only 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 on ⬅ to move it to the list of *assigned* view criteria. You will see the possible values below the criterion in the list. Tick the check boxes 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 will be immediately visible in the engineer's profile page.

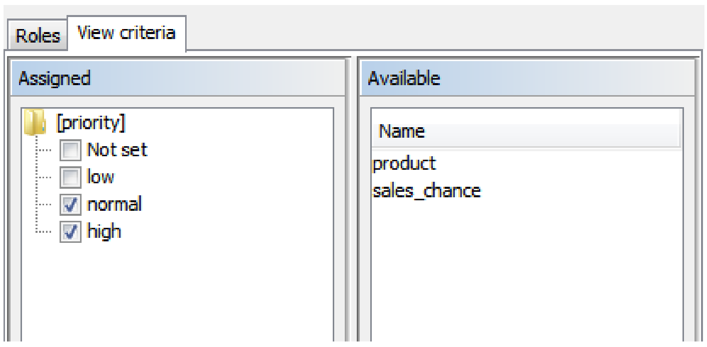


Fig. 3: ConSol*CM Admin Tool - Define Engineer-Specific View Criteria

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 into the Web Client. If you do not tick any values the engineer will see no tickets for this view. See section View Administration for details.

Attention:

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, 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.

4.3 Related Topics

- Roles
- Views

5 Role Administration

- Role Administration
    - Introduction to Role Administration
    - Role Administration Using the Admin Tool
        - Create a Role
            - Tab Queue Permissions
            - Tab Global Permissions
            - Tab Customer Group Permissions
            - Tab Views
        - Tab Engineer Functions
        - Delete a Role
        - Copy a Role
        - Edit a Role
    - Related Topics

5.1 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) the engineer obtains 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.

Information:

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, but you always grant access!

The sum of all granted permissions defines the final permissions for the engineer.

Roles define:

- **Access permissions to 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.
- **Access permissions to customer data**
  Read, write, modify, and delete permissions for each distinct customer group.
- **Views**
  To do lists of tickets which will be 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.*

5.2 Role Administration Using the Admin Tool

In the Role Administration, you see the list of all available roles on the left-hand side and the permissions which can be granted on the right-hand side. 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 with the access permissions of the role which has been selected in the list of roles. Only one role can be selected at a time. On the right-hand side, several tabs are available. During role management you switch between these tabs.

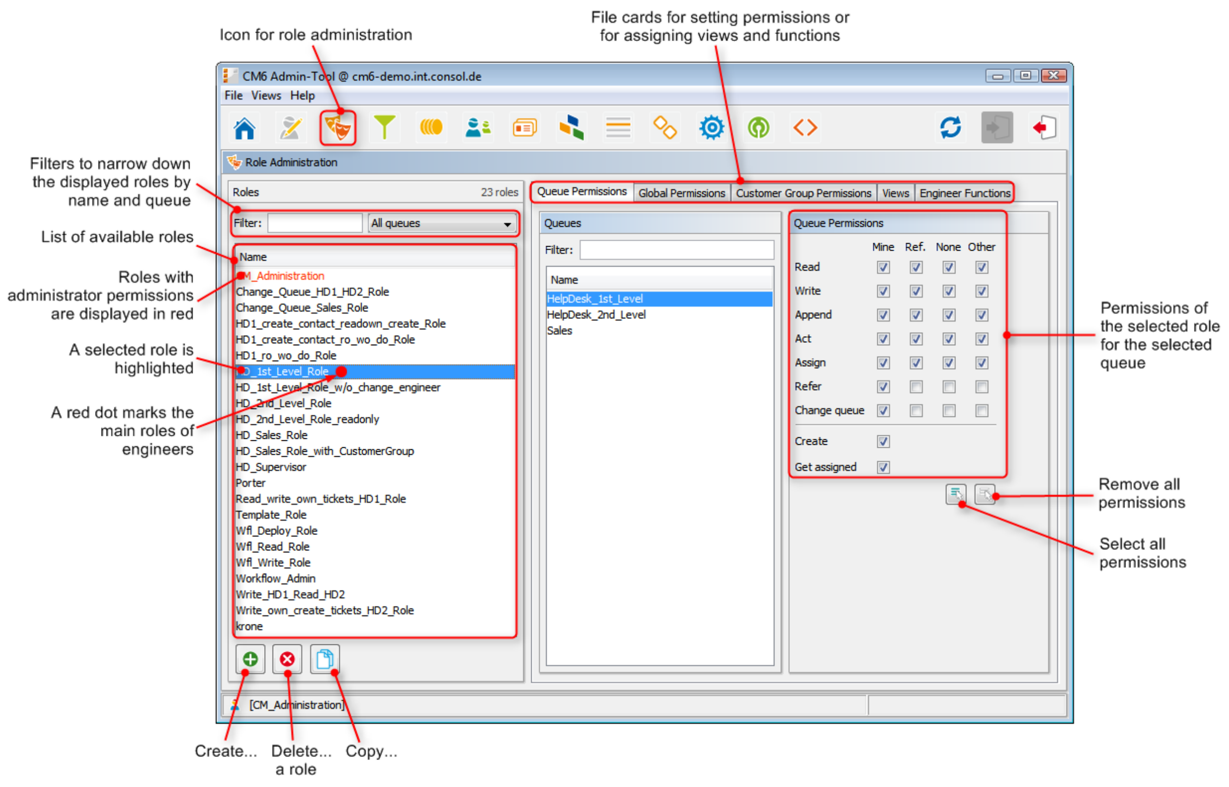

Icon for role administration
File cards for setting permissions or for assigning views and functions
Filters to narrow down the displayed roles by name and queue
List of available roles
Roles with administrator permissions are displayed in red
A selected role is highlighted
A red dot marks the main roles of engineers
Permissions of the selected role for the selected queue
Remove all permissions
Select all permissions
Create...
Delete... a role
Copy...
CM6 Admin-Tool @ cm6-demo.int.consol.de
File Views Help
Role Administration
Roles
23 roles
Filter:
All queues
Name
CM_Administration
Change_Queue_HD1_HD2_Role
Change_Queue_Sales_Role
HD1_create_contact_readown_create_Role
HD1_create_contact_ro_wo_do_Role
HD1_ro_wo_do_Role
HD_1st_Level_Role
HD_1st_Level_Role_w/o_change_engineer
HD_2nd_Level_Role
HD_2nd_Level_Role_readonly
HD_Sales_Role
HD_Sales_Role_with_CustomerGroup
HD_Supervisor
Porter
Read_write_own_tickets_HD1_Role
Template_Role
Wfl_Deploy_Role
Wfl_Read_Role
Wfl_Write_Role
Workflow_Admin
Write_HD1_Read_HD2
Write_own_create_tickets_HD2_Role
krone
Queue Permissions
Global Permissions
Customer Group Permissions
Views
Engineer Functions
Queues
Filter:
Name
HelpDesk_1st_Level
HelpDesk_2nd_Level
Sales
Queue Permissions
Mine Ref. None Other
Read
Write
Append
Act
Assign
Refer
Change queue
Create
Get assigned
[CM_Administration]

Fig. 1: ConSol*CM Admin Tool - Role Administration: Queue Permissions

Attention:

All changes on the Role Management tabs are immediately effective resp. after clicking the *OK* button. You do not have to click on in the icon bar. Engineers have to log in again to use their new roles. Views become effective after clicking *F5* (page refresh) in the Web Client.

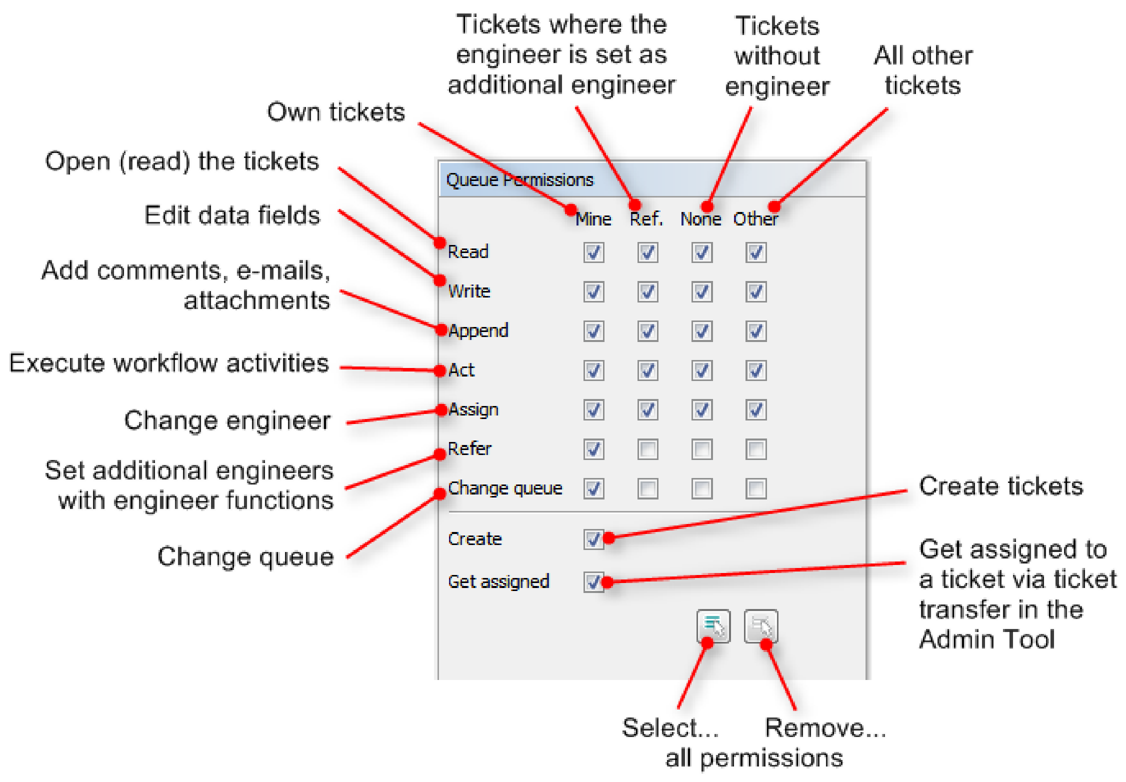
5.2.1 Create a Role

Click on below the role list to create a new role. A pop-up window appears where you can enter the role name. Afterwards you have to set the permissions of this role using the following tabs on the right side of the page (see also the preceding picture):

- Queue Permissions
- Global Permissions
- Customer Group Permissions
- Views
- Engineer Functions

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 or to act in the system.


Tickets where the engineer is set as additional engineer
Tickets without engineer
All other tickets
Own tickets
Open (read) the tickets
Edit data fields
Add comments, e-mails, attachments
Execute workflow activities
Change engineer
Set additional engineers with engineer functions
Change queue
Create tickets
Get assigned to a ticket via ticket transfer in the Admin Tool
Select... Remove... all permissions
Queue Permissions

	Mine	Ref.	None	Other
Read	✓	✓	✓	✓
Write	✓	✓	✓	✓
Append	✓	✓	✓	✓
Act	✓	✓	✓	✓
Assign	✓	✓	✓	✓
Refer	✓			
Change queue	✓			
Create	✓			
Get assigned	✓			

Fig. 2: ConSol*CM Admin Tool - Role Administration: Setting Queue Permissions

The following permissions can be set:

- **Read**
  Read tickets.

- **Write**
  Edit data fields (default fields, custom 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, e-mails, attachments, time booking entries), i.e. add content in the ticket protocol.
- **Act**
  Execute workflow activities, i.e. move the ticket forward in the workflow.
- **Assign**
  Assign tickets to another engineer and/or be assigned as engineer for tickets by other engineers.
- **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 (appropriate permissions for the *initial* and the *target queue* required).

Attention:

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 well.

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

- **Mine**
  Own tickets.
- **Ref**
  Tickets for 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 on the corresponding check box to assign one or more permissions to the desired ticket range.

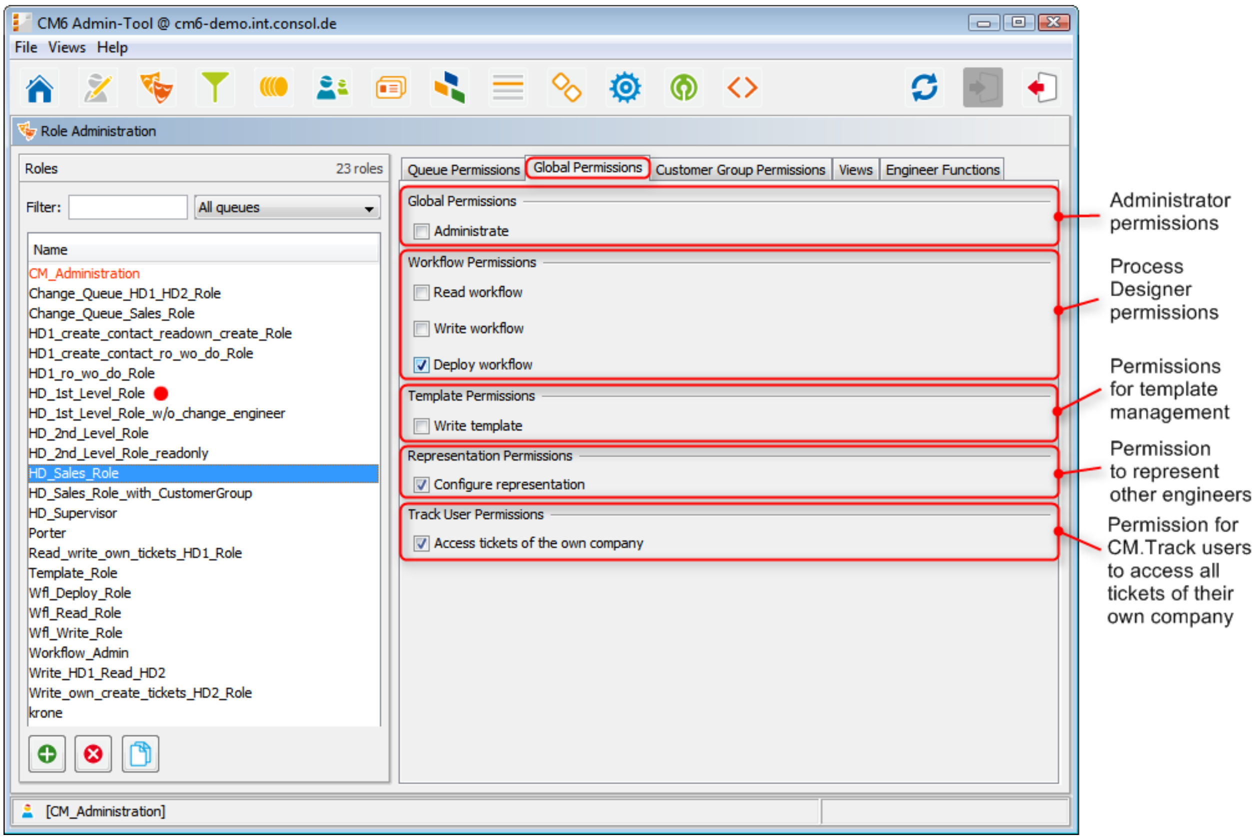
Two general permissions can also be set:

- **Create**
  An engineer is allowed to create tickets in this queue.
- **Get assigned**
  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 on below the list. Clicking on removes all selections.

Tab Global Permissions

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


CM6 Admin-Tool @ cm6-demo.int.consol.de
File Views Help
Role Administration
Roles 23 roles
Filter: All queues
Name
Queue Permissions
Global Permissions
Customer Group Permissions
Views
Engineer Functions
Administrator permissions
Process Designer permissions
Permissions for template management
Permission to represent other engineers
Permission for CM.Track users to access all tickets of their own company
[CM_Administration]

CM_Administration
Change_Queue_HD1_HD2_Role
Change_Queue_Sales_Role
HD1_create_contact_readown_create_Role
HD1_create_contact_ro_wo_do_Role
HD1_ro_wo_do_Role
HD_1st_Level_Role ●
HD_1st_Level_Role_w/o_change_engineer
HD_2nd_Level_Role
HD_2nd_Level_Role_readonly
HD_Sales_Role
HD_Sales_Role_with_CustomerGroup
HD_Supervisor
Porter
Read_write_own_tickets_HD1_Role
Template_Role
Wfl_Deploy_Role
Wfl_Read_Role
Wfl_Write_Role
Workflow_Admin
Write_HD1_Read_HD2
Write_own_create_tickets_HD2_Role
krone

Global Permissions
- [ ] Administrate

Workflow Permissions
- [ ] Read workflow
- [ ] Write workflow
- [✓] Deploy workflow

Template Permissions
- [ ] Write template

Representation Permissions
- [✓] Configure representation

Track User Permissions
- [✓] Access tickets of the own company

Fig. 3: ConSol*CM Admin Tool - Role Administration: Global Permissions

You can specify the following:

- **Global Permissions**
  - **Administrate**
    Provides administrator access to the system, this applies to the Admin Tool, the Process Designer, and admin access to the Web Client.
- **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 Template Manager which is used to create and edit e-mail and commentary templates, see section The ConSol*CM Template Manager for details.
    - to use the Doc 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.

Important information about representation configurations

In case a representation rule has been configured by an engineer (see *ConSol*CM User Manual* for details), all e-mails which are sent by CM are sent to the original recipient ⚠ and ⚠ his/her current representative. 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).

- **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.

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 user group(s) to one or more roles.

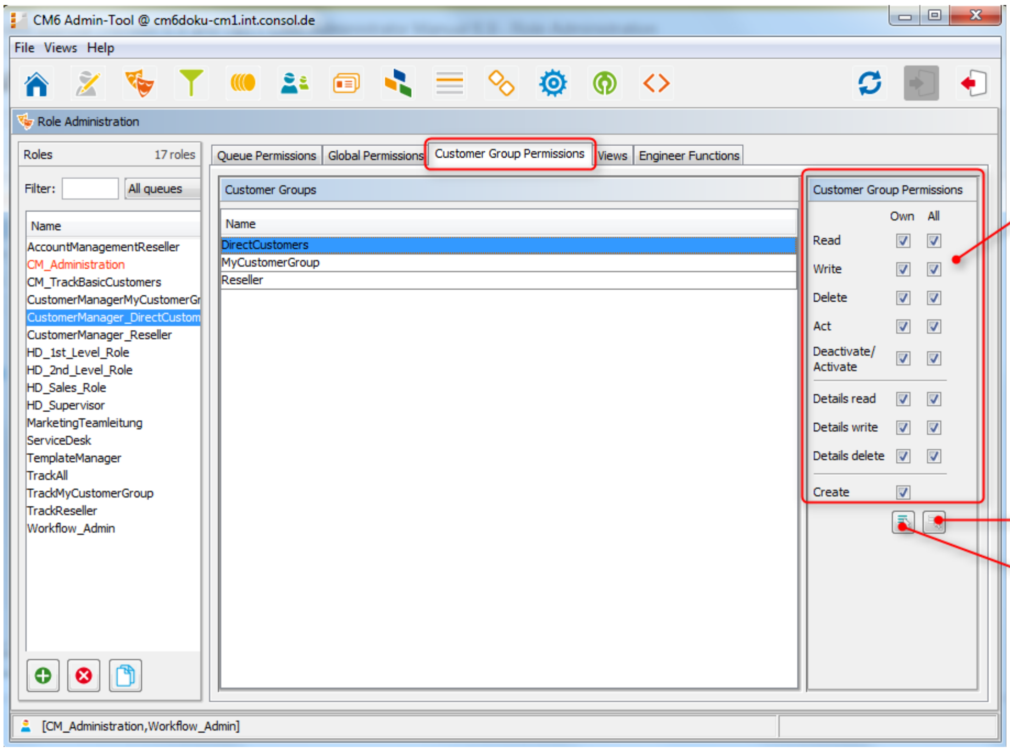


Fig. 4: ConSol* Admin Tool - Assigning Permissions for Customer Groups to a Role

The access rights which can be granted have been modified compared to previous ConSol*CM versions. New rights have been added which concern a new section of the customer page. The contact page as well as the company page in the Web Client have a new section, the *Details* section.

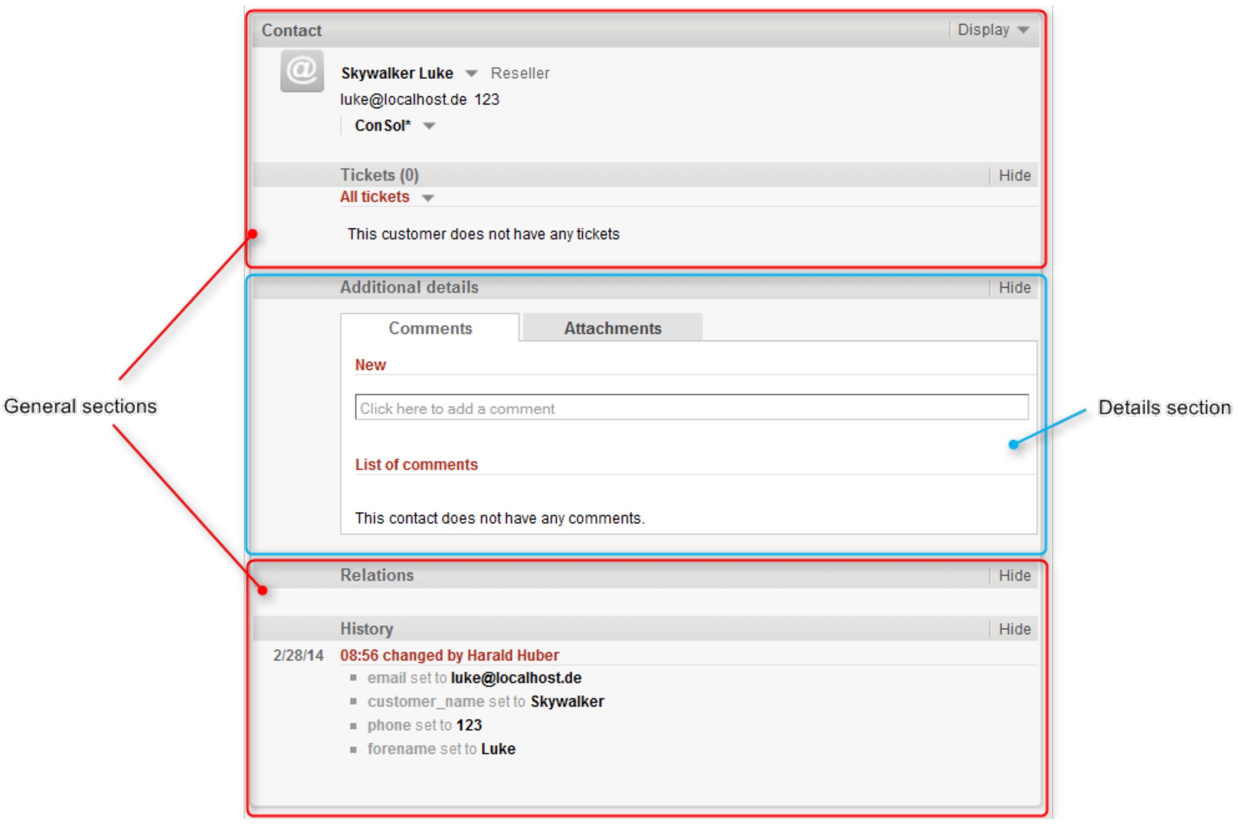


Fig. 5: ConSol*CM Web Client - Details Section of a Contact Page

The following access permissions can be granted:

- **Customer type**
  Refers to the tickets of the customer.
  - **Own**
    All customers (main or additional) which are contacts at tickets which are currently owned by the engineer or where the engineer is set as additional contact
  - **All**
    All customers.
- **General sections**
  - **Read**
    Read the customer data.
  - **Write**
    Write/modify the customer data.
  - **Delete**
    Delete a customer data set. Transfer all tickets associated with a customer of this customer group to another customer.
  - **Act**
    Execute actions for this customer (see section Action Framework for details about customer actions).
  - **Deactivate/activate**
    Deactivate and (re-)activate the customer or company. It is not possible to create tickets for a deactivated company.
- **Details section**
  - **Details read**
    Read customer data in the *Details* section.
  - **Details write**
    Write/modify customer data in the *Details* section.
  - **Details delete**
    Delete customer data in the *Details* section.
- **General**
  - **Create**
    Create a customer data set. In a two-level customer data model this refers to customer as well as to company data sets.

Attention:

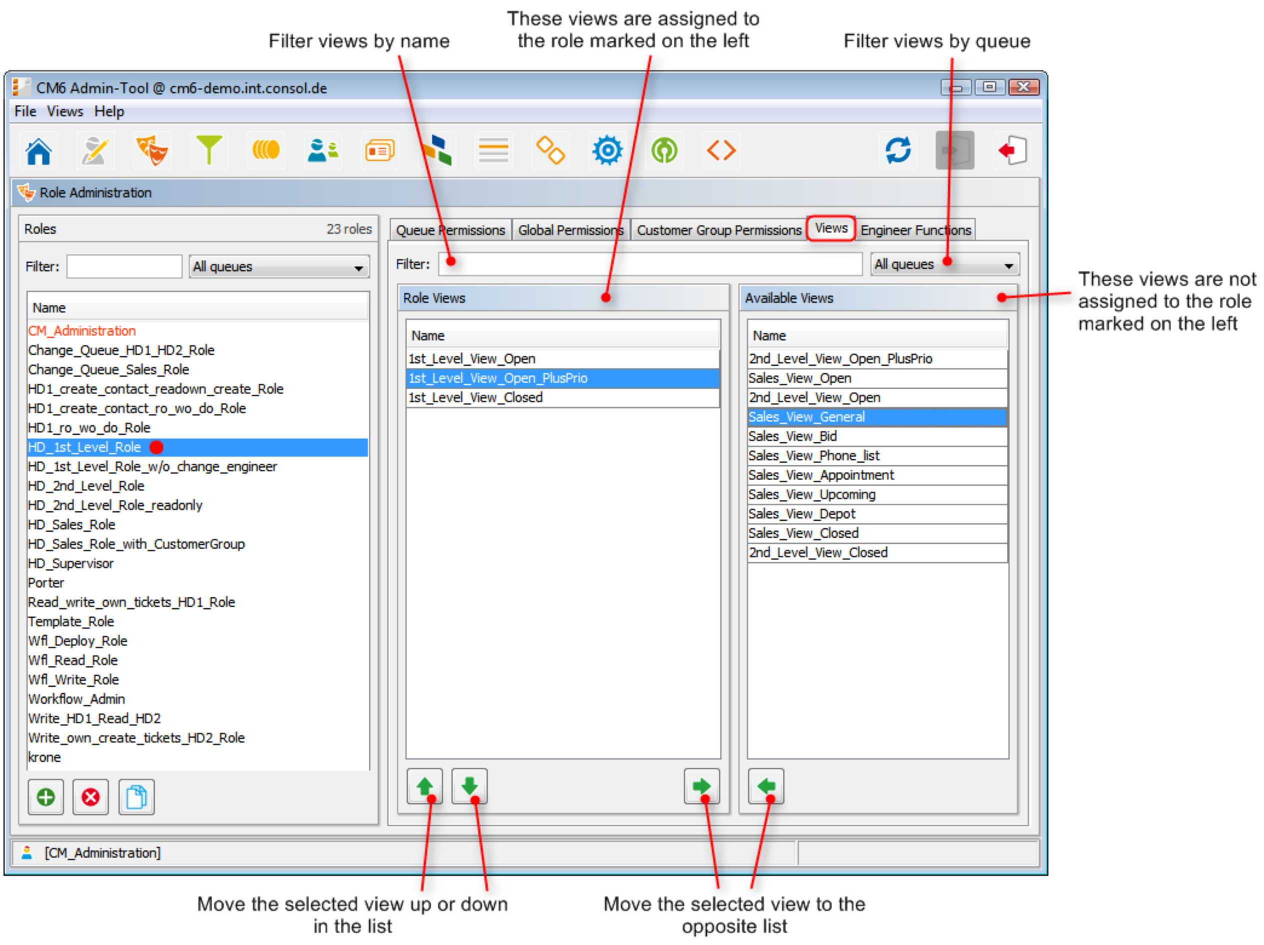
Please keep in mind that an engineer must have at least *read* permissions for a customer group to open and/or create tickets for customers in this group!

Tab 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 View Administration). The displayed views can be filtered by name and queue. Assigning views is optional.

Information:

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


Filter views by name
These views are assigned to the role marked on the left
Filter views by queue
These views are not assigned to the role marked on the left
Move the selected view up or down in the list
Move the selected view to the opposite list
CM6 Admin-Tool @ cm6-demo.int.consol.de
File Views Help
Role Administration
Roles 23 roles
Filter: All queues
Queue Permissions
Global Permissions
Customer Group Permissions
Views
Engineer Functions
Filter: All queues
Role Views
Available Views
[CM_Administration]

Roles:

- CM_Administration
- Change_Queue_HD1_HD2_Role
- Change_Queue_Sales_Role
- HD1_create_contact_readown_create_Role
- HD1_create_contact_ro_wo_do_Role
- HD1_ro_wo_do_Role
- HD_1st_Level_Role
- HD_1st_Level_Role_w/o_change_engineer
- HD_2nd_Level_Role
- HD_2nd_Level_Role_readonly
- HD_Sales_Role
- HD_Sales_Role_with_CustomerGroup
- HD_Supervisor
- Porter
- Read_write_own_tickets_HD1_Role
- Template_Role
- Wfl_Deploy_Role
- Wfl_Read_Role
- Wfl_Write_Role
- Workflow_Admin
- Write_HD1_Read_HD2
- Write_own_create_tickets_HD2_Role
- krone

Role Views:

Name
1st_Level_View_Open
1st_Level_View_Open_PlusPrio
1st_Level_View_Closed

Available Views:

Name
2nd_Level_View_Open_PlusPrio
Sales_View_Open
2nd_Level_View_Open
Sales_View_General
Sales_View_Bid
Sales_View_Phone_list
Sales_View_Appointment
Sales_View_Upcoming
Sales_View_Depot
Sales_View_Closed
2nd_Level_View_Closed

Fig. 6: ConSol*CM Admin Tool - Role Administration: 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 on 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 on .

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 View Administration. Only in case 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 arrow buttons and .

5.2.2 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 (see section Additional User Attributes for creation of engineer functions).

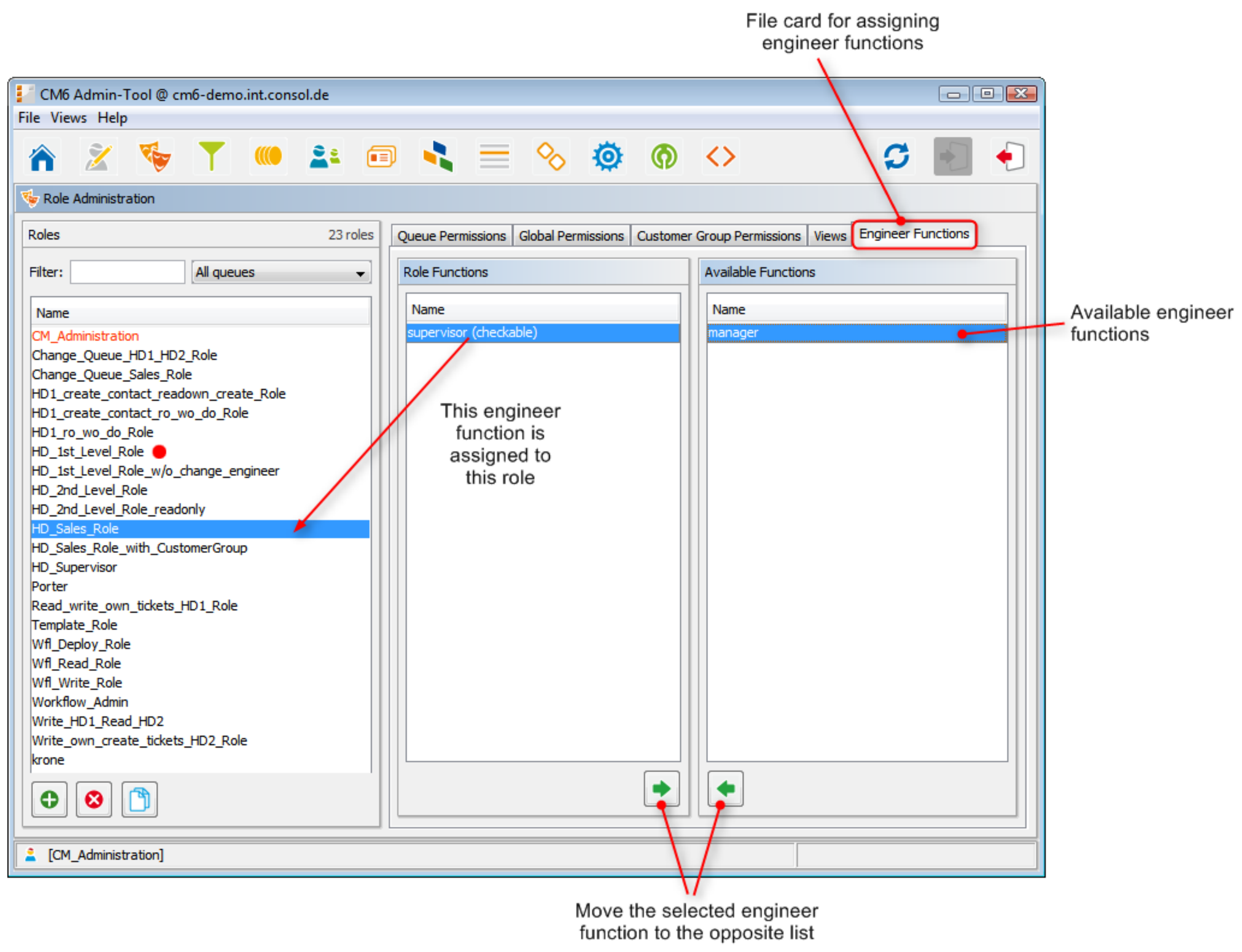

File card for assigning engineer functions
CM6 Admin-Tool @ cm6-demo.int.consol.de
File
Views
Help
Role Administration
Roles
23 roles
Filter:
All queues
Name
CM_Administration
Change_Queue_HD1_HD2_Role
Change_Queue_Sales_Role
HD1_create_contact_readown_create_Role
HD1_create_contact_ro_wo_do_Role
HD1_ro_wo_do_Role
HD_1st_Level_Role
HD_1st_Level_Role_w/o_change_engineer
HD_2nd_Level_Role
HD_2nd_Level_Role_readonly
HD_Sales_Role
HD_Sales_Role_with_CustomerGroup
HD_Supervisor
Porter
Read_write_own_tickets_HD1_Role
Template_Role
Wfl_Deploy_Role
Wfl_Read_Role
Wfl_Write_Role
Workflow_Admin
Write_HD1_Read_HD2
Write_own_create_tickets_HD2_Role
krone
Queue Permissions
Global Permissions
Customer Group Permissions
Views
Engineer Functions
Role Functions
Name
supervisor (checkable)
Available Functions
Name
manager
This engineer function is assigned to this role
Available engineer functions
[CM_Administration]
Move the selected engineer function to the opposite list

Fig. 7: ConSol*CM Admin Tool - Role Administration: 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 on 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 on .

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 will obtain the rights of a role immediately after assignment (without an additional update of the system).

5.2.3 Delete a Role

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

Attention:

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

5.2.4 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 on 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.

5.2.5 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.

5.3 Related Topics

- Engineer administration
- Customer groups
- Queues
- Views
- Additional user attributes

6 View Administration

- Introduction to View Administration
- View Administration Using the Admin Tool
    - Create a View
        - Queue Filter
        - Scope Filter
        - Static Criterion
        - Dynamic Criterion
    - Edit a View
    - Delete a View
    - Copy a View
- Related Topics

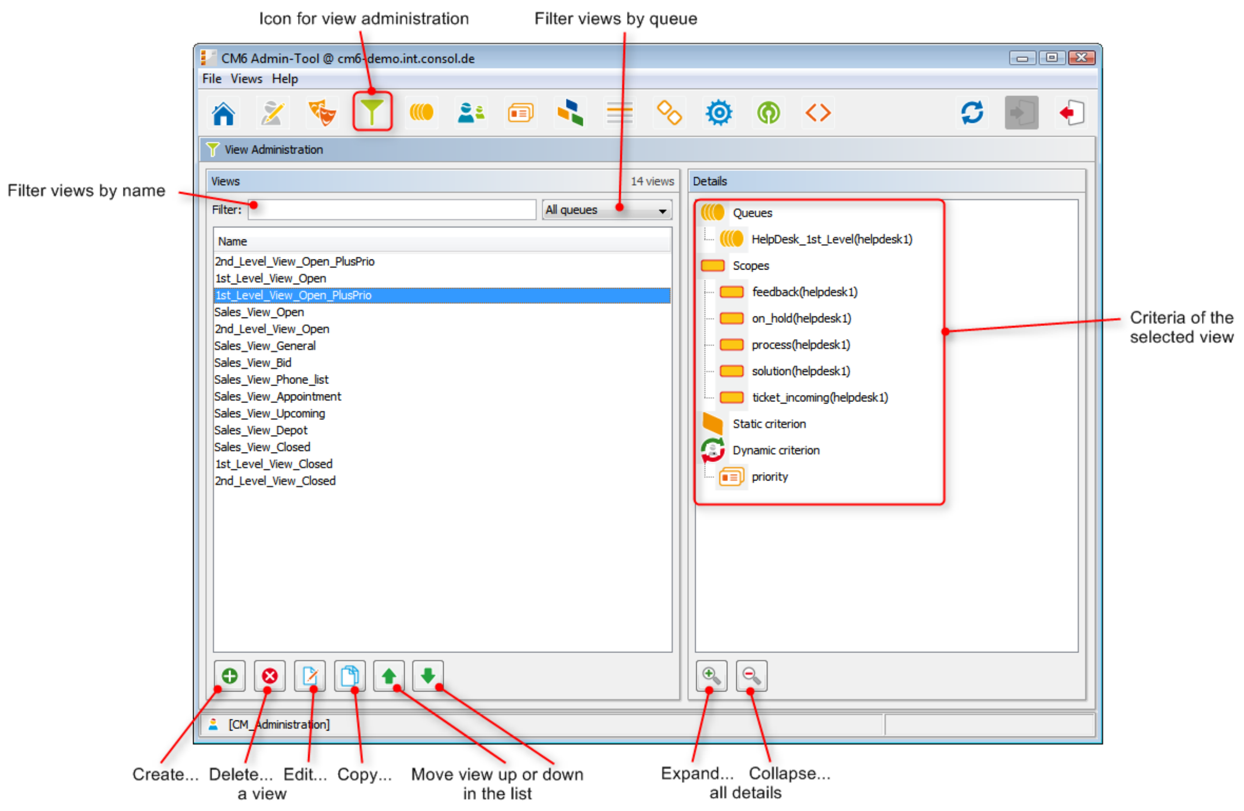
6.1 Introduction to View Administration

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.

6.2 View Administration Using the Admin Tool


Icon for view administration
Filter views by queue
Filter views by name
Criteria of the selected view
Create... Delete... Edit... Copy... a view
Move view up or down in the list
Expand... Collapse... all details
CM6 Admin-Tool @ cm6-demo.int.consol.de
File Views Help
View Administration
Views
14 views
Filter:
All queues
Name
Details
[CM_Administration]

2nd_Level_View_Open_PlusPrio
1st_Level_View_Open
1st_Level_View_Open_PlusPrio
Sales_View_Open
2nd_Level_View_Open
Sales_View_General
Sales_View_Bid
Sales_View_Phone_list
Sales_View_Appointment
Sales_View_Upcoming
Sales_View_Depot
Sales_View_Closed
1st_Level_View_Closed
2nd_Level_View_Closed

- Queues
  - HelpDesk_1st_Level(helpdesk1)
- Scopes
  - feedback(helpdesk1)
  - on_hold(helpdesk1)
  - process(helpdesk1)
  - solution(helpdesk1)
  - ticket_incoming(helpdesk1)
- Static criterion
- Dynamic criterion
  - priority

Fig. 1: ConSol*CM Admin Tool - View Administration

6.2.1 Create a View

After clicking on ⊕ 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 on 🌐 you can localize view name and description. The pop-up window *Localize* shows the available locales on the left side. Enter the corresponding view name or description in the *Value* field for each additional language on the right. After clicking *Save* the name or description will be displayed in the respective language of the engineer's locale.

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

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

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 on ⬅ . To remove an assigned queue, select it and click on ➡ . Continue with the *Next >* button, in order to define scope filters, too.

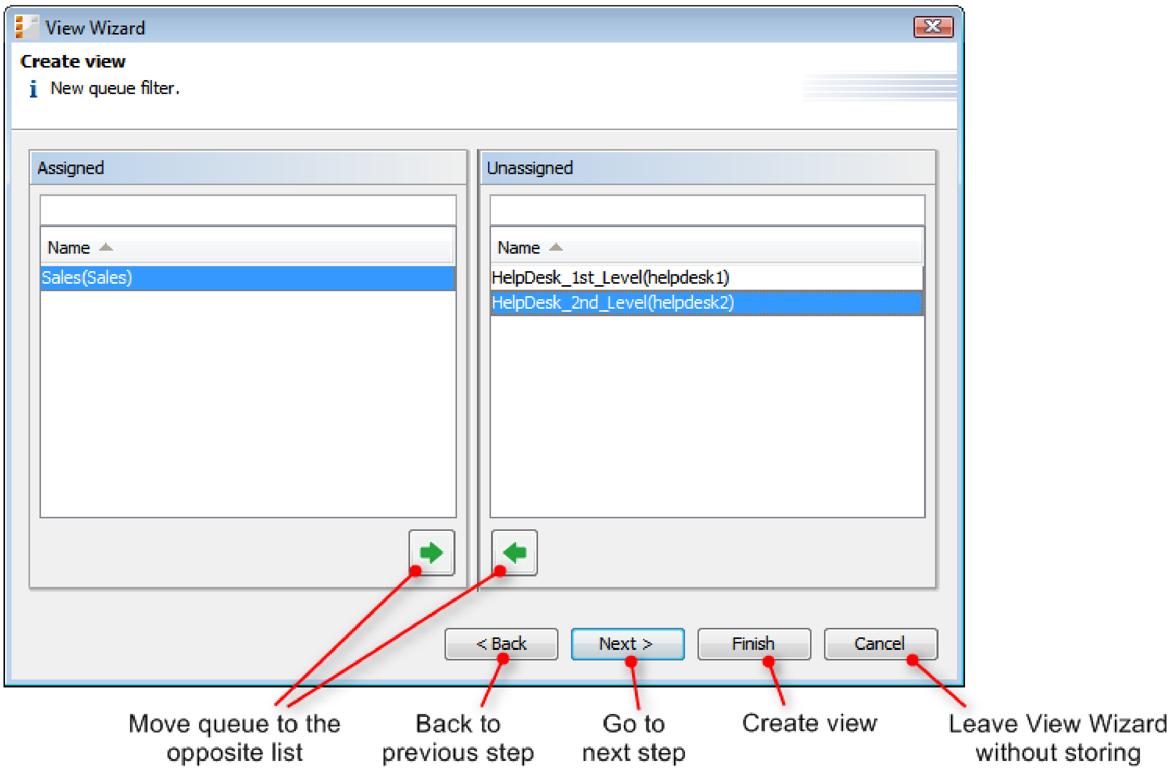


Fig. 2: ConSol*CM Admin Tool - View Wizard: Queue Filter

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.

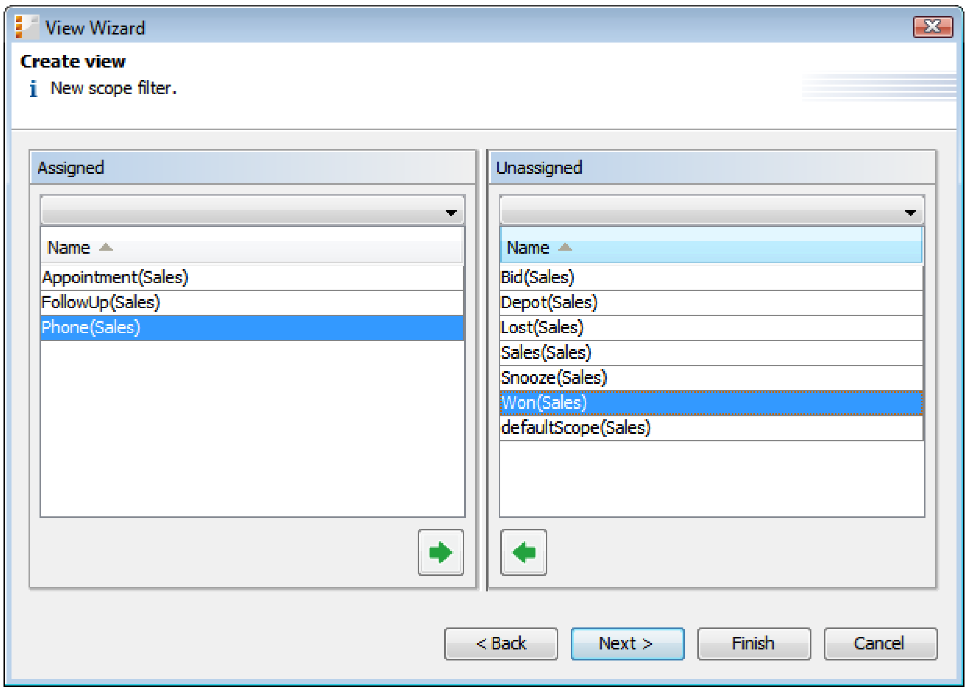
Select the desired scopes in the list *Unassigned* and move them to the list *Assigned* by clicking on ⬅ . To remove assigned scopes, select them and click on ➡ . Continue with the *Next >* button, if you want to define further criteria. Otherwise click on *Finish* to create the view.

Information:

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

Attention:

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, when 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!


View Wizard

Create view

New scope filter.

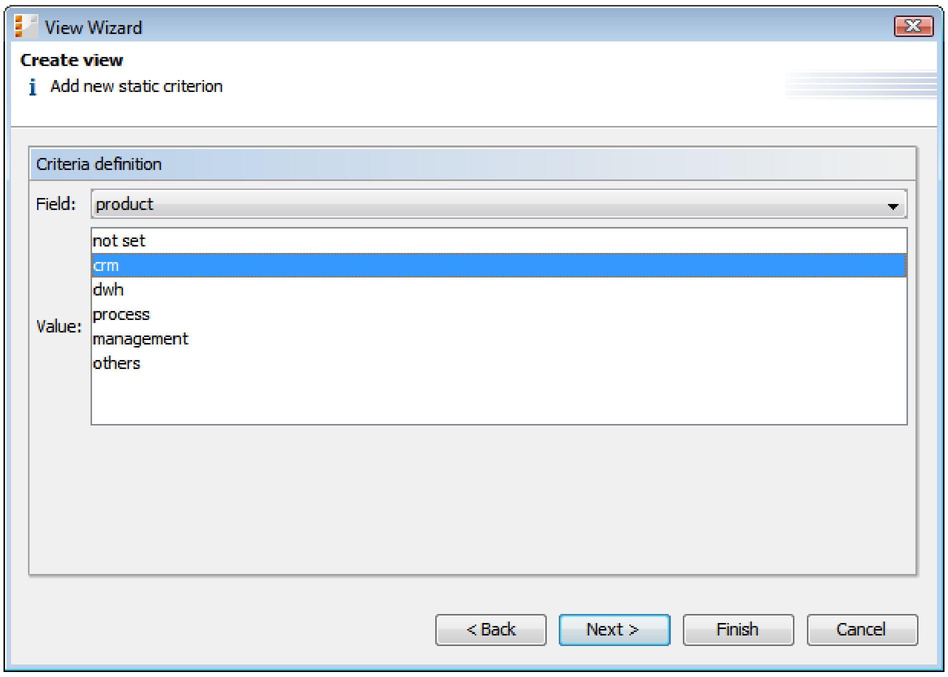
Assigned	Unassigned
Name	Name
Appointment(Sales)	Bid(Sales)
FollowUp(Sales)	Depot(Sales)
Phone(Sales)	Lost(Sales)
	Sales(Sales)
	Snooze(Sales)
	Won(Sales)
	defaultScope(Sales)

Fig. 3: ConSol*CM Admin Tool - View Wizard: Scope Filter

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.

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, too. Otherwise click on *Finish* to create the view.



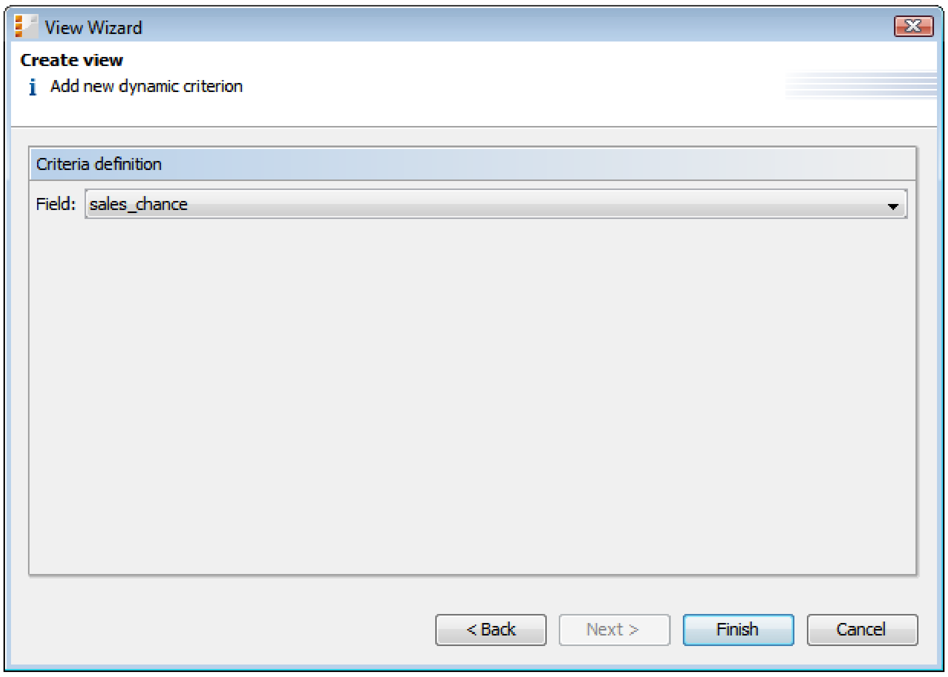
Field: product

Value: not set, crm (selected), dwh, process, management, others

Fig. 4: ConSol*CM Admin Tool - View Wizard: 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 parameter themselves. This can be done in the Web Client by editing the *User Profile*. Additionally, the administrator can adjust the value individually for each engineer on the *View criteria* tab of the engineer administration (see section Engineer Administration). Please see the *ConSol*CM User Manual* for a detailed description of working with views.



Field: sales_chance

Fig. 5: ConSol*CM Admin Tool - View Wizard: Dynamic Criterion

Click on *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.

Attention:

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 Engineer Administration), 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.

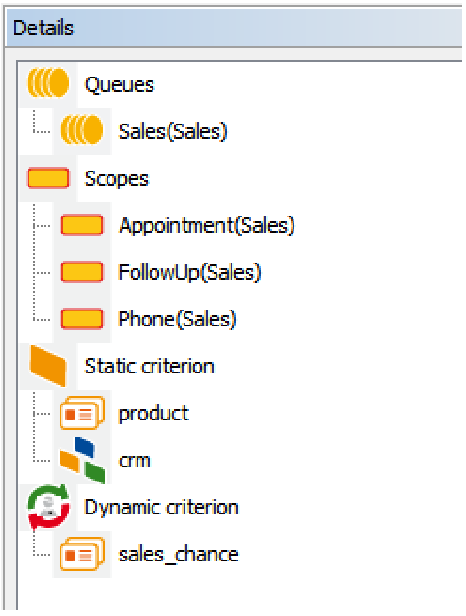


Fig. 6: ConSol*CM Admin Tool - View Administration: View Details

You can expand or collapse all details by clicking on or below the list.

Attention:

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 GUI 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.

6.2.2 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 on a filter criterion with the right mouse button. The following drop-down menu appears:

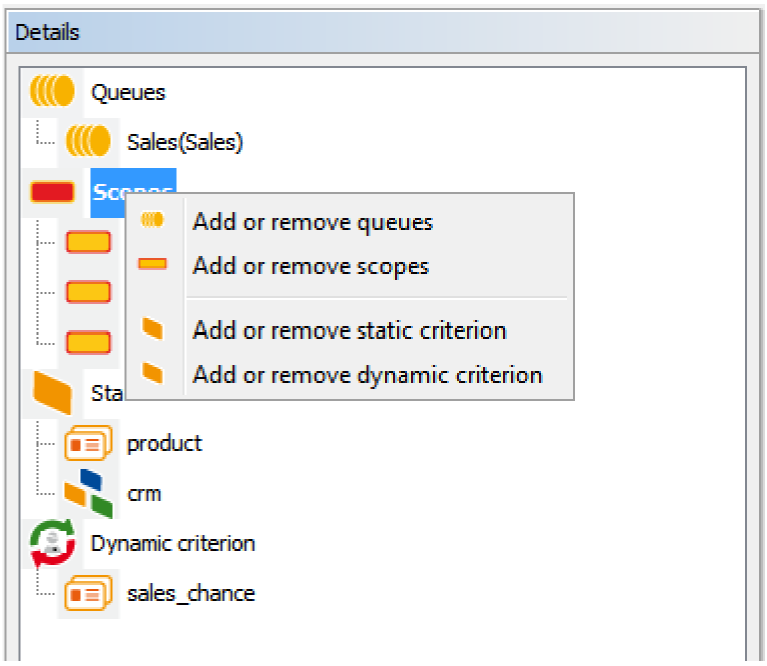


Fig. 7: ConSol*CM Admin Tool - View Administration: Edit a View

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 on 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*.

Information:

You cannot edit view criteria by clicking on . Here you can only modify name and description of a view.

6.2.3 Delete a View

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

6.2.4 Copy a View

The icon 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 on the icon.

6.3 Related Topics

- Queues
- Workflow scope (see separate document *ConSol*CM Process Designer Manual*)
- Roles
- Engineer administration

7 Queue Administration

- Introduction to Queue Administration
- Queue Administration Using the Admin Tool
    - Filter the Queue List
    - Create a Queue
    - Edit a Queue
    - Delete a Queue
    - Copy a Queue
    - Enable or Disable a Queue
- Related Topics

7.1 Introduction to Queue Administration

Queues are a central element of ConSol*CM. Tickets are grouped in queues, e.g. for certain tasks or work groups. To each queue a single workflow is assigned which controls the processing steps of all tickets in this queue. For example, there might be one queue *Helpdesk*, one queue *Marketing*, and one queue *Sales*.

In a queue you define:

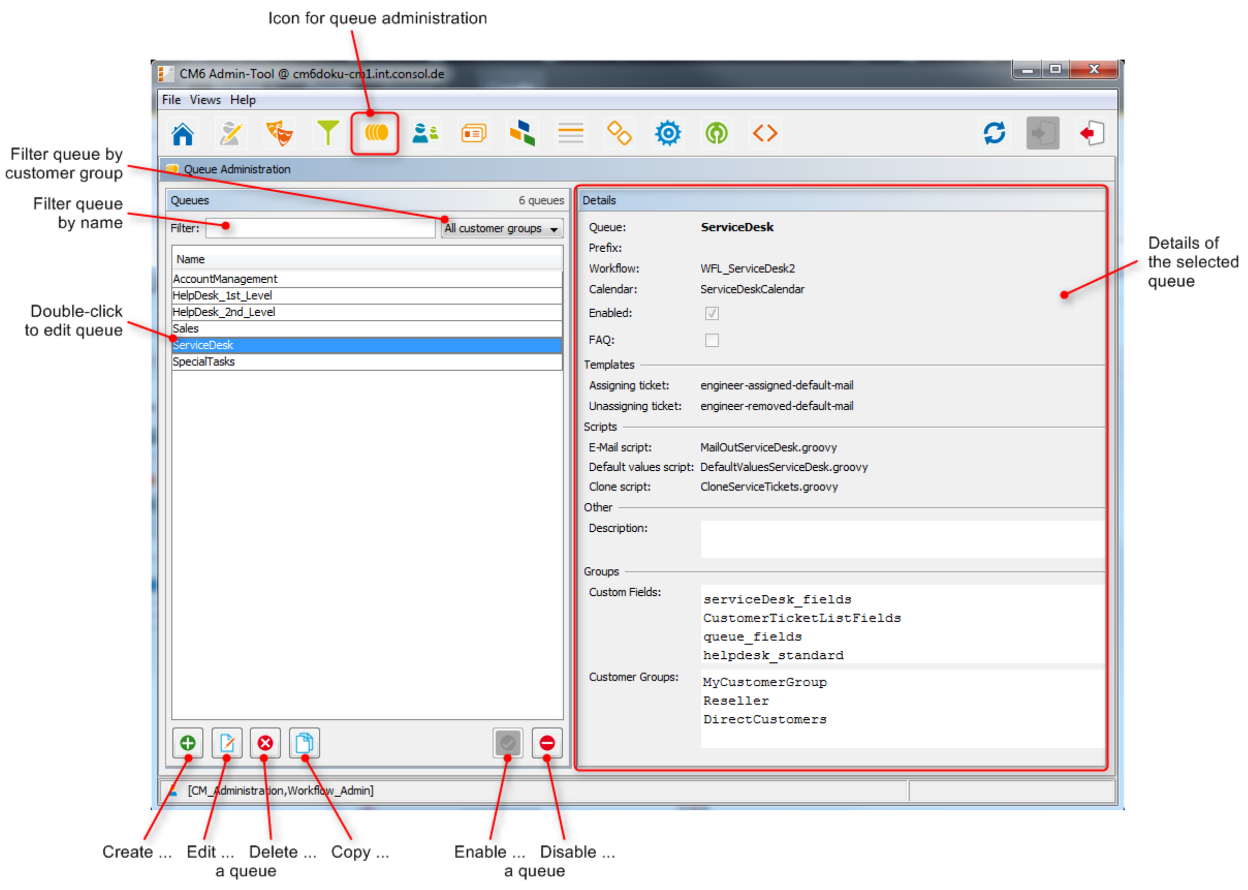
- The workflow of the queue (mandatory), i.e. the process which should be used for all tickets in the queue (e.g. all tickets of a department). A queue can only have one workflow but a workflow can be used by multiple queues.
- The template for the e-mails which are sent to engineers when a ticket is assigned or removed (optional).
- Several scripts that define the behavior of tickets in this queue (optional).
- One or more customer group(s) which are associated with the queue. Only for customers of those customer groups tickets can be created in the queue (one customer group is mandatory, more are optional).
- The business calendar (i.e. the working hours) which should be applied for tickets in this queue (optional).
- The data fields (Custom Fields) which should be available in tickets of the queue. They are defined by assigning custom field groups to the queue (some mandatory, some optional).
- The classes of text which should be available for tickets in this queue (optional).
- The project(s) which should be available for time booking in tickets of the queue (optional).

Information:

As a central element the queue uses various objects and elements that have been defined at another place, i.e. on another page of the Admin Tool, so usually the elements which are later required for the queue definition are defined first. However, except for the workflow, all parameters can be modified even after a queue definition has been saved. So you can configure the queue using an iterative approach if you like.

Furthermore, a queue is the basis for the assignment of access permissions, please see section Role Administration for details.

7.2 Queue Administration Using the Admin Tool


Icon for queue administration
Filter queue by customer group
Filter queue by name
Double-click to edit queue
Details of the selected queue
Create ... Edit ... Delete ... Copy ... a queue
Enable ... Disable ... a queue
CM6 Admin-Tool @ cm6doku-cm1.int.consol.de
File Views Help
Queue Administration
Queues
6 queues
Filter:
All customer groups
Name
Details
[CM_Administration,Workflow_Admin]

AccountManagement
HelpDesk_1st_Level
HelpDesk_2nd_Level
Sales
ServiceDesk
SpecialTasks

Queue:	ServiceDesk
Prefix:	
Workflow:	WFL_ServiceDesk2
Calendar:	ServiceDeskCalendar
Enabled:	✓
FAQ:	☐
Templates	
Assigning ticket:	engineer-assigned-default-mail
Unassigning ticket:	engineer-removed-default-mail
Scripts	
E-Mail script:	MailOutServiceDesk.groovy
Default values script:	DefaultValuesServiceDesk.groovy
Clone script:	CloneServiceTickets.groovy
Other	
Description:	
Groups	
Custom Fields:	serviceDesk_fields CustomerTicketListFields queue_fields helpdesk_standard
Customer Groups:	MyCustomerGroup Reseller DirectCustomers

Fig. 1: ConSol*CM Admin Tool - Queue Administration

7.2.1 Filter the Queue List

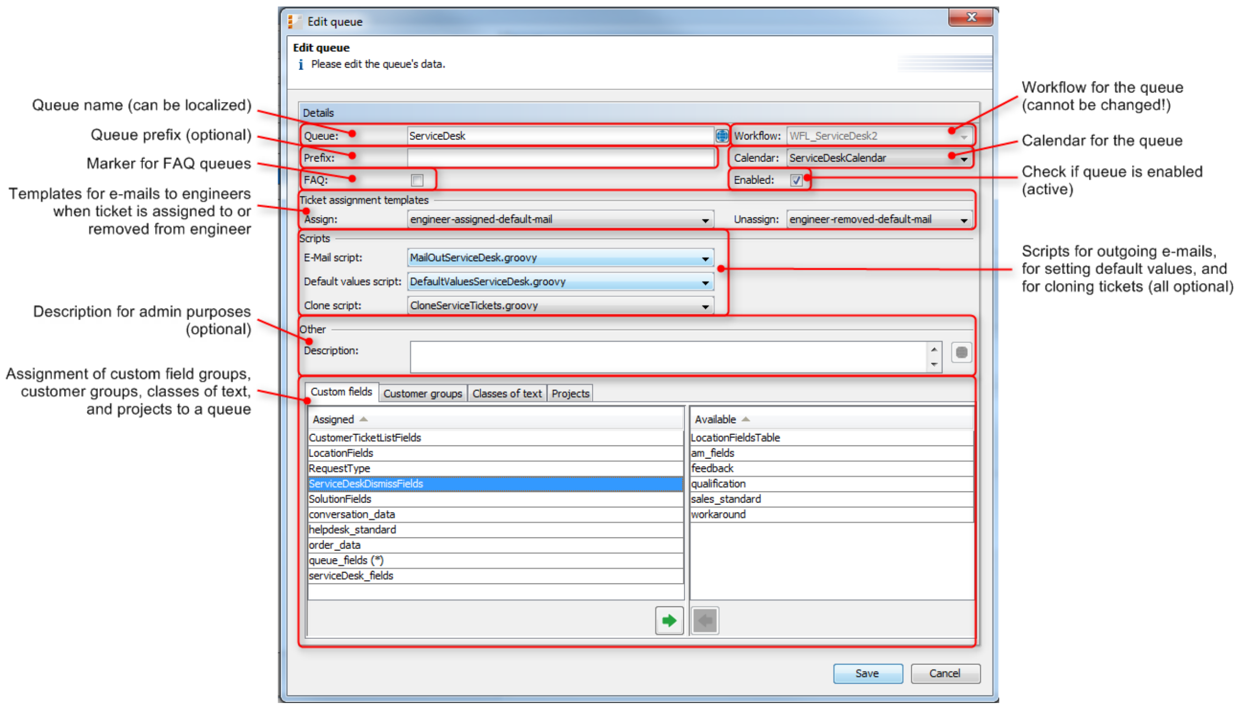
Queues you want to edit or copy can be found faster, if you enter filter information in the fields above the queue list.

You can filter for queues which

- contain a certain text string (blanks are interpreted, too) and/or
- are specific for customer groups.

7.2.2 Create a Queue

You create a new queue by clicking on ⊕ below the queue list. The following pop-up window appears:


Queue name (can be localized)
Queue prefix (optional)
Marker for FAQ queues
Templates for e-mails to engineers when ticket is assigned to or removed from engineer
Description for admin purposes (optional)
Assignment of custom field groups, customer groups, classes of text, and projects to a queue
Workflow for the queue (cannot be changed!)
Calendar for the queue
Check if queue is enabled (active)
Scripts for outgoing e-mails, for setting default values, and for cloning tickets (all optional)

Edit queue

Edit queue

Please edit the queue's data.

Details

Queue: ServiceDesk
Workflow: WFL_ServiceDesk2
Prefix:
Calendar: ServiceDeskCalendar
FAQ: ☐
Enabled: ☑

Ticket assignment templates

Assign: engineer-assigned-default-mail
Unassign: engineer-removed-default-mail

Scripts

E-Mail script: MailOutServiceDesk.groovy
Default values script: DefaultValuesServiceDesk.groovy
Clone script: CloneServiceTickets.groovy

Other

Description:

Custom fields | Customer groups | Classes of text | Projects

Assigned	Available
CustomerTicketListFields	LocationFieldsTable
LocationFields	am_fields
RequestType	feedback
ServiceDeskDismissFields	qualification
SolutionFields	sales_standard
conversation_data	workaround
helpdesk_standard	
order_data	
queue_fields (*)	
serviceDesk_fields	

Save | Cancel

Fig. 2: ConSol*CM Admin Tool - Queue Administration: Create a Queue

Here, you can define the queue details:

- **Queue:**
  Enter the technical queue name in this field. Click on 🌐 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 name will be displayed in the default language.
- **Workflow:**
  Choose the workflow for the queue from this list.

Best Practice:

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!

Warning:

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. if the ticket ID shall indicate to which queue or organizational structure it belongs.

  > **Attention:**
  >
  > The prefix will remain with the ticket name if the ticket is moved to another queue.

- **Calendar:**
  Choose the business calendar for the queue from the list. Calendars define working hours, holidays and the valid timezone (see Configuration - Tab 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 Tab Business Calendars the calendar is created and the acctive and vacation times are configured.
  - In the queue configuration page 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 the work with time triggers.
- **Enable:**
  If this check box is ticked, the queue is immediately available in the system after saving. If the check box is not ticked, the queue is disabled. In enabled queues, you can create tickets, in disabled queues, this is not possible.
- **FAQ:**
  Ticking this check box marks the queue as a knowledge base for CM.Track users. They can search for tickets of this queue in CM.Track, the ConSol*CM Web Portal. Please see also section CM.Track: FAQs in CM.Track on this topic.
- **Ticket assignment templates:**
  Here you can choose e-mail templates that shall be used for an automatic e-mail which is sent to the (new) engineer when a ticket is assigned to an engineer (*Assign*) or to the (old) engineer when a ticket is retrieved from an engineer (*Unassign*). When you have defined the templates in the *Script and Template Administration* of the Admin Tool (see section Admin Tool Templates) they will be available in the drop-down menu. When you do not want the CM system to send an automatic e-mail 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 Appendix C (System Properties) for details.

- **Scripts**:
  Scripts are used to automate recurring tasks and activities. They are managed and stored in the *Script and Template Administration* (see section Scripts). You can assign:
  - **E-mail script**
    Choose a script from the list if outgoing e-mails for this queue should be modified by the script, e.g. to contain default values like the sender or address fields. The e-mail script indicated here is the last script that processes an outgoing e-mail so all former settings will be overwritten (except for REPLY-TO, see warning below!) in case a variable has been set before. All scripts of type *E-mail* that are stored in the script section are available, please make sure to pick the correct one.

When you set the **REPLY-TO-address** in the queue-specific outgoing e-mail script, the *mail.reply.to* system property must not be set (because it would overwrite the configured value)! Leave the value for *mail.reply.to* empty.

That means when you use a queue-specific outgoing e-mail script for *one* queue you have to define outgoing e-mail scripts for *all* queues because the *mail.reply.to* property can no longer be used! In this case you will have to perform the following steps for each ⚠ queue where e-mails are relevant:

1. Create the queue-specific outgoing e-mail script for each specific queue, e.g. *ChangeOutgoingMail.groovy_Queue1,ChangeOutgoingMail.groovy_Queue2, ... ChangeOutgoingMail.groovy_Queue_n.*
2. Maybe create an outgoing e-mail script which should be used for all queues where no queue-specific script is required, e.g. *ChangeOutgoingMail.groovy_Queue_standard*
3. Assign this script to the queues in the Queue Administration.

In the **CM Web Client**, you will find the following behavior:

- When the REPLY-TO-address is set using the *mail.reply.to* system property, the *Reply to* field is displayed in the Ticket E-Mail Editor and pre-filled with the value of the system property, e.g. mymailaddress@consol.de. The engineer can use the field value or can also change it (which is not recommended!)
- When the *mail.reply.to* system property is empty, the *Reply to* field is not displayed in the Ticket E-Mail Editor in *edit* mode. In *display* mode, the REPLY-TO-address which was used is displayed. The REPLY-TO-address has to be set in the queue-specific outgoing e-mail script. If it is not set, either because an outgoing e-mail script is present where the REPLY-TO-address is not set or because there is no outgoing e-mail script defined for the queue, the e-mail is sent without a specific REPLY-TO-address (NULL).

- - **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. The script has to be present in the *Script and Template Administration* of the Admin Tool and has to be of type *Default values*.
  - **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 *Script and Template 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.
- **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.
- **Tab Custom fields:**
  In order to show data fields (custom fields) in tickets of the queue, you have to assign the respective custom field groups here.
- **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 User attributes page.

On each tab you can assign a selected entry by clicking on ← and remove it by clicking on → .

Click on *Save* afterwards to create the queue. The details of the new queue are displayed on the right side of the page.

7.2.3 Edit a Queue

If you want to edit a queue, select it in the list and click on or just double-click the name of the queue. Modify the queue details and click *Save* to store your modifications.

⚠ **Attention:**

You cannot change the workflow of a queue.

7.2.4 Delete a Queue

Select the queue you want to delete in the list and click on . If you confirm the following dialog with *Yes*, the queue will be deleted and is no longer available in the system.

> **Attention:**
>
> 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.

7.2.5 Copy a Queue

The icon 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 on the name or click on to open the edit window where you can modify the name and details of the queue. Click *Save* to store your modifications.

> **Attention:**
>
> You cannot change the workflow of the queue.

7.2.6 Enable or Disable a Queue

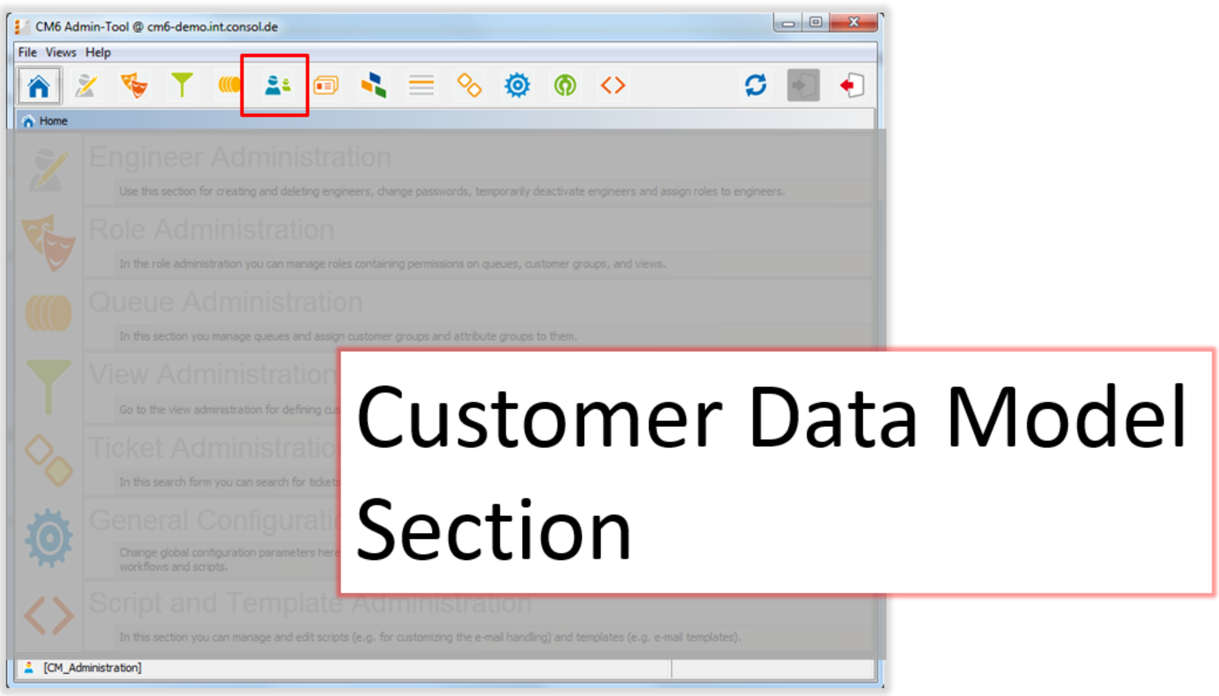
You can disable a queue to prevent that new tickets can be opened in this queue. That way you can re-activate the queue later and do not have to delete it. To disable a queue, select the queue in the queue list and click on . The entry in the list is now shown in italics. Just click on at the bottom of the page, if you want to enable the queue again.

You can still read tickets in a disabled queue (provided you have the read access rights for this queue), but you cannot process tickets, i.e. they cannot be moved to the next step in the process using workflow activities.

7.3 Related Topics

- Workflow (see *ConSol*CM Process Designer Manual*)
- Views
- Scripts and templates
- Customer groups
- Custom fields
- Classes of text
- Projects

8 Customer Data Model Section



Use this section for creating and deleting engineers, change passwords, temporarily deactivate engineers and assign roles to engineers.

In the role administration you can manage roles containing permissions on queues, customer groups, and views.

In this section you manage queues and assign customer groups and attribute groups to them.

Go to the view administration for defining cu

In this search form you can search for tickets

Change global configuration parameters here
workflows and scripts.

In this section you can manage and edit scripts (e.g. for customizing the e-mail handling) and templates (e.g. e-mail templates).

In this section, you learn how to set-up and manage the Flexible ConSol*CM Customer Data Model, FlexCDM and all related topics. The following subjects are explained:

- The CM Customer Data Model: FlexCDM
    - Introduction to FlexCDM
    - A Short Introduction to FlexCDM-Specific Web Client Functionalities
    - Setting Up the Customer Data Model
    - Data Object Group Field Management and GUI Design
    - Templates for Customer Data
- Managing Customer Groups
- Customer Relations
- Action Framework
- Additional User Attributes

9 CM6 Administrator Manual 6.9.4 - The CM Customer Data Model - FlexCDM

9.1 The CM Customer Data Model: FlexCDM

Starting with version 6.9, ConSol*CM offers a very flexible and powerful customer administration based on the *FlexCDM*, the Flexible Customer Data Model.

In the following sections, all aspects of the new data model are explained.

- Introduction to FlexCDM
- A Short Introduction to FlexCDM-Specific Web Client Functionalities
- Setting Up the Customer Data Model
- Data Object Group Field Management and GUI Design
- Templates for Customer Data

9.2 Introduction to FlexCDM

- FlexCDM at a Glance
  - Flexible Customer Data Model
- Introduction to FlexCDM Objects
  - Important Terms
- Management of FlexCDM Objects Using the Admin Tool

Because the customer data model *FlexCDM*, which has been introduced to ConSol*CM with version 6.9, is rather complex and very powerful, a separate introduction chapter will help you to understand all the details.

9.2.1 FlexCDM at a Glance

Flexible Customer Data Model

As the name *FlexCDM* suggests, the ConSol*CM customer data model offers a very high degree of flexibility. Various **customer groups** can be defined, each with its particular data model.

Within a customer group, there might be ...

- a *contact* and a *company* level:
  two-level customer model (where a company can contain several contacts)
- only a *contact* or only a *company* level:
  one-level customer model

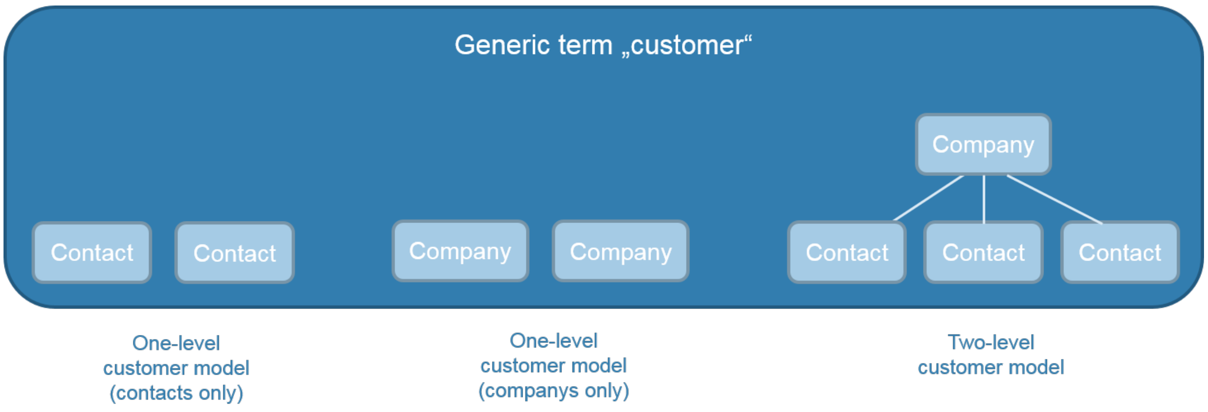


Fig. 1: Types of Customer Data Models in ConSol*CM

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, e-mail, and room number for a contact).

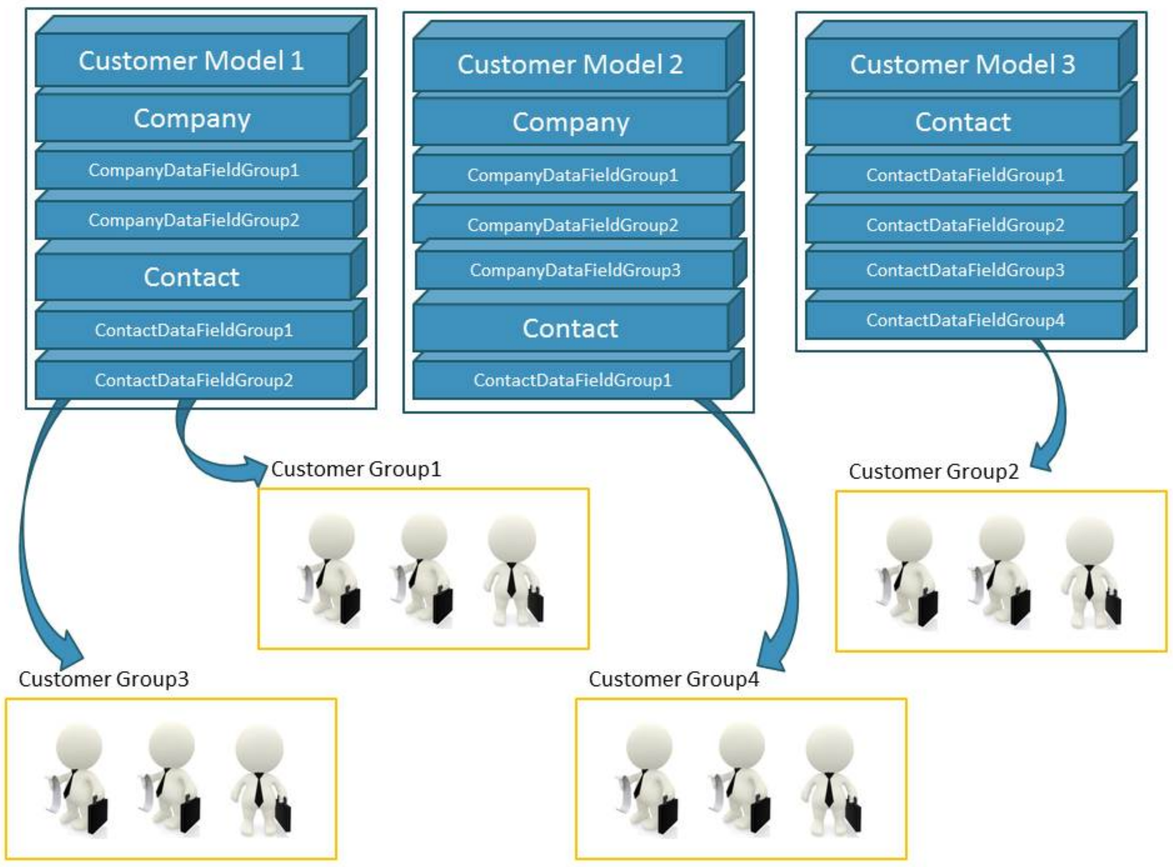


Fig. 2: ConSol*CM FlexCDM - General Principle

Information:

For a two-level customer model:
The terms *company* and *contact* are used to indicate the hierarchical level of an object within FlexCDM. 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 model:
The customer objects in a one-level customer model are either of type *contact* or of type *company*.

The customers which are managed by your ConSol*CM system, the levels and names of all components entirely depend on the configuration of FlexCDM.

Using FlexCDM you can build different realms where each includes a specific customer group and the respective data and processes.

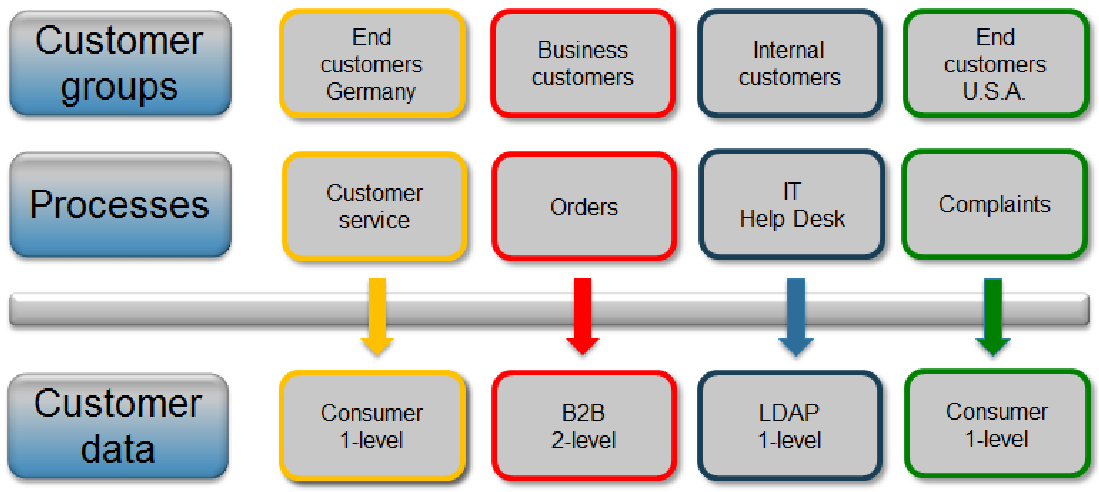


Fig. 3: ConSol*CM FlexCDM - Customer Data Model

Please see section Setting Up the Customer Data Model for a detailed description of the customer management.

9.2.2 Introduction to FlexCDM Objects

In this section, we will give you an overview of all objects which are relevant for the FlexCDM.

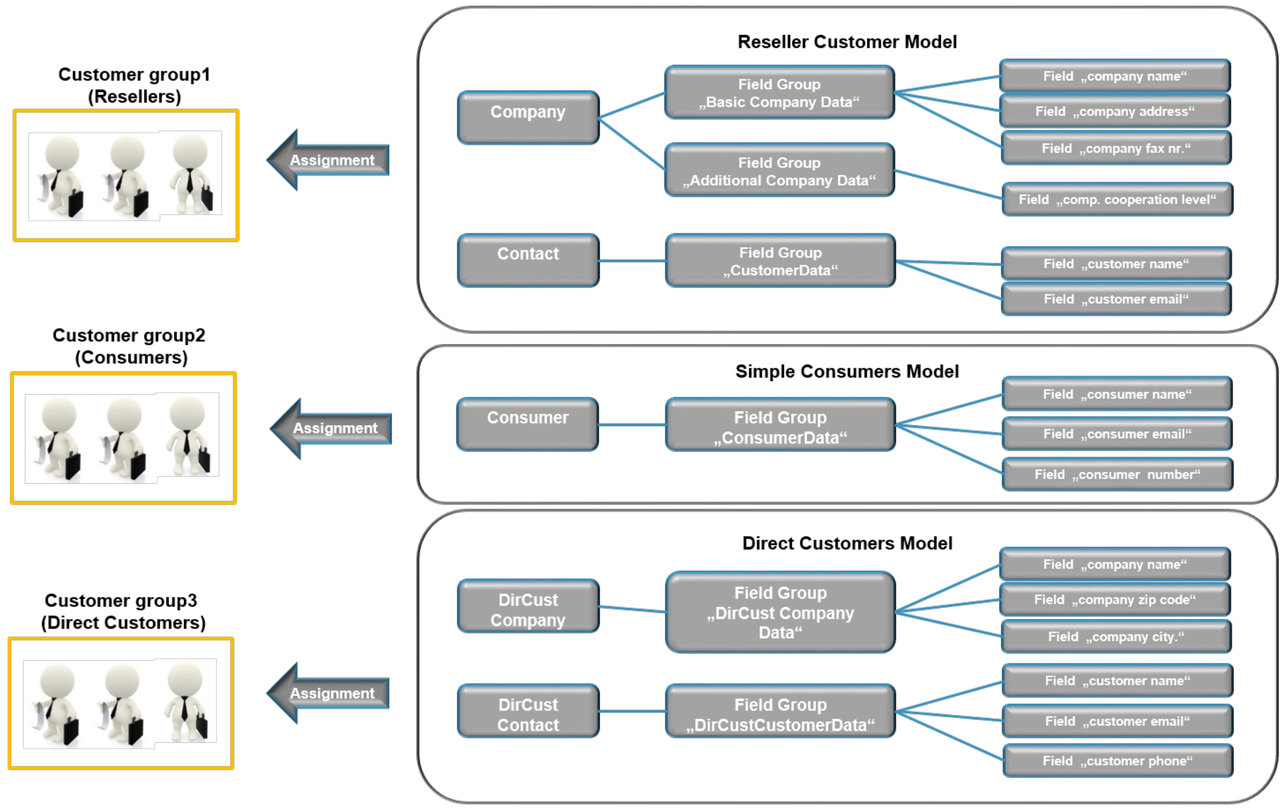


Fig. 4: ConSol*CM FlexCDM - Example Configuration

Important Terms

Here are some important terms for the FlexCDM:

- **Customer**
  General term for customer objects, can be of type *contact* or of type *company*.
- **Company**
  Data object of type *company*, company level.
- **Contact**
  Data object of type *contact*, contact level.
- **Customer group**
  A group of customers with a specific customer data model. The engineer permissions (via roles) for customer management are assigned based on customer groups.
- **Data object**
  An object within the customer data model. The object type can be *customer* (often times a person) or *company*. The technical (Groovy) equivalent is an object of class *Unit*.

- **Data object definition**
  All definitions pertaining to the unit. For a company these are e.g. all data object groups, all group annotations, and the assignment of all templates (for the display of customer data in the Web Client, not to be confused with other templates in ConSol*CM!)
- **Data object group**
  A group comprising one or more data field(s) (data object group fields), analog to a custom field group for ticket data. A data object group can be shown or hidden or it can be displayed as a tab in the (new) customer data group section.
- **Data object group field**
  A single data field (types like custom field types) that can contain customer data, analog to custom fields when defining data fields for ticket data.
- **Customer data model**
  The whole data model that can be assigned to a customer group.
  The data model can have:
    - one level (only contact or only company)
    - two levels (company and contact)

  The customer data model also contains the definitions of data object groups and data object group fields.
- **Customer relations**
  Relations between a company and contacts or between companies or between contacts. All relations in their entirety represent the customer relations network.

9.2.3 Management of FlexCDM Objects Using the Admin Tool

In the Admin Tool, most of the new FlexCDM configuration options are to be found in the *User attributes* tabs.

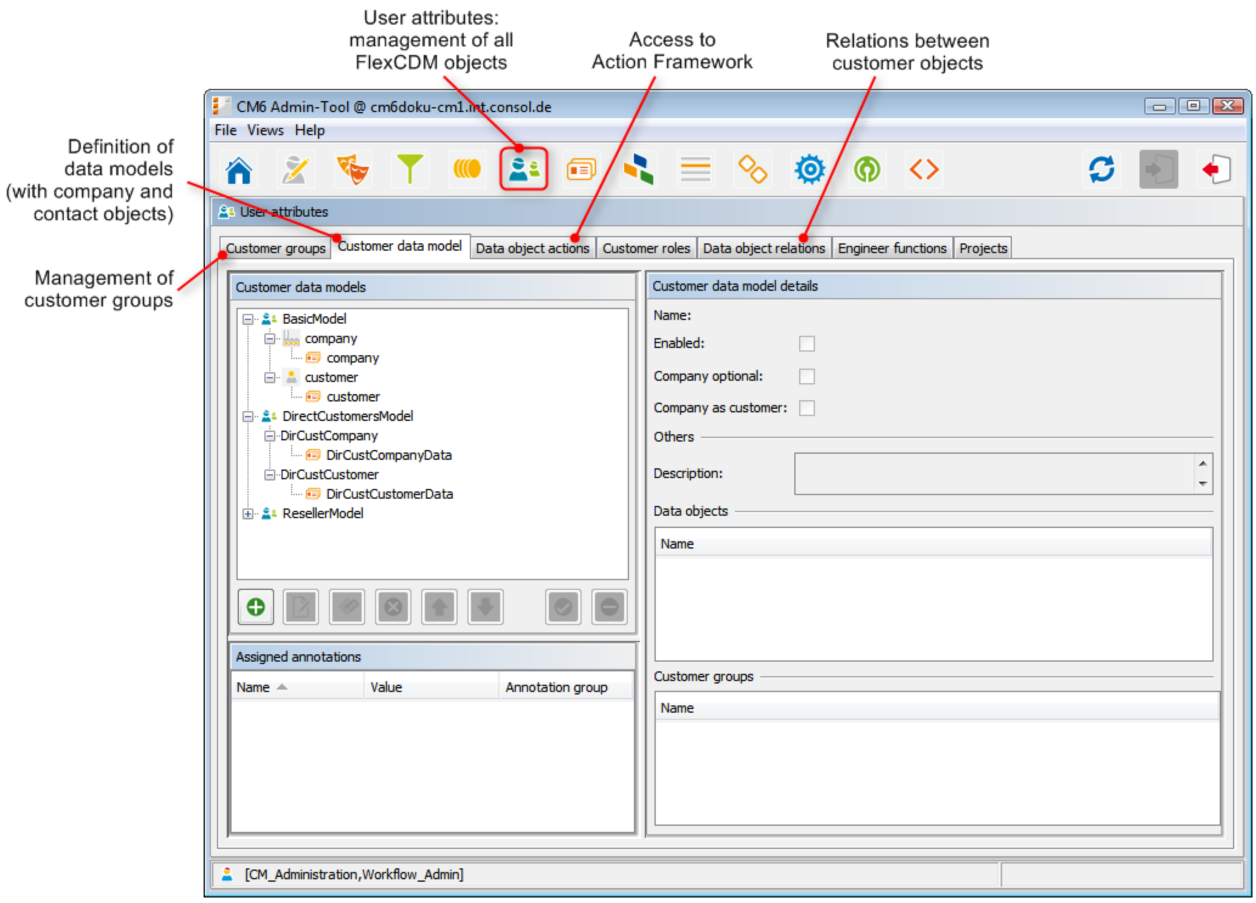


Fig. 5: ConSol*CM Admin Tool - User Attributes Tabs Relevant in FlexCDM

Three tabs are relevant for the definition and management of the customer data model:

- **Customer data model**
  Definition of the data model, 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 GUI Design for details.
- **Data object actions**
  Definition of company and contact actions, please see section Action Framework for details.
- **Data object relations**
  Definition of relation types for references between customer (i.e. contact and company) objects, please see section Customer (Data Object) Relations for details.

The assignment of the data model to customer groups is done in the tab:

- **Customer groups**
  See section Managing Customer Groups

9.3 A Short Introduction to FlexCDM-Specific Web Client Functionalities

- Introduction
- Working with the ConSol*CM Web Client with FlexCDM
  - Example 1: Selecting the Customer Group
  - Example 2: Creating a New Company and Contact
  - Example 3: Using Company and Contact Page
    - Company Page
    - Contact Page
  - Example 4: Setting a Company as Main Customer of a Ticket
  - Example 5: Using Company and Contact Actions
  - Example 6: Setting Relations between Contacts and Companies
  - Example 7: Deactivate a Customer (i.e. a Company or a Contact)
  - Example 8: Using the Ticket Filters on Company or Contact Pages
    - Ticket Filter on Company Page
    - Ticket Filter on Contact Page
  - Example 9: Customer Group Displayed in Quick Search

9.3.1 Introduction

You as an administrator might wonder why a Web Client GUI introduction is provided in an administrator's manual. However, when you want to work with the new customer data model, the *FlexCDM*, you have to know the effects of all administration actions. And of course, those actions are visible in the Web Client. So in this section, we will take the role of an engineer and show several examples for the work with the new customer data model.

All configuration details which are required to understand the system's behavior will be explained in the corresponding sections of the manual.

9.3.2 Working with the ConSol*CM Web Client with FlexCDM

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 drop-down list in the main menu. The name which is displayed is the localized name of the customer group.

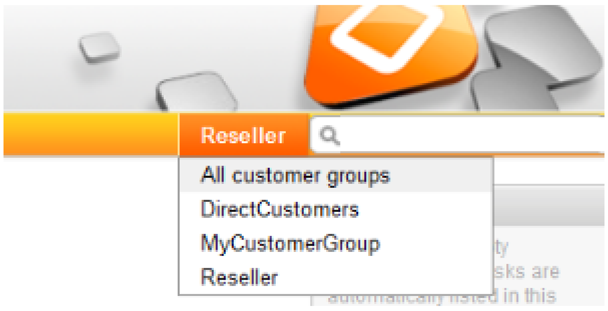


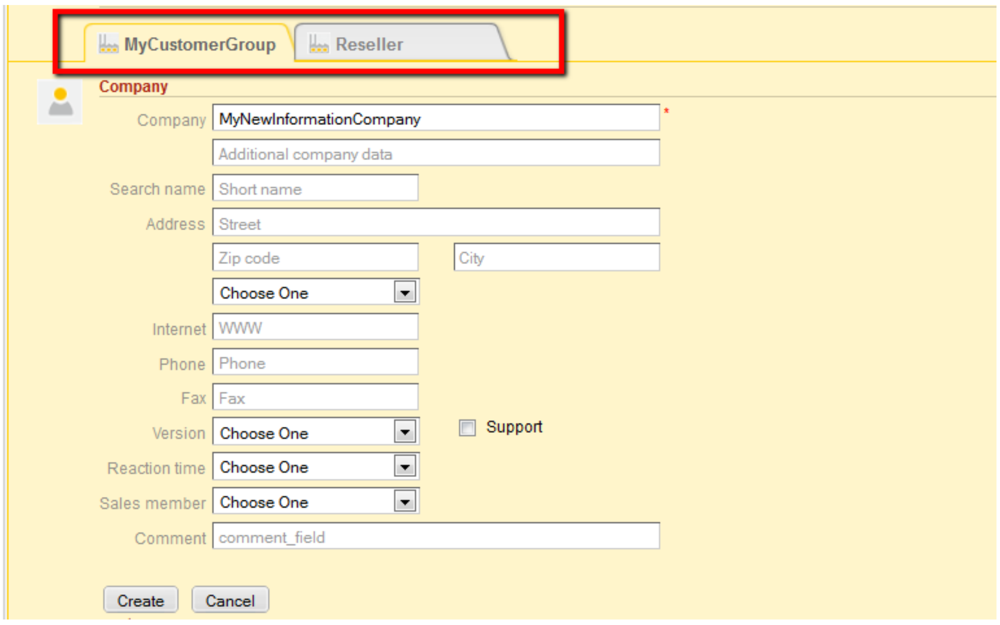
Fig. 1: ConSol*CM Web Client - Selecting a Customer Group

The selection influences the following actions:

- The quick search is performed only within the selected customer group.
- In the detail search, the criterion *customer group* is only offered when *All customer groups* has been selected in the drop-down menu. Otherwise the search uses implicitly only data from the selected customer group.
- In the detail 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 for which the selected customer group has been assigned.
- In the ticket list, only views are available which contain tickets from queues for which the selected customer group has been assigned.

Example 2: Creating a New Company and Contact

In case engineers have access to several customer groups (and have selected *All customer groups* 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. In case 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.


MyCustomerGroup | Reseller

Company

Company	MyNewInformationCompany *
	Additional company data
Search name	Short name
Address	Street
	Zip code / City
	Choose One
Internet	WWW
Phone	Phone
Fax	Fax
Version	Choose One ☐ Support
Reaction time	Choose One
Sales member	Choose One
Comment	comment_field

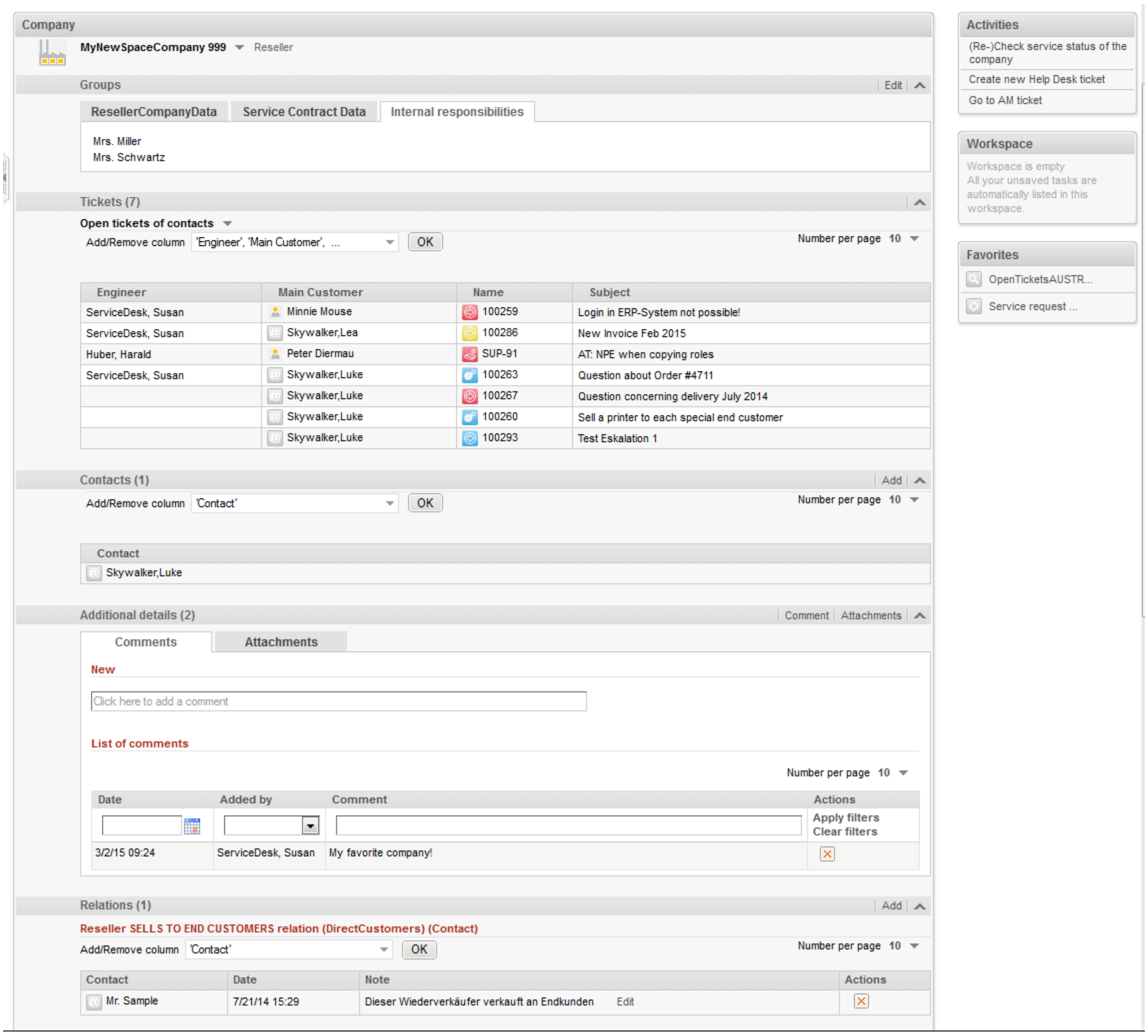
Fig. 2: ConSol*CM Web Client - Creating a New Company within a Customer Group

Example 3: Using Company and Contact Page

Provided there is a two-level customer data model (company and contact) there is **a separate company and a contact page**. 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 which improves the usability.

Company Page



Company

MyNewSpaceCompany 999 Reseller

Groups Edit

ResellerCompanyData | Service Contract Data | Internal responsibilities

Mrs. Miller
Mrs. Schwartz

Tickets (7)

Open tickets of contacts

Add/Remove column 'Engineer', 'Main Customer', ... OK Number per page 10

Engineer	Main Customer	Name	Subject
ServiceDesk, Susan	Minnie Mouse	100259	Login in ERP-System not possible!
ServiceDesk, Susan	Skywalker,Lea	100286	New Invoice Feb 2015
Huber, Harald	Peter Diermau	SUP-91	AT: NPE when copying roles
ServiceDesk, Susan	Skywalker,Luke	100263	Question about Order #4711
	Skywalker,Luke	100267	Question concerning delivery July 2014
	Skywalker,Luke	100260	Sell a printer to each special end customer
	Skywalker,Luke	100293	Test Eskalation 1

Contacts (1) Add

Add/Remove column 'Contact' OK Number per page 10

Contact
Skywalker,Luke

Additional details (2) Comment | Attachments

Comments | Attachments

New

Click here to add a comment

List of comments

Number per page 10

Date	Added by	Comment	Actions
			Apply filters Clear filters
3/2/15 09:24	ServiceDesk, Susan	My favorite company!	☒

Relations (1) Add

Reseller SELLS TO END CUSTOMERS relation (DirectCustomers) (Contact)

Add/Remove column 'Contact' OK Number per page 10

Contact	Date	Note	Actions
Mr. Sample	7/21/14 15:29	Dieser Wiederverkäufer verkauft an Endkunden Edit	☒

Activities

(Re-)Check service status of the company
Create new Help Desk ticket
Go to AM ticket

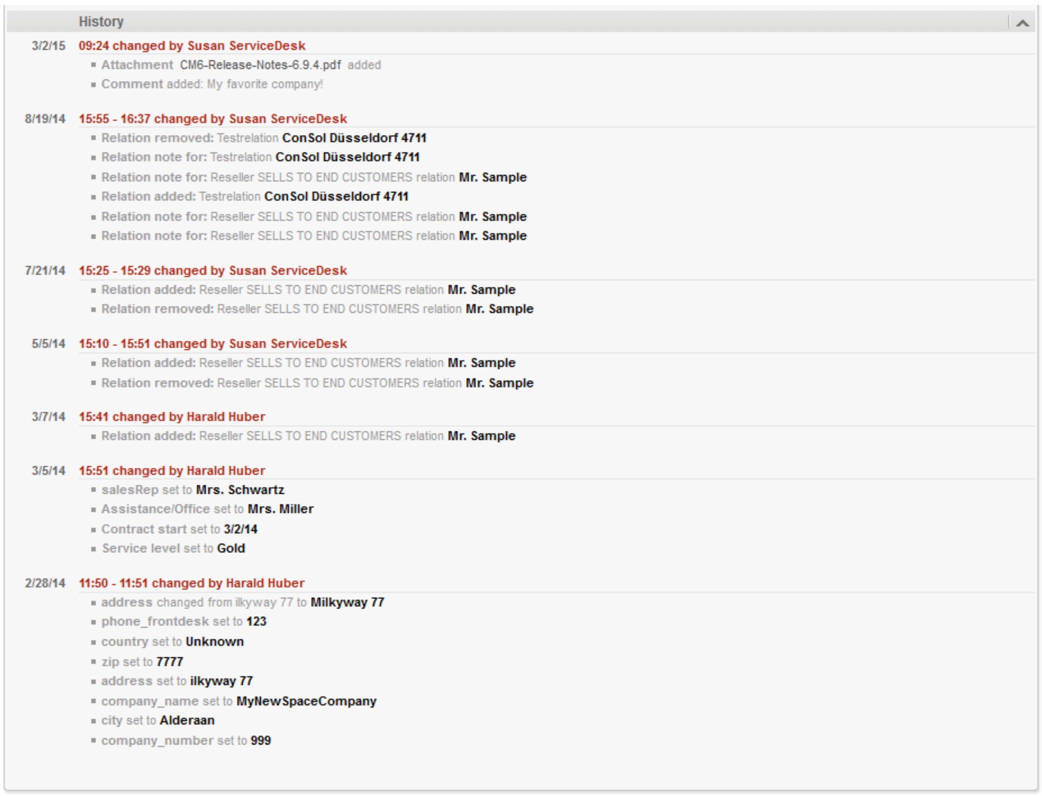
Workspace

Workspace is empty
All your unsaved tasks are automatically listed in this workspace.

Favorites

OpenTicketsAUSTR...
Service request ...

Fig. 3: ConSol*CM Web Client - Company Page, top section


History

3/2/15 09:24 changed by Susan ServiceDesk
- Attachment CM6-Release-Notes-6.9.4.pdf added
- Comment added: My favorite company!

8/19/14 15:55 - 16:37 changed by Susan ServiceDesk
- Relation removed: Testrelation **ConSol Düsseldorf 4711**
- Relation note for: Testrelation **ConSol Düsseldorf 4711**
- Relation note for: Reseller SELLS TO END CUSTOMERS relation **Mr. Sample**
- Relation added: Testrelation **ConSol Düsseldorf 4711**
- Relation note for: Reseller SELLS TO END CUSTOMERS relation **Mr. Sample**
- Relation note for: Reseller SELLS TO END CUSTOMERS relation **Mr. Sample**

7/21/14 15:25 - 15:29 changed by Susan ServiceDesk
- Relation added: Reseller SELLS TO END CUSTOMERS relation **Mr. Sample**
- Relation removed: Reseller SELLS TO END CUSTOMERS relation **Mr. Sample**

5/5/14 15:10 - 15:51 changed by Susan ServiceDesk
- Relation added: Reseller SELLS TO END CUSTOMERS relation **Mr. Sample**
- Relation removed: Reseller SELLS TO END CUSTOMERS relation **Mr. Sample**

3/7/14 15:41 changed by Harald Huber
- Relation added: Reseller SELLS TO END CUSTOMERS relation **Mr. Sample**

3/5/14 15:51 changed by Harald Huber
- salesRep set to **Mrs. Schwartz**
- Assistance/Office set to **Mrs. Miller**
- Contract start set to **3/2/14**
- Service level set to **Gold**

2/28/14 11:50 - 11:51 changed by Harald Huber
- address changed from ilkyway 77 to **Milkyway 77**
- phone_frontdesk set to **123**
- country set to **Unknown**
- zip set to **7777**
- address set to **ilkyway 77**
- company_name set to **MyNewSpaceCompany**
- city set to **Alderaan**
- company_number set to **999**

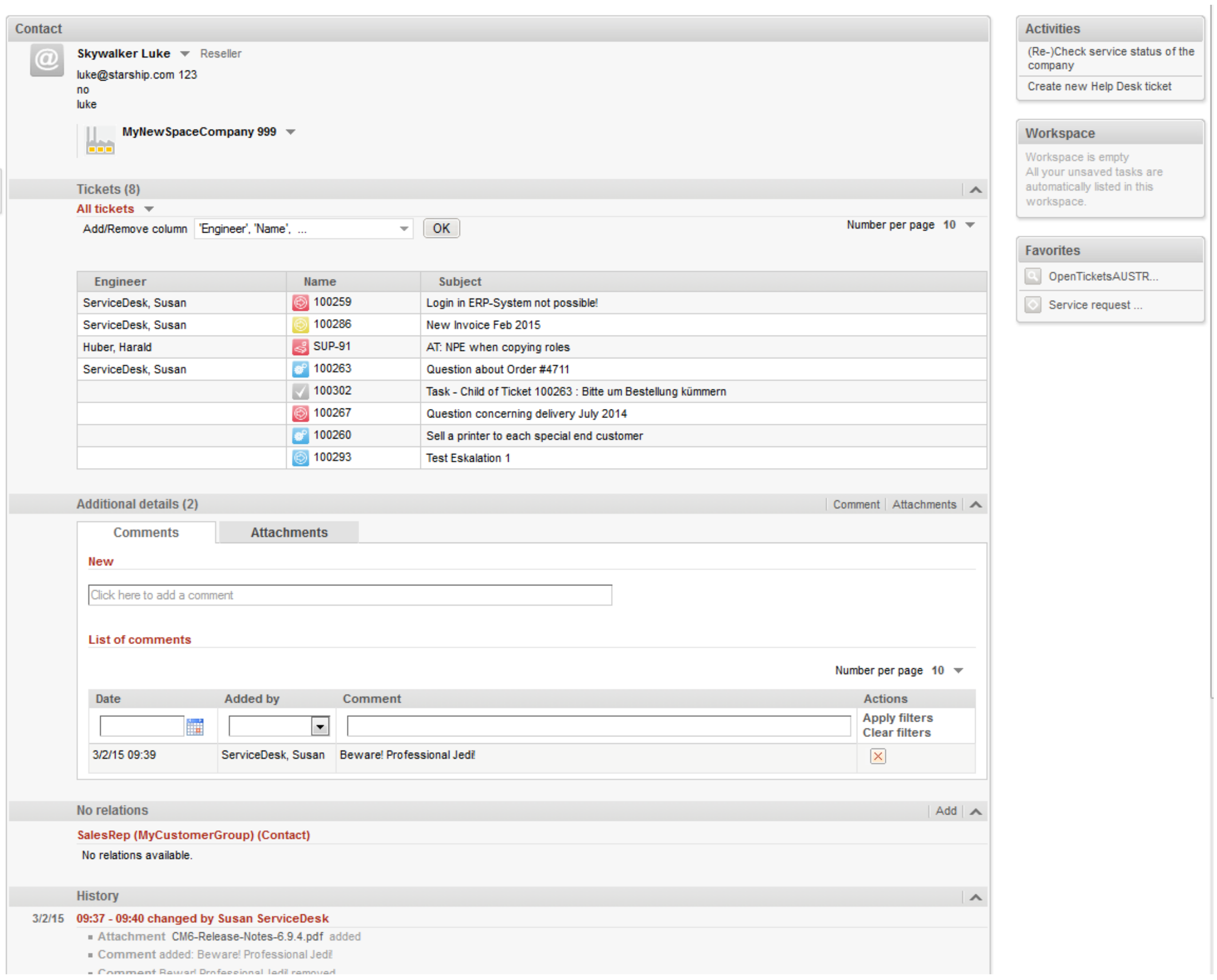
Fig. 4: ConSol*CM Web Client - Company Page, bottom section

The *Company* page contains the following sections:

- **Company data**
  company data like address, phone number, service data (i.e. the data object group fields you have defined in the the Customer Data Model). All company data might be placed in the ticket section or there might be one or more tabs in the *Groups* section.
- **Tickets**
  All tickets for the company are listed in this section. Starting with version 6.9, it is possible to assign a ticket to a company directly. In the customer data model the option *Company as customer* has to be set to activate this functionality.
- **Contacts**
  This section shows a list of all contacts belonging to this company. Click on a contact name to open the contact page.
- **Additional details**
  There are two tabs:
  - **Comments**
    Here, all comments concerning this company are listed.
  - **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**
  Here, all relations to and from this company are listed.
- **History**
  In this section, all actions which have been performed with this company object are listed, e.g. the change of a name or any other value of one of the data object group fields.

Please refer to the *ConSol*CM User Manual* for a detailed introduction of how to work with companies.

Contact Page



Contact

Skywalker Luke Reseller
luke@starship.com 123
no
luke

MyNewSpaceCompany 999

Tickets (8)
All tickets
Add/Remove column 'Engineer', 'Name', ... OK
Number per page 10

Engineer	Name	Subject
ServiceDesk, Susan	100259	Login in ERP-System not possible!
ServiceDesk, Susan	100286	New Invoice Feb 2015
Huber, Harald	SUP-91	AT: NPE when copying roles
ServiceDesk, Susan	100263	Question about Order #4711
	100302	Task - Child of Ticket 100263 : Bitte um Bestellung kümmern
	100267	Question concerning delivery July 2014
	100260	Sell a printer to each special end customer
	100293	Test Eskalation 1

Additional details (2) Comment | Attachments

Comments | Attachments

New
Click here to add a comment

List of comments
Number per page 10

Date	Added by	Comment	Actions
			Apply filters Clear filters
3/2/15 09:39	ServiceDesk, Susan	Beware! Professional Jedi!	

No relations Add
SalesRep (MyCustomerGroup) (Contact)
No relations available.

History
3/2/15 09:37 - 09:40 changed by Susan ServiceDesk
- Attachment CM6-Release-Notes-6.9.4.pdf added
- Comment added: Beware! Professional Jedi!

Activities
(Re-)Check service status of the company
Create new Help Desk ticket

Workspace
Workspace is empty
All your unsaved tasks are automatically listed in this workspace.

Favorites
OpenTicketsAUSTR...
Service request ...

Fig. 5: ConSol*CM Web Client - Contact Page

These are the sections on the *Contact* page:

- **Contact data**
  - contact data like address, phone number, service data (i.e. the data object group fields you have defined in the the Customer Data Model). All contact data might be placed in the ticket section or there might be one or more tabs in the *Groups* section.
- **Tickets**
  All tickets for the contact are listed in this section.
- **Additional details**
  There are two tabs:
  - **Comments**
    Here, all comments concerning this contact are listed.
  - **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**
  Here, all relations to and from this contact are listed, e.g. if the contact is the CEO of a company.
- **History**
  In this section, all actions which have been performed with this contact object are listed, e.g. the change of a name or any other value of one of the data object group 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.

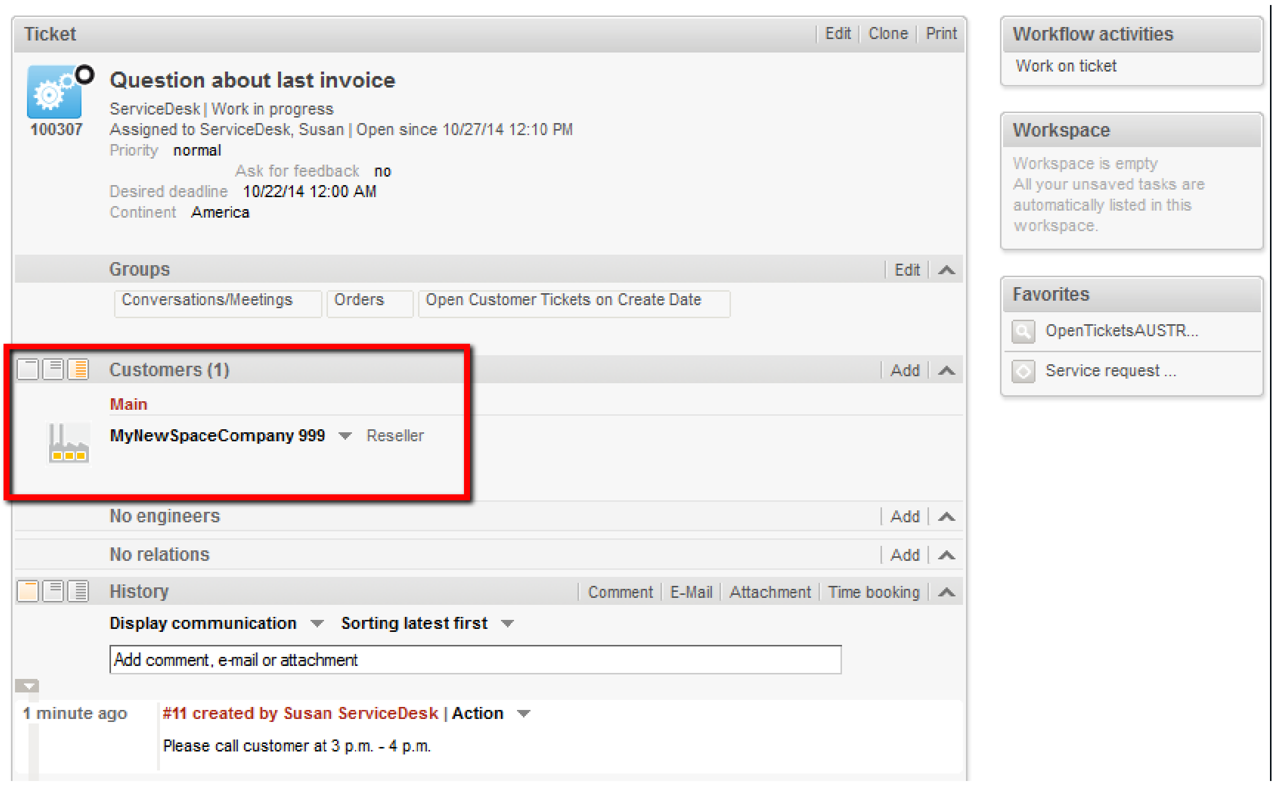
Attention:

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 *Additional details* section of tickets:

- Details read
- Details write
- Details delete

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.


Ticket | Edit | Clone | Print

Question about last invoice
100307
ServiceDesk | Work in progress
Assigned to ServiceDesk, Susan | Open since 10/27/14 12:10 PM
Priority normal
Ask for feedback no
Desired deadline 10/22/14 12:00 AM
Continent America

Groups | Edit
Conversations/Meetings | Orders | Open Customer Tickets on Create Date

Customers (1) | Add
Main
MyNewSpaceCompany 999 Reseller

No engineers | Add
No relations | Add
History | Comment | E-Mail | Attachment | Time booking
Display communication | Sorting latest first
Add comment, e-mail or attachment
1 minute ago #11 created by Susan ServiceDesk | Action
Please call customer at 3 p.m. - 4 p.m.

Workflow activities
Work on ticket

Workspace
Workspace is empty
All your unsaved tasks are automatically listed in this workspace.

Favorites
OpenTicketsAUSTR...
Service request ...

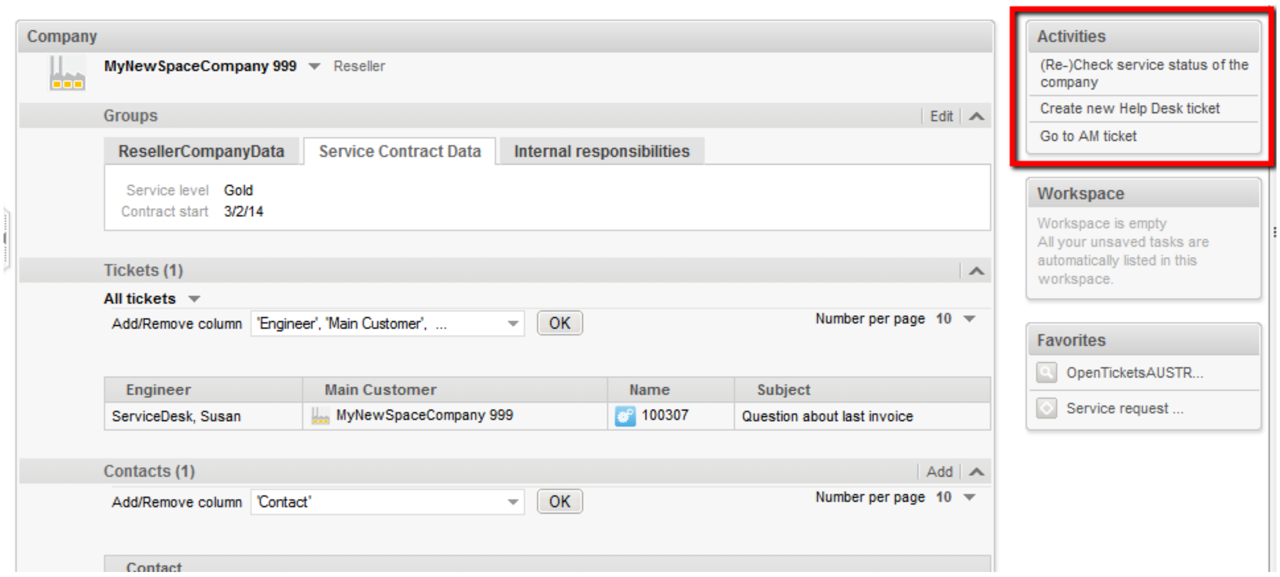
Fig. 6: ConSol*CM Web Client - Using the Company as Main Customer for a Ticket

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 following figure), or can update the contact data from another database, or create a ticket for the contact (see figure after next).

Automatic actions can be performed when a system action takes place (create/update/delete of a customer). The company and contact actions are based on the *Action Framework*.


Company
MyNewSpaceCompany 999 Reseller
Groups Edit
ResellerCompanyData | Service Contract Data | Internal responsibilities
Service level Gold
Contract start 3/2/14
Tickets (1)
All tickets
Add/Remove column 'Engineer', 'Main Customer', ... OK
Number per page 10

Engineer	Main Customer	Name	Subject
ServiceDesk, Susan	MyNewSpaceCompany 999	100307	Question about last invoice

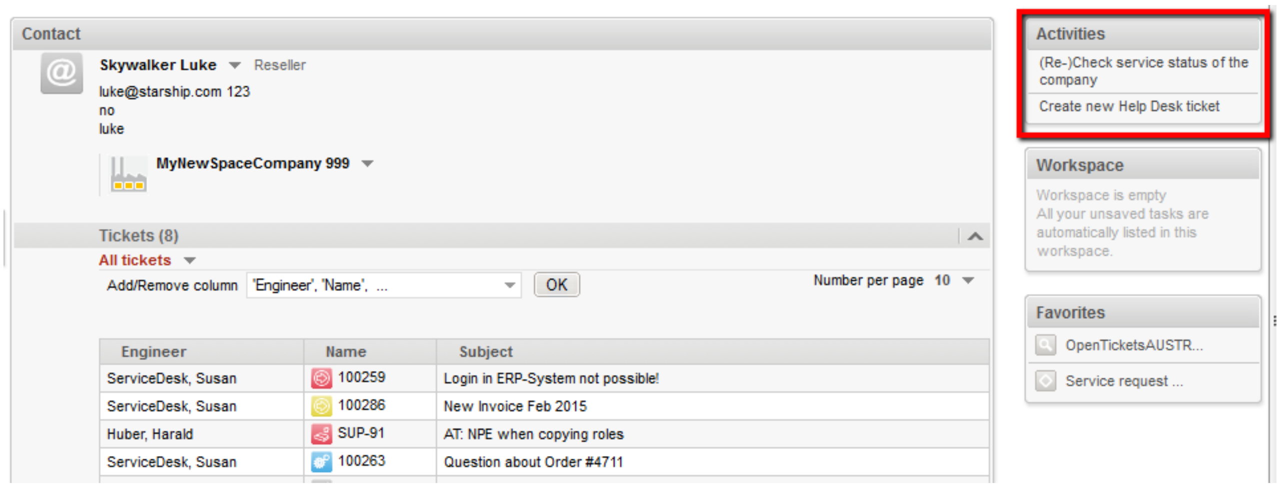
Contacts (1) Add
Add/Remove column 'Contact' OK
Number per page 10
Contact

Activities
(Re-)Check service status of the company
Create new Help Desk ticket
Go to AM ticket

Workspace
Workspace is empty
All your unsaved tasks are automatically listed in this workspace.

Favorites
OpenTicketsAUSTR...
Service request ...

Fig. 7: ConSol*CM Web Client - Manual Company Actions


Contact
Skywalker Luke Reseller
luke@starship.com 123
no
luke
MyNewSpaceCompany 999
Tickets (8)
All tickets
Add/Remove column 'Engineer', 'Name', ... OK
Number per page 10

Engineer	Name	Subject
ServiceDesk, Susan	100259	Login in ERP-System not possible!
ServiceDesk, Susan	100286	New Invoice Feb 2015
Huber, Harald	SUP-91	AT: NPE when copying roles
ServiceDesk, Susan	100263	Question about Order #4711

Activities
(Re-)Check service status of the company
Create new Help Desk ticket

Workspace
Workspace is empty
All your unsaved tasks are automatically listed in this workspace.

Favorites
OpenTicketsAUSTR...
Service request ...

Fig. 8: ConSol*CM Web Client - Manual Contact Action

Attention:

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

Example 6: Setting Relations between Contacts and Companies

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 which improves *Customer Relationship Management* (CRM) functionalities.

In the Web Client, relations between companies and/or contacts are established and displayed similar to ticket relations. In the example, *MyNewSpaceCompany* sells products to the end customer *Mr. Sample*.

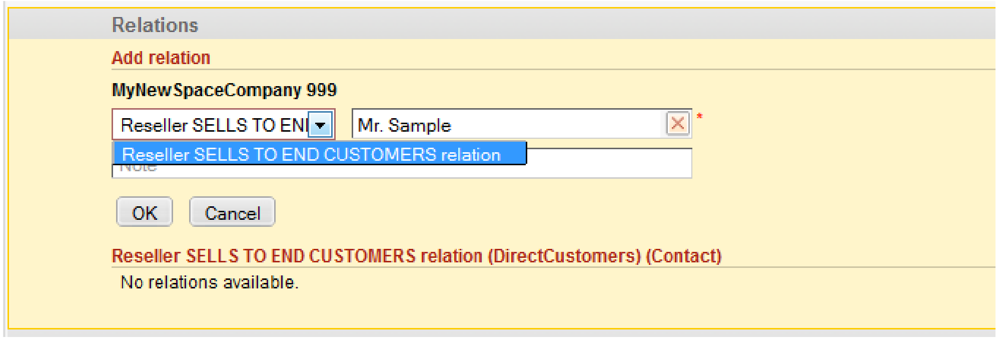
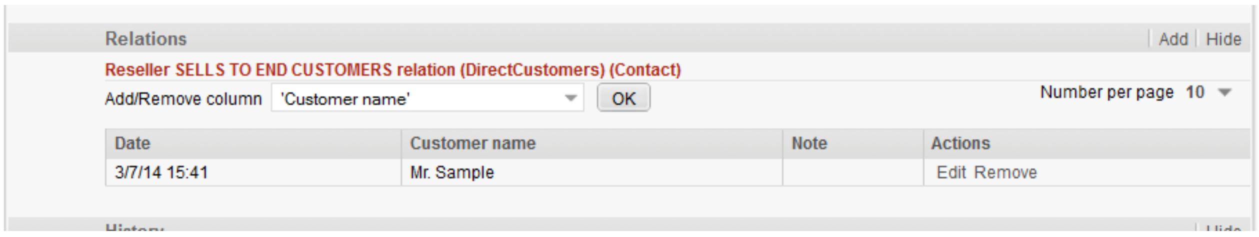

Relations
Add relation
MyNewSpaceCompany 999
Reseller SELLS TO ENI
Mr. Sample
Reseller SELLS TO END CUSTOMERS relation
Note
OK
Cancel
Reseller SELLS TO END CUSTOMERS relation (DirectCustomers) (Contact)
No relations available.

Fig. 9: ConSol*CM Web Client - Establishing a Company-Contact Relation


Relations
Add Hide
Reseller SELLS TO END CUSTOMERS relation (DirectCustomers) (Contact)
Add/Remove column 'Customer name' OK
Number per page 10

Date	Customer name	Note	Actions
3/7/14 15:41	Mr. Sample		Edit Remove

Fig. 10: ConSol*CM Web Client - Display of Company-Contact Relation

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. In case the customer has to be deleted, all of its 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 to deactivate (and reactivate) companies and/or customers (contacts), i.e. only then the *Deactivate/Activate* menu items will be displayed in the Web Client:

- Deactivate/activate

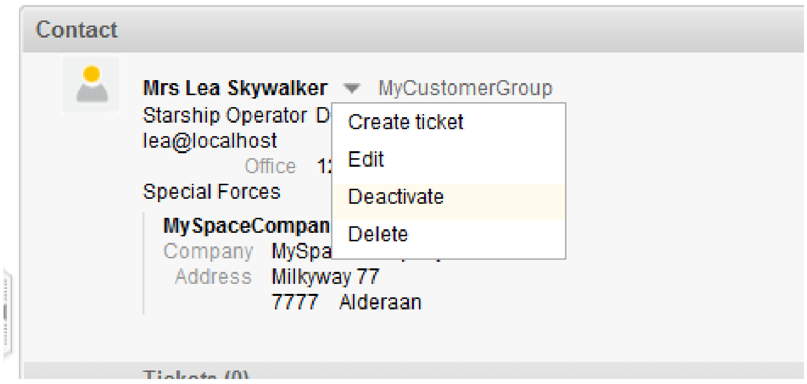


Fig. 11: ConSol*CM Web Client - Deactivating a Contact

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 ticket to another customer.

The following actions **cannot** be performed for a deactivated customer:

- Create a new ticket.
- Assign a ticket to this customer.
- 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).

Attention:

Deactivation in a two-level model
When a company (or more generally spoken: an object on company level) is deactivated, all assigned contacts are deactivated automatically.

There are two use cases:

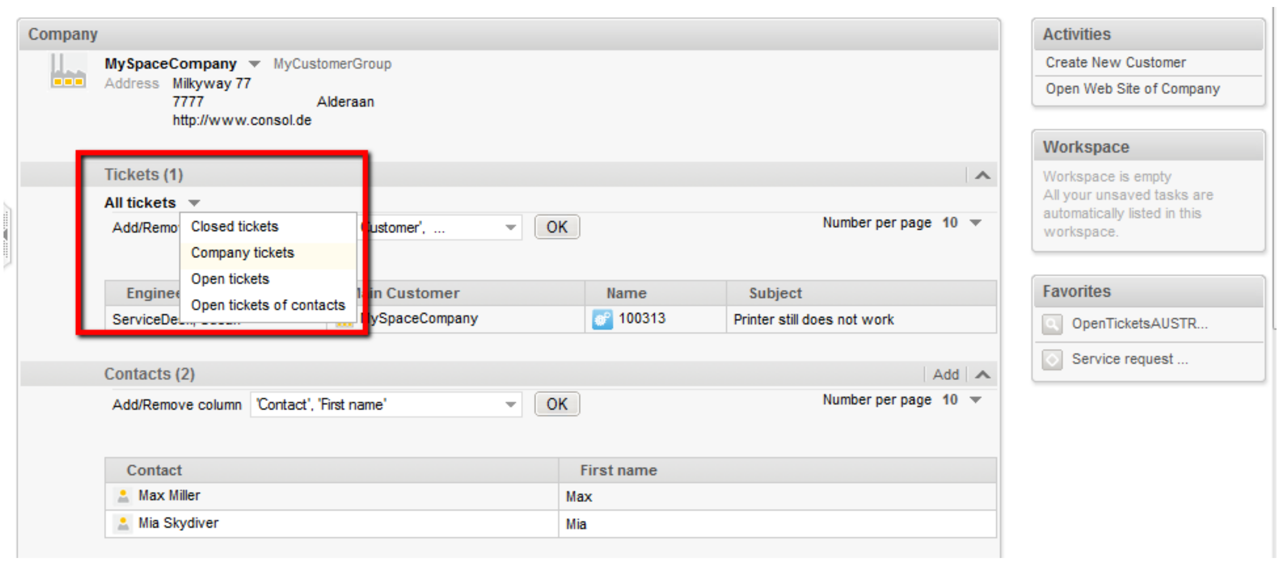
1. **All** contacts of the company **can** be deactivated (no open ticket assigned).
   In this case the company **and** all assigned contacts will be deactivated. Afterwards the company page will be reloaded, company and contact data are marked as deactivated.
2. The company has still contacts which **cannot** be deactivated because of open tickets.
   Here, the deactivation of a company is **not** allowed. The *deactivated* option is not selectable.

Reactivation in a two-level model
In case a company is reactivated, the assigned contacts will **not** be reactivated **automatically**. They have to be reactivated **manually**.

Example 8: 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 Filter on Company Page



MySpaceCompany MyCustomerGroup
Address Milkyway 77
7777 Alderaan
http://www.consol.de

Engine	in Customer	Name	Subject
ServiceDe...	MySpaceCompany	100313	Printer still does not work

'Contact', 'First name'

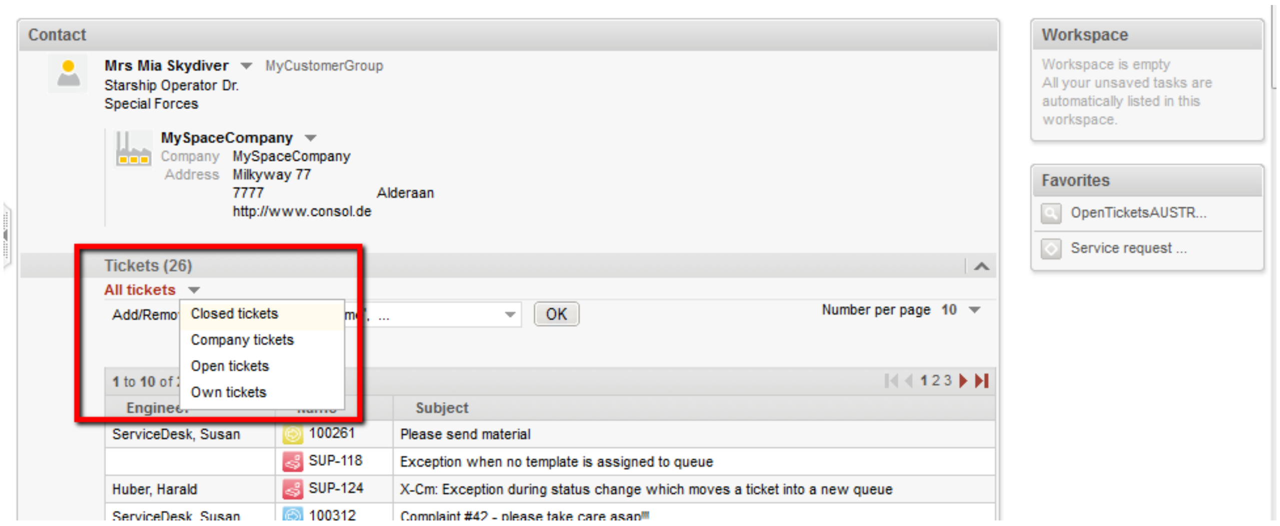
Contact	First name
Max Miller	Max
Mia Skydiver	Mia

Fig. 12: ConSol*CM Web Client - Ticket Filter on Company Page

Available options:

- **Closed tickets**
  Closed tickets where the company is the main customer or additional customer.
- **Company tickets**
  Tickets where the company is the main customer.
- **Open tickets**
  Open tickets where the company is the main customer or additional customer.
- **Open tickets of contacts**
  Open tickets of contacts of this company.

Ticket Filter on Contact Page



Engineer	Name	Subject
ServiceDesk, Susan	100261	Please send material
	SUP-118	Exception when no template is assigned to queue
Huber, Harald	SUP-124	X-Cm: Exception during status change which moves a ticket into a new queue
ServiceDesk, Susan	100312	Complaint #42 - please take care asap!!!

Fig. 13: ConSol*CM Web Client - Ticket Filter on Contact Page

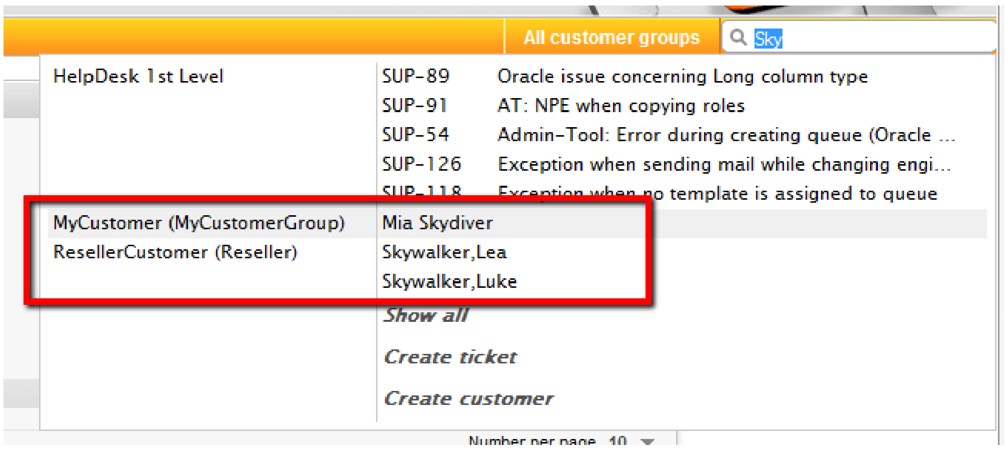
Available options:

- **Closed tickets**
  Closed tickets where the contact is the main customer or additional customer.
- **Company tickets**
  Tickets where the company of the customer is the main customer or an additional customer.
- **Open tickets**
  Open tickets where the contact is the main customer or additional customer.
- **Own tickets**
  Tickets where the contact is the main customer.

Example 9: 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 data object group> (<localized name of customer group>)
```



HelpDesk 1st Level	SUP-89	Oracle issue concerning Long column type
	SUP-91	AT: NPE when copying roles
	SUP-54	Admin-Tool: Error during creating queue (Oracle ...
	SUP-126	Exception when sending mail while changing engi...
	SUP-118	Exception when no template is assigned to queue
MyCustomer (MyCustomerGroup)	Mia Skydiver	
ResellerCustomer (Reseller)	Skywalker,Lea	
	Skywalker,Luke	

Fig. 14: ConSol*CM Web Client - Search Results for Quick Search

9.4 Setting Up the Customer Data Model

- Introduction to Setting Up the Customer Data Model Based on FlexCDM
- Managing Contacts and Companies Using the Admin Tool
    - Creating a New Two-Level Customer Data Model
        - Step 1: Create the Customer Data Model with the First Data Object
        - Step 2: Create Another Data Object
        - Step 3: Configuring the Parameters for the Defined Objects
            - Parameters for the Customer Data Model
            - Parameters for the Data Object
            - Parameters for the Data Object Group
    - Creating a New Customer Group Using the New Customer Data Model
    - Assigning Access Rights for Customer Groups with the New Model to Roles
    - Assign the New Customer Groups to Queues

9.4.1 Introduction to Setting Up the Customer Data Model Based on FlexCDM

With *FlexCDM* various customer data models can be implemented. Please refer to section Introduction to FlexCDM for a detailed introduction. To work with a new customer data model within a certain customer group, the following steps have to be performed:

1. Create a customer data model.
   (This implies you have already decided if 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.

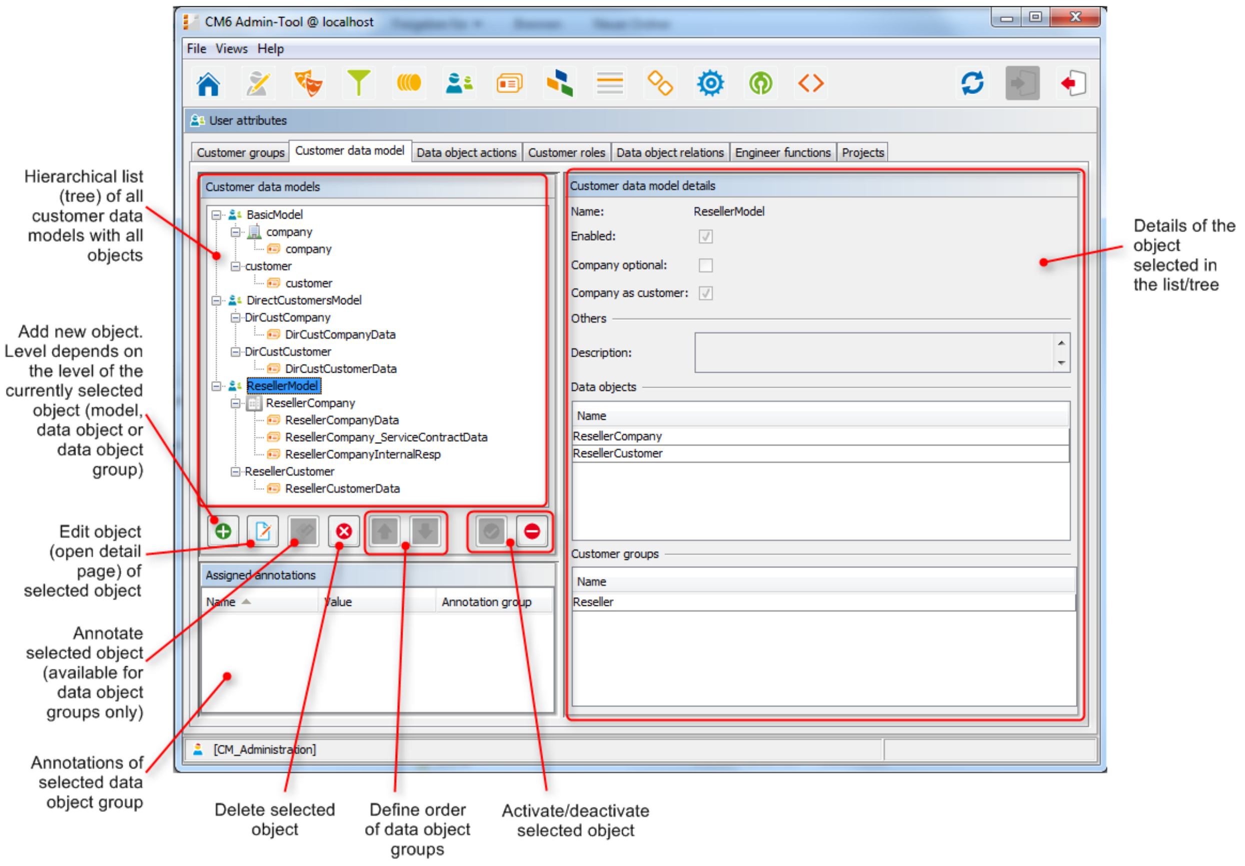
A customer data model comprises objects on three model levels:

1. **The customer data model definition**
2. **The data objects within this model**
   A data 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 data object group fields**
   These are the data fields for the data objects, i.e. either the data object group fields for company data (e.g. ZIP, address, phone) or the data object group fields for contact data (e.g. name, forename, e-mail address).

9.4.2 Managing Contacts and Companies Using the Admin Tool

In the Admin Tool, most of the FlexCDM configuration is done within the *User attributes* section. For the definition of customer data models use the tab *Customer data model*.


CM6 Admin-Tool @ localhost
File Views Help
User attributes
Customer groups
Customer data model
Data object actions
Customer roles
Data object relations
Engineer functions
Projects
Customer data models
Customer data model details
Assigned annotations
Name
Value
Annotation group
[CM_Administration]
Hierarchical list (tree) of all customer data models with all objects
Add new object. Level depends on the level of the currently selected object (model, data object or data object group)
Edit object (open detail page) of selected object
Annotate selected object (available for data object groups only)
Annotations of selected data object group
Delete selected object
Define order of data object groups
Activate/deactivate selected object
Details of the object selected in the list/tree

```
BasicModel
  company
    company
  customer
    customer
DirectCustomersModel
  DirCustCompany
    DirCustCompanyData
  DirCustCustomer
    DirCustCustomerData
ResellerModel
  ResellerCompany
    ResellerCompanyData
    ResellerCompany_ServiceContractData
    ResellerCompanyInternalResp
  ResellerCustomer
    ResellerCustomerData
```

Name: ResellerModel
Enabled: ✓
Company optional: ☐
Company as customer: ✓
Others
Description:
Data objects

Name
ResellerCompany
ResellerCustomer

Customer groups

Name
Reseller

Fig. 1: ConSol*CM Admin Tool - Customer Data Model Definiton

To explain how to work with the customer data model, we will walk you through an example in the next sections.

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 reseller 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 data object (1st level)
- the data object group fields for the company
- the contact data object (2nd level)

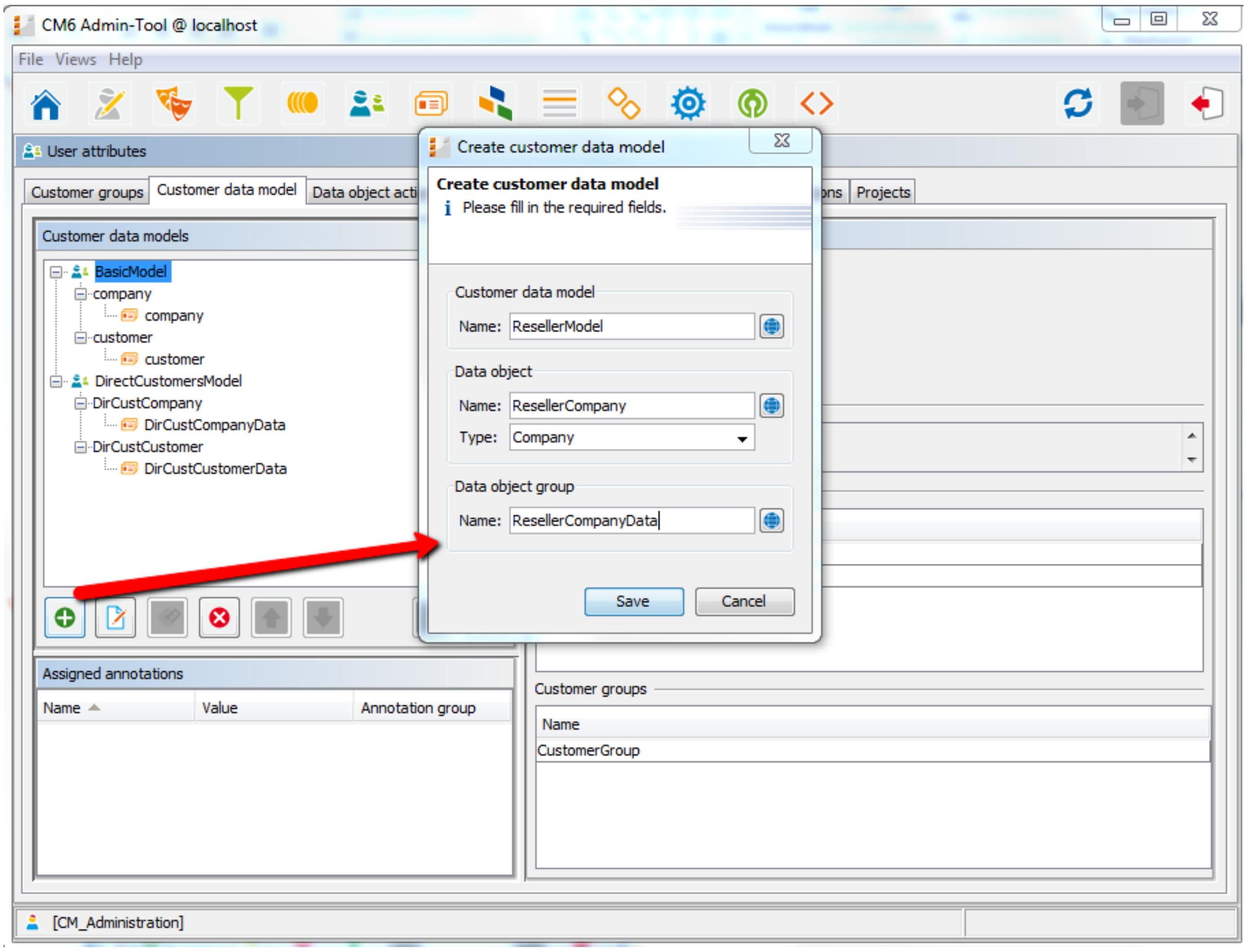
- the data object group fields for the contact

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 Data Object

When you create a new customer data model, you have to add a data object and the respective data object group 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 button to open the pop-up window.



Create customer data model

Create customer data model

Please fill in the required fields.

Customer data model

Name: ResellerModel

Data object

Name: ResellerCompany

Type: Company

Data object group

Name: ResellerCompanyData

Save Cancel

Fig. 2: ConSol*CM Admin Tool - Creating a New Customer Data Model

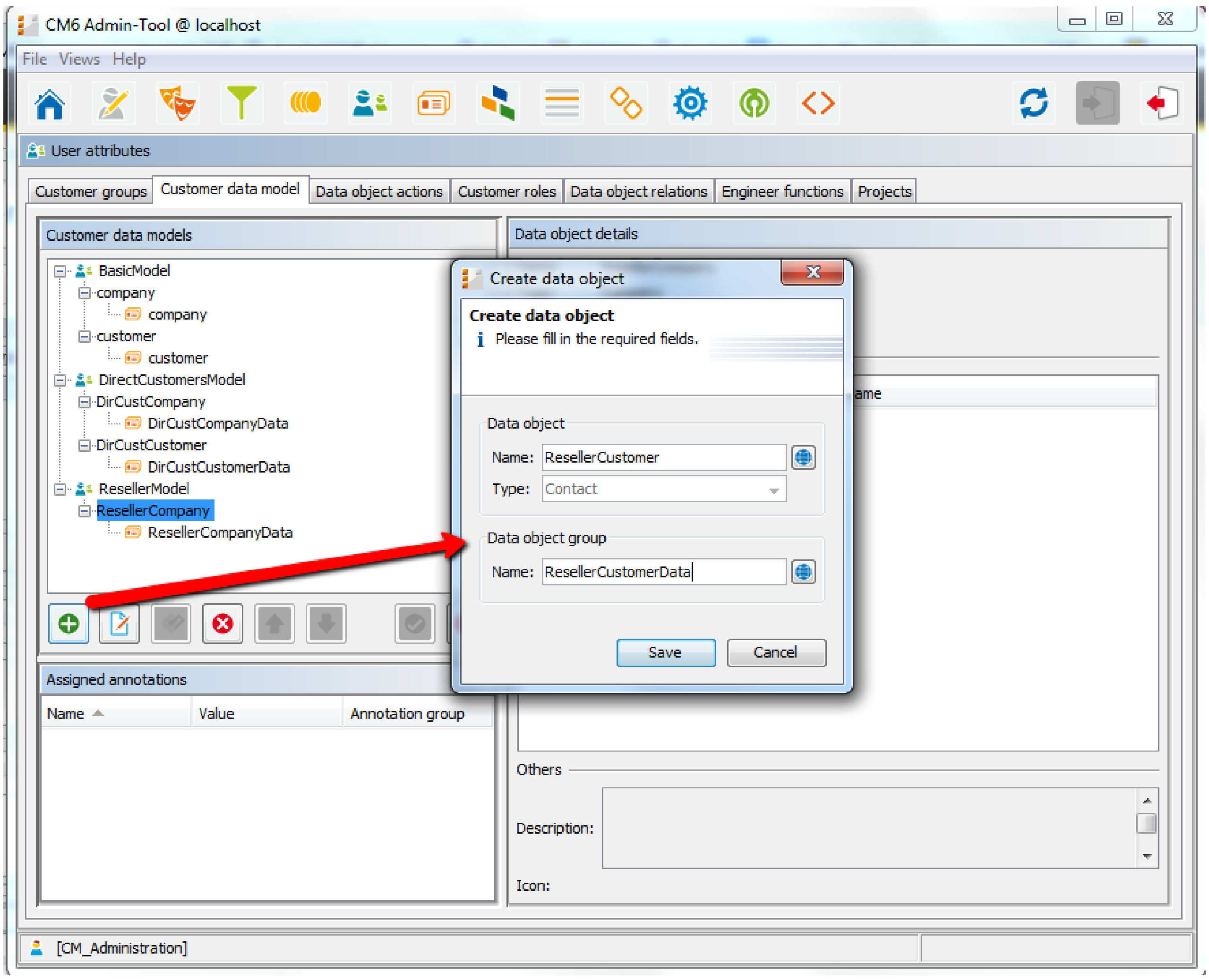
You have to fill-in the following fields:

- **Customer data model**
    - **Name**
      The name of the new customer data model. As usual in ConSol*CM6, there is one (unique) technical object name. By using the button you can open a pop-up window where you can enter the values for the name in various languages.

- **Data object**
    - **Name**
      The unique technical name of the company/contact object, can also be localized using the button.
    - **Type**
      Select *Contact* or *Customer*. There can be only one company object and one contact object within one customer data model.
- **Data object group**
    - **Name**
      The unique technical name of the first data object group for company data within the defined data object. More data object groups can be added later on.

Step 2: Create Another Data Object

In the next step, you have to add the contact object. Select the object *ResellerCompany* (to set the correct level for the following *Add* operation) and use the button to open the pop-up window.



Create data object

Please fill in the required fields.

Data object
Name: ResellerCustomer
Type: Contact

Data object group
Name: ResellerCustomerData

Fig. 3: ConSol*CM Admin Tool - Adding a New Data Object

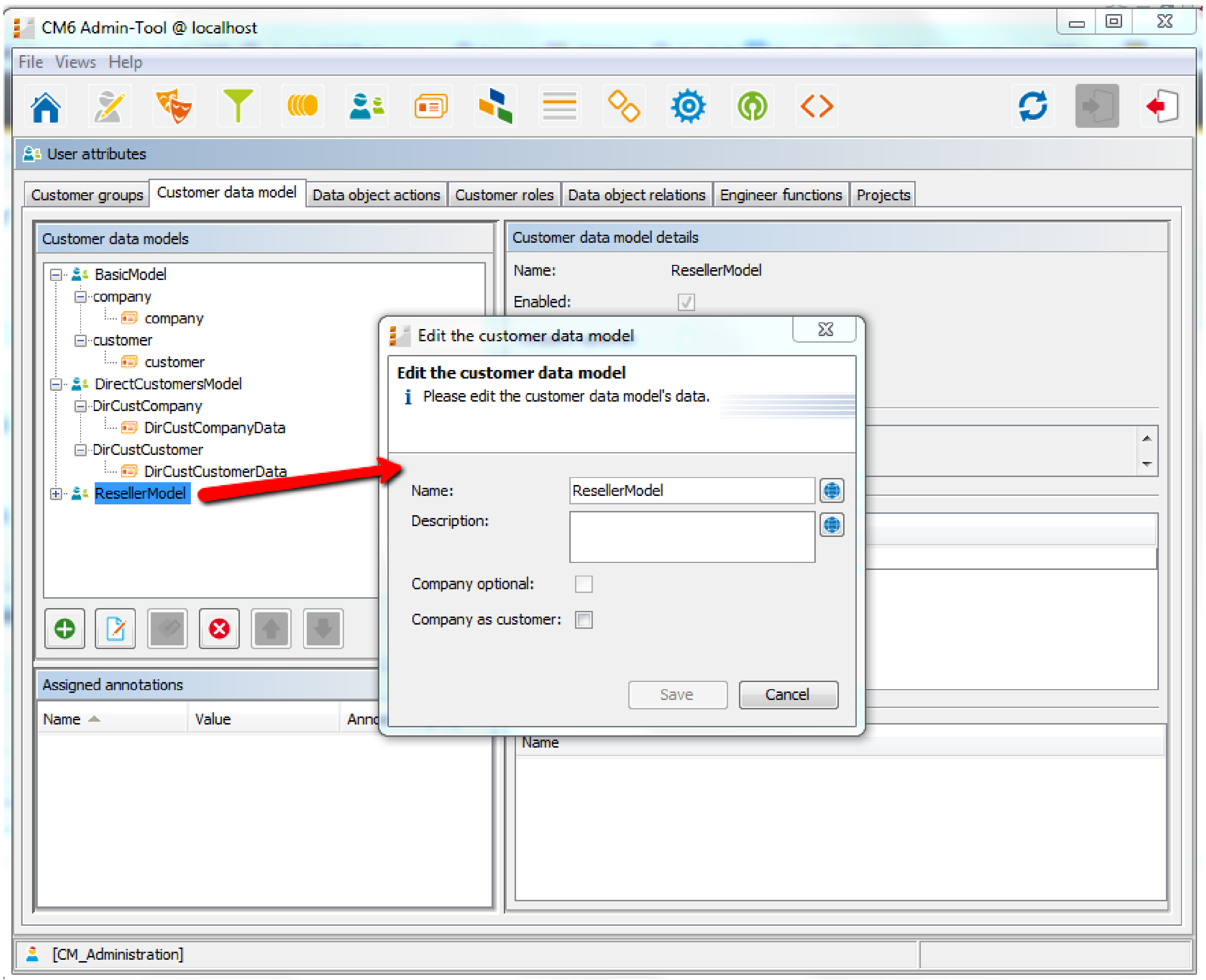
You have to fill-in the following fields:

- **Data object**
  - **Name**
    The unique technical name of the contact object, can also be localized using the button.
  - **Type**
    Here, *Contact* is pre-selected and cannot be modified, because a company object is already present in the customer data model.
- **Data object group**
  - **Name**
    The unique technical name of the first data object group for contact data within the defined data object. More data object groups can be added later on.

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 (*ResellerModel* in our example) or mark the customer data model in the list and click on to open the pop-up window where you can define the parameters for the model.



Name: ResellerModel
Enabled: ✓

Edit the customer data model

Edit the customer data model
Please edit the customer data model's data.

Name: ResellerModel
Description:
Company optional: ☐
Company as customer: ☐
Save Cancel

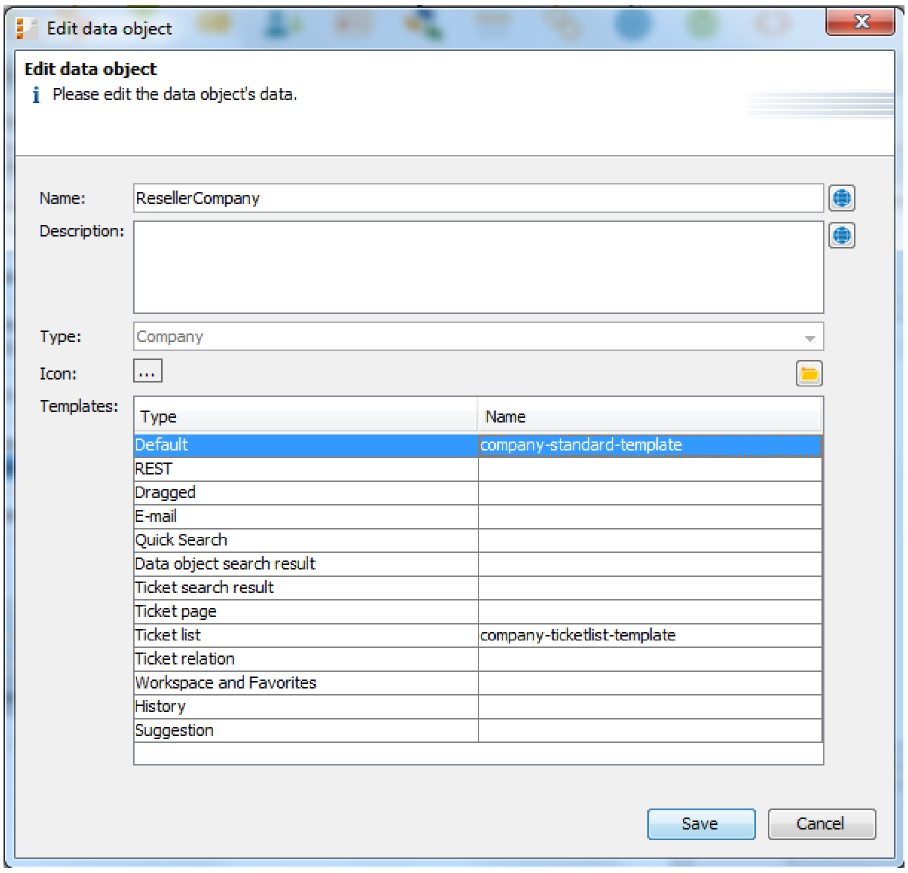
Fig. 4: ConSol*CM Admin Tool - Parameters for a Customer Data Model

You can fill-in the following fields:

- **Name**
  Check or modify the existing name of the model.
- **Description**
  Optional description.
- **Company optional**
  If this check box is marked 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 the system to work with single contacts, even within a two-level customer data model.
- **Company as customer**
  Mark this check box if it should be allowed to create tickets not only for contacts but also for companies within the model.

Parameters for the Data Object

Double-click on the name of a data object, e.g. the *ResellerCompany*, to open the pop-up menu where you can configure the parameters for this object.



Name: ResellerCompany
Description:
Type: Company
Icon: ...
Templates:

Type	Name
Default	company-standard-template
REST	
Dragged	
E-mail	
Quick Search	
Data object search result	
Ticket search result	
Ticket page	
Ticket list	company-ticketlist-template
Ticket relation	
Workspace and Favorites	
History	
Suggestion	

Fig. 5: ConSol*CM Admin Tool - Parameters for a Data Object

You can fill-in the following fields:

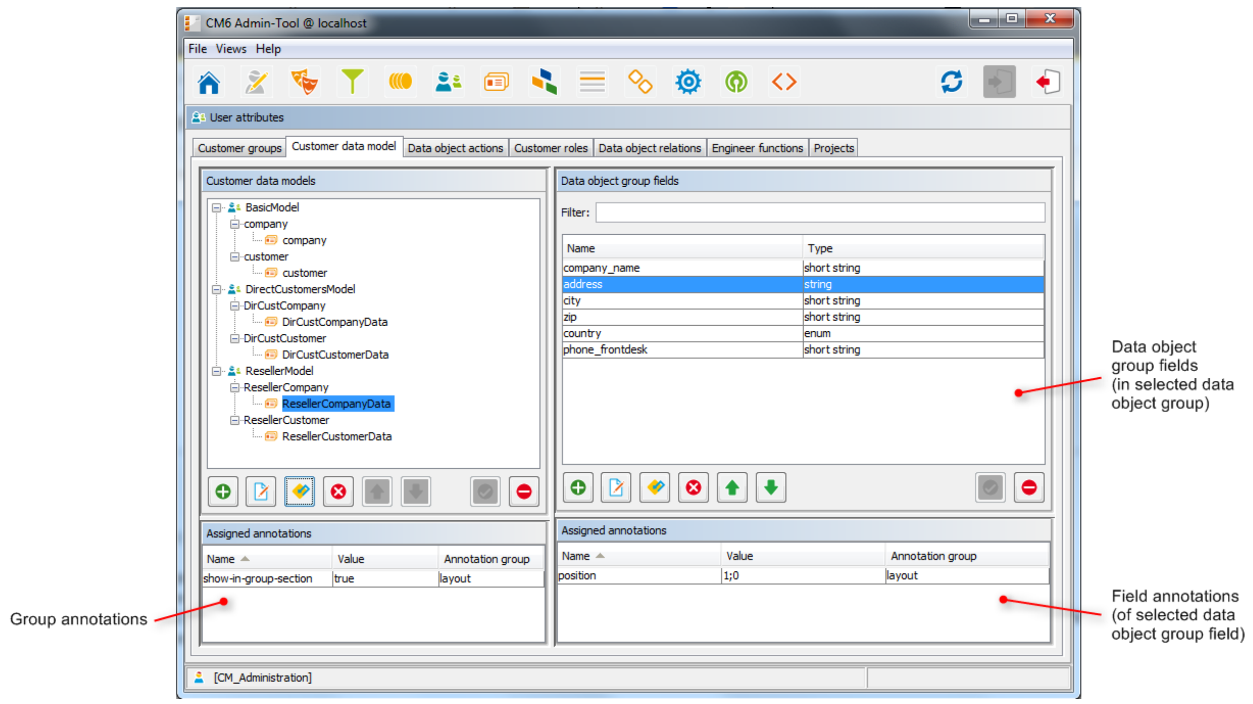
- **Name**
  The unique technical name of the data object with technical name and localized name(s).

- **Description**
  The description of the data object. Will be used in future ConSol*CM versions.
- **Type**
  A read-only field which displays the type (*contact* or *company*) of the data object.
- **Icon**
  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 from the file system using the file explorer button .
- **Templates**
  The templates which are used to render the data of the data 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 Data Object Group

Double-click on the name of the data object group to change the technical and/or localized name(s) of the group.

To define the data object group fields, use the GUI elements on the right-hand side. In the following figure, an example for the definition of data object group fields for the *ResellerCompany* is shown. Here, only one data object group (*ResellerCompanyData*) is used. You can use as many data object groups in one data object as you consider suitable for your system.



- BasicModel
  - company
    - company
  - customer
    - customer
- DirectCustomersModel
  - DirCustCompany
    - DirCustCompanyData
  - DirCustCustomer
    - DirCustCustomerData
- ResellerModel
  - ResellerCompany
    - ResellerCompanyData
  - ResellerCustomer
    - ResellerCustomerData

Name	Type
company_name	short string
address	string
city	short string
zip	short string
country	enum
phone_frontdesk	short string

Name	Value	Annotation group
show-in-group-section	true	layout

Name	Value	Annotation group
position	1;0	layout

Fig. 6: ConSol*CM Admin Tool - Parameters for the Data Object Group

The definition of data object group fields within customer data models is based on the same principles as the definition of custom fields for ticket data. For a detailed introduction to the definition and management of custom fields, please refer to sections Custom Field Administration and Data Object Group Field Management and GUI Design.

The available data object group field annotations are listed in section Appendix A (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*).

⚠ **Attention:**

Please make sure that the annotation *field indexed* is set for all fields which should be searchable. This concerns the quick search, the detail search, and all auto-complete operations! See also section Tab Index (Search and Indexer 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 ahead to assign the model to one or more customer group(s).

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 *Resellers* which will use the new *ResellerModel*.

Use the tab *Customer groups* in the *User attributes* section of the Admin Tool to create a new customer group and to assign the desired customer data model.

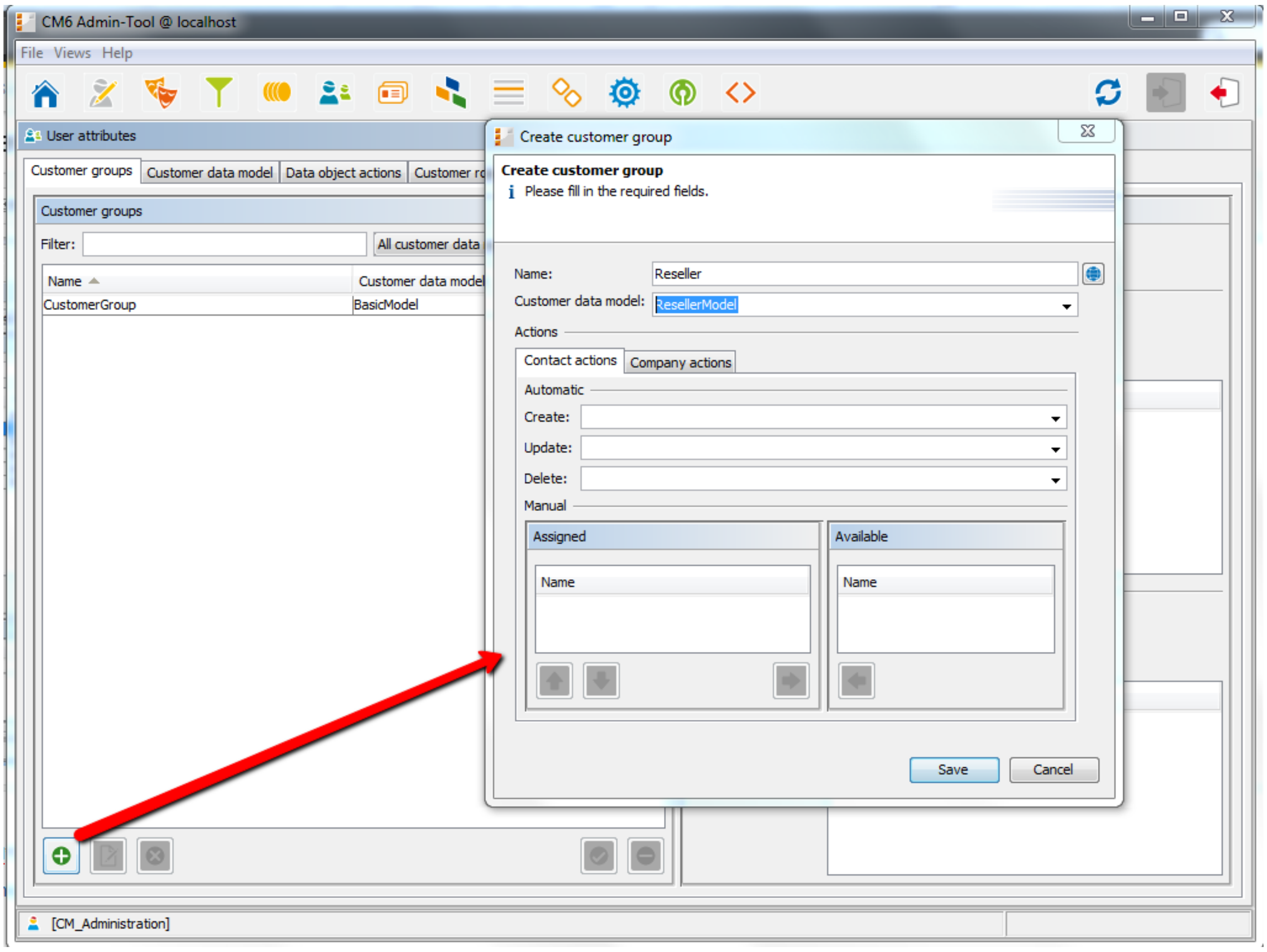


Fig. 7: ConSol*CM Admin Tool - Definition of a New Customer Group

You can fill-in the following fields:

- **Name**
  The unique technical name (and localized name) of the new customer group.
- **Customer data model**
  Select the desired data model from the drop-down menu.
- **Actions**
  Here, the customer actions can be defined. This will be explained in detail in section Action Framework.

For an engineer who has access permissions for two customer groups, the Web Client will look as shown in the following figure.

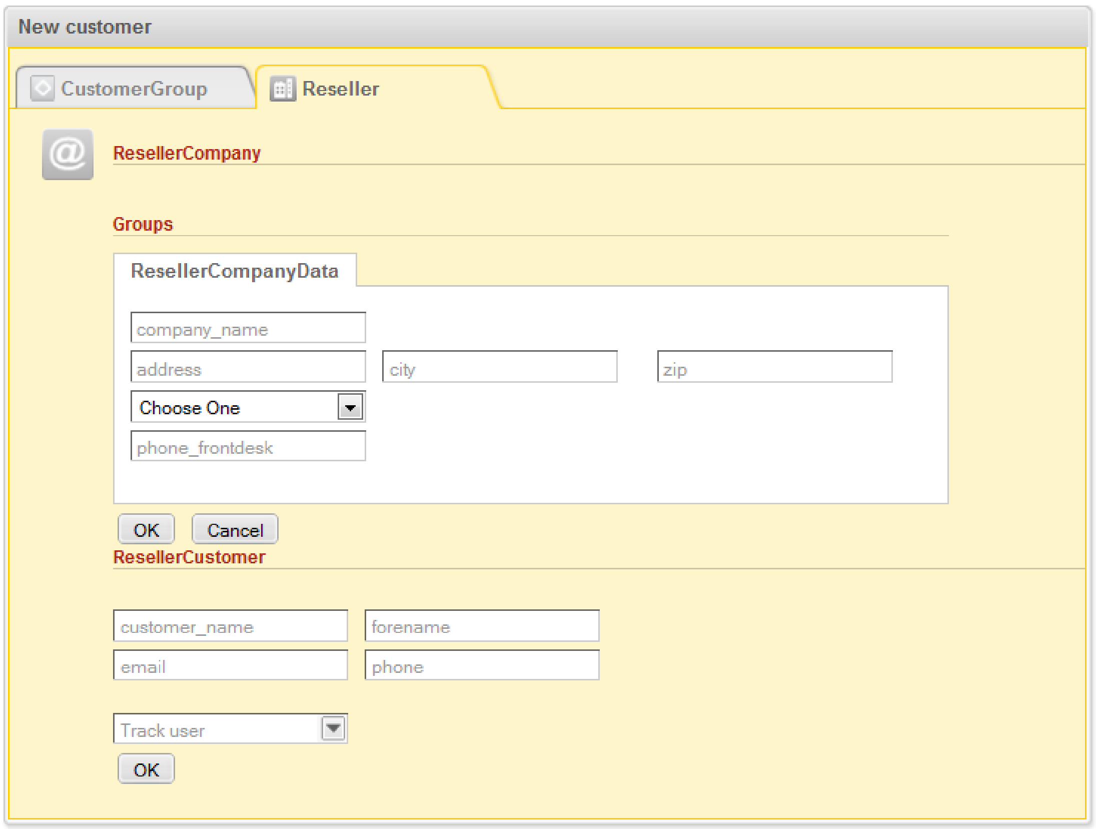


Fig. 8: ConSol*CM Web Client - Creating a New Company and Customer

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 reseller data sets or to modify them, you have to grant access permissions for the user groups to one or more roles.

See section Access Rights for Customer Groups.

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 Queue Administration for details.

9.5 Data Object Group Field Management and GUI Design for Customer Data

- Introduction
- Defining Data Object Group Fields for Customer Data Using the Admin Tool
  - Admin Tool GUI
  - Data Object Groups
  - Data Object Group Field Definition
    - Labels for Data Object Group Fields
      - Label mode after an update from CM version prior to 6.9.4 to a version 6.9.4 and up
      - Default mode for new data object groups and data object group fields
- Scripting Using Objects from the FlexCDM
  - New Scripting Features Since ConSol*CM Version 6.9
  - Extension of the Custom Field Expression Language (CFEL)
- Changes in Scripting from Consol*CM Version 6.8 to Version 6.9
  - AbstractField
  - ActivityFormFieldsSet
  - ContentFile
  - ContentResource
  - FieldLogEntry
  - Ticket
  - TimerTrigger
  - Unit
- New (Convenience) Methods
- New Objects in ConSol*CM 6.9 and Up

9.5.1 Introduction

One feature of ConSol*CM version 6.9 is the great flexibility as far as customer data model (*FlexCDM*) and GUI design are concerned. You as an administrator can define any data field (data object group field) which is required and place it on the user interface wherever it is suitable. The basic principle is now the same as the one you know for ticket custom fields: full flexibility.

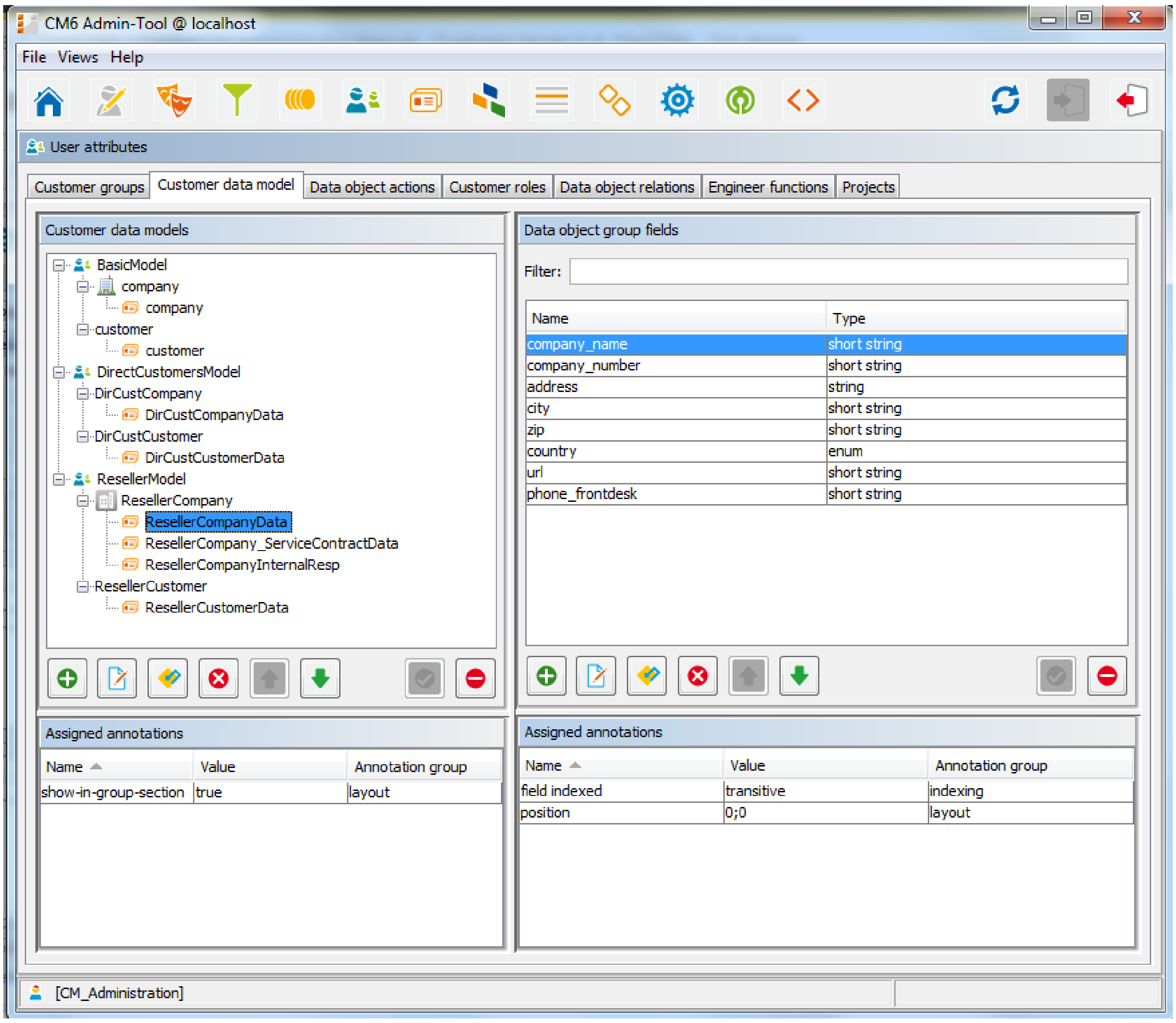
The management of the ticket data model and GUI design is explained in section Custom Field Administration. The management of objects within the customer model is explained in section Setting Up the Customer Data Model. Please refer to those sections for a detailed explanation. In this chapter, we assume that you have a good knowledge of those topics.

9.5.2 Defining Data Object Group Fields for Customer Data Using the Admin Tool

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.

Data fields for data objects within the customer data model are called *data object group fields*. All data object group fields are defined in the *User attributes* section of the Admin Tool, tab *Customer data model*. The work with data object group fields is based on the same principle as the work with ticket data fields (custom fields): data fields are managed in groups and the groups as well as the single fields can be annotated.



- BasicModel
  - company
    - company
  - customer
    - customer
- DirectCustomersModel
  - DirCustCompany
    - DirCustCompanyData
  - DirCustCustomer
    - DirCustCustomerData
- ResellerModel
  - ResellerCompany
    - ResellerCompanyData
    - ResellerCompany_ServiceContractData
    - ResellerCompanyInternalResp
  - ResellerCustomer
    - ResellerCustomerData

Name	Type
company_name	short string
company_number	short string
address	string
city	short string
zip	short string
country	enum
url	short string
phone_frontdesk	short string

Name	Value	Annotation group
show-in-group-section	true	layout

Name	Value	Annotation group
field indexed	transitive	indexing
position	0;0	layout

Fig. 1: ConSol*CM Admin Tool - Definition of Data Object Group Fields

Data Object Groups

Like the custom fields in previous versions, data object group fields are placed in groups, the *data object groups*. Each data object within a customer data model can have as many data object groups as required. For example, for a reseller company, there can be a data object 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 data object group with general data is defined.

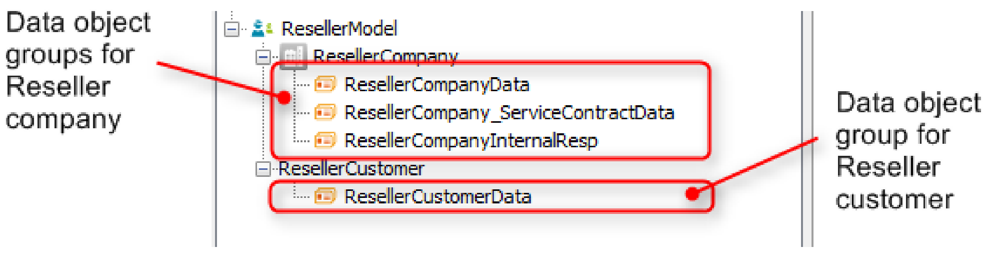
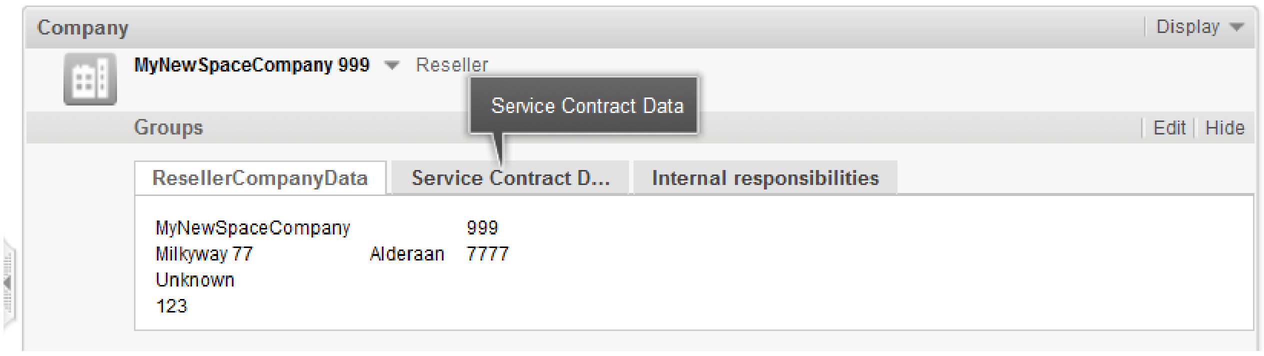

Data object groups for Reseller company
ResellerModel
ResellerCompany
ResellerCompanyData
ResellerCompany_ServiceContractData
ResellerCompanyInternalResp
ResellerCustomer
ResellerCustomerData
Data object group for Reseller customer

Fig. 2: ConSol*CM Admin Tool - Customer Data Model with Several Data Object 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 data object group ...

- can be faded in and out in the GUI during the process, but only the whole group, not single fields (= data object group fields) of it.
- can be displayed as tab or in the customer data section. The title (and mouse-over) of the tab is the (localized) name of the data object group.
- is configured by the group annotations.
- is placed on the GUI based on its position in the data object group list (defines e.g. the order of tabs).



MyNewSpaceCompany 999
Milkyway 77 Alderaan 7777
Unknown
123

Fig. 3: ConSol*CM Web Client - Company Data Organized in Tabs (Based on Data Object Groups)

Data Object Group Field Definition

The definition of data object group fields (i.e. data fields like *name*, *address*, or *phone number*) is based on the principle which has been used for custom fields in previous ConSol*CM versions.

A data object group field ...

- is defined by a data type.
- is configured using custom field annotations (e.g. *position* or *field-indexed*), see Annotations.

Labels for Data Object Group Fields

In CM versions prior to 6.9.4, the labels for data object group fields (e.g. name, address) are displayed in the Web Client as watermarks per default. If a distinct label is required, a separate data object group field of type *string* with the annotation *text-type* = *label* has to be used, as shown in the following figures.

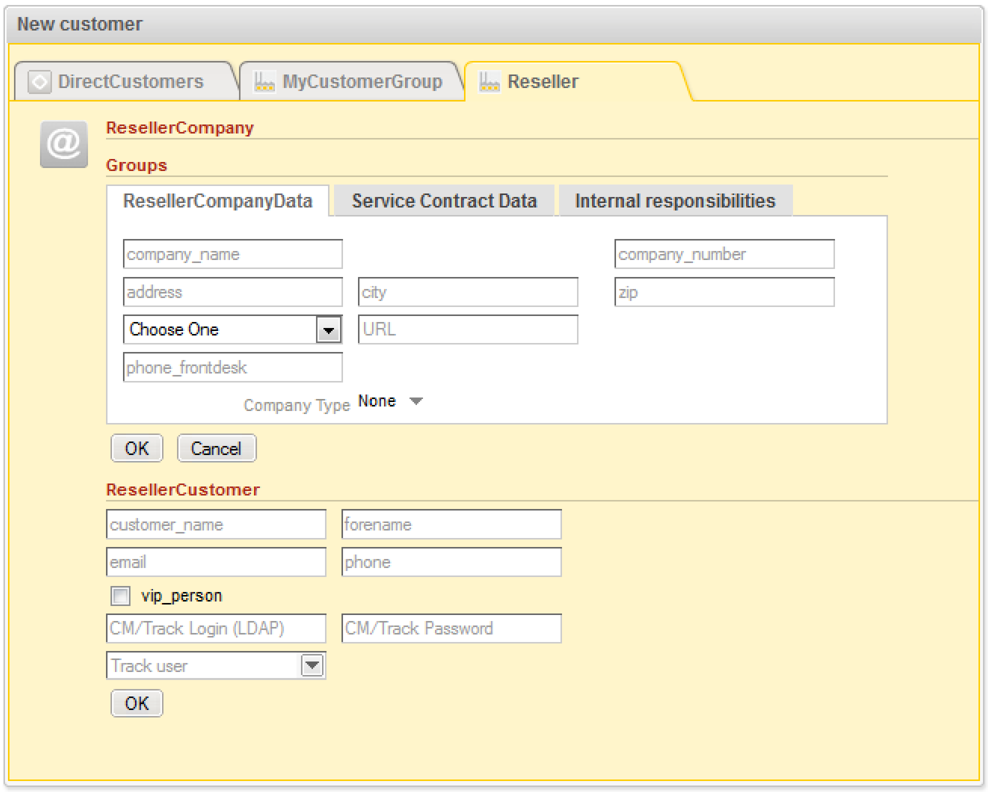
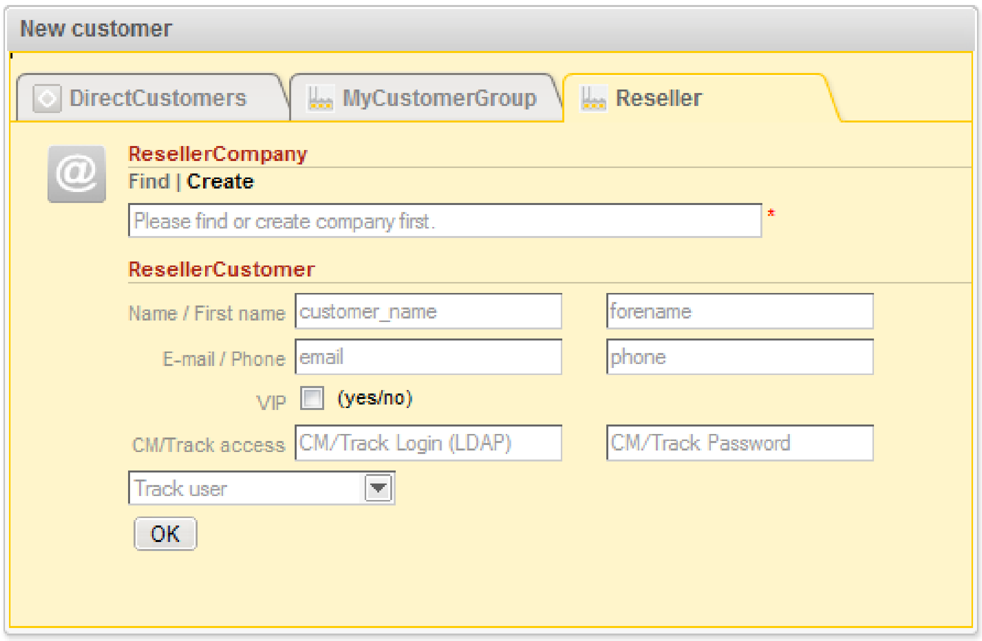


Fig. 4: Web Client, default display of data object group field labels in versions prior to 6.9.4


New customer

DirectCustomers | MyCustomerGroup | Reseller

ResellerCompany
Find | Create
Please find or create company first. *

ResellerCustomer

Name / First name	customer_name	forename
E-mail / Phone	email	phone
VIP	☐ (yes/no)	
CM/Track access	CM/Track Login (LDAP)	CM/Track Password

Track user

OK

Fig. 5: Web Client, labels as distinct data object group fields in versions prior to 6.9.4

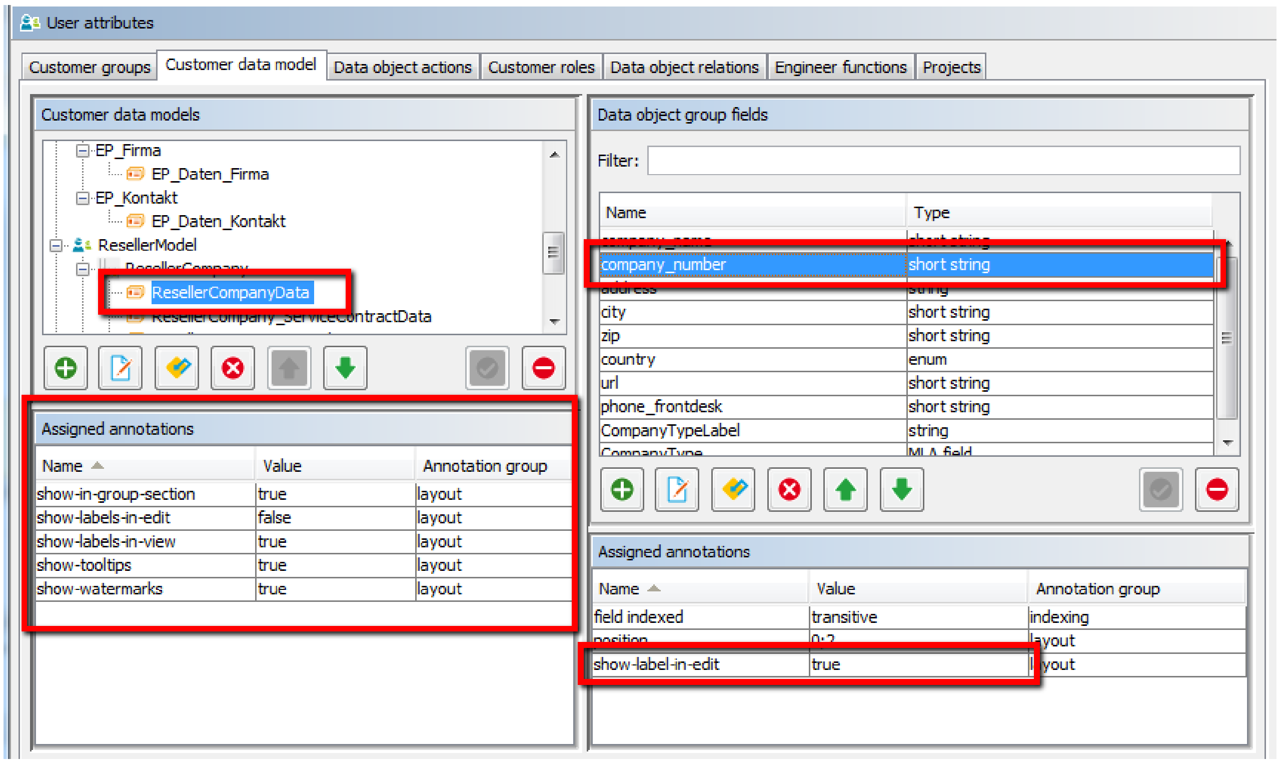
Now (as of CM version 6.9.4) you as an administrator can decide if watermarks or labels (or both) should be used, labels are now implemented similar to the labels for custom fields (i.e. the data fields for ticket data). Furthermore, tool tips can be added. The display modes of labels for data object group fields is controlled by annotations.Annotations can be set on data object group level and on data object group field level. The field annotations overwrite the group annotations. In tis way, you can define a group display behavior but can modify one or more single fields. Please see the example which is explained below.

Annotations for data object 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 data object group fields:

- layout:show-label-in-view
- layout:show-label-in-edit
- layout:show-watermark
- layout:show-tooltip



Name	Type
company_name	short string
company_number	short string
address	string
city	short string
zip	short string
country	enum
url	short string
phone_frontdesk	short string
CompanyTypeLabel	string
CompanyType	MLA field

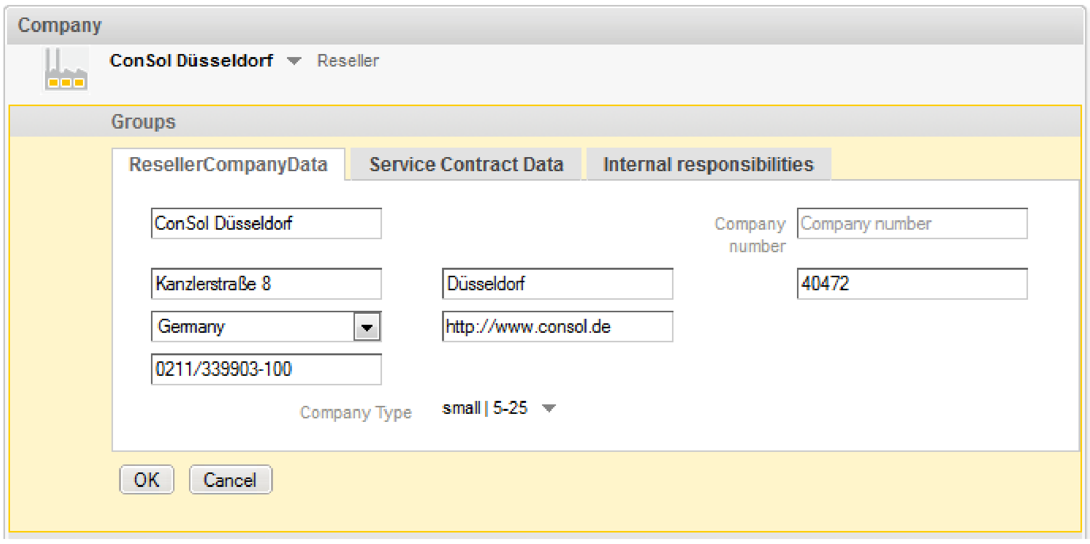
Assigned annotations (ResellerCompanyData)

Name	Value	Annotation group
show-in-group-section	true	layout
show-labels-in-edit	false	layout
show-labels-in-view	true	layout
show-tooltips	true	layout
show-watermarks	true	layout

Assigned annotations (company_number)

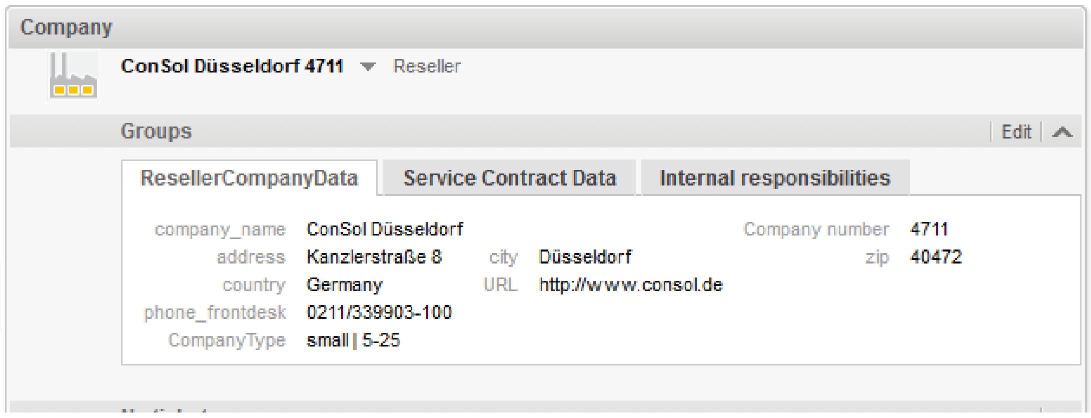
Name	Value	Annotation group
field indexed	transitive	indexing
position	0;2	layout
show-label-in-edit	true	layout

Fig. 6: Admin Tool configuration for Reseller data object group, one field configured field-specific



	ConSol Düsseldorf
Company number	Company number
	Kanzlerstraße 8
	Düsseldorf
	40472
	Germany
	http://www.consol.de
	0211/339903-100
Company Type	small | 5-25

Fig. 7: Web Client (edit mode) view for Reseller data object group, one field configured field-specific



ConSol Düsseldorf 4711 Reseller

company_name	ConSol Düsseldorf			Company number	4711
address	Kanzlerstraße 8	city	Düsseldorf	zip	40472
country	Germany	URL	http://www.consol.de		
phone_frontdesk	0211/339903-100				
CompanyType	small | 5-25				

Fig. 8: Web Client (view mode) view for Reseller data object group, one field configured field-specific

As you might know from the work with custom fields, you can use three columns for data object group fields, i.e. for the annotation *position*, you can use the columns 0, 1, and 2. The labels are inserted automatically, so in the end, there might be six columns as maximum.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

Fig. 9: Example for data object group field positions in the grid

Label mode after an update from CM version prior to 6.9.4 to a version 6.9.4 and up

The layout of the customer model will be preserved during the update! All annotations will be set accordingly and can be modified later.

For all existing data object groups the annotation setting layout:*show-labels-in-edit* = *false* (which will hide standard labels in edit mode) is made during the update.

Default mode for new data object groups and data object group fields

All new data object groups will be rendered with the new default configuration. This means:

- standard labels
- watermarks disabled
- tool tip enabled

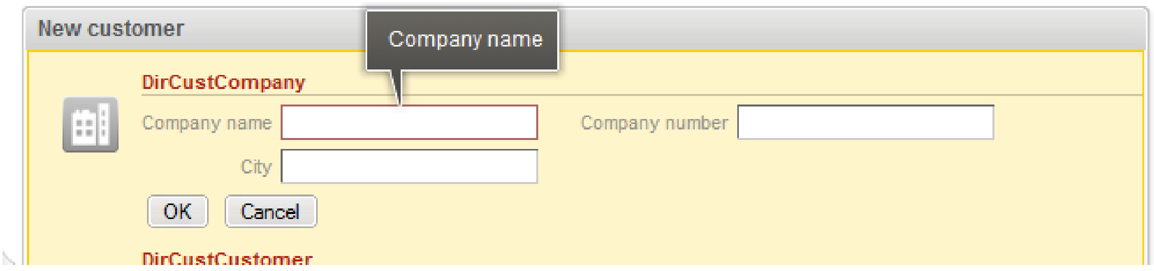


Fig. 10: Web Client, default display mode for data object group fields

9.5.3 Scripting Using Objects from the FlexCDM

In this book we use the terms *data object* and *data object definition*. However, the names of the corresponding Java classes are *Unit* and *UnitDefinition*. All other Java classes which deal with customer data objects also are still named *Unit..*. Please keep that in mind when you work on the administrator level as well as on the programmer's level. Please refer to the *ConSol*CM Java API Doc* and to the *ConSol*CM Process Designer Manual* for details.

New Scripting Features Since ConSol*CM Version 6.9

Up to ConSol*CM version 6.8, a *Unit* could have only one custom field group. The expression *unit.get ("name")* was always valid because one custom field with a specified name could exist only once, for example *"group1.name"*.

Starting with ConSol*CM version 6.9, a data object (*Unit*) may have one or more data object group fields with the same field name, e.g *"name"* can be represented as *"group1.name"*, *"group2.name"* . In such cases, the expression *Unit.get("name")* is not valid and throws an exception.

Attention:

For backwards compatibility the code *unit.get("name")* will work as long as the custom field *"name"* is unique. When another custom field with the same name is added, the code *unit.get("name")* will no longer work!

Now use the following notation to retrieve unit field (data object group field) data:

- **For one field:**
  `unit.get("group1 :name ")`
- **For numerous fields:**
  `unit.get(" group1:field1.group2:field2 ")`

Example to get contact company name

```
unit.get("contactFields:companyReference.companyFields:name")
```

Extension of the Custom Field Expression Language (CFEL)

Starting with ConSol*CM version 6.9, the *Custom Field Expression Language* (CFEL) has been improved to provide simple access to many objects. This concerns scripting for ticket and other objects as well as objects from the customer data model, i.e. Units (data objects: objects on contact or company level). Here, the improvements concerning units (data objects) will be explained. For a detailed explanation on how to write Java or Groovy code which uses customer data model objects, please refer to the *ConSol*CM Process Designer Manual*.

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
  - E-mail
  - Clone
  - Default values
  - Data object action
  - Data object condition
  - Workflow

9.5.4 Changes in Scripting from Consol*CM Version 6.8 to Version 6.9

In ConSol*CM version 6.9, some methods were declared deprecated and have to be replaced by new methods. This is only relevant if you have scripts from version 6.8 or older. In this case, the scripts have to be redesigned using the new methods.

AbstractField

Removed custom value accessors for each custom field type.

Removed/changed method	Replacement
StringField.getStringValue()	StringField.getValue()
NumberField.getNumberValue()	NumberField.getValue()

ActivityFormFieldsSet

Removed accessors with plain *FieldDefinition*, use *ActivityFormElement* instead.

Removed/changed method	Replacement
ActivityFormFieldsSet.addFieldDefinition(new FieldDefinition)	ActivityFormFieldsSet.addElement(new ActivityFormElement(new FieldDefinition()))
ActivityFormFieldsSet.getFields() returns List<FieldDefinition>	ActivityFormFieldsSet.getElements() returns List<ActivityFormElement>
ActivityFormFieldsSet.setFields (List<FieldDefinition>)	ActivityFormFieldsSet.setElements (List<ActivityFormElement>)
ActivityFormFieldsSet.removeFieldDefinition (FieldDefinition)	ActivityFormFieldsSet.removeElement (ActivityFormElement)
ActivityFormFieldsSet.removeAllFieldDefinitions()	ActivityFormFieldsSet.removeAllElements()
ActivityFormFieldsSet.getFieldDefinition(index) returns FieldDefinition	ActivityFormFieldsSet.getElement(index) returns ActivityFormElement
ActivityFormFieldsSet.addFieldDefinition(new FieldDefinition, index)	ActivityFormFieldsSet.setElements(ordered list of elements)

ContentFile

Added size parameter to input stream methods.

Removed/changed method	Replacement
new ContentFile(filename, inputstream)	new ContentFile(filename, inputstream, streamsize)
ContentFile.setInputStream(inputstream)	ContentFile.setInputStream(inputstream, streamsize)

ContentResource

Same changes as in *ContentFile*.

Removed/changed method	Replacement
new ContentResource(filename, inputstream)	new ContentResource(filename, inputstream, streamsize)

Removed/changed method	Replacement
ContentResource.setInputStream(inputstream)	ContentResource.setInputStream(inputstream, streamsize)

FieldLogEntry

Removed modification accessors.

Removed/changed method	Replacement
FieldLogEntry.setModification(Modification)	FieldLogEntry.setValue(value) + FieldLogEntry.setPreviousValue(value)
FieldLogEntry.getModification()	FieldLogEntry.getValue() + FieldLogEntry.getPreviousValue()

Ticket

Removed renamed custom fields accessors and other changes.

Removed/changed method	Replacement
Ticket.getField(), Ticket.setFieldValue(), Ticket.removeField()	Previously mixing *groupName* and *fieldName* parameters worked, now only the order *groupName, fieldName* is accepted.
Ticket.setField(AbstractField)	Ticket.addField(AbstractField)
Ticket.addOrUpdateField(AbstractField)	Ticket.setFieldValue(pGroupName, pFieldName, Object pValue)
Ticket.getEnumValue	EnumValue enumValue = getFieldValue(String pGroupName, String pFieldName) String enumName = enumValue.getName();

Removed/changed method	Replacement

Ticket.setEnumValue(fieldName, groupName, enumName)	EnumValue enumValue = enumService. getEnumValue(enumGroupName, enumValueName);
	
	Ticket.setFieldValue(pGroupName, pFieldName, enumValue);
	
	For workflow usage:
	
	Ticket.setFieldValue(pGroupName, pFieldName, getEnumValueByName(enumGroupName, enumValueName));

TimerTrigger

Removed *setDuedate* method.

Removed/changed method	Replacement
TimerTrigger.setDuedate	TimerTrigger.setDueTime

Unit

Removed renamed custom fields accessors.

Removed/changed method	Replacement
Unit.getFieldsSet()	Unit.getFields()
Unit.setFieldsMap(Map)	Unit.addFields(Set)
Unit.setField(AbstractField)	Unit.addField(AbstractField)

New (Convenience) Methods

Examples for new 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).
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.

Example 1: Search for the tickets of a contact or of a company

```
TicketCriteria ticketCriteria = new TicketCriteria();
Unit patternContact = new Unit("contact", customerGroup);
mdcmCriteriaBuilder.setReferencedContactCriteria(ticketCriteria, patternContact);
```

Example 2: Search for the tickets of the contact who is member of a certain 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);
```

Example 3: Search for contacts of a certain company

```
UnitCriteria unitCriteria = new UnitCriteria();Unit c
ompanyPattern = new Unit("company", customerGroup);
mdcmCriteriaBuilder.setReferencedCompanyCriteria(unitCriteria, companyPattern);
```

Information:

For detailed information about the methods, including input parameters (method signatures) and output data type, please refer to the *ConSol*CM Java API Doc*.

9.5.5 New Objects in ConSol*CM 6.9 and Up

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:

Starting point	Script	Objects	Example
Company page	data object action script	unit represents the company	def contacts = unit.get ("contacts()")
Contact page	data object 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()

9.6 Templates for Customer Data

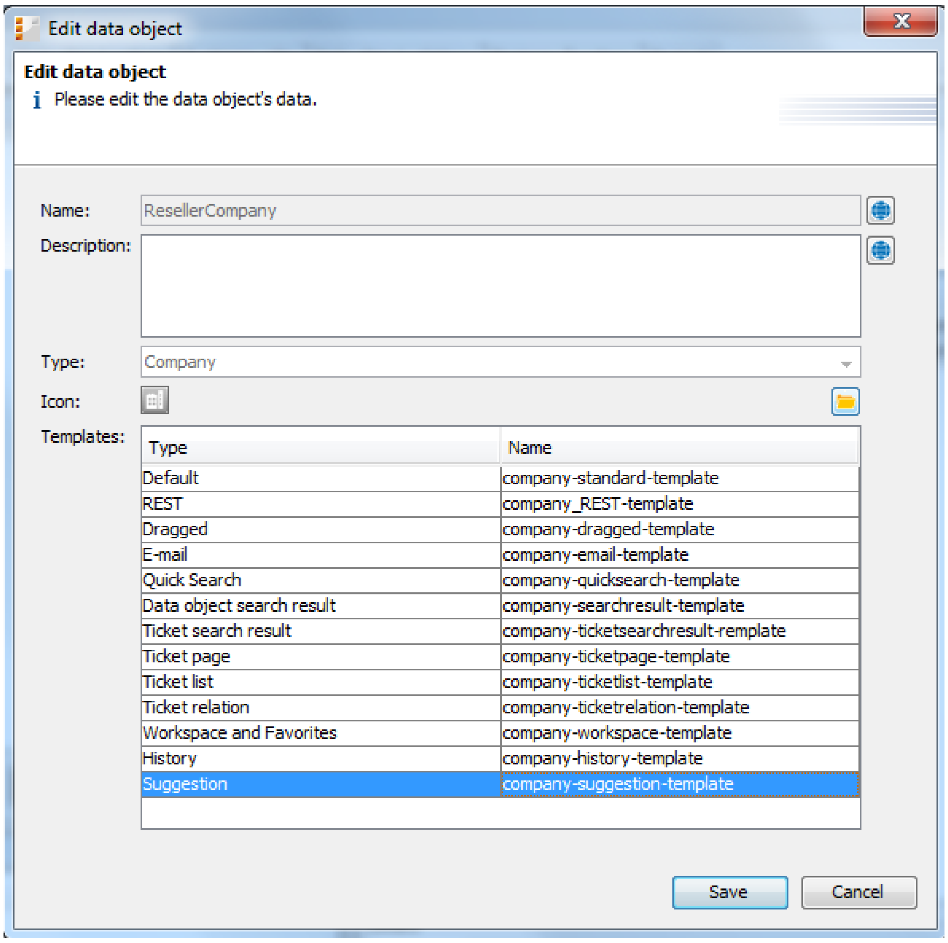
- Introduction to Using Templates for the Display of Customer Data
- Coding Templates
  - Examples for Templates
- Template Types
  - Standard (Default)
  - REST
  - Dragged
  - E-Mail
  - Quick Search
  - Data Object Search Result
  - Ticket Search Result
  - Ticket Page
  - Ticket List
  - Ticket Relation
  - Workspace and Favorites
  - History
  - Suggestion
  - CM.Phone Customer Details
  - CM.Phone Customer List

9.6.1 Introduction to Using Templates for the Display of Customer Data

In the ConSol*CM Web Client, contact data sets are displayed in short form at various locations, based on templates. For example in the ticket list, the contact name and company name might be required whereas in the contact data section of the ticket, the name, first name, and phone number of a contact might be needed. This section will show you where short forms are used and how the respective templates are configured using the Admin Tool.

The configuration is based on the following principle (very similar to the principle used in previous ConSol*CM6 versions):

- An annotation is assigned to a data object, i.e. to a contact or company definition, in the tab *Customer data model* of the *User attributes* section in the Admin Tool. It defines for which section of the Web Client the annotation is valid.
- The referenced template must be the name of a template which is stored in the *Script and Template* section of the Admin Tool. You as an administrator are free to assign names to the templates. All you have to make sure is that the referenced name in the *Edit data object* window and the template name in the *Script andTemplate* section are identical.



Edit data object

Please edit the data object's data.

Name: ResellerCompany

Description:

Type: Company

Icon:

Templates:

Type	Name
Default	company-standard-template
REST	company_REST-template
Dragged	company-dragged-template
E-mail	company-email-template
Quick Search	company-quicksearch-template
Data object search result	company-searchresult-template
Ticket search result	company-ticketsearchresult-remplate
Ticket page	company-ticketpage-template
Ticket list	company-ticketlist-template
Ticket relation	company-ticketrelation-template
Workspace and Favorites	company-workspace-template
History	company-history-template
Suggestion	company-suggestion-template

Fig. 1: ConSol*CM Admin Tool - Template Annotations for Data Object (Here: Company)

In the following paragraphs, the syntax and coding for templates and all possible template types are explained.

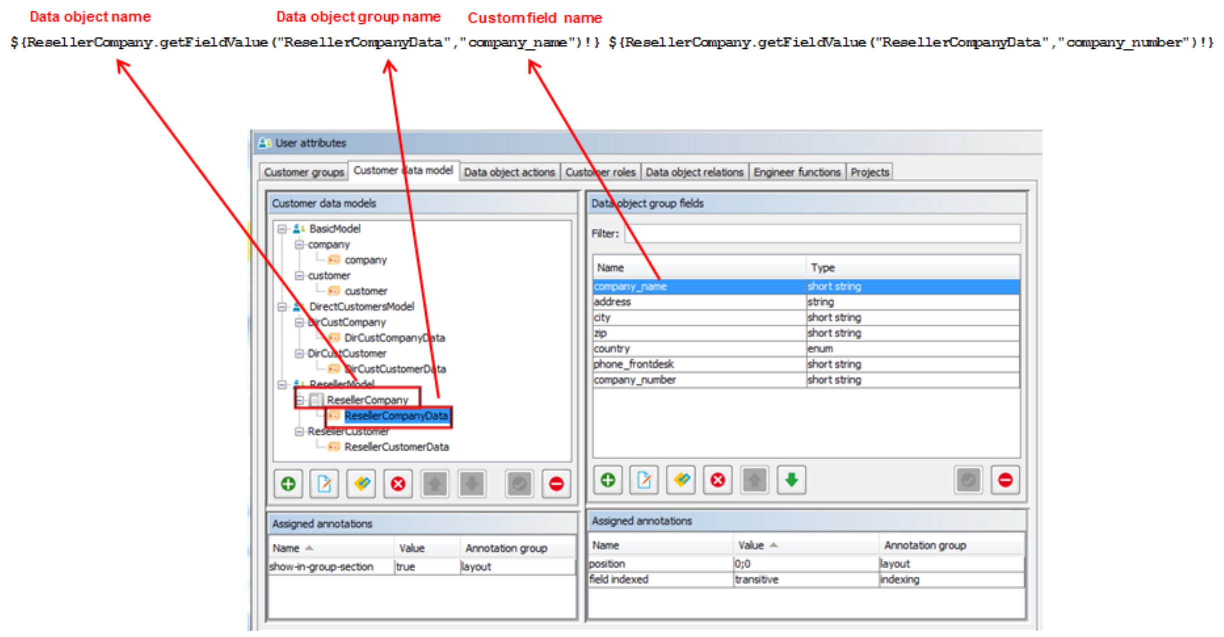
9.6.2 Coding Templates

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:

1. **Name of the data object**
   i.e. the name of a company or contact object.
2. **Name of the data object group**
   within the data object (a data object group is similar to a custom field group for ticket data).
3. **Name of the data object group fields**
   within the data object group.

See the following figure for an example:



```
${ResellerCompany.getFieldValue("ResellerCompanyData","company_name")!} ${ResellerCompany.getFieldValue("ResellerCompanyData","company_number")!}
```

```
BasicModel
  company
    company
  customer
    customer
DirectCustomersModel
  DirCustCompany
    DirCustCompanyData
  DirCustCustomer
    DirCustCustomerData
ResellerModel
  ResellerCompany
    ResellerCompanyData
  ResellerCustomer
    ResellerCustomerData
```

Name	Value	Annotation group
show-in-group-section	true	layout

Name	Type
company_name	short string
address	string
city	short string
zip	short string
country	enum
phone_frontdesk	short string
company_number	short string

Name	Value	Annotation group
position	0;0	layout
field indexed	transitive	indexing

Fig. 2: ConSol*CM Admin Tool - Writing Customer Templates

Attention:

The customer templates must be single-row! They must not contain line-breaks!

Examples for Templates

Here are some examples for templates:

Example for engineer description template name (has to be written in one line!)

```
<#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>
```

Example for company-contact-template (has to be written in one line!)

```
${company.getFieldValue("company", "name1")!}${company.getFieldValue("company", "name2")!}
${company.getFieldValue("company", "mainaddr_city")!},${customer.getFieldValue("customer", "firs
tname")!}
${customer.getFieldValue("customer", "name")!}
```

Example for search-customer-template (has to be written in one line!)

```
<#if company??>${company.getFieldValue("company", "name1")!}${company.getFieldValue("company", "
name2")!}
${company.getFieldValue("company", "mainaddr_city")!},
</#if>${customer.getFieldValue("customer", "firstname")!}${customer.getFieldValue("customer", "n
ame")!}
```

Setting number format: removing "." in number display

```
<#setting number_format="#"/>${customerModelCompany.getFieldValue("groupName", "numberValueField
")!}
```

9.6.3 Template Types

Standard (Default)

This template, respectively format, is used at all following locations if no special templates have been defined, i.e. all other annotations could be omitted if a standard template is defined.

Attention:

If no template is defined for a certain Web Client location and no standard template has been defined either, there will be an error in the log file and *-- unknown --* will be displayed in the Web Client.

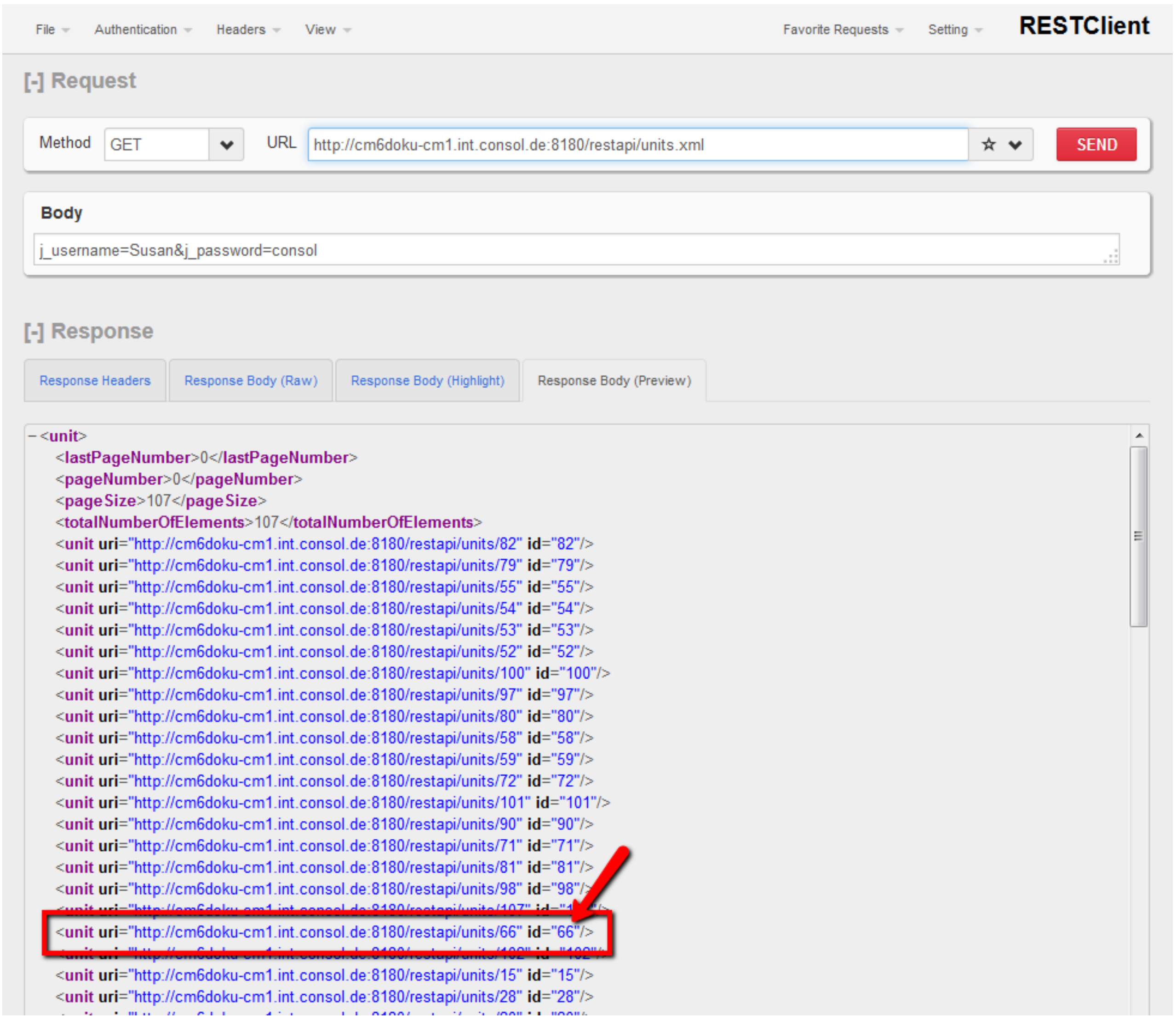
So make sure that at least a standard template is defined. In a two-level customer data model, this has to be done for the company and for the contact level!

REST

The display of customer data using the *REST API*. In the standard configuration, no customer data are displayed in the CM portal, CM.Track which is based on the REST API. This template will only be effective when you address the REST API directly, e.g. for programming CM interfaces. The following example shows a REST client request for customer data and the resulting answer of the CM server via REST API. The value within the <mark> tag in the XML output is the customer information which is formatted using the REST template. In the example, the company name ("ConSol*") and the company number ("4711") are part of the template.

Example REST template

```
${ResellerCompany.getFieldValue("ResellerCompanyData","company_name")!} ${ResellerCompany.
getFieldValue("ResellerCompanyData","company_number")!}
```



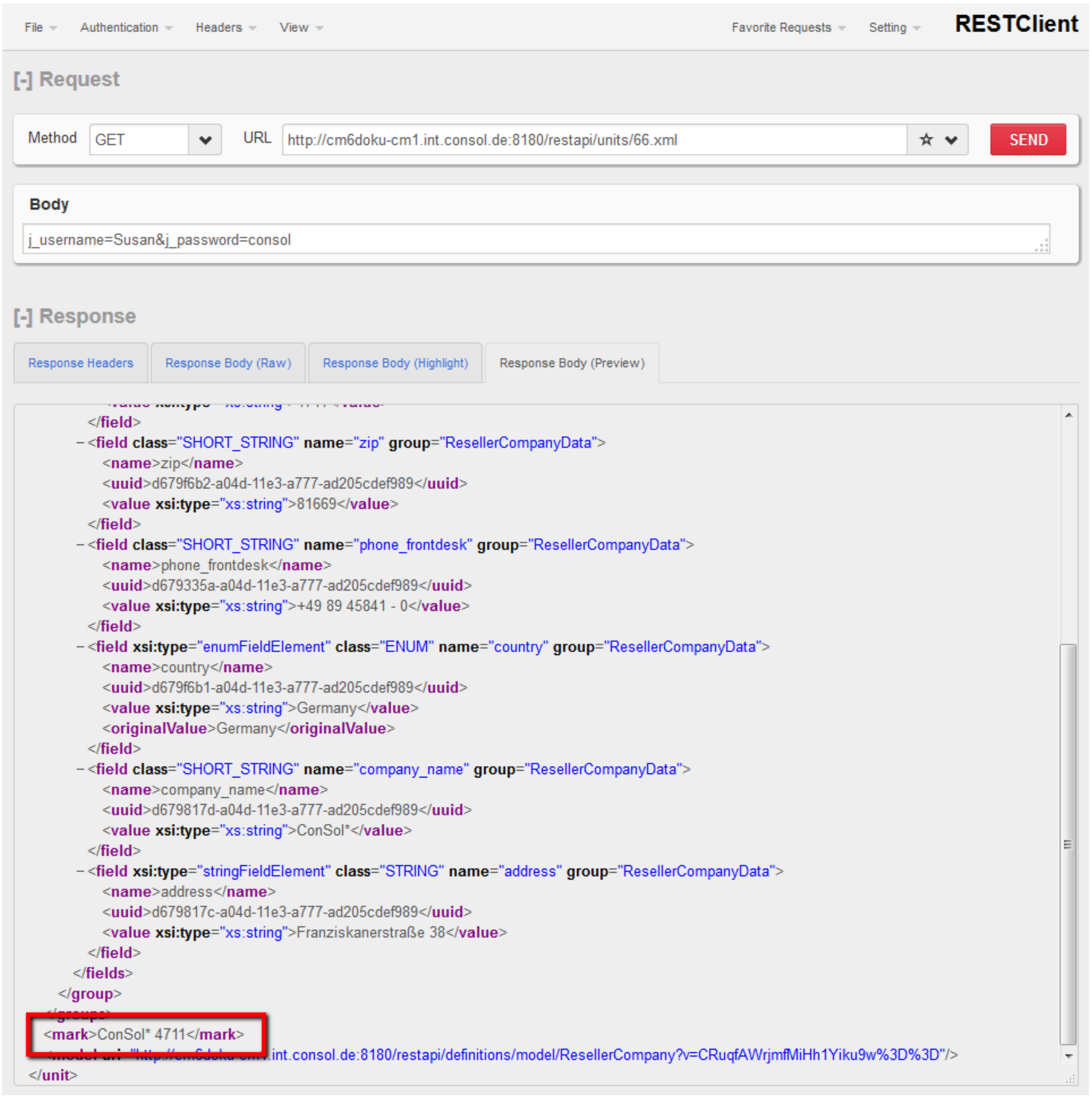
Method: GET
URL: http://cm6doku-cm1.int.consol.de:8180/restapi/units.xml

Body

```
j_username=Susan&j_password=consol
```

```
- <unit>
    <lastPageNumber>0</lastPageNumber>
    <pageNumber>0</pageNumber>
    <pageSize>107</pageSize>
    <totalNumberOfElements>107</totalNumberOfElements>
    <unit uri="http://cm6doku-cm1.int.consol.de:8180/restapi/units/82" id="82"/>
    <unit uri="http://cm6doku-cm1.int.consol.de:8180/restapi/units/79" id="79"/>
    <unit uri="http://cm6doku-cm1.int.consol.de:8180/restapi/units/55" id="55"/>
    <unit uri="http://cm6doku-cm1.int.consol.de:8180/restapi/units/54" id="54"/>
    <unit uri="http://cm6doku-cm1.int.consol.de:8180/restapi/units/53" id="53"/>
    <unit uri="http://cm6doku-cm1.int.consol.de:8180/restapi/units/52" id="52"/>
    <unit uri="http://cm6doku-cm1.int.consol.de:8180/restapi/units/100" id="100"/>
    <unit uri="http://cm6doku-cm1.int.consol.de:8180/restapi/units/97" id="97"/>
    <unit uri="http://cm6doku-cm1.int.consol.de:8180/restapi/units/80" id="80"/>
    <unit uri="http://cm6doku-cm1.int.consol.de:8180/restapi/units/58" id="58"/>
    <unit uri="http://cm6doku-cm1.int.consol.de:8180/restapi/units/59" id="59"/>
    <unit uri="http://cm6doku-cm1.int.consol.de:8180/restapi/units/72" id="72"/>
    <unit uri="http://cm6doku-cm1.int.consol.de:8180/restapi/units/101" id="101"/>
    <unit uri="http://cm6doku-cm1.int.consol.de:8180/restapi/units/90" id="90"/>
    <unit uri="http://cm6doku-cm1.int.consol.de:8180/restapi/units/71" id="71"/>
    <unit uri="http://cm6doku-cm1.int.consol.de:8180/restapi/units/81" id="81"/>
    <unit uri="http://cm6doku-cm1.int.consol.de:8180/restapi/units/98" id="98"/>
    <unit uri="http://cm6doku-cm1.int.consol.de:8180/restapi/units/107" id="1[illegible]"/>
    <unit uri="http://cm6doku-cm1.int.consol.de:8180/restapi/units/66" id="66"/>
    <unit uri="http://cm6doku-cm1.int.consol.de:8180/restapi/units/102" id="102"/>
    <unit uri="http://cm6doku-cm1.int.consol.de:8180/restapi/units/15" id="15"/>
    <unit uri="http://cm6doku-cm1.int.consol.de:8180/restapi/units/28" id="28"/>
```

Fig. 3: REST API: request for all units



Method GET URL http://cm6doku-cm1.int.consol.de:8180/restapi/units/66.xml

Body
j_username=Susan&j_password=consol

```
        <value xsi:type="xs:string">4711</value>
      </field>
    - <field class="SHORT_STRING" name="zip" group="ResellerCompanyData">
        <name>zip</name>
        <uuid>d679f6b2-a04d-11e3-a777-ad205cdef989</uuid>
        <value xsi:type="xs:string">81669</value>
      </field>
    - <field class="SHORT_STRING" name="phone_frontdesk" group="ResellerCompanyData">
        <name>phone_frontdesk</name>
        <uuid>d679335a-a04d-11e3-a777-ad205cdef989</uuid>
        <value xsi:type="xs:string">+49 89 45841 - 0</value>
      </field>
    - <field xsi:type="enumFieldElement" class="ENUM" name="country" group="ResellerCompanyData">
        <name>country</name>
        <uuid>d679f6b1-a04d-11e3-a777-ad205cdef989</uuid>
        <value xsi:type="xs:string">Germany</value>
        <originalValue>Germany</originalValue>
      </field>
    - <field class="SHORT_STRING" name="company_name" group="ResellerCompanyData">
        <name>company_name</name>
        <uuid>d679817d-a04d-11e3-a777-ad205cdef989</uuid>
        <value xsi:type="xs:string">ConSol*</value>
      </field>
    - <field xsi:type="stringFieldElement" class="STRING" name="address" group="ResellerCompanyData">
        <name>address</name>
        <uuid>d679817c-a04d-11e3-a777-ad205cdef989</uuid>
        <value xsi:type="xs:string">Franziskanerstraße 38</value>
      </field>
    </fields>
  </group>
 </groups>
 <mark>ConSol* 4711</mark>
 <model uri="http://cm6doku-cm1.int.consol.de:8180/restapi/definitions/model/ResellerCompany?v=CRuqfAWrjmfMiHh1Yiku9w%3D%3D"/>
</unit>
```

Fig. 4: REST API: Request for one unit by Id

Dragged

This defines the format of the contact/company data of a customer data set while the data set is dragged, e. g. from the *Customers* section to the *Favorites* section.

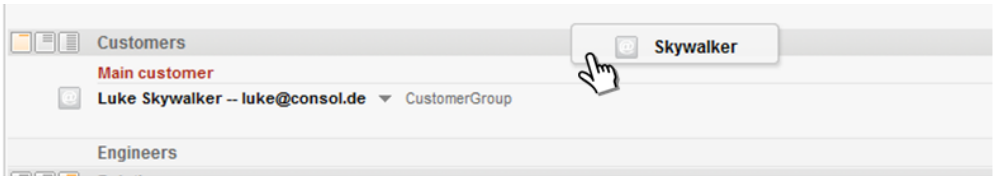
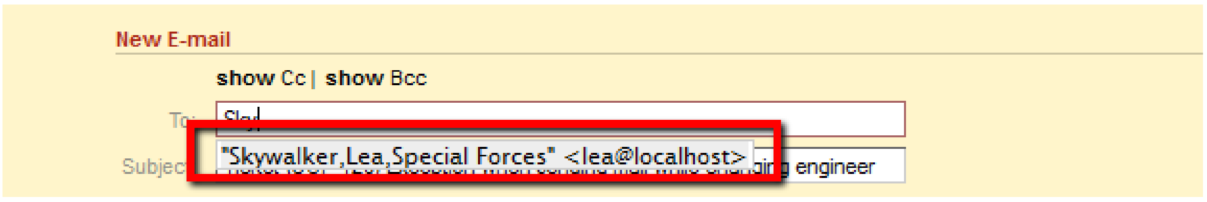


Fig. 5: ConSol*CM Web Client - Customer Dragged Template

E-Mail

In the Ticket E-Mail Editor an automatic search provides search results in-line in form of a drop-down list. The format of those search results can be configured using this template.



"Skywalker,Lea,Special Forces" <lea@localhost>

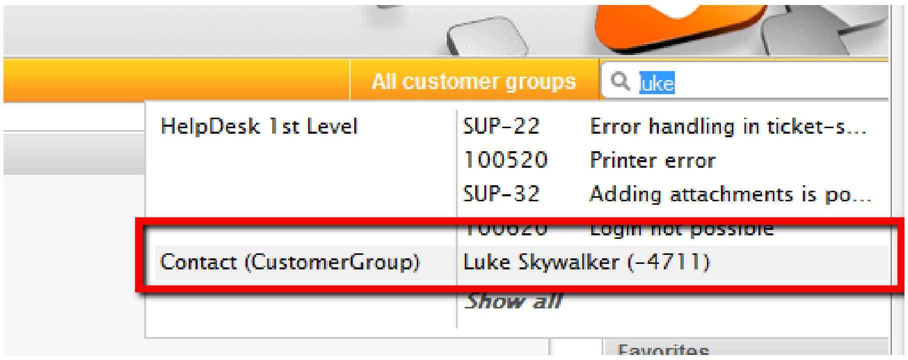
Fig. 6: ConSol*CM Web Client - Customer E-Mail Template

E-mail template (has to be written in one line!)

```
<#if customer.getFieldValue("customer","name")?has_content
&& customer.getFieldValue("customer","firstname")?has_content
&& customer.getFieldValue("customer","division")?has_content>
${customer.getFieldValue("customer","name")!},
${customer.getFieldValue("customer","firstname")!},
${customer.getFieldValue("customer","division")!}
<#else> ${customer.getFieldValue("customer","name")!},
${customer.getFieldValue("customer","division")!}</#if>
```

Quick Search

This template defines the format of the search result for contacts or companies in the quick search. The template has to be short and single-line.

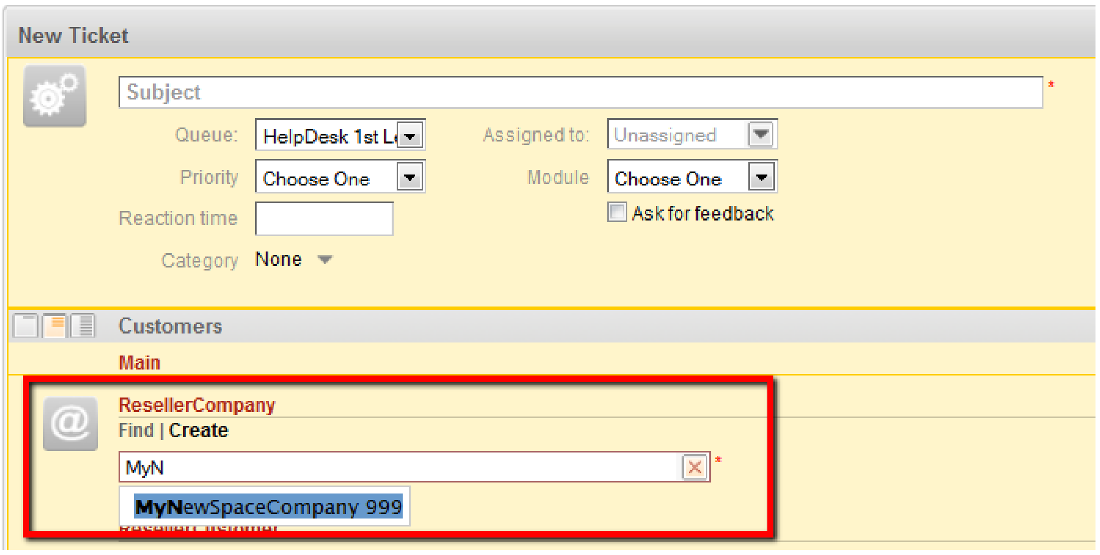


HelpDesk 1st Level	SUP-22	Error handling in ticket-s...
	100520	Printer error
	SUP-32	Adding attachments is po...
	100620	Login not possible
Contact (CustomerGroup)	Luke Skywalker (-4711)	

Fig. 7: ConSol*CM Web Client - Customer Quick Search Template

Data Object Search Result

This template defines the search results for automatic searches in auto-complete fields.


New Ticket

Subject *

Queue: HelpDesk 1st L… Assigned to: Unassigned

Priority Choose One Module Choose One

Reaction time ☐ Ask for feedback

Category None

Customers

Main

ResellerCompany

Find | Create

MyN *

MyNewSpaceCompany 999

Fig. 8: ConSol*CM Web Client - Customer Data Object Search Result Template

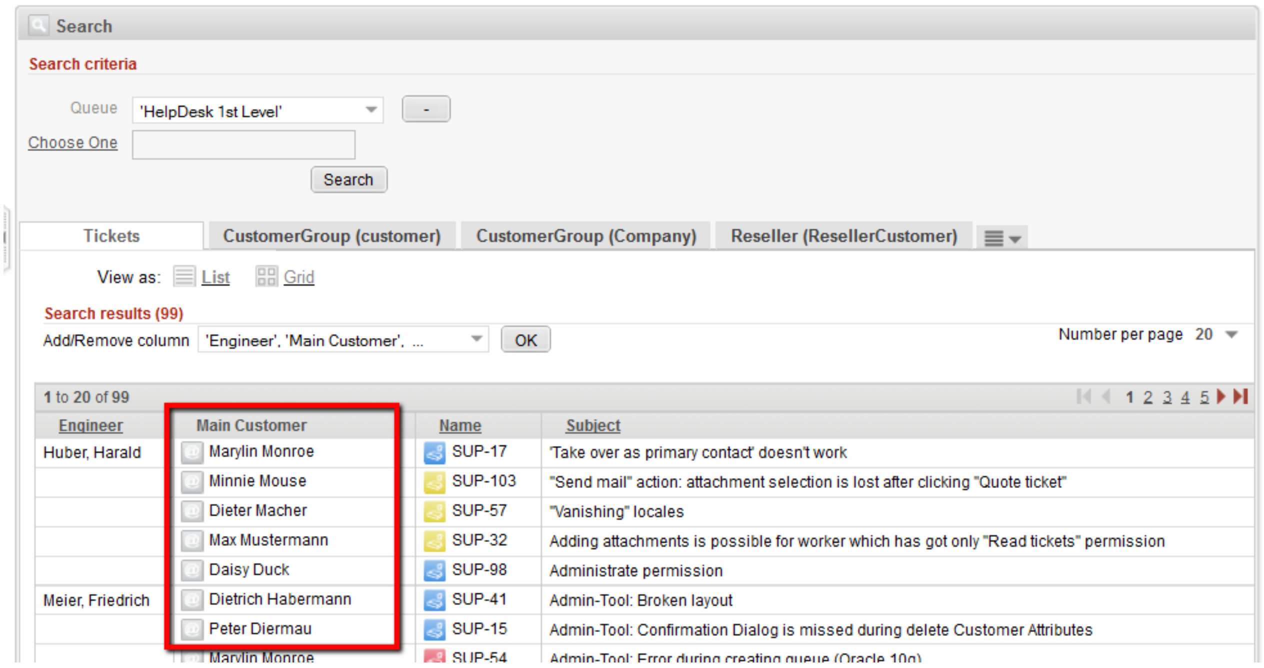
This is the applied template:

ResellerCompany search result template (has to be written in one line!)

```
${ResellerCompany.getFieldValue("ResellerCompanyData","company_name")!}
${ResellerCompany.getFieldValue("ResellerCompanyData","company_number")!}
```

Ticket Search Result

In the results page of the detail search the tickets found by the search are displayed as a list. One column of this list contains the main contact of the ticket. The *Ticket search result* template defines the layout of the customer data in this column.


Search

Search criteria

Queue 'HelpDesk 1st Level'

Choose One

Search

Tickets | CustomerGroup (customer) | CustomerGroup (Company) | Reseller (ResellerCustomer)

View as: List Grid

Search results (99)

Add/Remove column 'Engineer', 'Main Customer', ... OK

Number per page 20

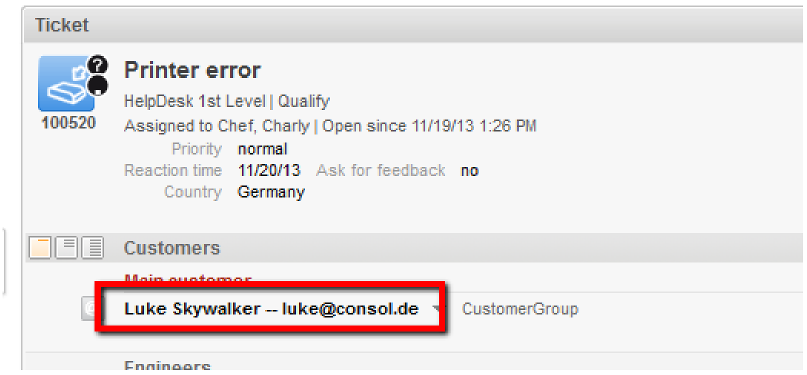
1 to 20 of 99 1 2 3 4 5

Engineer	Main Customer	Name	Subject
Huber, Harald	Marylin Monroe	SUP-17	'Take over as primary contact' doesn't work
	Minnie Mouse	SUP-103	"Send mail" action: attachment selection is lost after clicking "Quote ticket"
	Dieter Macher	SUP-57	"Vanishing" locales
	Max Mustermann	SUP-32	Adding attachments is possible for worker which has got only "Read tickets" permission
	Daisy Duck	SUP-98	Administrate permission
Meier, Friedrich	Dietrich Habermann	SUP-41	Admin-Tool: Broken layout
	Peter Diermau	SUP-15	Admin-Tool: Confirmation Dialog is missed during delete Customer Attributes
	Marylin Monroe	SUP-54	Admin-Tool: Error during creating queue (Oracle 10g)

Fig. 9: ConSol*CM Web Client - Customer Ticket Search Result Template

Ticket Page

This template defines the presentation of the contact/company data in the *Customers* section of a ticket.


Ticket

Printer error

100520

HelpDesk 1st Level | Qualify

Assigned to Chef, Charly | Open since 11/19/13 1:26 PM

Priority normal

Reaction time 11/20/13 Ask for feedback no

Country Germany

Customers

Main customer

Luke Skywalker -- luke@consol.de CustomerGroup

Engineers

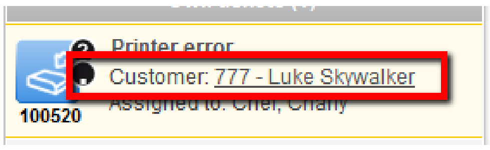
Fig. 10: ConSol*CM Web Client - Customer Ticket Page Template

Ticket List

This template defines the presentation of the contact data in the ticket list.

Attention:

If you would like to work with this template type, please make sure that the page customization parameter *accordionTicketList.mainCustomerDescriptionVisible* has been set to *true*. Otherwise contact data cannot be displayed in the ticket list.


Printer error

Customer: 777 - Luke Skywalker

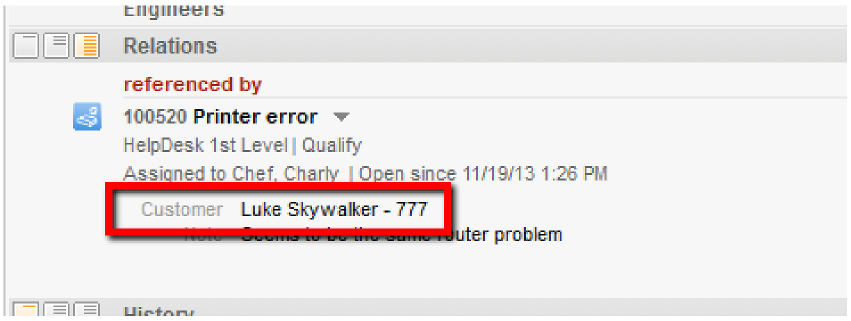
Assigned to: Chef, Charly

100520

Fig. 11: ConSol*CM Web Client - Customer Ticket List Template

Ticket Relation

This template defines the presentation of the contact data in ticket references in the *Relations* section of a ticket. Please keep in mind that contact data of referenced tickets are only displayed in the extended display level.


Relations

referenced by

100520 Printer error

HelpDesk 1st Level | Qualify

Assigned to Chef, Charly | Open since 11/19/13 1:26 PM

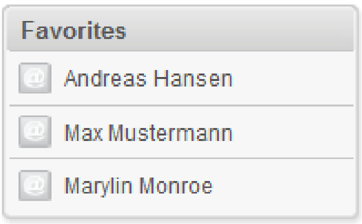
Customer Luke Skywalker - 777

Note Seems to be the same router problem

Fig. 12: ConSol*CM Web Client - Customer Ticket Relation Template

Workspace and Favorites

This template defines the presentation of contact data in the *Favorites* section.


Favorites

Andreas Hansen

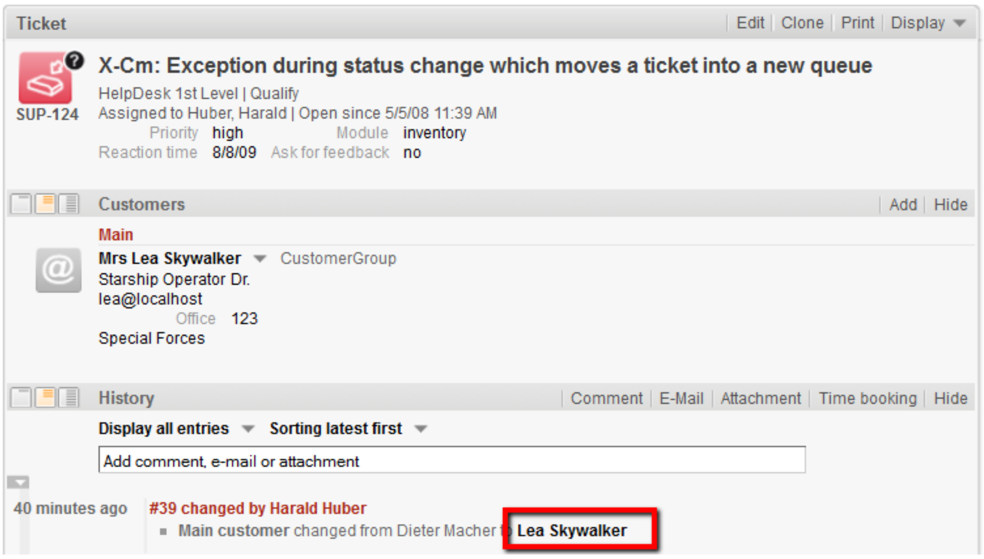
Max Mustermann

Marylin Monroe

Fig. 13: ConSol*CM Web Client - Customer Favorites Template

History

This template defines the presentation of contact data in the ticket protocol, i.e. in the *History* section of a ticket.


Ticket | Edit | Clone | Print | Display

SUP-124

X-Cm: Exception during status change which moves a ticket into a new queue

HelpDesk 1st Level | Qualify

Assigned to Huber, Harald | Open since 5/5/08 11:39 AM

Priority high Module inventory

Reaction time 8/8/09 Ask for feedback no

Customers | Add | Hide

Main

Mrs Lea Skywalker CustomerGroup

Starship Operator Dr.

lea@localhost

Office 123

Special Forces

History | Comment | E-Mail | Attachment | Time booking | Hide

Display all entries Sorting latest first

Add comment, e-mail or attachment

40 minutes ago #39 changed by Harald Huber

- Main customer changed from Dieter Macher to Lea Skywalker

Fig. 14: ConSol*CM Web Client - Customer History Template

Suggestion

This template defines the presentation of contact data for suggestions which are displayed when a ticket is created.

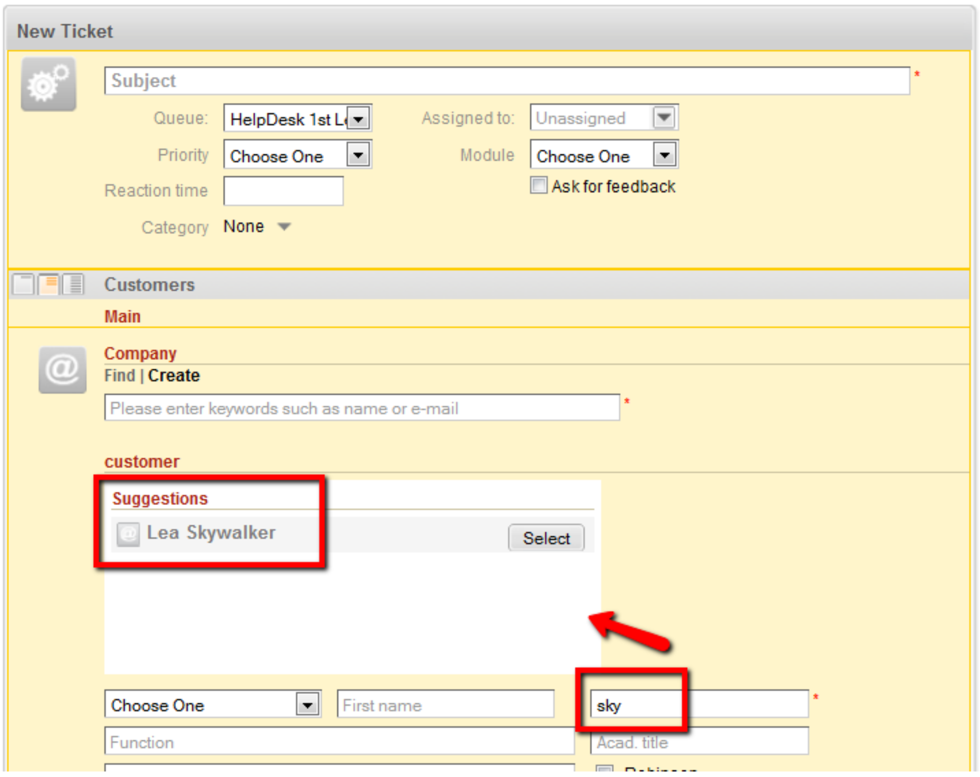


Fig. 15: ConSol*CM Web Client - Customer Suggestion Template

CM.Phone Customer Details

See section CTI with ConSol*CM: CM.Phone.

CM.Phone Customer List

See section CTI with ConSol*CM: CM.Phone.

10 Managing Customer Groups

- Basic Principle for Customer Data Models and Customer Groups
- Managing Customer Groups Using the Admin Tool
    - Customer Groups List
    - Customer Group Details
    - Creating a New Customer Group
    - Editing a Customer Group
    - Deleting a Customer Group
    - Disabling and (Re-)Enabling a Customer Group
- Assigning Access Rights for Customer Groups

10.1 Basic Principle for Customer Data Models and Customer Groups

In a ConSol*CM system several customer groups can be used.

Principles:

There can be any number of customer groups and any number of customer data models.

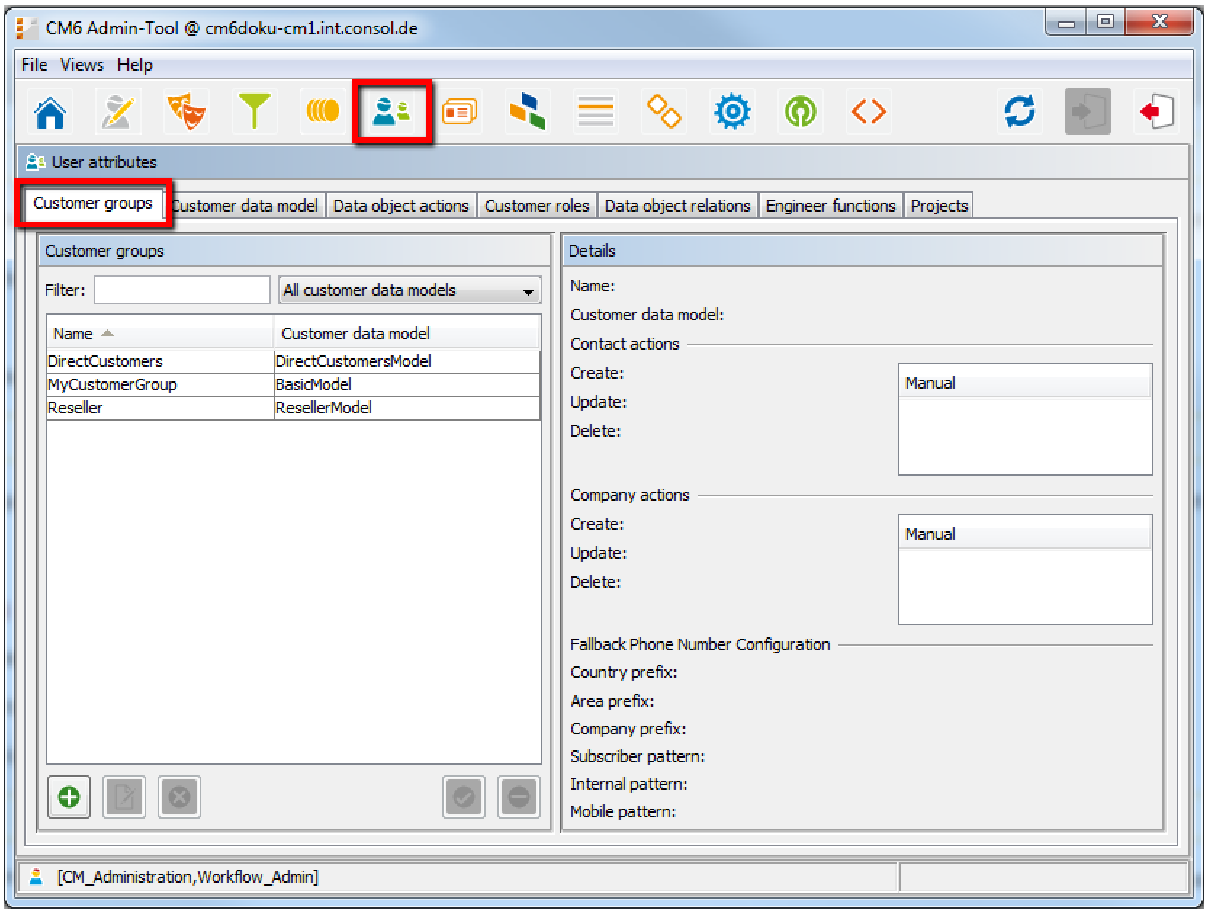
Each customer group has exactly one customer data model.

Each customer data model can be assigned to any number of customer groups.

In the following example, the system contains three customer groups, each with its specific customer data model.

10.2 Managing Customer Groups Using the Admin Tool

In the Admin Tool, customer groups are managed using the *Customer groups* tab in the *User attributes* section.



Name	Customer data model
DirectCustomers	DirectCustomersModel
MyCustomerGroup	BasicModel
Reseller	ResellerModel

Contact actions: Manual

Company actions: Manual

Fig. 1: ConSol*CM Admin Tool - Managing Customer Groups

10.2.1 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.

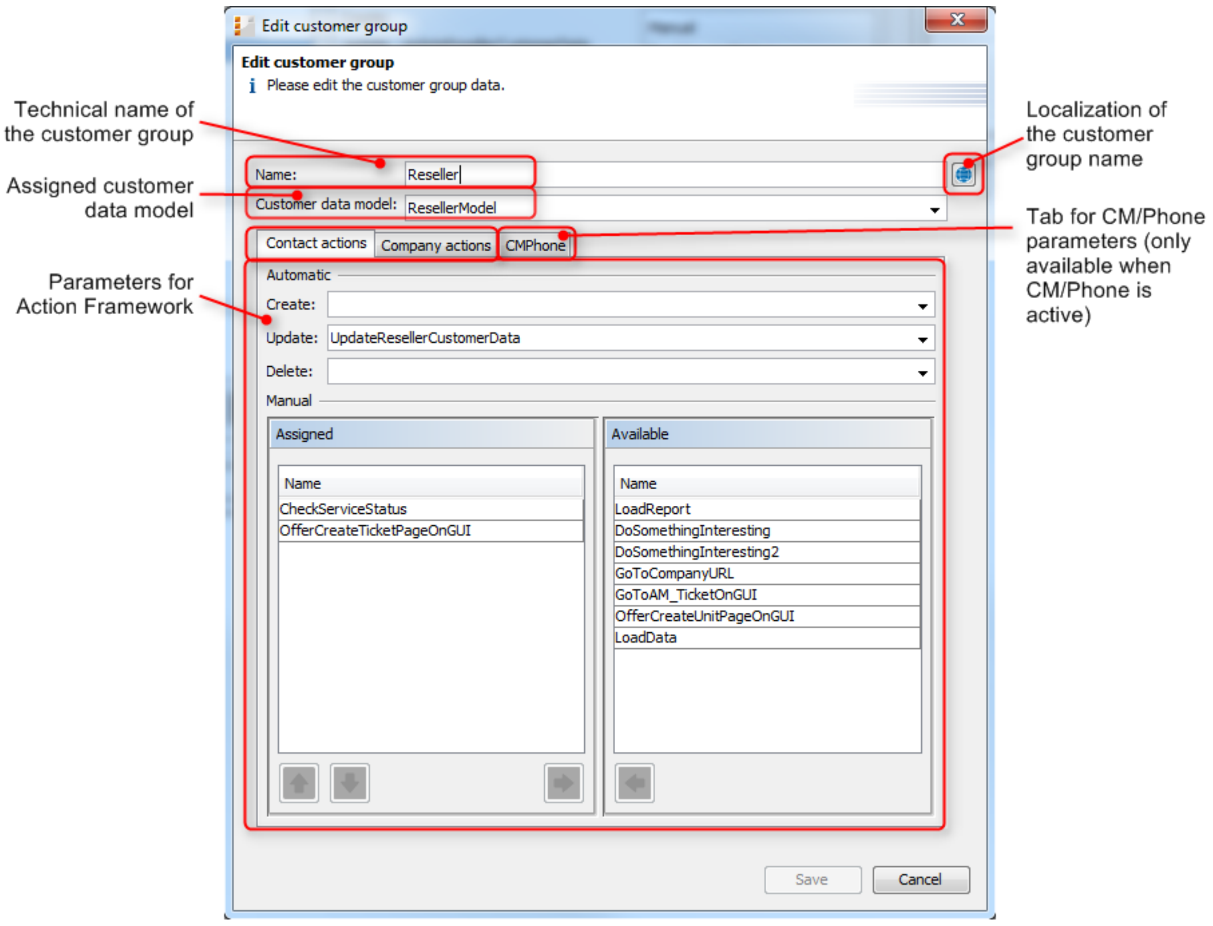
- **Customer data model filter**
  Select a customer data model from the drop-down list. Only customer groups with the selected data model will be displayed in the list.

10.2.2 Customer Group Details

On the right side, the details of the customer group, which is selected in the list, will be displayed. An explanation of all parameters is given in the following section.

10.2.3 Creating a New Customer Group

You create a new customer group by clicking on ⊕ below the group list. A pop-up window is opened where you have to enter the customer group parameters.


Technical name of the customer group
Assigned customer data model
Parameters for Action Framework
Localization of the customer group name
Tab for CM/Phone parameters (only available when CM/Phone is active)
Edit customer group
Edit customer group
Please edit the customer group data.
Contact actions
Company actions
CMPhone
Automatic
Manual
Assigned
Available
Save
Cancel

Name: Reseller
Customer data model: ResellerModel
Create:
Update: UpdateResellerCustomerData
Delete:

Name
CheckServiceStatus
OfferCreateTicketPageOnGUI

Name
LoadReport
DoSomethingInteresting
DoSomethingInteresting2
GoToCompanyURL
GoToAM_TicketOnGUI
OfferCreateUnitPageOnGUI
LoadData

Fig. 2: ConSol*CM Admin Tool - Parameters for a Customer Group

- **Name**
  The technical and unique name of the customer group. Click on 🌐 to enter the localized name of the customer group for all languages that are available in the system. The localized queue name will be displayed in the Web Client in the ticket header. If no localized values are provided, the name will be displayed in the default language.

- **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.
- **Contact actions**
  Part of the Action Framework.
- **Company actions**
  Part of the Action Framework.
- **CMPhone**
  Tab for all CM.Phone parameters. Only available if CM.Phone is active, see section CTI with ConSol*CM: CM.Phone.

10.2.4 Editing a Customer Group

If you want to edit a customer group, select it in the list and click on or just double-click the name of the customer group. Modify the customer group parameters and click *Save* to store your modifications.

10.2.5 Deleting a Customer Group

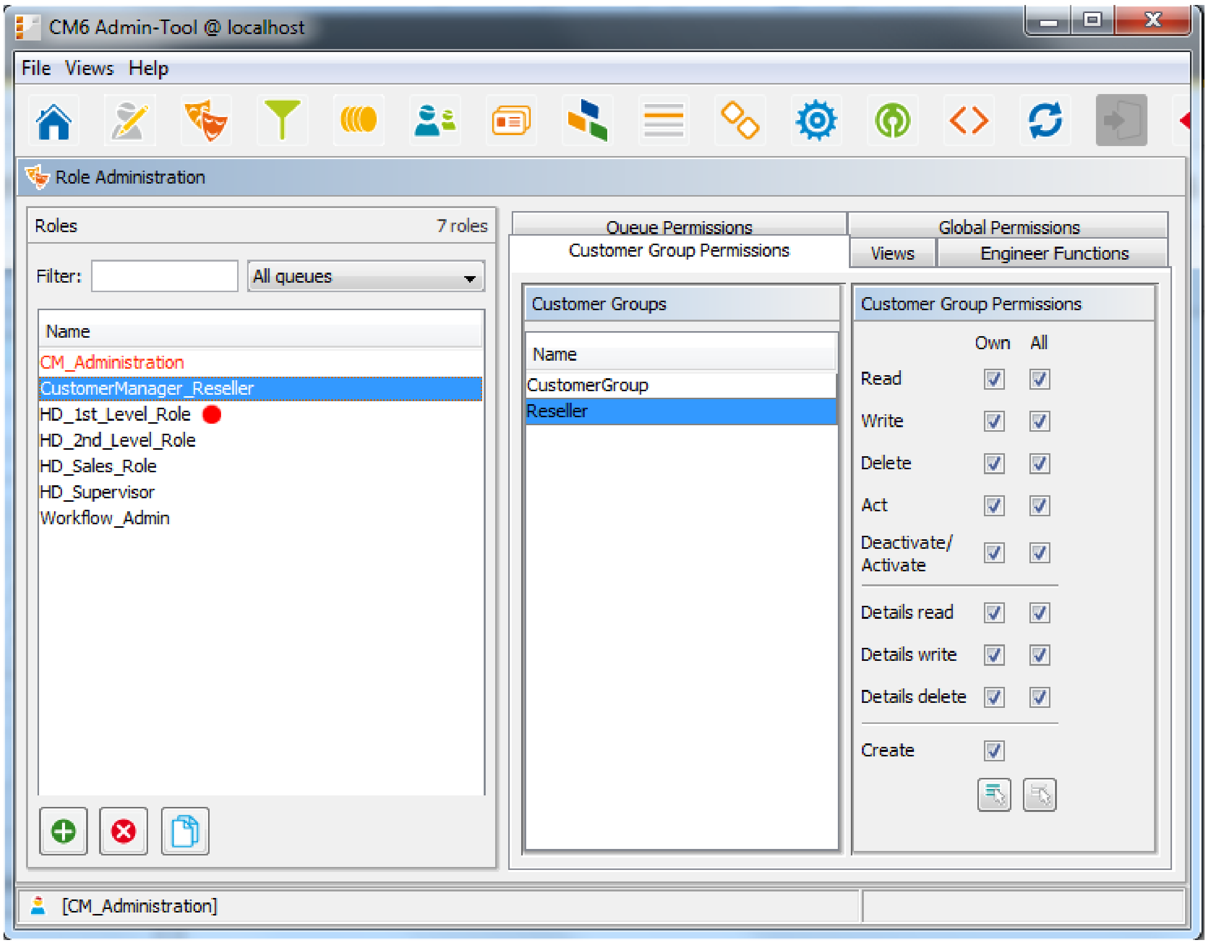
Select the customer group you want to delete in the list and click on . If you confirm the following dialog with *Yes*, the customer group will be deleted and is no longer available in the system. A customer group can only be deleted if it is not assigned to a queue and if there are no tickets for customers 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.

10.2.6 Disabling and (Re-)Enabling a Customer Group

To disable a customer group, select the customer group in the list and click on . The entry in the list is now shown in italics. Just click on at the bottom of the page, if you want to enable the customer group again. When 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.

10.3 Assigning Access Rights for Customer Groups

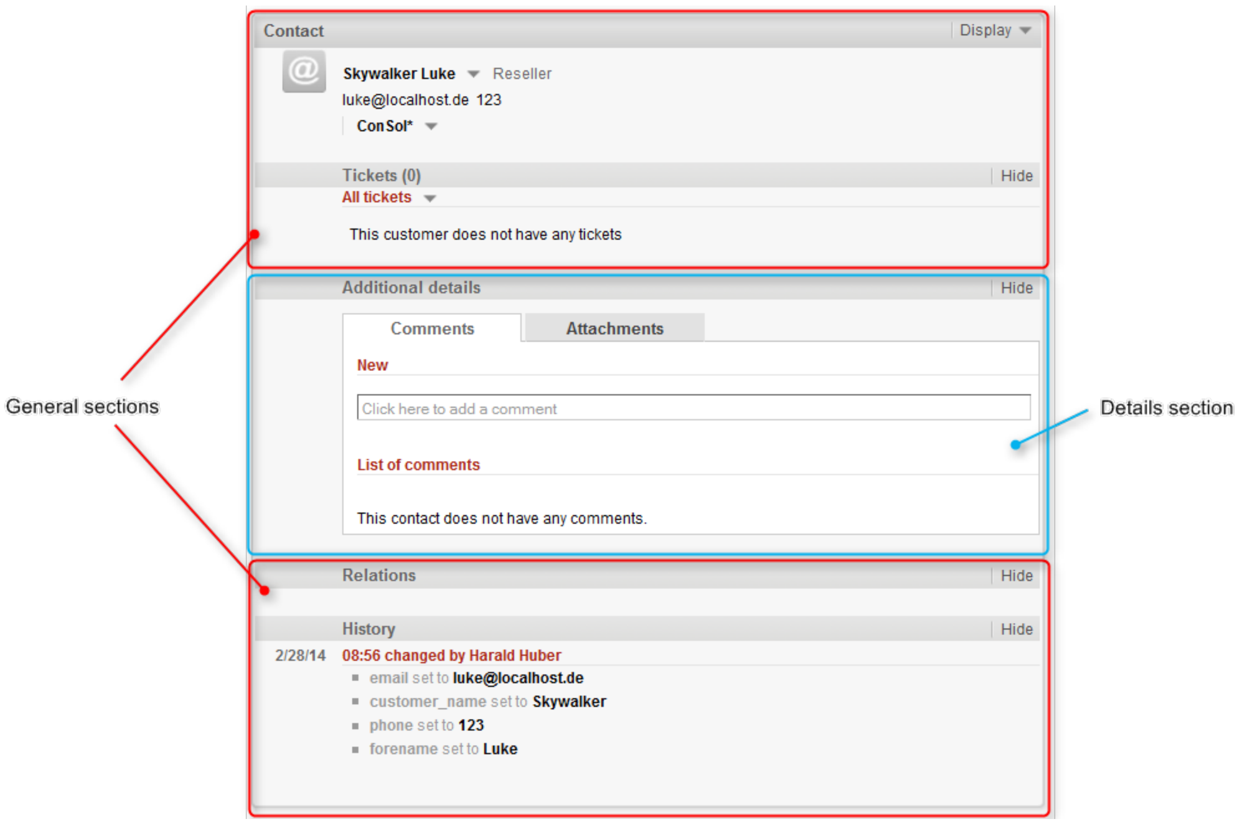
In order to let engineers work with customer data of a customer group, e.g. to create new reseller data sets or to modify them, you have to grant access permissions for the user groups to one or more roles.



	Own	All
Read	✓	✓
Write	✓	✓
Delete	✓	✓
Act	✓	✓
Deactivate/ Activate	✓	✓
Details read	✓	✓
Details write	✓	✓
Details delete	✓	✓
Create	✓	

Fig. 3: ConSol* Admin Tool - Assigning Permissions for Customer Groups to a Role

The access rights which can be granted have been modified compared to previous ConSol*CM versions. New rights have been added which concern a new section of the customer page. The customer page in the Web Client has a new section, the *Details* section.



This customer does not have any tickets

This contact does not have any comments.

2/28/14 08:56 changed by Harald Huber
- email set to luke@localhost.de
- customer_name set to Skywalker
- phone set to 123
- forename set to Luke

Fig. 4: ConSol*CM Web Client - Contact Page: Details Section

The following access permissions can be granted:

- **Customer type**
  Refers to the tickets of the customer.
  - **Own**
    All customers which are contacts at tickets which are currently assigned to the engineer.
  - **All**
    All customers.
- **General sections**
  - **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 for details about customer actions).
  - **Deactivate/activate**
    Deactivate and (re-)activate the customer or company. It is not possible to create tickets for a deactivated company.

- **Details section**
    - **Details read**
      Read customer data in the *Details* section.
    - **Details write**
      Write/modify customer data in the *Details* section.
    - **Details delete**
      Delete customer data in the *Details* section.
- **General**
    - **Create**
      Create a customer data set. In a two-level customer data model this refers to customer as well as to company data sets.

Attention:

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!

11 Customer (Data Object) Relations

- Introduction to Customer (Data Object) Relations
- Management of Customer Relations Using the Admin Tool
- Creating Customer Relations Using the Web Client
- Scripting Using Relations

11.1 Introduction to Customer (Data Object) Relations

Customer relations represent relations between customers (data objects), 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.

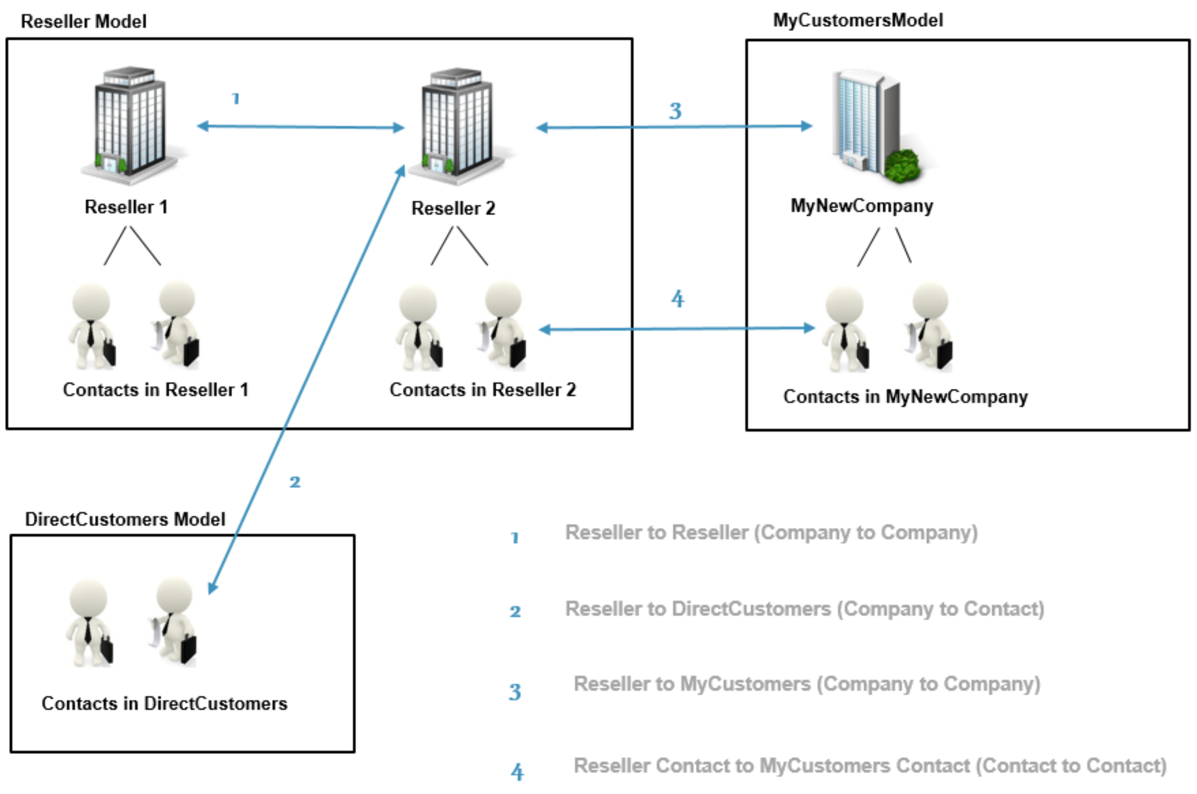
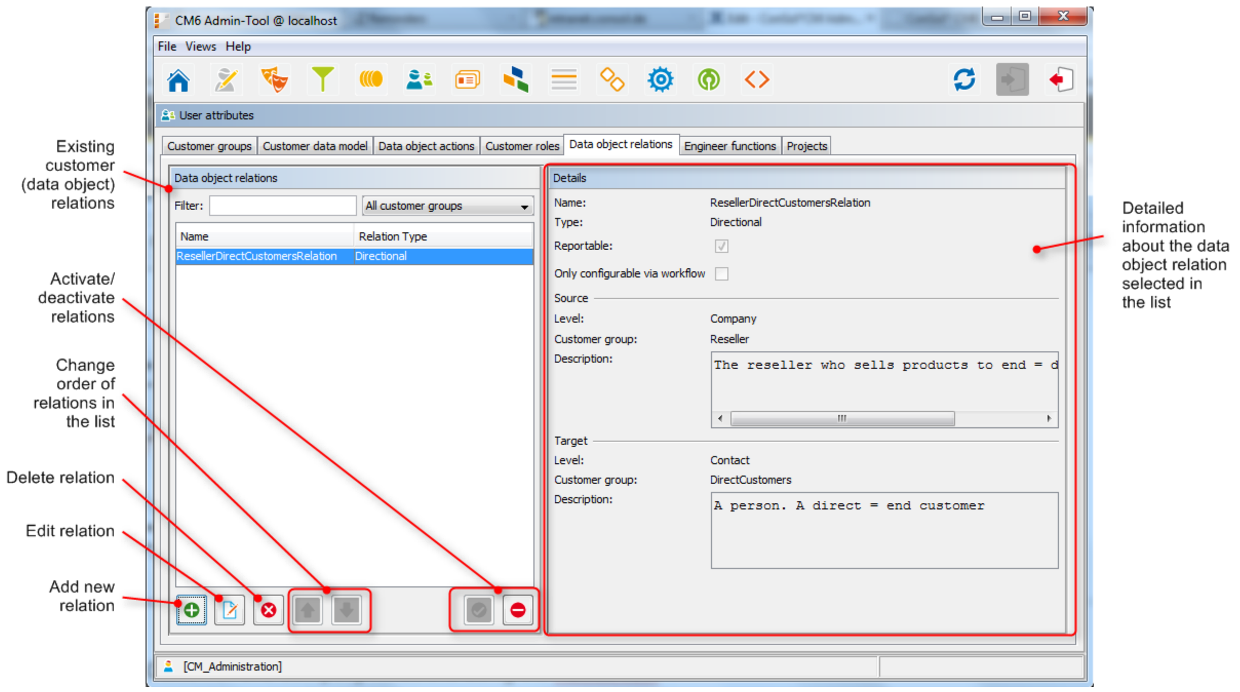

Reseller Model
MyCustomersModel
1
3
Reseller 1
Reseller 2
MyNewCompany
4
Contacts in Reseller 1
Contacts in Reseller 2
Contacts in MyNewCompany
2
DirectCustomers Model
Contacts in DirectCustomers
1 Reseller to Reseller (Company to Company)
2 Reseller to DirectCustomers (Company to Contact)
3 Reseller to MyCustomers (Company to Company)
4 Reseller Contact to MyCustomers Contact (Contact to Contact)

Fig. 1: ConSol*CM FlexCDM - Examples of Customer Relations

11.2 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 tab *Data object relations* in the *User Attributes* section. All relations are listed, new relations can be added, or old ones can be deleted.



Name	Relation Type
ResellerDirectCustomersRelation	Directional

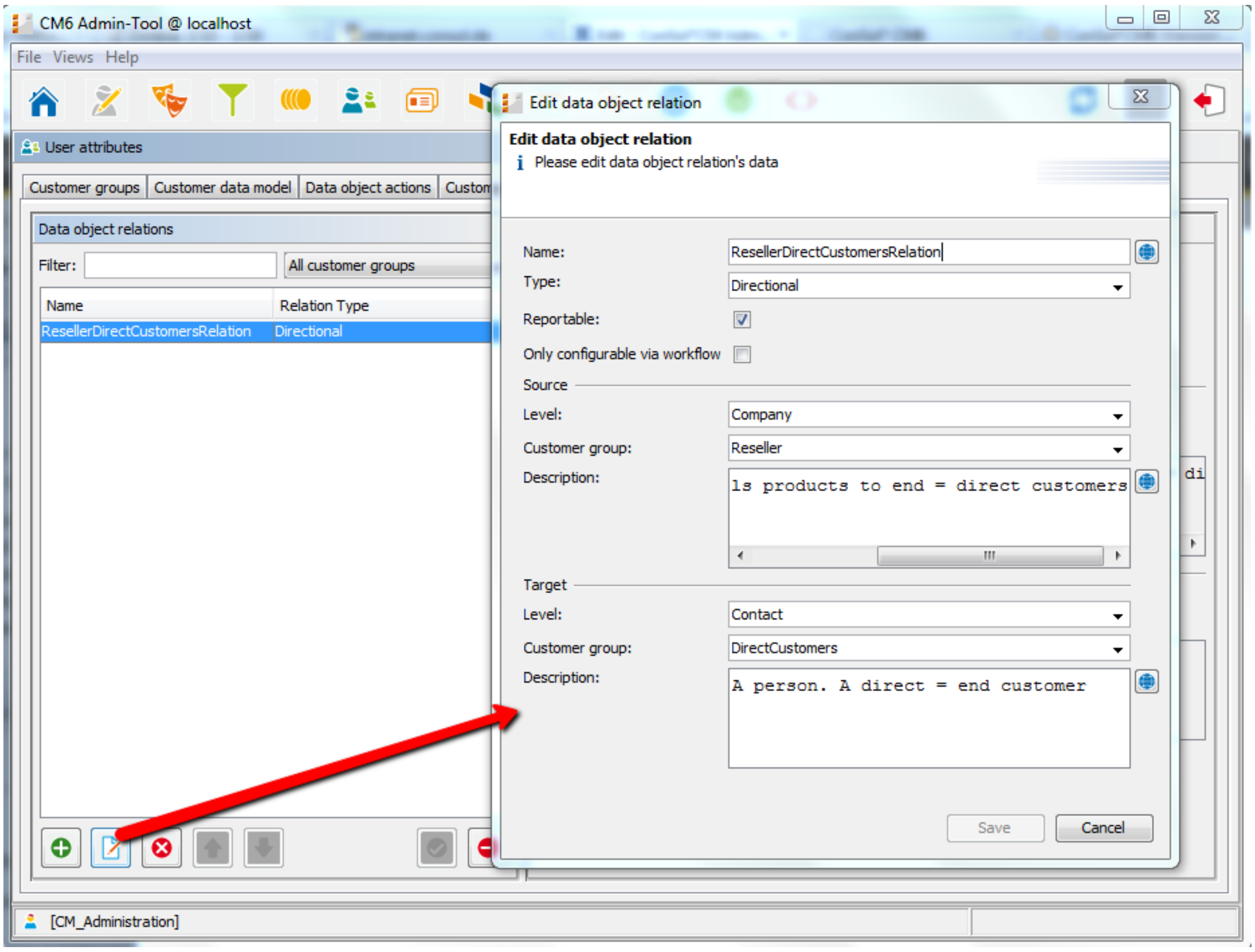
Name: ResellerDirectCustomersRelation
Type: Directional
Reportable: ✓
Only configurable via workflow: ☐
Source
Level: Company
Customer group: Reseller
Description: The reseller who sells products to end = d
Target
Level: Contact
Customer group: DirectCustomers
Description: A person. A direct = end customer

Fig. 2: ConSol*CM Admin Tool - Managing Customer Relations

The following elements are available:

- **List of relations**
- **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.
- **Add button**
  Add a new relation. The pop-up window *Create data object relation* with the details fields (see next section) is opened.
- **Edit button**
  Modify the parameters of a relation. The pop-up window *Edit data object relation* with the details fields (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).

- **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.



Name	Relation Type
ResellerDirectCustomersRelation	Directional

Name:	ResellerDirectCustomersRelation
Type:	Directional
Reportable:	☑
Only configurable via workflow	☐
Source	
Level:	Company
Customer group:	Reseller
Description:	ls products to end = direct customers
Target	
Level:	Contact
Customer group:	DirectCustomers
Description:	A person. A direct = end customer

Fig. 3: ConSol*CM Admin Tool - Details of a Customer Relation

To create a new relation, use the button, to edit a relation use the 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), the localized name will be displayed in the Web Client as for most fields in ConSol*CM.

- **Type**
  Select one of two types:
  - **Directional**
    A directional relation has a defined source and a defined target. A data object 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). Or a relation between two contacts of a company: *is boss of*. The other direction *works for* can also be used. However, a consistent structure for the entire system should be designed to avoid misunderstandings.
  - **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 check box is marked the relation is not available in the Web Client but can only be created via workflow scripts. Therefore 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 data object.
    - **Description**
      Will be displayed in the Web Client as 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 data object.
    - **Description**
      Will be displayed in the Web Client as description of the relation on the target side.

11.3 Creating Customer Relations Using the Web Client

In spite of this book 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 data object to another in the *Relations* section of the source data object. 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 data object-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 products to* from a company in the *Reseller* customer group to a contact in the *Direct 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 press *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 the work with customer relations.

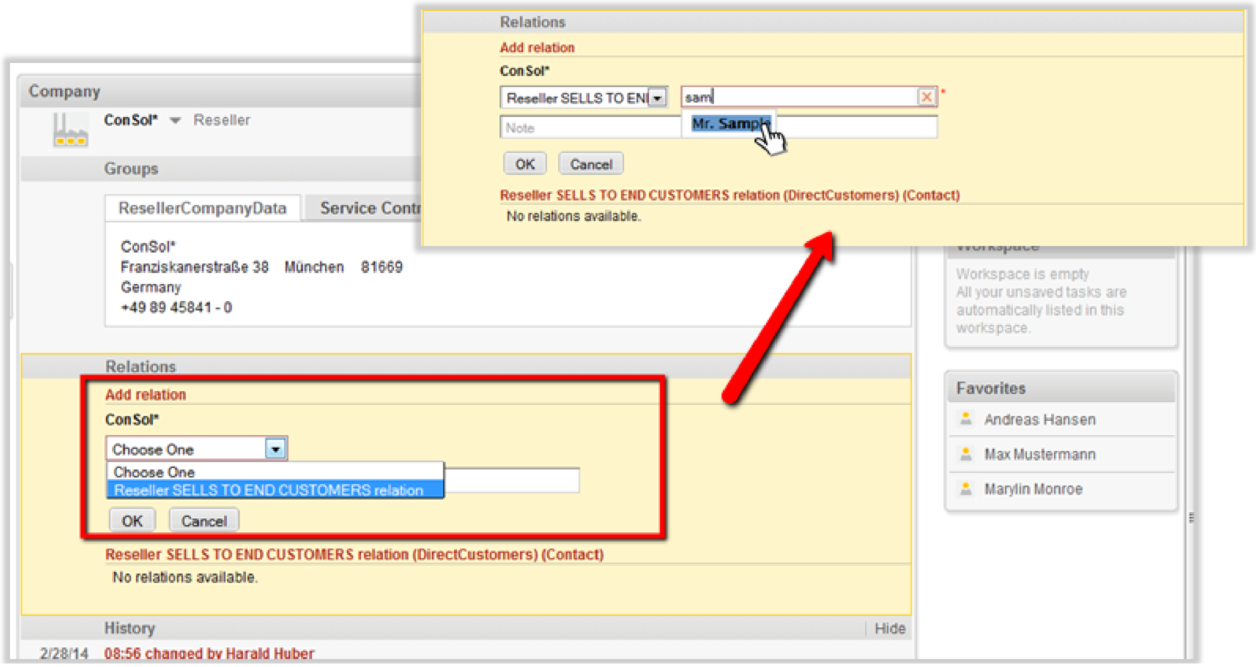


Fig. 4: ConSol*CM Web Client - Setting a Relation

11.4 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 *data object* and *data object definition*. However, the names of the corresponding Java classes are *Unit* and *UnitDefinition*. All other Java classes which deal with customer data objects also are still named *Unit..*. Please keep that in mind when you work on the administrator level as well as on the programmer's level. Please refer to the 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.

12 Action Framework

- Introduction to Data Object Actions
- Managing Data Object Actions Using the Admin Tool
    - Step 1: Write the Data Object Action Script
    - Step 2: Create Data Object Action(s) Which Use the Script
    - Step 3: Assign Data Object Action(s) to Customer Group(s)
- Using Data Object Actions as an Engineer (User)
- Examples for Data Object Action Scripts
    - Example 1: Simple Manual Action
    - Example 2: New Ticket for Contact
- Scripts for the Action Framework: Programming Data Object Actions
    - Data Object Action Scripts
        - Automatic Data Object Action Scripts
        - Manual Data Object Action Scripts
            - Create a Unit
            - Create a Ticket
            - Open a Unit Page
            - Open a Ticket Page
            - Open a Web Page
            - Object UnitActionScriptResult
    - Data Object Condition Scripts

12.1 Introduction to Data Object Actions

Data object actions are actions which can be performed for a data object, i.e. a contact or a company. 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 data object actions for use cases like the following:

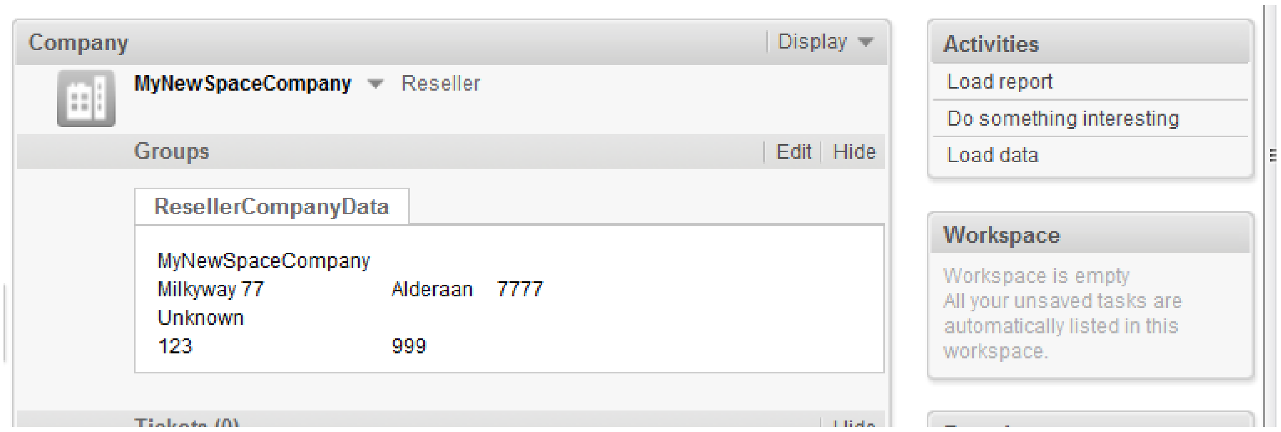
- Load additional data into a company's data set.
- Build an automatic report about the company-specific KPIs.
- Transfer ConSol*CM data to another system (e.g. an ERP system).
- Create/update a Google Maps link from the address data.

You can use the following types of data object actions:

- **Automatic** actions which are performed by the system after one of the following data object operations:
  - CREATE
  - UPDATE
  - DELETE
- **Manual** actions which are performed by the engineer using *Activities* links in the data object page (*Company* or *Contact* page) of the Web Client (similar to *Workflow activities* for tickets). Manual actions are executed for the data object which is 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 data object actions, i.e. only then the *Activities* will be displayed in the Web Client:

- Act



MyNewSpaceCompany
Milkyway 77 Alderaan 7777
Unknown
123 999

Fig. 1: ConSol*CM Web Client - Example for Manual Data Object Activities

Data object actions are defined as Groovy scripts which are stored in the *Script and Template* section of the Admin Tool.

The execution of data object actions can be controlled using condition scripts, i.e. you can implement a condition script which is executed before the data object action itself. Only when this script returns *true*, the following action script is executed.

So there are two types of scripts you have to deal with when you use the ConSol*CM Action Framework:

- **Data object action scripts**
  Defines the action which should be performed.
- **Data object 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 use a data object action you have to proceed in three steps:

1. Create data object action script (either action script only or action script and condition script).
2. Create the data object action(s) which use(s) the script(s).
3. Assign the data object action(s) to the customer group(s) where they should be available.

In the following sections, all three steps are explained in detail.

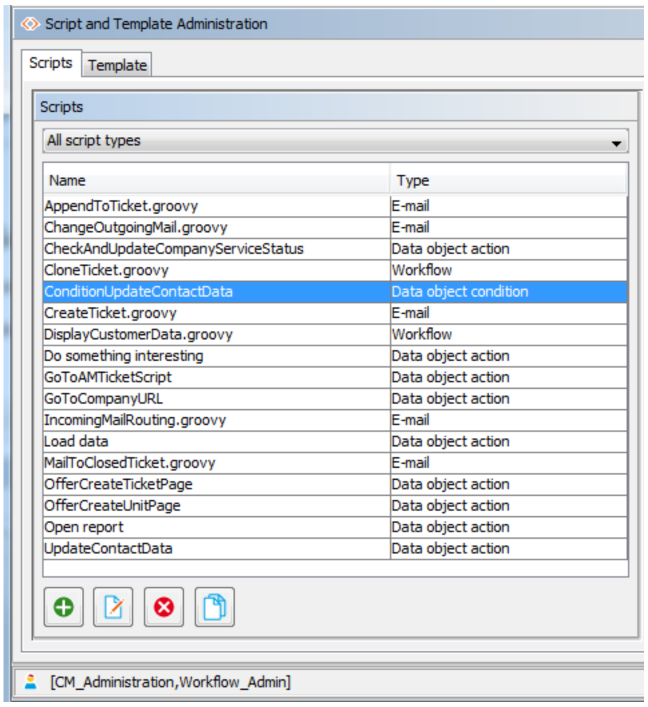
12.2 Managing Data Object Actions Using the Admin Tool

In this book we will use the terms *data object* and *data object definition*. However, the names of the corresponding Java classes are *Unit* and *UnitDefinition*. All other Java classes which deal with customer data objects are also still named *Unit..*. Please keep that in mind when you work on the administrator level as well as on the programmer's level. Please refer to the the *ConSol**CM Java API Doc* for details.

12.2.1 Step 1: Write the Data Object Action Script

Create a new Admin Tool script of type *Data object action*. If required, create another script of type *Data object 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 data object actions, please read section Scripts for the Action Framework in this chapter.



Name	Type
AppendToTicket.groovy	E-mail
ChangeOutgoingMail.groovy	E-mail
CheckAndUpdateCompanyServiceStatus	Data object action
CloneTicket.groovy	Workflow
ConditionUpdateContactData	Data object condition
CreateTicket.groovy	E-mail
DisplayCustomerData.groovy	Workflow
Do something interesting	Data object action
GoToAMTicketScript	Data object action
GoToCompanyURL	Data object action
IncomingMailRouting.groovy	E-mail
Load data	Data object action
MailToClosedTicket.groovy	E-mail
OfferCreateTicketPage	Data object action
OfferCreateUnitPage	Data object action
Open report	Data object action
UpdateContactData	Data object action

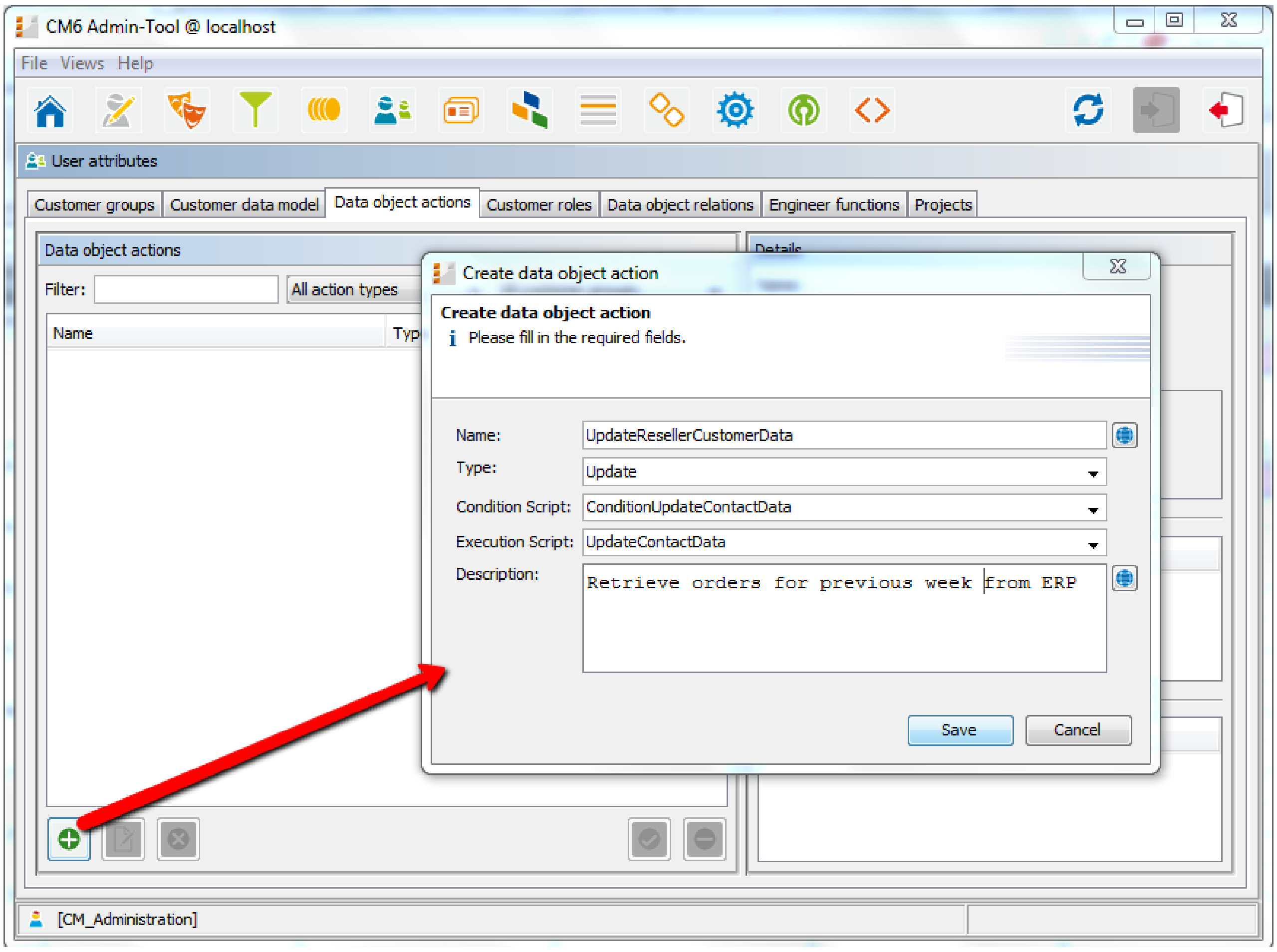
Fig. 2: ConSol*CM Admin Tool - Scripts for Data Object Actions

Example data object action script:

```
//do something with the unit (data object)
unit.setFieldValue("personalData", "name", "Kowalski")
unitServiceImpl.update(unit);
//create and return action result that will tell the web to create new ticket with unit as
//a main contact
def queueId = queueService.getByName("Helpdesk").getId();
Map<String, Object> valuesMap = new HashMap<String, Object>
valuesMap.put(PostActionParameter.UNIT_ID, unit.getId())
valuesMap.put(PostActionParameter.QUEUE_ID, queueId)
return unitActionScriptResultFactory.getPostAction(PostActionType.CREATE_TICKET, valuesMap)
```

12.2.2 Step 2: Create Data Object Action(s) Which Use the Script

Open the tab *Data object actions* under *User attributes* in the Admin Tool and add a new action using the *Plus* button .



Create data object action

Please fill in the required fields.

Name: UpdateResellerCustomerData
Type: Update
Condition Script: ConditionUpdateContactData
Execution Script: UpdateContactData
Description: Retrieve orders for previous week from ERP

Fig. 3: ConSol*CM Admin Tool - Creating Data Object Actions

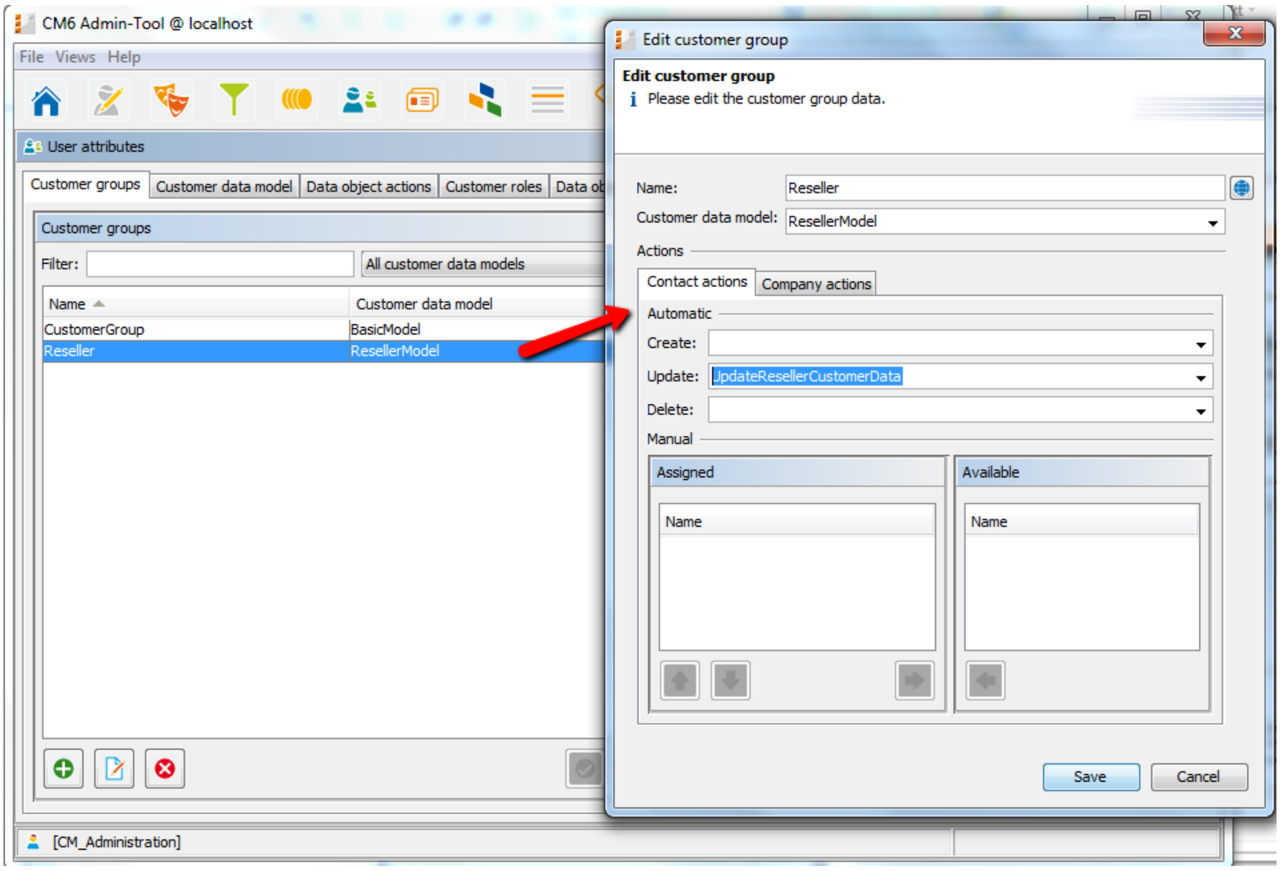
In the pop-up window, the parameters for the new action have to be defined:

- **Name**
  The unique technical name of the action. Can be localized using the *Localize* button .The localization is required for manual actions, because the localized name is displayed under *Activities* in the Web Client.
- **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.
  - **Delete**
    This script will be executed automatically when the contact/company is deleted.
  - **Manual**
    This script will be offered on the contact/company page as manual activity.
- **Condition Script**
  In case a condition script should be executed before the action script, the name of the condition script has to be entered here. Only when the condition script has returned *true*, the action script will be executed. If there is no condition, just leave this field empty.
- **Execution Script**
  The name of the action script which should be executed. This has to be the exact name under which the script is stored in the *Script and Template* 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 customer groups. Please see following step.

12.2.3 Step 3: Assign Data Object Action(s) to Customer Group (s)

Open the tab *Customer groups* in the *User attributes* section of the Admin Tool. Select the customer group you would like to edit and click on to open the pop-up window to assign the data object actions. Only data object actions which have been stored as Admin Tool scripts will be offered here.



Customer groups

Filter: | All customer data models

Name	Customer data model
CustomerGroup	BasicModel
Reseller	ResellerModel

Name:	Reseller
Customer data model:	ResellerModel
Create:	
Update:	UpdateResellerCustomerData
Delete:	

Fig. 4: ConSol*CM Admin Tool - Assigning Data Object Actions to a Customer Group

You can define the following action types:

- **Contact actions**
  The script will be executed for the contact. Manual actions are offered on the contact page only.
- **Company actions**
  The script will be executed for the company. Manual actions are offered on the company page only.

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.
- **Delete**
  This script will be executed automatically when the contact/company is deleted.
- **Manual**
  This script will be offered on the contact/company page as manual activity.

12.3 Using Data Object Actions as an Engineer (User)

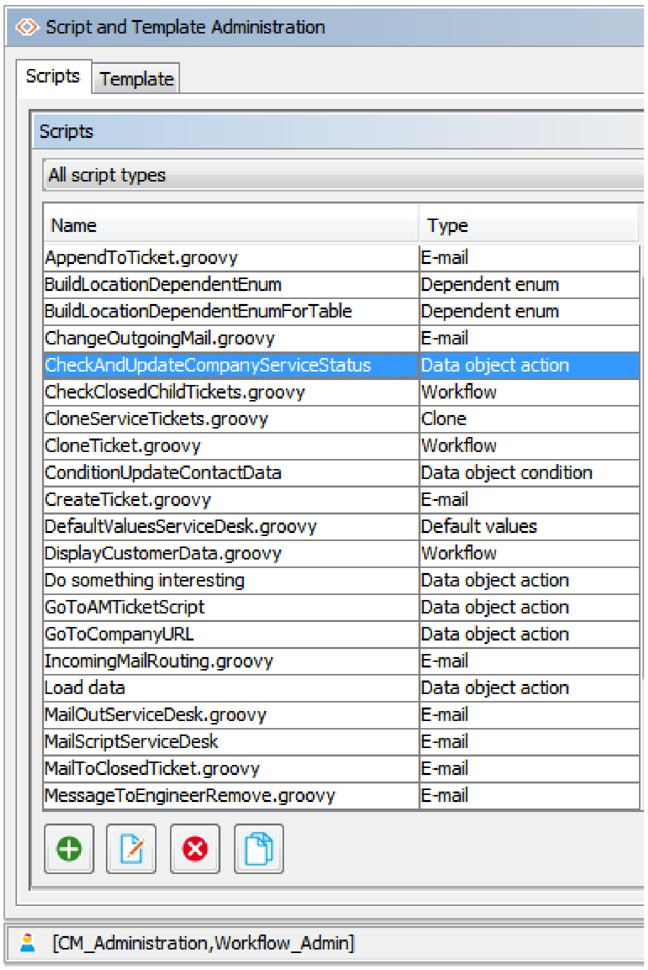
As an engineer (user), only the data object action type *manual* is relevant for you. The *DELETE*, *UPDATE*, and *CREATE* scripts run in the background.

Manual actions are offered in the Web Client similar to workflow activities for a ticket. Please see *Example 1* in the next section.

12.4 Examples for Data Object Action Scripts

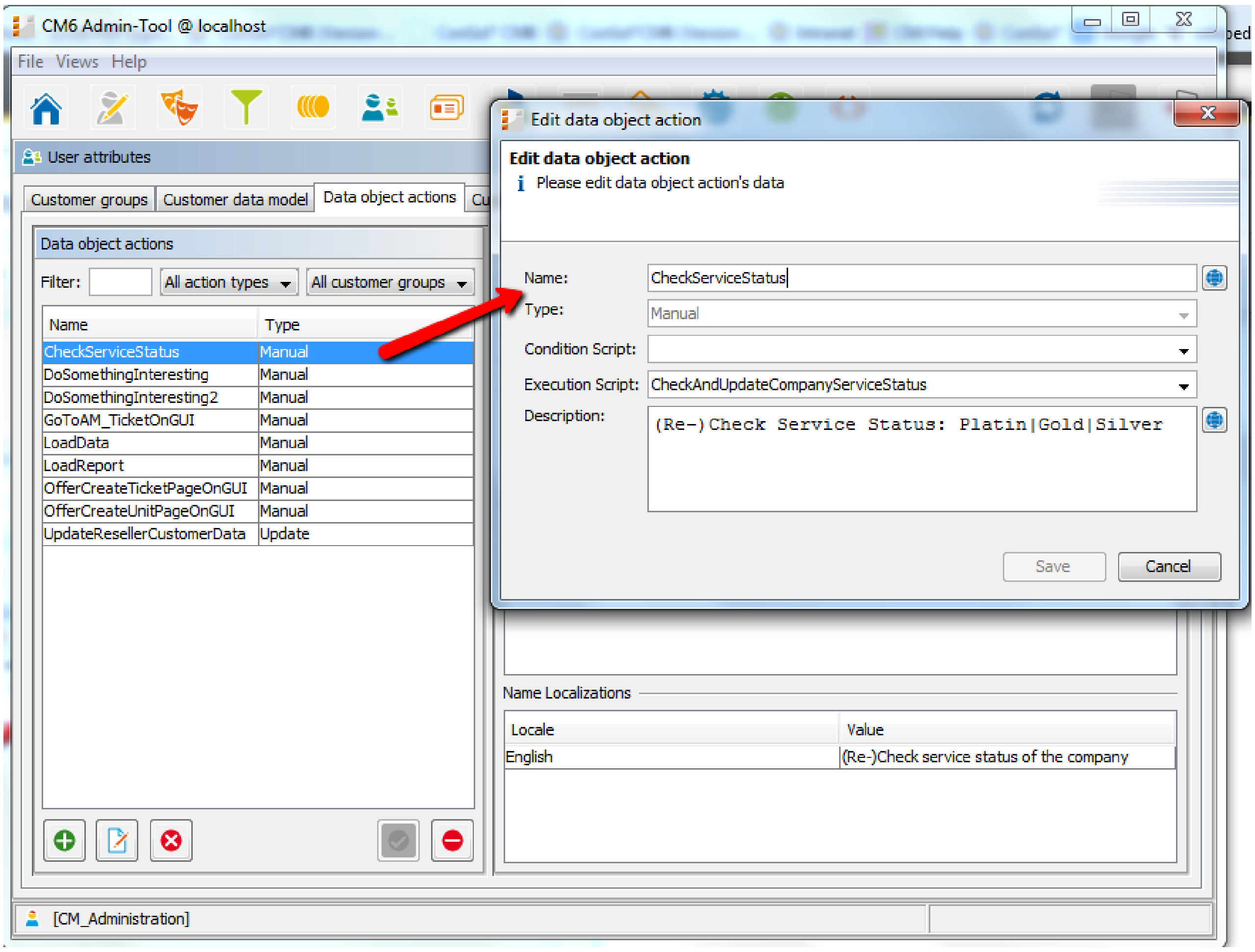
12.4.1 Example 1: Simple Manual Action

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.



Name	Type
AppendToTicket.groovy	E-mail
BuildLocationDependentEnum	Dependent enum
BuildLocationDependentEnumForTable	Dependent enum
ChangeOutgoingMail.groovy	E-mail
CheckAndUpdateCompanyServiceStatus	Data object action
CheckClosedChildTickets.groovy	Workflow
CloneServiceTickets.groovy	Clone
CloneTicket.groovy	Workflow
ConditionUpdateContactData	Data object condition
CreateTicket.groovy	E-mail
DefaultValuesServiceDesk.groovy	Default values
DisplayCustomerData.groovy	Workflow
Do something interesting	Data object action
GoToAMTicketScript	Data object action
GoToCompanyURL	Data object action
IncomingMailRouting.groovy	E-mail
Load data	Data object action
MailOutServiceDesk.groovy	E-mail
MailScriptServiceDesk	E-mail
MailToClosedTicket.groovy	E-mail
MessageToEngineerRemove.groovy	E-mail

Fig. 5: ConSol*CM Admin Tool - Data Object Action Script to Be Used as Company Script



Edit data object action

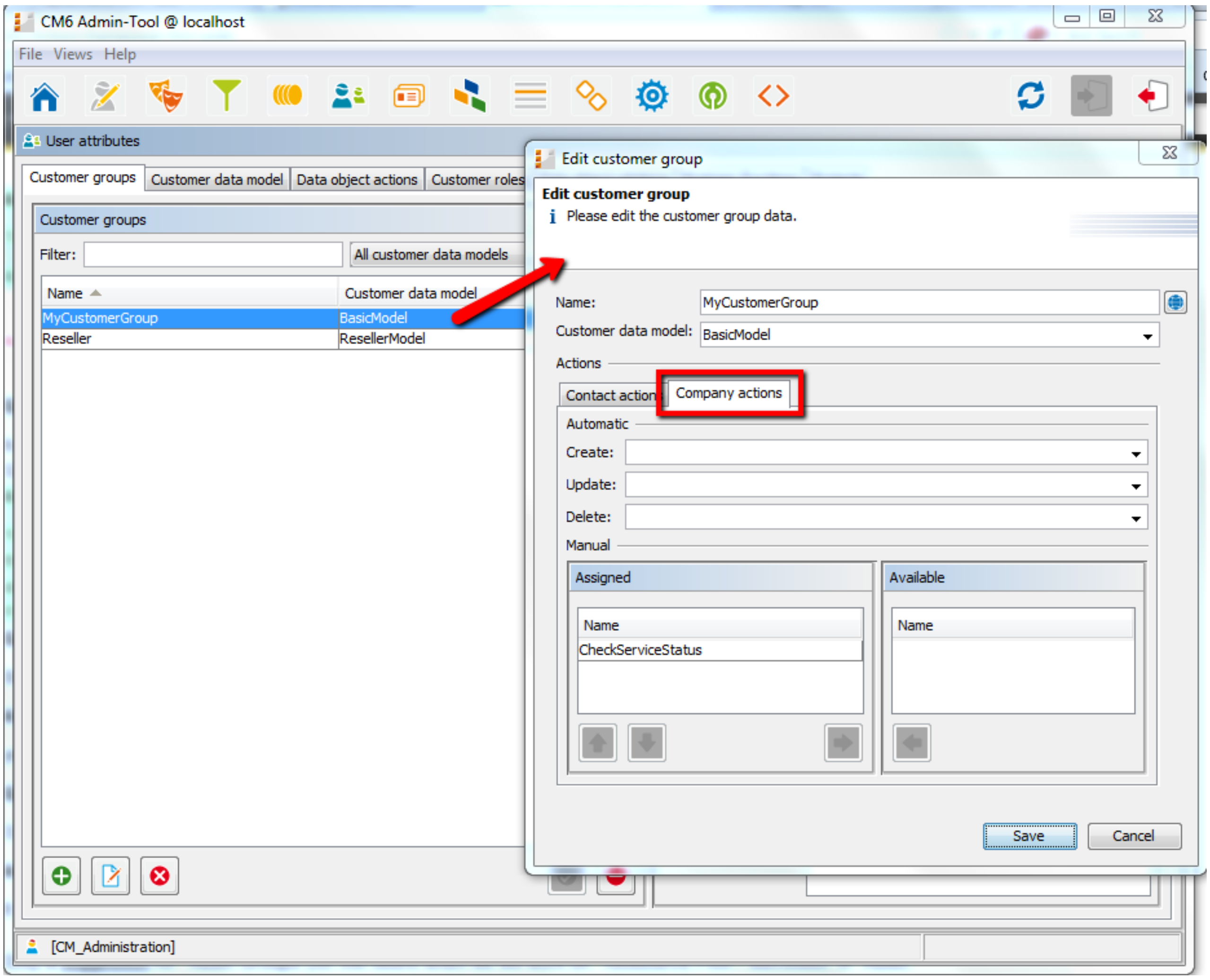
Please edit data object action's data

Name	Type
CheckServiceStatus	Manual
DoSomethingInteresting	Manual
DoSomethingInteresting2	Manual
GoToAM_TicketOnGUI	Manual
LoadData	Manual
LoadReport	Manual
OfferCreateTicketPageOnGUI	Manual
OfferCreateUnitPageOnGUI	Manual
UpdateResellerCustomerData	Update

Name:	CheckServiceStatus
Type:	Manual
Condition Script:	
Execution Script:	CheckAndUpdateCompanyServiceStatus
Description:	(Re-)Check Service Status: Platin|Gold|Silver

Locale	Value
English	(Re-)Check service status of the company

Fig. 6: ConSol*CM Admin Tool - Defining the Company Action



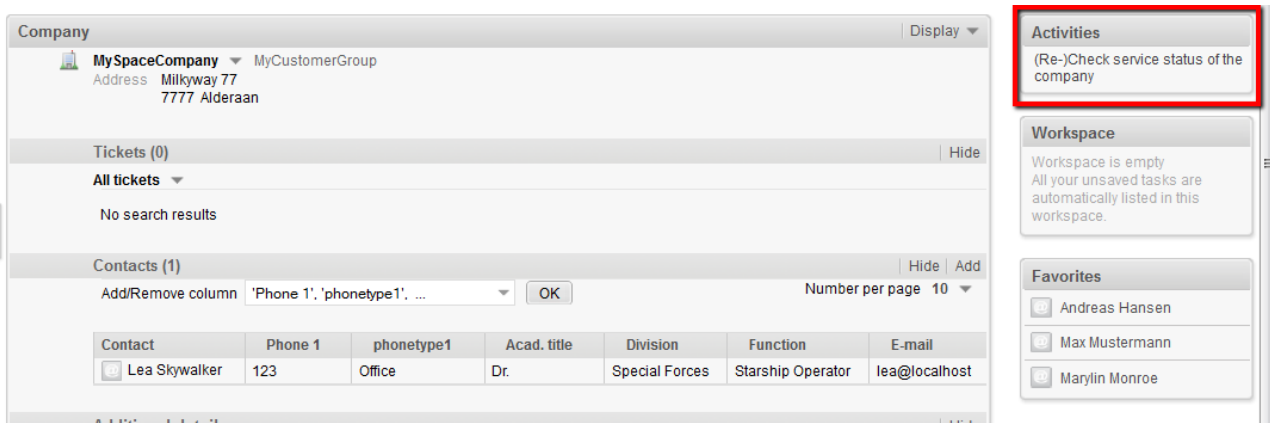
Name	Customer data model
MyCustomerGroup	BasicModel
Reseller	ResellerModel

Name: MyCustomerGroup
Customer data model: BasicModel
Create:
Update:
Delete:

Name
CheckServiceStatus

Fig. 7: ConSol*CM Admin Tool - Assigning a Company Action

The engineer can use the action manually in the Web Client:



MySpaceCompany MyCustomerGroup
Address Milkyway 77
7777 Alderaan

Tickets (0)
All tickets
No search results

Contacts (1)
Add/Remove column 'Phone 1', 'phonetype1', ...
Number per page 10

Contact	Phone 1	phonetype1	Acad. title	Division	Function	E-mail
Lea Skywalker	123	Office	Dr.	Special Forces	Starship Operator	lea@localhost

(Re-)Check service status of the company

Workspace is empty
All your unsaved tasks are automatically listed in this workspace.

Andreas Hansen
Max Mustermann
Marylin Monroe

Fig. 8: ConSol*CM Web Client - Using a Manual Company Action

12.4.2 Example 2: New Ticket for Contact

This script opens the *Create ticket* page for the contact from which the action has been performed. The target queue is *Reminders*. That way a new reminder ticket can be created in no time for the open contact. For an introduction to Admin Tool scripts for the Action Framework, please read the following section.

Example contact script:

```
import com.consol.cmas.common.model.scripting.unit.PostActionParameterimport
import com.consol.cmas.core.server.service.UnitActionScriptResultFactoryimport
import com.consol.cmas.common.model.scripting.unit.PostActionType
def queueId = queueService.getByName("Reminders").getId();
Map<String, Object> valuesMap = new HashMap<String, Object>()
valuesMap.put(PostActionParameter.UNIT_ID, unit.getId())
valuesMap.put(PostActionParameter.QUEUE_ID, queueId)
return unitActionScriptResultFactory.getPostAction("createTicket", valuesMap)
```

12.5 Scripts for the Action Framework: Programming Data Object Actions

Data object actions are defined by Admin Tool scripts, i.e. by Groovy scripts which are stored in the *Script and Template* 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:

- Data object action scripts
- Data object condition scripts

12.5.1 Data Object Action Scripts

The actions in this script are executed either triggered automatically by the system operations *CREATE*, *UPDATE*, or *DELETE* or by a manual action (using *Activities* in the Web Client) of the engineer.

Automatic Data Object Action Scripts

Example script: Set a value in customer data and update the unit

```
unit.setFieldValue("personalData", "name", "Skywalker")
unitService.update(unit)
```

Attention:

When you use *unitService.update(unit)* as in the example above, you can use a data object condition script to avoid infinite loops. See note in section *Data Object Condition Scripts*.

Manual Data Object Action Scripts

For manual data object action scripts you can make use of some specific methods and objects:

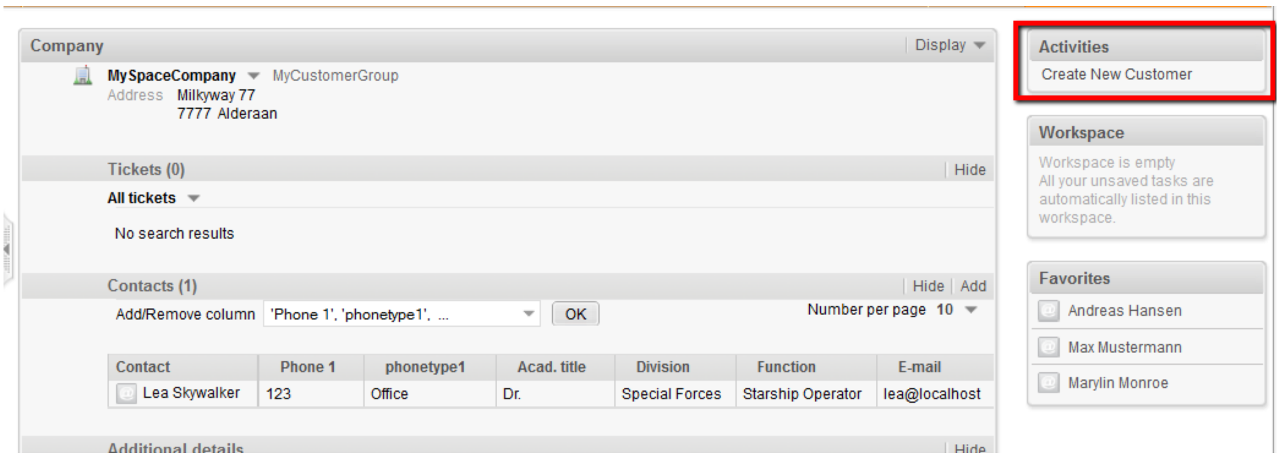
- **Methods (fields of the Interface PostActionType):**
  - **CREATE_UNIT**
    Create a unit.
  - **CREATE_TICKET**
    Create a ticket.
  - **GOTO_UNIT**
    Open unit page.
  - **GOTO_TICKET**
    Open ticket page.
  - **GOTO_PAGE**
    Open a web page (URL).
- **Objects:**
  - UnitActionScriptResult

Create a Unit

(*PostActionType.CREATE_UNIT*) redirects the user to the create unit page. It uses the optional parameter *PostActionParameter.CUSTOMER_GROUP_ID* to decide for which customer group a new unit has to be created and optionally the map of data object group fields (*PostActionParameter.FIELDS_MAP*) to fill the unit's data object group fields with the passed values.

Example company action script which fills some unit data

```
import com.consol.cmas.common.model.customfield.meta.FieldKey
import com.consol.cmas.common.model.customfield.AbstractField
import com.consol.cmas.common.model.customfield.StringField
import com.consol.cmas.common.model.scripting.unit.PostActionParameter
import com.consol.cmas.common.model.scripting.unit.PostActionType
Map<FieldKey, AbstractField<?>> fieldsMap = new HashMap<FieldKey, AbstractField<?>>()
FieldKey firstName = new FieldKey("customer", "firstname")
FieldKey name = new FieldKey("customer", "name")
fieldsMap.put(firstName, new StringField(firstName, "Han"))
fieldsMap.put(name, new StringField(name, "Solo"))
Map<String, Object> valuesMap = new HashMap<String, Object>()
valuesMap.put(PostActionParameter.CUSTOMER_GROUP_ID, unit.getCustomerGroup().getId())
valuesMap.put(PostActionParameter.FIELDS_MAP, fieldsMap)
return unitActionScriptResultFactory.getPostAction(PostActionType.CREATE_UNIT, valuesMap)
```



MySpaceCompany MyCustomerGroup
Address Milkyway 77
7777 Alderaan

No search results

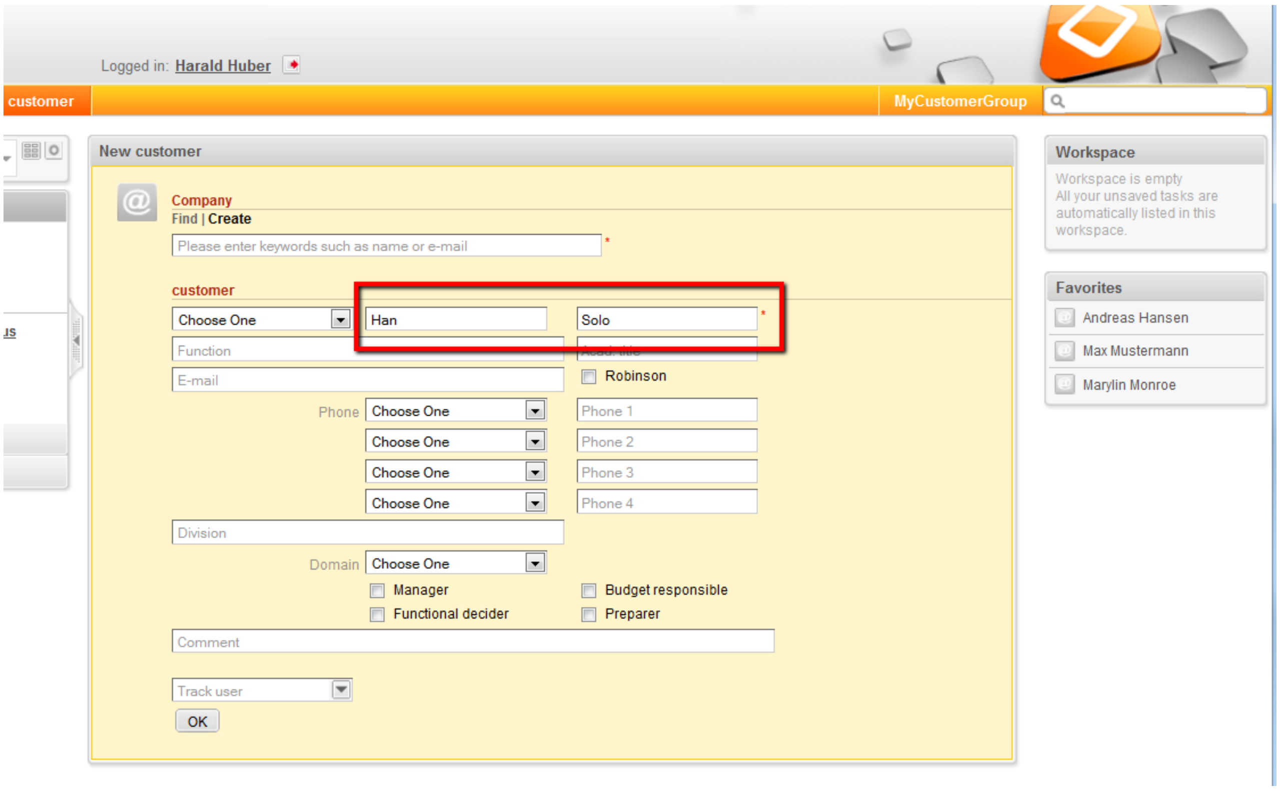
'Phone 1', 'phonetype1', ...

Contact	Phone 1	phonetype1	Acad. title	Division	Function	E-mail
Lea Skywalker	123	Office	Dr.	Special Forces	Starship Operator	lea@localhost

Workspace is empty
All your unsaved tasks are automatically listed in this workspace.

Andreas Hansen
Max Mustermann
Marylin Monroe

Fig. 9: ConSol*CM Web Client - Company Action Script


Logged in: Harald Huber

customer | MyCustomerGroup

New customer

Company
Find | Create

Please enter keywords such as name or e-mail

customer

Choose One	Han	Solo
Function		Acad. title
E-mail		Robinson
Phone	Choose One	Phone 1
	Choose One	Phone 2
	Choose One	Phone 3
	Choose One	Phone 4
Division		
Domain	Choose One	
	Manager	Budget responsible
	Functional decider	Preparer
Comment		
Track user		

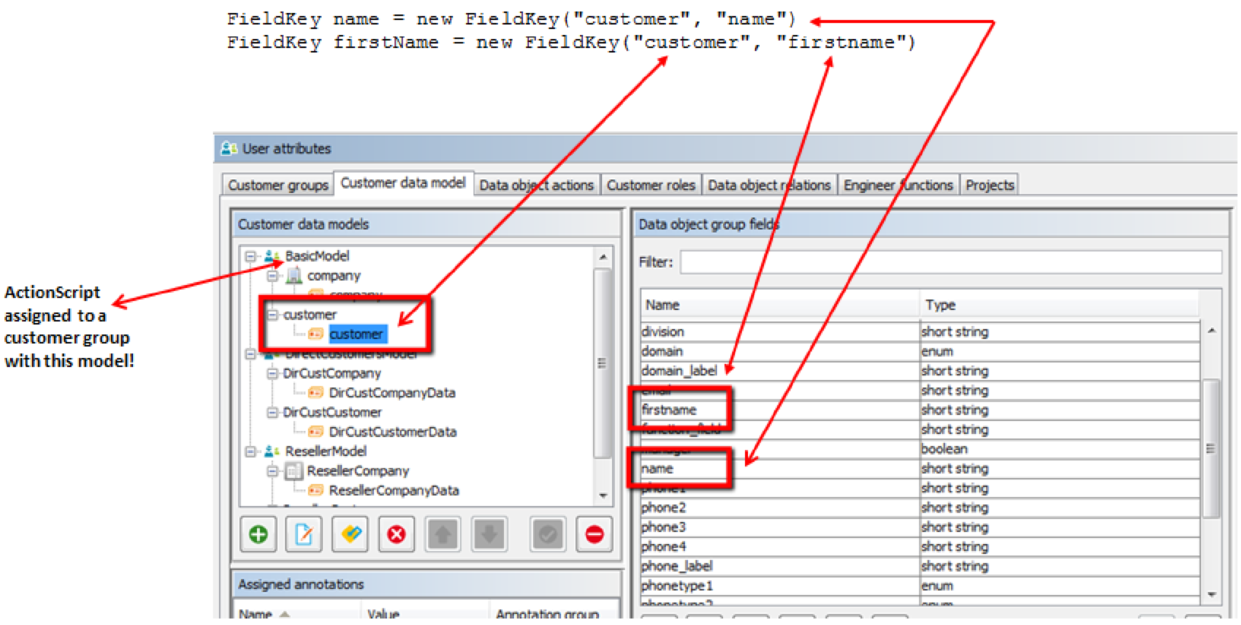
OK

Workspace
Workspace is empty
All your unsaved tasks are automatically listed in this workspace.

Favorites
Andreas Hansen
Max Mustermann
Marylin Monroe

Fig. 10: ConSol*CM Web Client - Create Unit Page Opened and Pre-Filled by Company Action Script

Of course, the names of the data object group 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.


```
FieldKey name = new FieldKey("customer", "name")
FieldKey firstName = new FieldKey("customer", "firstname")
```

Name	Type
division	short string
domain	enum
domain_label	short string
email	short string
firstname	short string
function_field	short string
manager	boolean
name	short string
phone1	short string
phone2	short string
phone3	short string
phone4	short string
phone_label	short string
phonetype1	enum
phonetype2	enum

Fig. 11: ConSol*CM Admin Tool - Script for Creating Unit Page as Company Action

Create a Ticket

(*PostActionType.CREATE_TICKET*) redirects the user to a create ticket page. It uses the optional *PostActionParameter.UNIT_ID* with the ID of the main contact, *PostActionParameter.QUEUE_ID* with the ID of the queue, and the custom fields map *PostActionParameter.FIELDS_MAP*.

Script creates and returns action result that will tell the web to create a new ticket with unit as the main contact

```
import com.consol.cmas.common.model.scripting.unit.PostActionType
import com.consol.cmas.common.model.scripting.unit.PostActionParameter
def queueId = queueService.getByName("HelpDesk_1st_Level").getId()
Map<String, Object> valuesMap = new HashMap<String, Object>()
valuesMap.put(PostActionParameter.UNIT_ID, unit.getId())
valuesMap.put(PostActionParameter.QUEUE_ID, queueId)
return unitActionScriptResultFactory.getPostAction(PostActionType.CREATE_TICKET, valuesMap)
```

Attention:

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 (*HelpDesk_1st_Level* in the example).

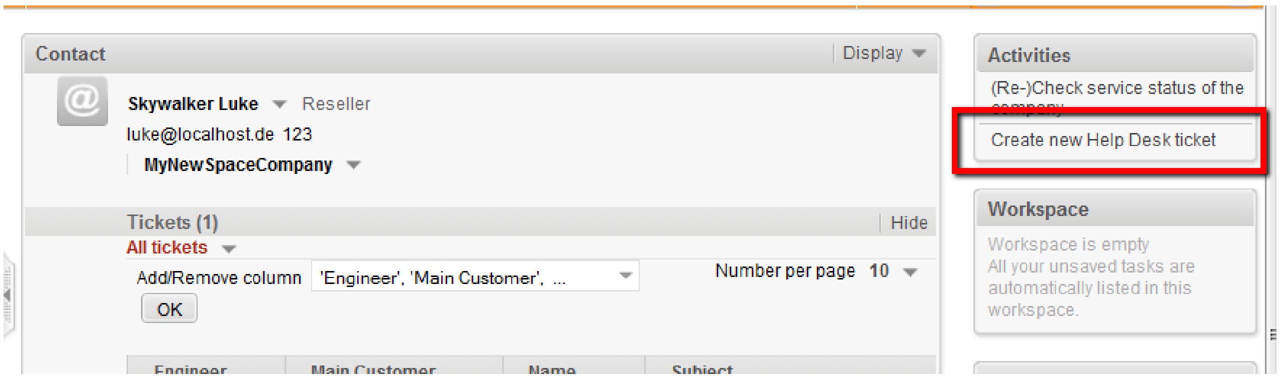


Fig. 12: ConSol*CM Web Client - Example Contact Action Script

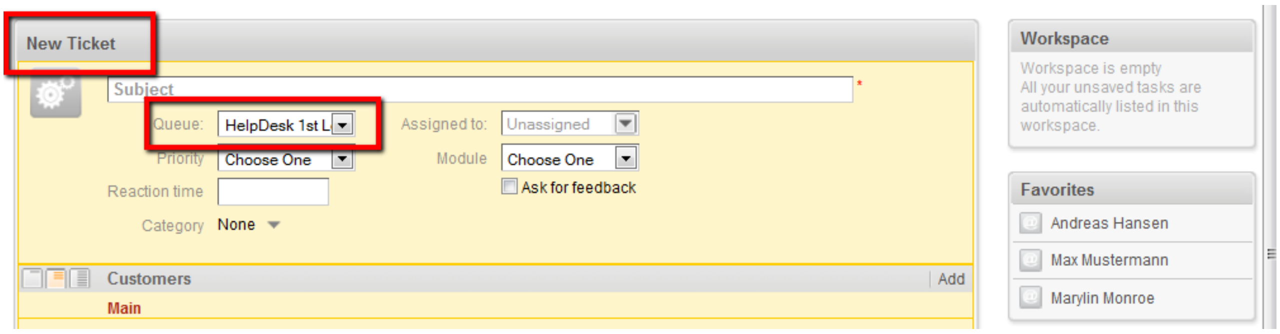


Fig. 13: ConSol*CM Web Client - Create Ticket Page Opened and Pre-Filled by Contact Action Script

Open a Unit Page

(*PostActionType.GOTO_UNIT*) redirects to a unit page. It uses the obligatory parameter *PostActionParameter.UNIT_ID* with the ID of the unit.

Go to company page

```
import com.consol.cmas.common.model.scripting.unit.PostActionType
import com.consol.cmas.common.model.scripting.unit.PostActionParameter
Map<String, Object> valuesMap = new HashMap<String, Object>()
valuesMap.put(PostActionParameter.UNIT_ID, unit.get("company()").getId())
return unitActionScriptResultFactory.getPostAction(PostActionType.GOTO_UNIT, valuesMap)
```

Open a Ticket Page

(*PostActionType.GOTO_TICKET*) redirects to a ticket page. It uses the obligatory parameter *PostActionParameter.TICKET_ID* with the ID of the ticket.

Go to ticket page

```
import com.consol.cmas.common.model.scripting.unit.PostActionType
import com.consol.cmas.common.model.scripting.unit.PostActionParameter
import com.consol.cmas.common.model.customfield.Unit
import com.consol.cmas.common.model.ticket.TicketCriteria
import com.consol.cmas.common.model.customfield.ListField
import com.consol.cmas.common.model.customfield.ContactReferenceField
import com.consol.cmas.common.model.customfield.UnitReferenceSearchField
import com.consol.cmas.common.model.customfield.ContactReferenceSearchField
import com.consol.cmas.common.model.customfield.meta.FieldKey
import com.consol.cmas.common.model.ticket.Ticket
import com.consol.cmas.common.model.ContactTicketRole
import com.consol.cmas.common.model.customfield.StringField
import com.consol.cmas.common.model.scripting.unit.UnitActionScriptResult
//get AM queue for search
def q_id = (workflowApi.getQueueByName("AccountManagement")).id
def q_ids = new HashSet()
q_ids.add(q_id)
//find AM ticket for the company
def crit = new TicketCriteria()
crit.setQueueIds(q_ids)
// create list field key
def contactSearchListFieldKey  = new FieldKey("queue_fields","contacts")
// prepare list field
def contactsListField = new ListField(contactSearchListFieldKey )
// create member field key
def contactSearchFieldKey = new FieldKey("queue_fields","contacts_member")
// create unit member field with Unit and ticket main role
def contactsMember = new
ContactReferenceSearchField(contactSearchFieldKey, unit,
ContactTicketRole.MAIN_ROLE)
// put member field in Unit list field
contactsListField.addChild(contactsMember)
// put field(s) into the criteria
crit.setFields([contactsListField] as Set)
// seek and find
def foundTickets = ticketService.getByCriteria(crit)
if ( foundTickets ) {
  def AM_tic = foundTickets.first()
  def AM_tic_id = AM_tic.id
  // go to AM ticket
  Map<String, Object> valuesMap = new HashMap<String, Object>()
  valuesMap.put(PostActionParameter.TICKET_ID, AM_tic_id)
  return unitActionScriptResultFactory.getPostAction(PostActionType.GOTO_TICKET, valuesMap)
}
// Default: found nothing
return null
```

Attention:

When you use the script above, please keep in mind that the ticket search requires that the data object group fields of the company be indexed (annotation *field-indexed* = *true*).

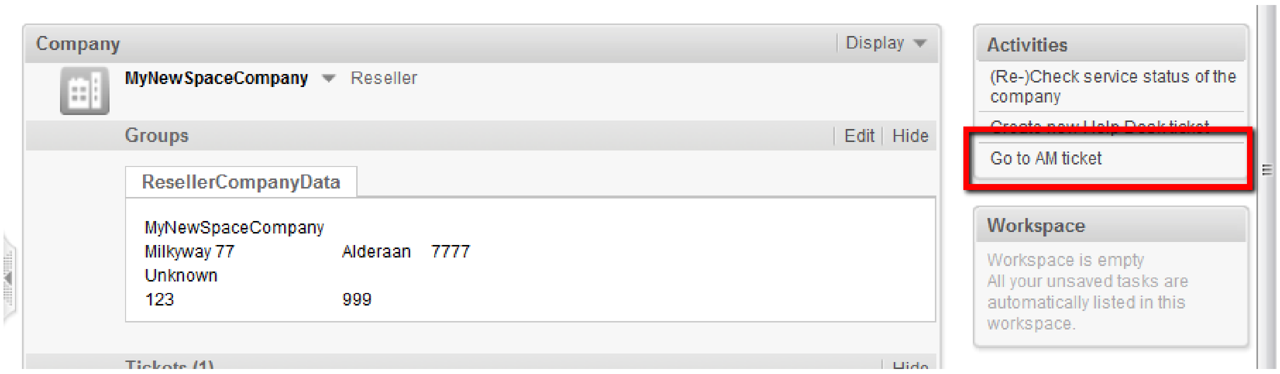


Fig. 14: ConSol*CM Web Client - Company Action Available on Company Page (1)

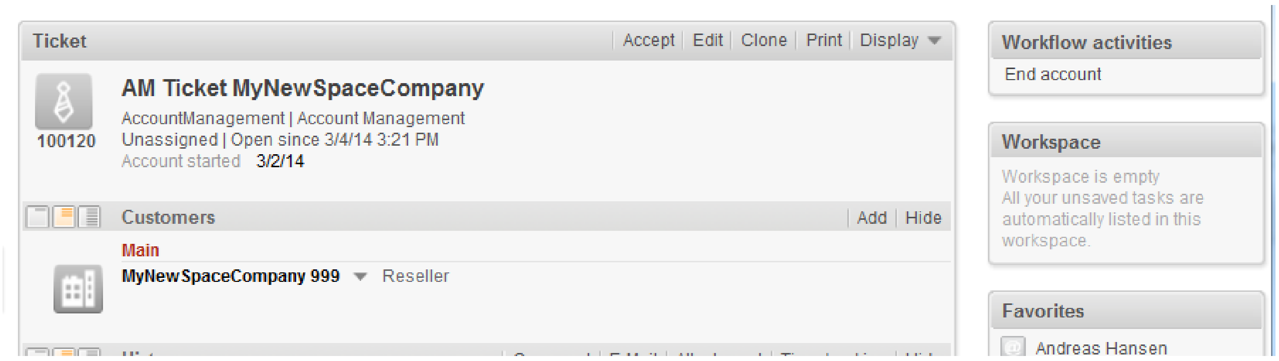

Ticket
Accept | Edit | Clone | Print | Display
AM Ticket MyNewSpaceCompany
100120
AccountManagement | Account Management
Unassigned | Open since 3/4/14 3:21 PM
Account started 3/2/14
Customers
Add | Hide
Main
MyNewSpaceCompany 999
Reseller
Workflow activities
End account
Workspace
Workspace is empty
All your unsaved tasks are automatically listed in this workspace.
Favorites
Andreas Hansen

Fig. 15: ConSol*CM Web Client - AM Page Opened after Company Action Go to Ticket

Open a Web Page

(*PostActionType.GOTO_PAGE*) redirects to URL. It uses the obligatory *PostActionParameter.URL* with the URL.

The following code shows a simple example with a fixed URL for each company.

Go to URL page

```
import com.consol.cmas.common.model.scripting.unit.PostActionType
import com.consol.cmas.common.model.scripting.unit.PostActionParameter
Map<String, Object> valuesMap = new HashMap<String, Object>()
valuesMap.put(PostActionParameter.URL, unit.get("company:www"))
return unitActionScriptResultFactory.getPostAction(PostActionType.GOTO_PAGE, valuesMap)
```

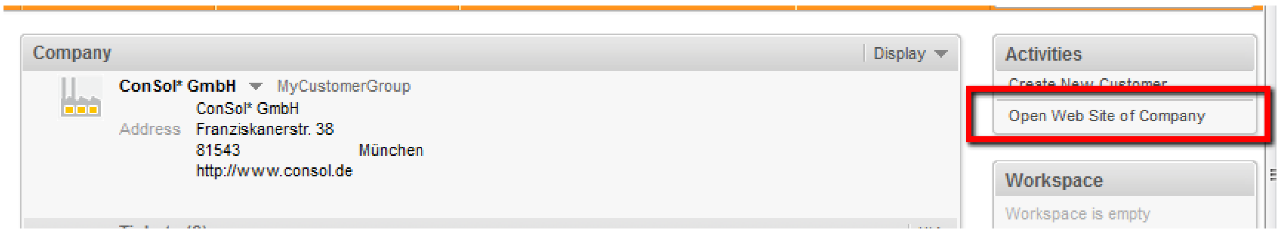


Fig. 16: ConSol*CM Web Client - Company Action Available on Company Page (2)

Attention:

To open a fixed URL, you can use a data object group field of type *string* with the annotation text-type *url*. This will automatically create a hyperlink. So the use of the *GOTO_URL* parameter in data object action scripts is recommended when a URL is built dynamically within a script.

Object UnitActionScriptResult

The object *UnitActionScriptResult* is only taken into account for manual actions. For actions like *CREATE*, *UPDATE*, or *DELETE* it is not available. The *UnitActionScriptResult* object is created by the *unitActionScriptResultFactory.getPostAction(String, Map<String, Object>)* method. This class (resp. the object) is used to store information that will influence the process flow of the Web Client after the manual action has been executed. The *UnitActionScriptResult* object contains the manual action type, the IDs of the ticket, the unit, the queue, and the customer group. After having executed the manual action the user can be redirected to a different page.

12.5.2 Data Object Condition Scripts

A data object condition script defines whether the action should be shown in the Web Client or not. It is executed before executing the data object action script. If it returns *false*, the data object action script will not be executed.

Example data object condition script

```
if(unit.getFieldValue("customer.personalData") == null) {
    return true
} else {
    return false
}
```

Attention:

The data object actions *CREATE*, *UPDATE*, and *DELETE* are executed in the core methods *create* , *update*, and *delete* of the object *unitService*.

So if the update action script updates the data object using the *unitService.update(Unit)* method then a *java.lang.StackOverflowError* error can be thrown, because the update action will be executed infinitely. In that case a data object condition script can be used to avoid such infinite loops.

13 CM6 Administrator Manual 6.9.4 - Additional User Attributes

13.1 Additional User Attributes: Customer Roles, Engineer Functions , and Projects

- Introduction
- Tab Card Customer Roles
  - Create or Edit a Customer Role
  - Delete a Customer Role
  - Disable or Enable a Customer Role
  - Localize a Customer Role
- Tab Engineer Functions
  - Create or Edit an Engineer Function
  - Delete an Engineer Function
  - Disable or Enable an Engineer Function
- Tab Projects
  - Create or Edit a Project
  - Delete a Project
  - Disable or Enable a Project
  - Localize a Project
- Related Topics

13.1.1 Introduction

Most of the *User attributes* settings have been explained in the sections Setting Up the Customer Data Model, Action Framework, and Customer (Data Object) Relations, because they are related to FlexCDM, the ConSol*CM Flexible Customer Data Model. In this section, the four non-FlexCDM user attributes will be explained:

1. **Customer roles**
   This attribute is used for additional customers and has to be assigned in the Web Client. See section Customer Roles.
2. **Engineer functions**
   This attribute is used for additional engineers and has to be assigned to a role. See section Engineer Functions.
3. **Projects**
   This attribute is used for time booking and has to be assigned to a queue. See section Projects.
4. **Autocomplete adress**
   This feature can be used to work with imported data from external address databases to make work with cutsomer data easier and less error-prone. See section for Tab Autocomplete Address details.

13.1.2 Tab Card Customer Roles

On this tab, you can create customer roles.

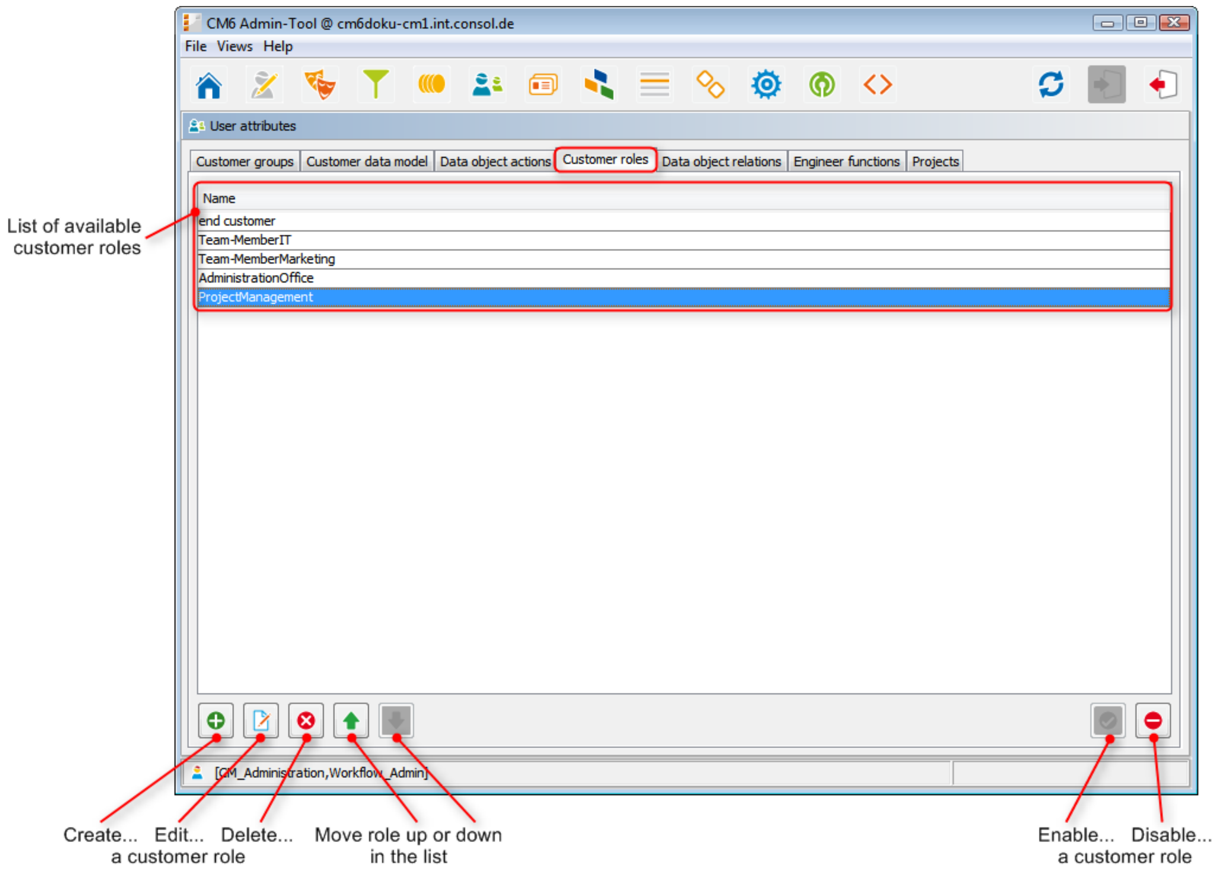


Fig. 1: ConSol*CM Admin Tool - User Attributes: Customer Roles

In the Web Client these 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. Information in the Web Client is provided (e.g. you would not want to send a log file to a *manager* but to the *IT contact*).
2. The customer role can be used in workflow programming to control the process flow (e.g. send an e-mail to all *IT contacts* but not to contacts with other roles).

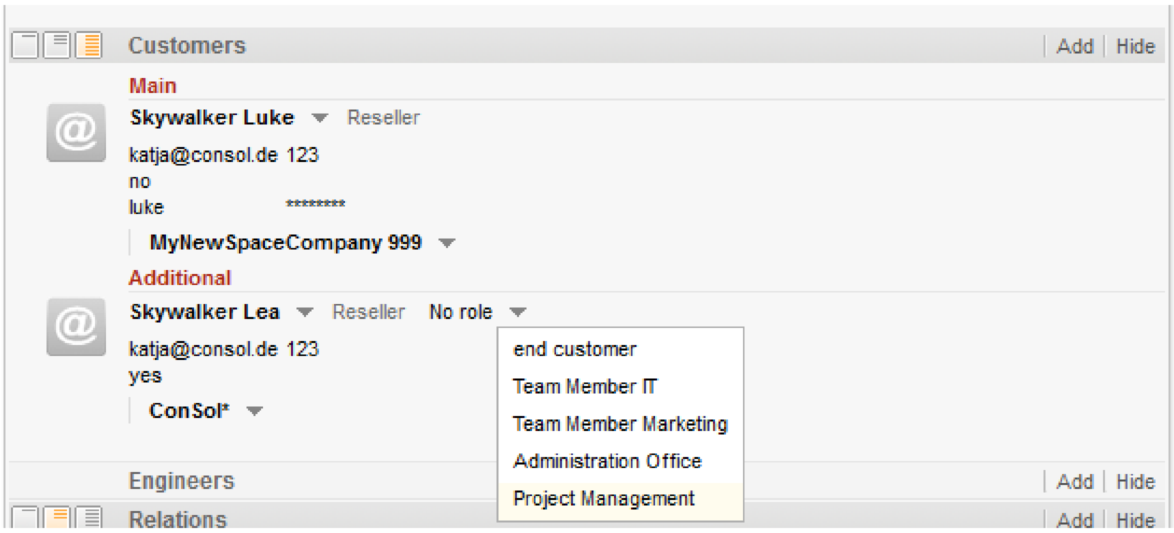


Fig. 2: ConSol*CM Web Client - Setting a Customer Role for an Additional Contact

Create or Edit a Customer Role

A customer role is defined by its name. By clicking on a pop-up window appears where you can enter the name. Using the globe icon next to the name field you can localize the name subsequently (see below). The check box *Enabled* is already selected to set the customer role active in the system (see also Disable or Enable a Customer Role). You will get the same window when you click on in order to edit a customer role.

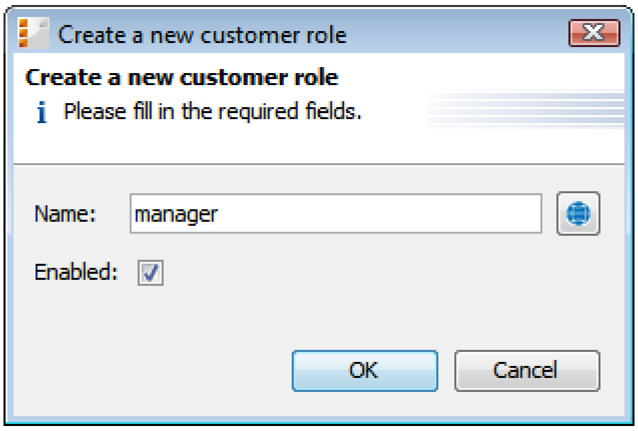

Create a new customer role
Create a new customer role
Please fill in the required fields.
Name:
manager
Enabled:
OK
Cancel

Fig. 3: ConSol*CM Admin Tool - User Attributes: Create or Edit a Customer Role

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 you can only disable this customer role (see below).

In order to delete a customer role, select it in the list and click on . After choosing *Yes* in the confirmation dialog the customer role will be removed from the list and the system.

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 on . The entry in the list is shown in italics afterwards. The customer role cannot be assigned anymore. Just click on at the bottom of the page, if you want to enable the role again.

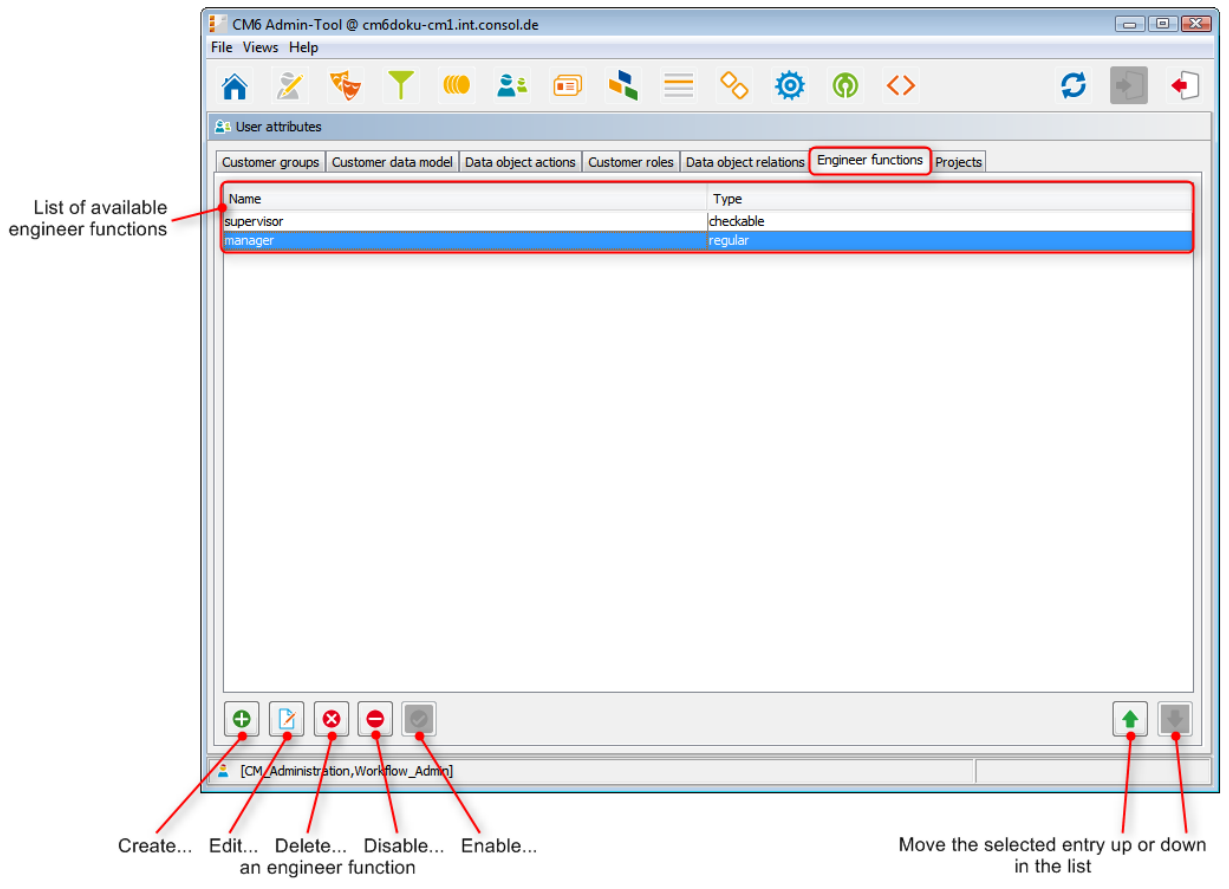
You can also enable or disable a customer role in the window used for editing customer roles by selecting or de-selecting the check box *Enabled*. When you create a customer role this check box is automatically selected.

Localize a Customer Role

Click on the globe icon in the create or edit window to enter the localized name of a customer role. In the pop-up window *Localize* all languages that are available in the system are listed. Enter the customer role name in the *Value* field for each additional language on the right and click *Save*. The localized customer role name, according to the locale of the web browser, will be displayed in the Web Client. If no localized value is found, the default value is displayed. This is the value of the default language. If this has not been defined either, the technical name of the customer group is displayed.

13.1.3 Tab Engineer Functions

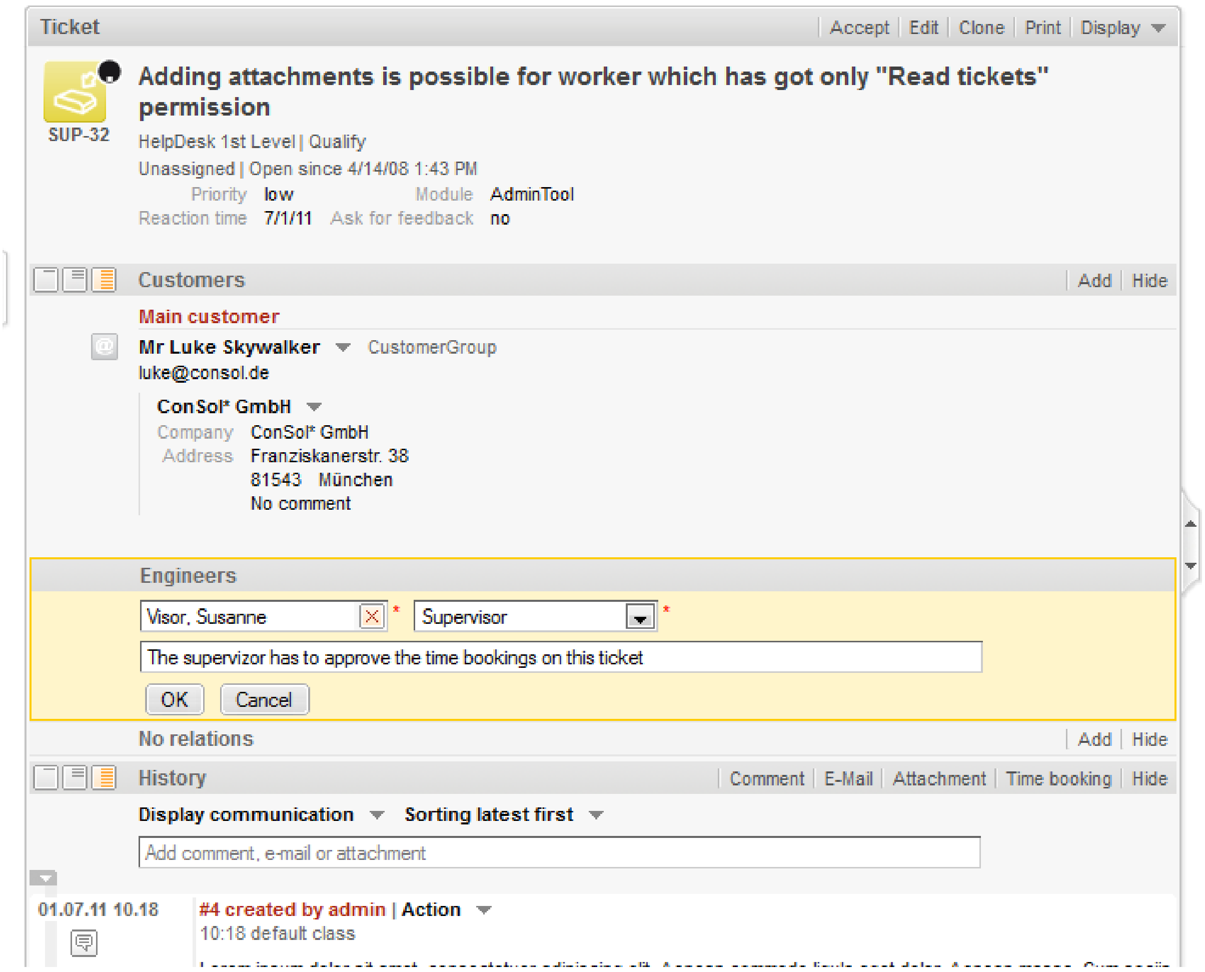
Engineer functions are used if you need an additional engineer for a ticket, e.g. a supervisor who has to decide what to do, before the ticket can be moved on in the workflow.


CM6 Admin-Tool @ cm6doku-cm1.int.consol.de
File Views Help
User attributes
Customer groups
Customer data model
Data object actions
Customer roles
Data object relations
Engineer functions
Projects
[CM_Administration,Workflow_Admin]
List of available engineer functions
Create... Edit... Delete... Disable... Enable...
an engineer function
Move the selected entry up or down in the list

Name	Type
supervisor	checkable
manager	regular

Fig. 4: ConSol*CM Admin Tool - User Attributes: 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.



Adding attachments is possible for worker which has got only "Read tickets" permission

HelpDesk 1st Level | Qualify
Unassigned | Open since 4/14/08 1:43 PM
Priority low Module AdminTool
Reaction time 7/1/11 Ask for feedback no

Main customer
Mr Luke Skywalker CustomerGroup
luke@consol.de
ConSol* GmbH
Company ConSol* GmbH
Address Franziskanerstr. 38
81543 München
No comment

Visor, Susanne * Supervisor *
The supervizor has to approve the time bookings on this ticket

OK
Cancel
No relations
Add | Hide
History
Comment | E-Mail | Attachment | Time booking | Hide
Display communication
Sorting latest first
Add comment, e-mail or attachment

01.07.11 10.18 #4 created by admin | Action
10:18 default class

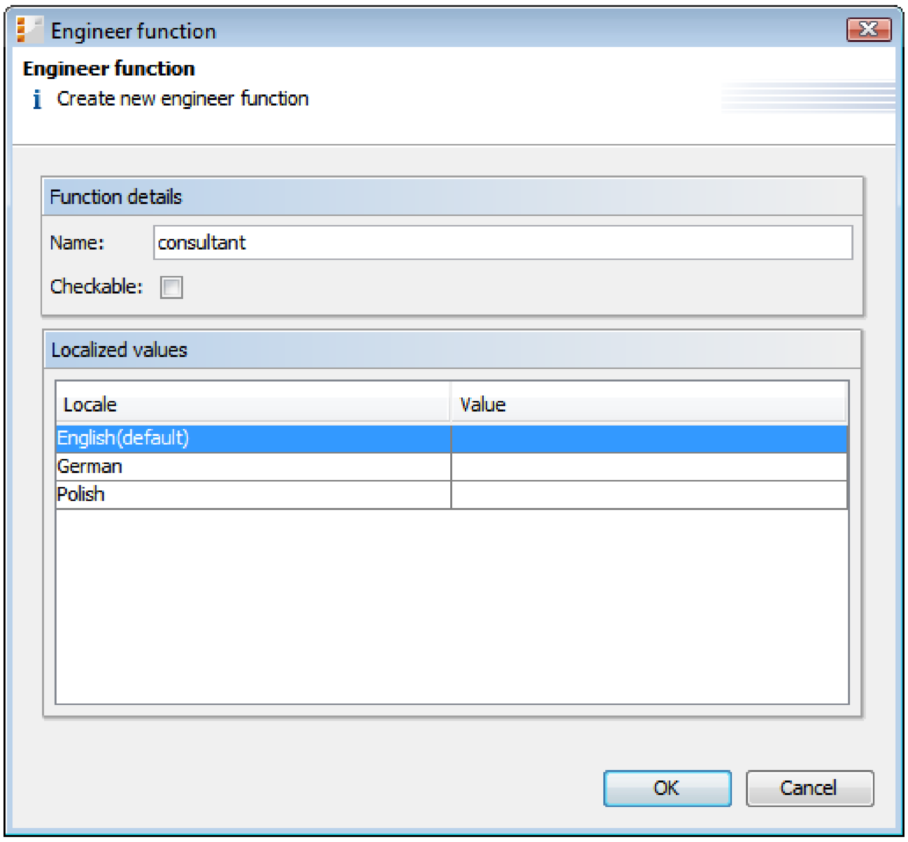
Fig. 5: ConSol*CM Web Client - Assigning an (Additional) Engineer with an Engineer Function

Create or Edit an Engineer Function

An engineer function is defined by its name. By clicking on a pop-up window appears where you can enter the name. You will get the same window when you click on in order to edit an engineer function. The check box *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.

Attention:

After creation of an engineer function the check box *Checkable* cannot be de-selected anymore.



Engineer function

Create new engineer function

Function details

Name: consultant

Checkable: ☐

Localized values

Locale	Value
English(default)	
German	
Polish	

Fig. 6: ConSol*CM Admin Tool - User Attributes: Create or Edit an Engineer Function

You can also localize the name of an engineer function in this window. The available locales are shown on the left side of the table. Enter the corresponding engineer function name in the *Value* field for each additional language on the right. After clicking *OK* the engineer function is created and the name will be displayed in the respective language of the engineer's locale.

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 on ⊗ . After choosing *Yes* in the confirmation dialog the engineer function will be removed from the list and the system.

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 on ⊖ . The entry in the list is shown in italics afterwards. The engineer function cannot be assigned anymore. Just click on ✓ at the bottom of the page, if you want to enable the function again.

13.1.4 Tab Projects

With ConSol*CM you can book amounts of time on projects, please see section Time Booking for a detailed explanation. The projects are created on this tab and have to be assigned to queues. In the Web Client you can book amounts of time on tickets that are in one of the queues where the project has been assigned. An engineer can see his/her time bookings on the engineer profile page.

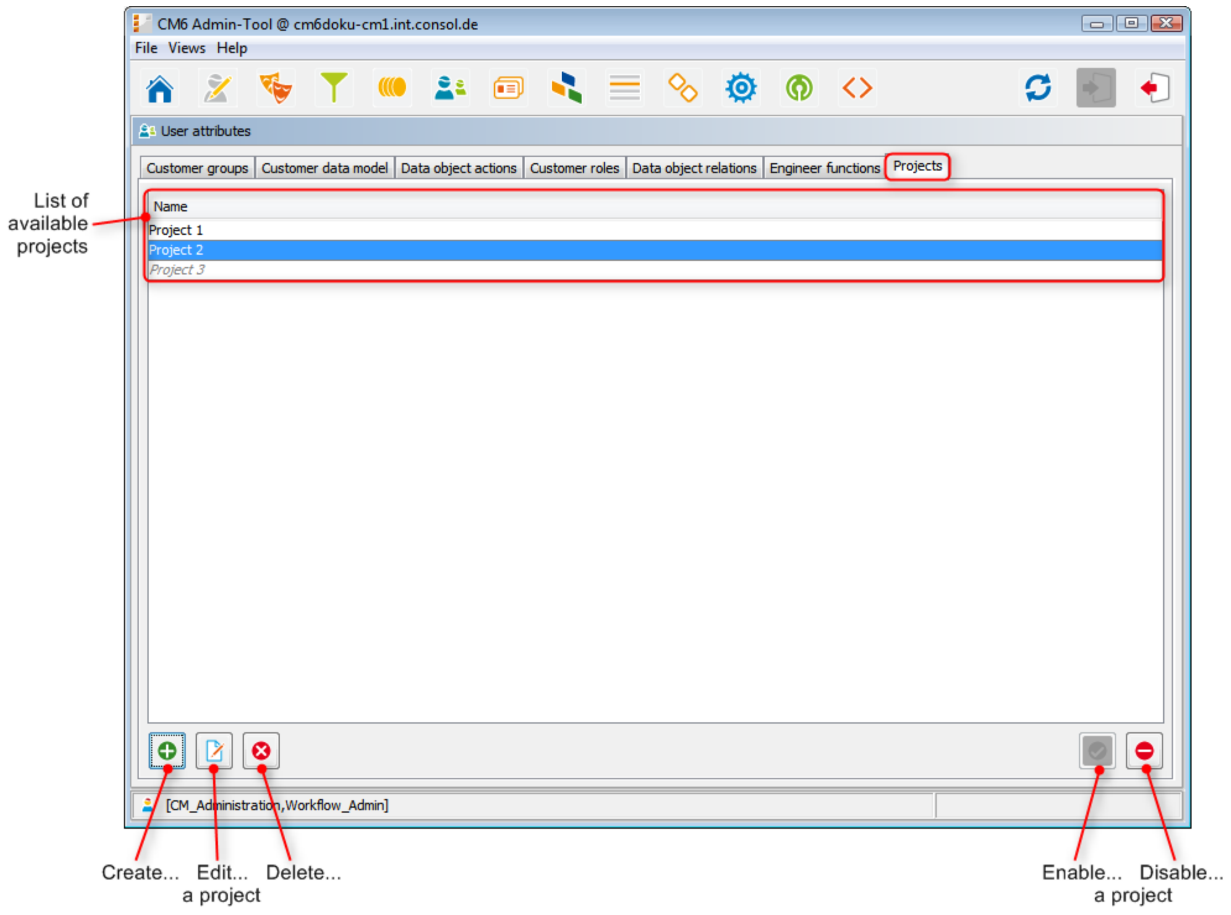
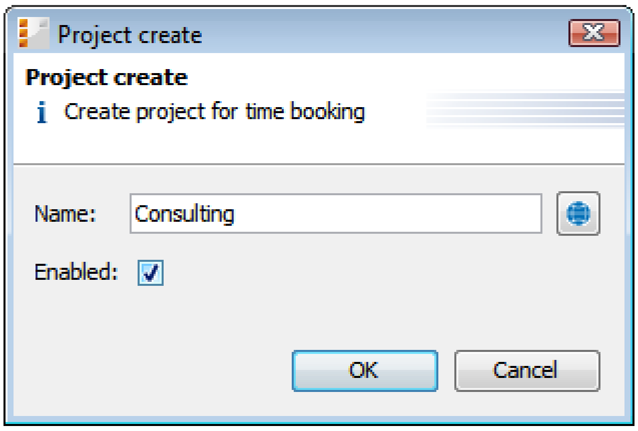


Fig. 7: ConSol*CM Admin Tool - User Attributes: Projects

Create or Edit a Project

A project is defined by its name. By clicking on a pop-up window appears where you can enter the name. Using the globe icon next to the name field you can localize the name subsequently (see below). The check box *Enabled* is already selected to set the project active in the system (see also Disable or Enable a Project). You will get the same window when you click on in order to edit a project.



Name: Consulting

Enabled: ☑

Fig. 8: ConSol*CM Admin Tool - User Attributes: Create or Edit a Project

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 you can only disable this project (see below).

In order to delete a project, select it in the list and click on . After choosing *Yes* in the confirmation dialog the project will be removed from the list and the system.

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 on . The entry in the list is shown in italics afterwards. The project is not available for time bookings anymore. Just click on 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 de-selecting the check box *Enabled*. When you create a project this check box is automatically selected.

Localize a Project

Click on the globe icon in the create or edit window to enter the localized name of a project. In the pop-up window *Localize* all languages that are available in the system are listed. Enter the project name in the *Value* field for each additional language on the right and click *Save*. The localized project name, according to the locale of the web browser, will be displayed in the Web Client in the time booking section of a ticket. If no localized value is found, the default value is displayed. This is the value of the default language. If this has not been defined either, the technical name of the project is displayed.

13.1.5 Related Topics

- Queues
- Roles
- Engineer administration

13.2 Tab Autocomplete Address

13.2.1 Introduction

In some ConSol*CM systems it is required that engineers enter or edit a great number of customer data manually. In that case it can be helpful to have system support to

- provide suggestions for the input into some data fields (like e.g. zip or address) to ensure that the entered address really exists
- avoid entering duplicates

To provide those functionalities, ConSol*CM offers the feature *Autocomplete Address*. 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 data object group fields:

- one or more fields which contain the zip code
- one or more fields that contain the city
- one or more fields that contain the address (street and number)

This applies to the following operations in the CM Web Client:

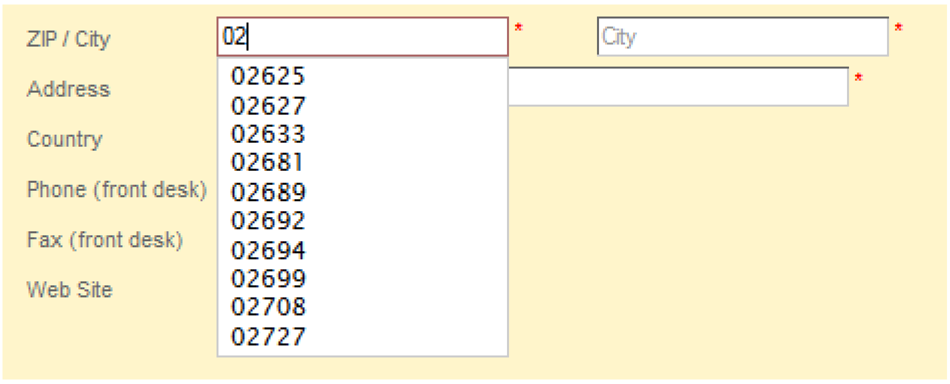
- Create a customer on the create customer page
- Create a customer in the *Customers* section of a ticket
- Edit a customer on the customer page
- Edit a customer in the *Customers Section* of a ticket
- Entering customer data on the Detail Search page
- Entering customer data in an ACF (Activity Control 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/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 *Autocomplete Address* 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 CM Web Client to learn how 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 a Autocomplete Strategy Definition, please see section Define the Autocomplete Strategy using the ConSol*CM Admin Tool for details.

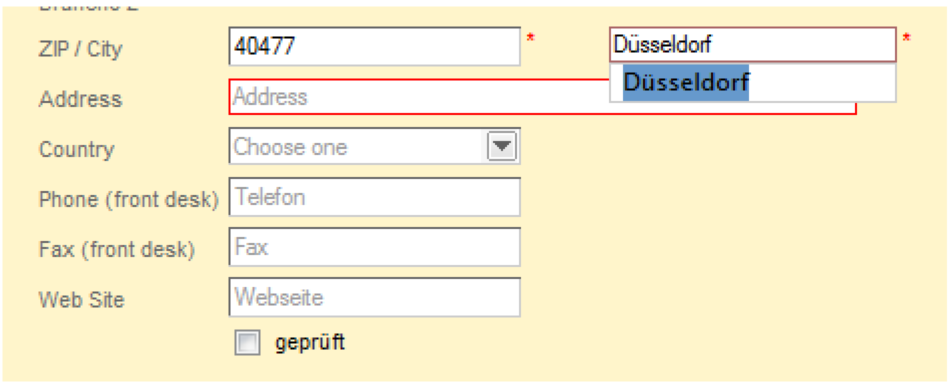


02

02625
02627
02633
02681
02689
02692
02694
02699
02708
02727

Fig. 1: CM Web Client: Suggestions of Autocomplete Address feature, Example 1

Example 2: The *ZIP* field has already been filled. For the *City* field, only the correct possible values are suggested.



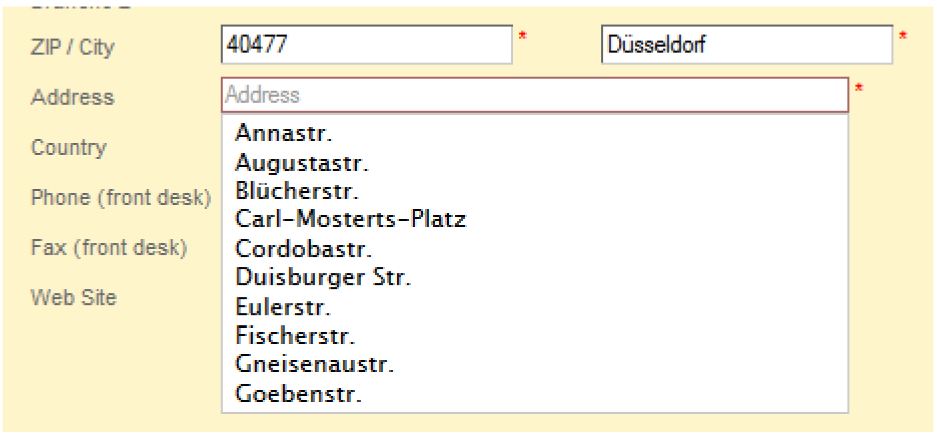
40477

Düsseldorf

Düsseldorf

Fig. 2: CM Web Client: Suggestions of Autocomplete Address 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.



40477

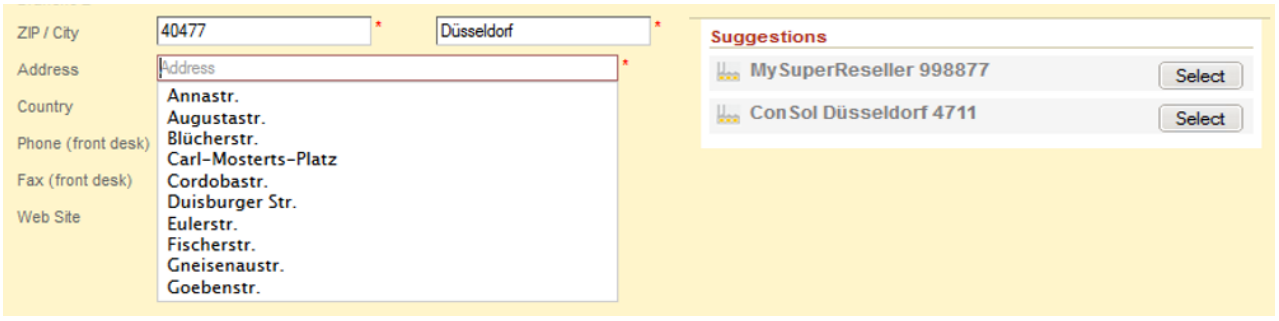
Düsseldorf

Annastr.
Augustastr.
Blücherstr.
Carl-Mosterts-Platz
Cordobastr.
Duisburger Str.
Eulerstr.
Fischerstr.
Gneisenaustr.
Goebenstr.

Fig. 3: CM Web Client: Suggestions of Autocomplete Address 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 CM standard feature *Autocomplete Search* works and will display suggestions for the customers which are already part of the CM database.



ZIP / City	40477 *	Düsseldorf *
Address	Address *	
Country		
Phone (front desk)		
Fax (front desk)		
Web Site		

Annastr.
Augustastr.
Blücherstr.
Carl-Mosterts-Platz
Cordobastr.
Duisburger Str.
Eulerstr.
Fischerstr.
Gneisenaustr.
Goebenstr.

Suggestions

MySuperReseller 998877	Select
ConSol Düsseldorf 4711	Select

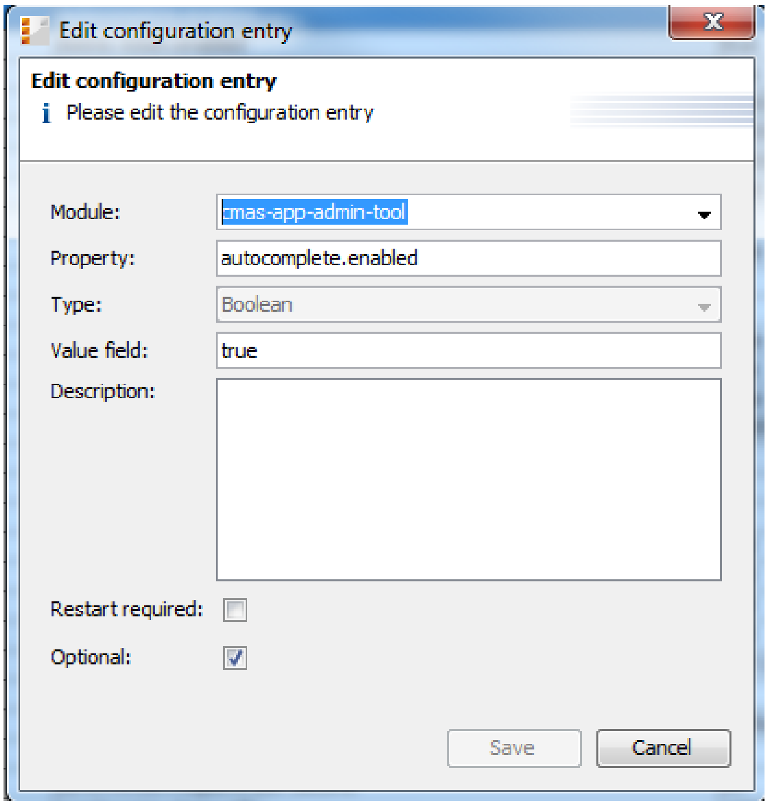
Fig. 4: CM Web Client: Standard Autocomplete / Suggestion Feature in combination with Autocmplete Address

To configure your ConSol*CM system for this feature, the following steps have to be performed:

1. Switch on the *Autocomplete Addresses* feature using the Admin Tool, see section Switch on the *Autocomplete Address* Feature using the Admin Tool
2. Import zip/city/address data into the CM database. See section Import Zip/City/Address Data into the CM Database
3. Define the autocomplete strategy using the ConSol*CM Admin Tool, see section Define the Autocomplete Strategy using the ConSol*CM Admin Tool
4. Refresh the index, see section Refresh the Index

13.2.2 Switch on the Autocomplete Address Feature using the Admin Tool

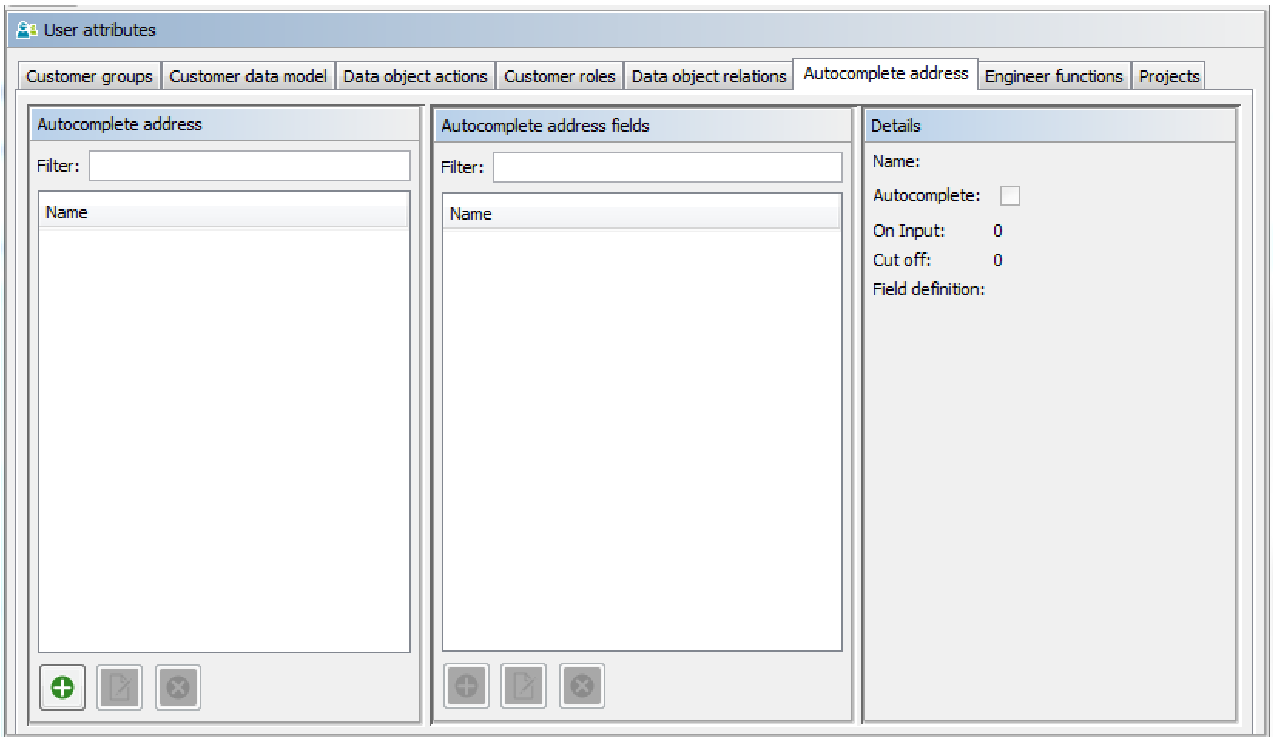
Enter the system property *cmas-app-admin-tool*, *autocomplete.enabled* and set its value to *true*. The system property is not present in a standard CM installation and has to be added manually.



Module: cmas-app-admin-tool
Property: autocomplete.enabled
Type: Boolean
Value field: true
Description:
Restart required: ☐
Optional: ☑

Fig. 5: Admin Tool: System Property for the Autocmplete Addresses feature

Restart the Admin Tool. You will then see the new tab *Autocomplete address* in the *User attributes* section. You will learn how to configure the autocomplete strategy in section Define the Autocomplete Strategy using the ConSol*CM 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.



Name:
Autocomplete:
On Input: 0
Cut off: 0
Field definition:

Fig. 6: Admin Tool: Autocomplete Address tab in User attributes section

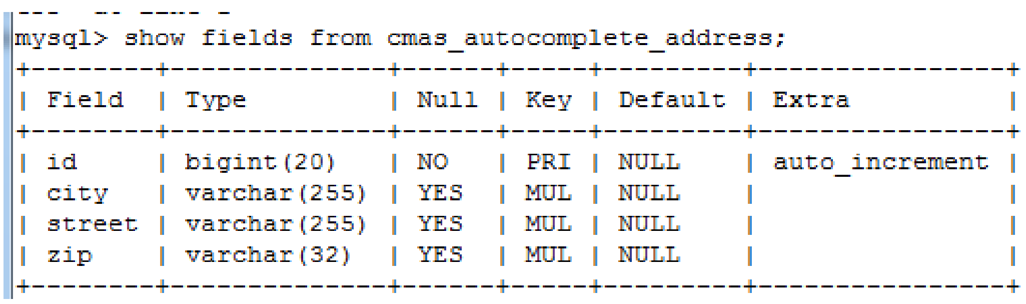
13.2.3 Import Zip/City/Address Data into the CM Database

Import the data into the table *cmas_autocomplete_address* in your 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    |                |
+--------+--------------+------+-----+---------+----------------+
```

Fig. 7: ConSol*CM database: Table cmas_autocomplete_address

13.2.4 Define the Autocomplete Strategy using the ConSol*CM Admin Tool

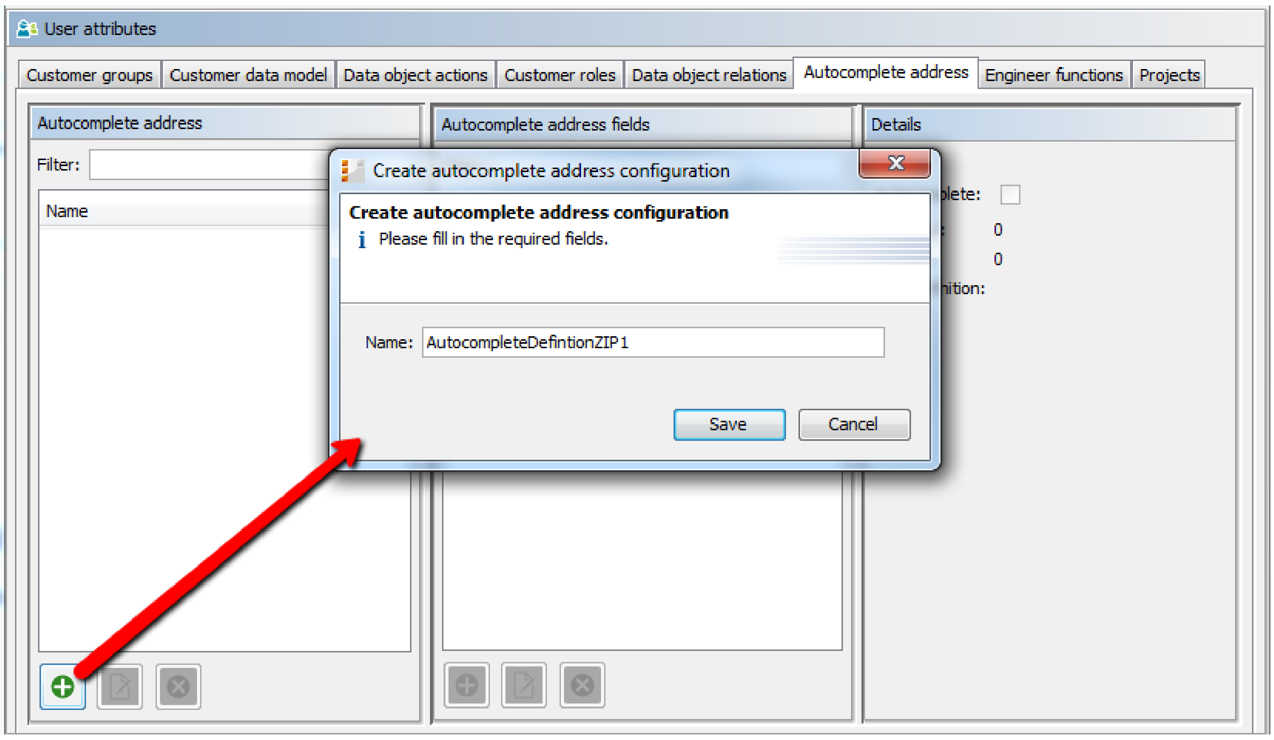
To define the autocomplete strategy, you have to perform the following steps:

1. Create one or more Autocomplete Strategy Definitions
2. Configure the behavior for each Autocomplete Strategy Definition

Create One or More Autocomplete Strategy Definitions

An Autocomplete Strategy Definition represents a mapping of a data object group field to one of the key fields in the *cmas_autocomplete_address* table, e.g. you want to define the following definition "When the engineer starts the input in the field *ResellerCompanyData:zip*, the system should search in *ZIP*, and when the engineer starts typing in the data object group field *ResellerCompanyData:city*, the system should search in the city field".

You have to define each of the Autocomplete Strategy Definitions in the Admin Tool in the tab Autocomplete address in the User attributes section. Add a new Autocomplete Strategy Definition by pressing the PLUS button. Enter the name of the new strategy definition and press *Save*.



Create autocomplete address configuration

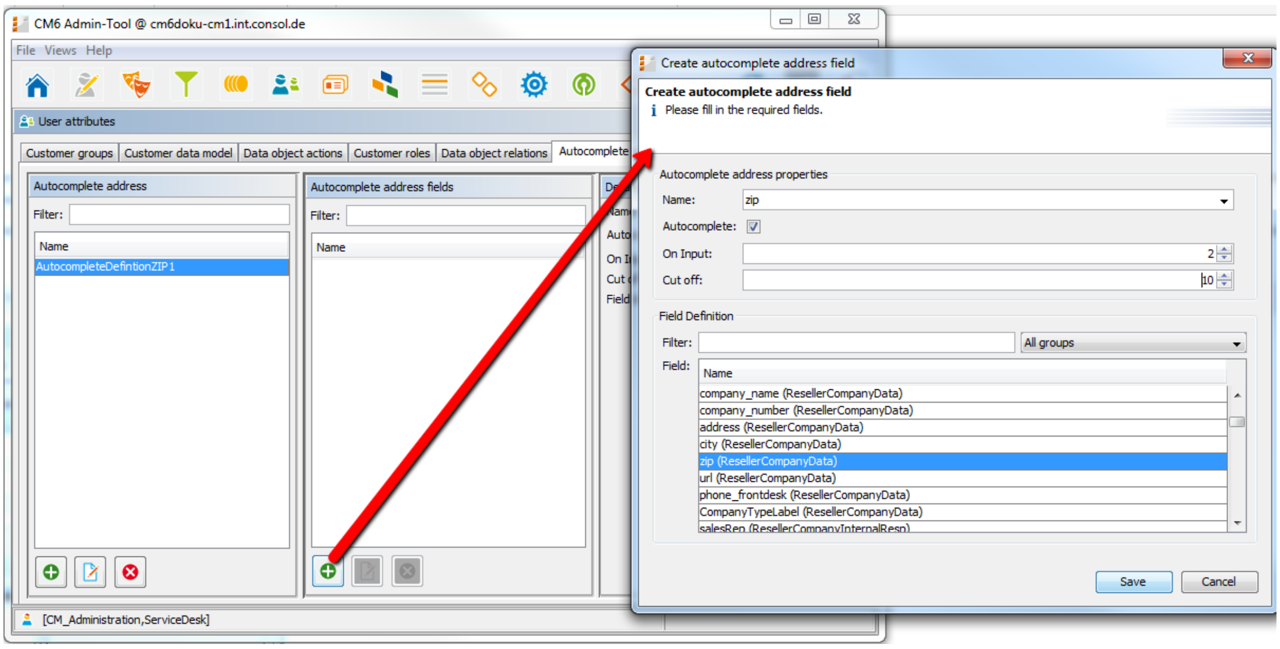
Please fill in the required fields.

Name: AutocompleteDefintionZIP1

Fig. 8: Admin Tool: Adding a new Autocomplete Strategy Definition

Configure the Behavior for Each Autocomplete Strategy Definition

To configure an Autocomplete Strategy Definition, mark the definition in the list and add one or more Autocomplete Address Fields. These are the mapping rules from the data object group fields to the key fields in the *cmas_autocomplete_address* table.


CM6 Admin-Tool @ cm6doku-cm1.int.consol.de

File Views Help

User attributes

Customer groups | Customer data model | Data object actions | Customer roles | Data object relations | Autocomplete

Autocomplete address

Filter:

Name
AutocompleteDefintionZIP1

Autocomplete address fields

Filter:

Name

[CM_Administration,ServiceDesk]

Create autocomplete address field

Create autocomplete address field

Please fill in the required fields.

Autocomplete address properties

Name: zip

Autocomplete: ✓

On Input: 2

Cut off: 10

Field Definition

Filter: All groups

Field:

Name
company_name (ResellerCompanyData)
company_number (ResellerCompanyData)
address (ResellerCompanyData)
city (ResellerCompanyData)
zip (ResellerCompanyData)
url (ResellerCompanyData)
phone_frontdesk (ResellerCompanyData)
CompanyTypeLabel (ResellerCompanyData)
salesRep (ResellerCompanyInternalResp)

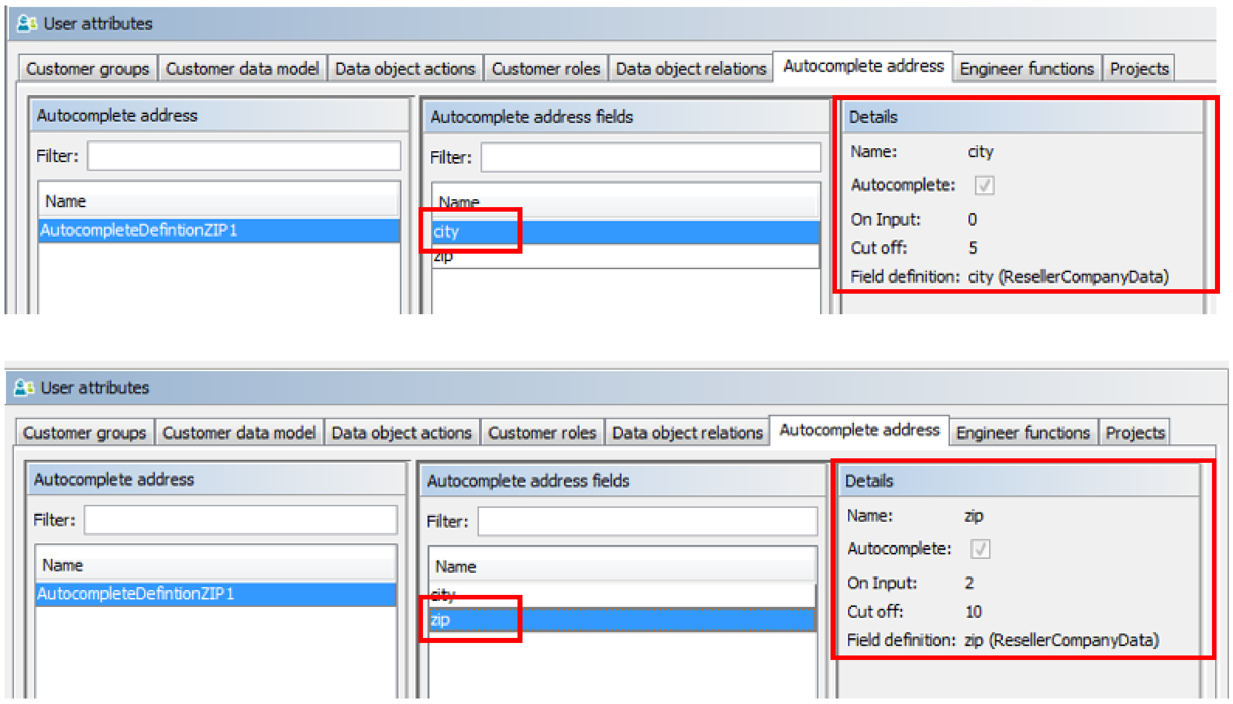
Save Cancel

Fig. 9: Admin Tool: Definition of Autocomplete address Fields

The following fields have to be defined for each autocomplete address 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 to other fields.
  if switched off: The field will not be an autocomplete field itself but will be filled by dependence 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 react to engineer input itself.
- **On Input:**
  Here you define the number of characters the engineer has to type into the data object group 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 data object group field in the Web Client, leave the configuration field here empty.
- **Cut off:**
  Here you define the number of suggestions in the (drop-down) list.
- **Field Definition:**
  Here you select one data object group field which should be sensitive for the autocomplete input. Only single selection is possible.

In the following example, two Autocomplete Address Fields have been defined for the Autocomplete Address definition *AutocompleteDefinitionZIP1*, see the following figure.


User attributes

Customer groups | Customer data model | Data object actions | Customer roles | Data object relations | Autocomplete address | Engineer functions | Projects

Autocomplete address
Filter:

Name
AutocompleteDefintionZIP1

Autocomplete address fields
Filter:

Name
city
zip

Details

Name:	city
Autocomplete:	✓
On Input:	0
Cut off:	5
Field definition:	city (ResellerCompanyData)

User attributes

Customer groups | Customer data model | Data object actions | Customer roles | Data object relations | Autocomplete address | Engineer functions | Projects

Autocomplete address
Filter:

Name
AutocompleteDefintionZIP1

Autocomplete address fields
Filter:

Name
city
zip

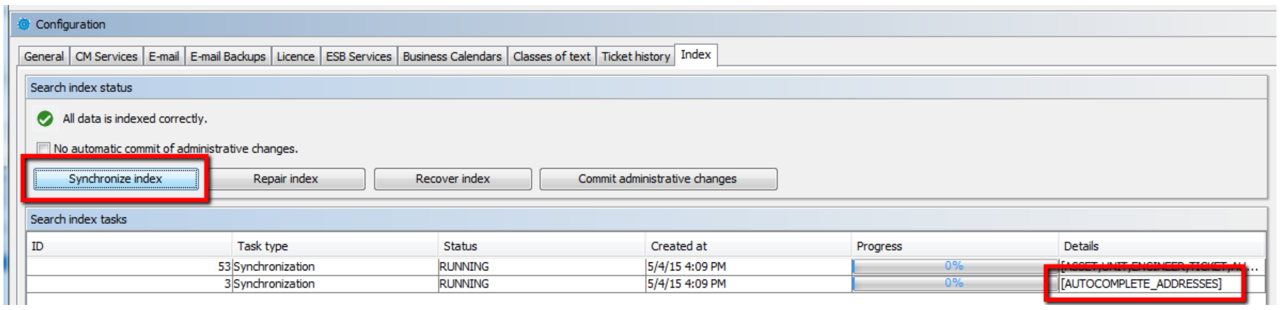
Details

Name:	zip
Autocomplete:	✓
On Input:	2
Cut off:	10
Field definition:	zip (ResellerCompanyData)

Fig. 10: Admn Tool: Complete Autocomplete Address Definition

Refresh the Index

When you have entered and configured all required Autocomplete Strategy Definitions, you have to refresh the index. For details about the index, please refer to section Tab Index - Search and Indexer Configuration.


Configuration

General | CM Services | E-mail | E-mail Backups | Licence | ESB Services | Business Calendars | Classes of text | Ticket history | Index

Search index status

✓ All data is indexed correctly.

☐ No automatic commit of administrative changes.

Search index tasks

ID	Task type	Status	Created at	Progress	Details
53	Synchronization	RUNNING	5/4/15 4:09 PM	0%	[ASSET,UNIT,ENGINEER,TICKET,AL...
3	Synchronization	RUNNING	5/4/15 4:09 PM	0%	[AUTOCOMPLETE_ADDRESSES]

Fig. 11: Admin Tool: Index Refresh

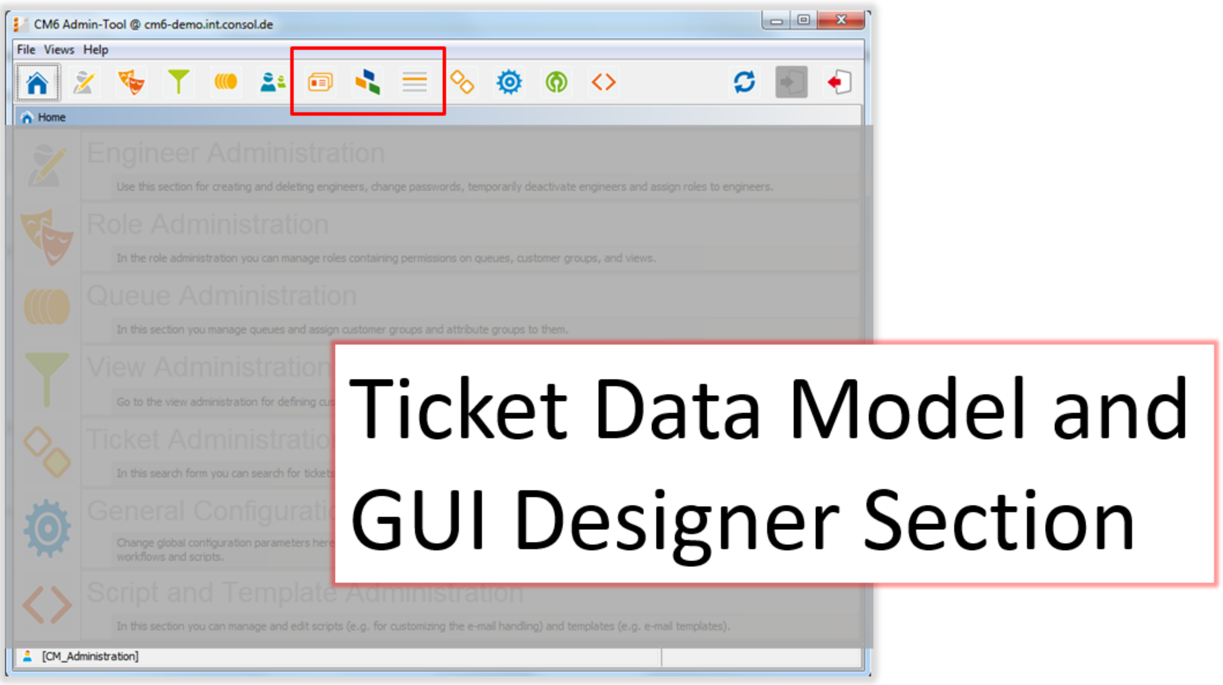
13.2.5 Edit an Autocomplete Strategy Definition

In order to edit an existing Autocomplete Strategy Definition, use the EDIT button. You can edit the Autocomplete Strategy Definition and/or each of the Autocomplete Address Fields.

13.2.6 Delete an Autocomplete Strategy or Autocomplete Address Fields

In order to delete an Autocomplete Strategy Definition or Autocomplete Address Fields, mark the strategy definition or the field in the respective list and press the DELETE ⊗ button.

14 Ticket Data Model and GUI Designer Section


CM6 Admin-Tool @ cm6-demo.int.consol.de
File Views Help
Home
Engineer Administration
Use this section for creating and deleting engineers, change passwords, temporarily deactivate engineers and assign roles to engineers.
Role Administration
In the role administration you can manage roles containing permissions on queues, customer groups, and views.
Queue Administration
In this section you manage queues and assign customer groups and attribute groups to them.
View Administration
Ticket Administration
General Configuration
Script and Template Administration
In this section you can manage and edit scripts (e.g. for customizing the e-mail handling) and templates (e.g. e-mail templates).
[CM_Administration]
Ticket Data Model and GUI Designer Section

In this section, you learn how to define the data model for ticket data, thus this is about Custom Field definition and the positioning of the custom fields on the web Client GUI. However, some of the data structures (enums, MLAs) might also be used for data object group fields, see section Data Object Group Field Management and GUI Design. The Web Client Dashboard, a new feature in versions as of 6.9.4 is included, as well as the Page Customization, a powerful mechanism to configure the layout and functionalities of the Web Client.

In this section, the following topics are covered:

- Custom Field Administration
- Managing Sorted Lists: Enum Administration
- MLA Administration
- Configuration of the Web Client Dashboard
- Page Customization

15 Custom Field Administration

- Introduction to Custom Field Administration
- Custom Field Administration Using the Admin Tool
  - Tab Ticket Data
    - Create a Custom Field Group
    - Edit a Custom Field Group
    - Annotate a Custom Field Group
    - Delete a Custom Field Group
    - Enable or Disable a Custom Field Group
    - Create a Custom Field
    - Edit a Custom Field
    - Annotate a Custom Field
    - Delete a Custom Field
    - Enable or Disable a Custom Field
  - Tab Activity Form Data
    - Short Introduction to Activity Control Forms (ACFs)
    - Definition of Activity Control Forms (ACFs)
    - Create an Activity Control Form
    - Edit an Activity Control Form
    - Delete an Activity Control Form
    - Enable or Disable an Activity Control Form
    - Localize an Activity Control Form
  - Frequently Used Annotations
- Related Topics

15.1 Introduction to Custom Field Administration

Custom fields are fields that contain ticket data (e.g. *priority*, *software module*, *reaction time*, or *sales potential*) of the ConSol*CM system.

Attention:

Custom fields (CFs) are defined as distinct fields, but the system configuration concerning custom fields is always based on custom field groups (CF groups), never on single CFs.

A **custom field group** (CF group) ...

- can be assigned to a queue, e.g. the CF group *helpdesk_data_fields* is assigned to the queue *HelpDesk*.
- can be faded in and out in the GUI during the process; only the whole group, not single fields (CFs).
- can be displayed as tab or in the *Groups* section of a ticket. The title (and mouse-over) of the tab is the (localized) name of the CF group.
- is configured by the group annotations. Annotations are used to define special parameters and characteristics for a CF, e.g. the position in the user interface. Please see section Appendix A for a list of all available annotations.
- is placed on the GUI based on its position in the CF group list (defines e.g. the order of tabs).

A **custom field** (CF) ...

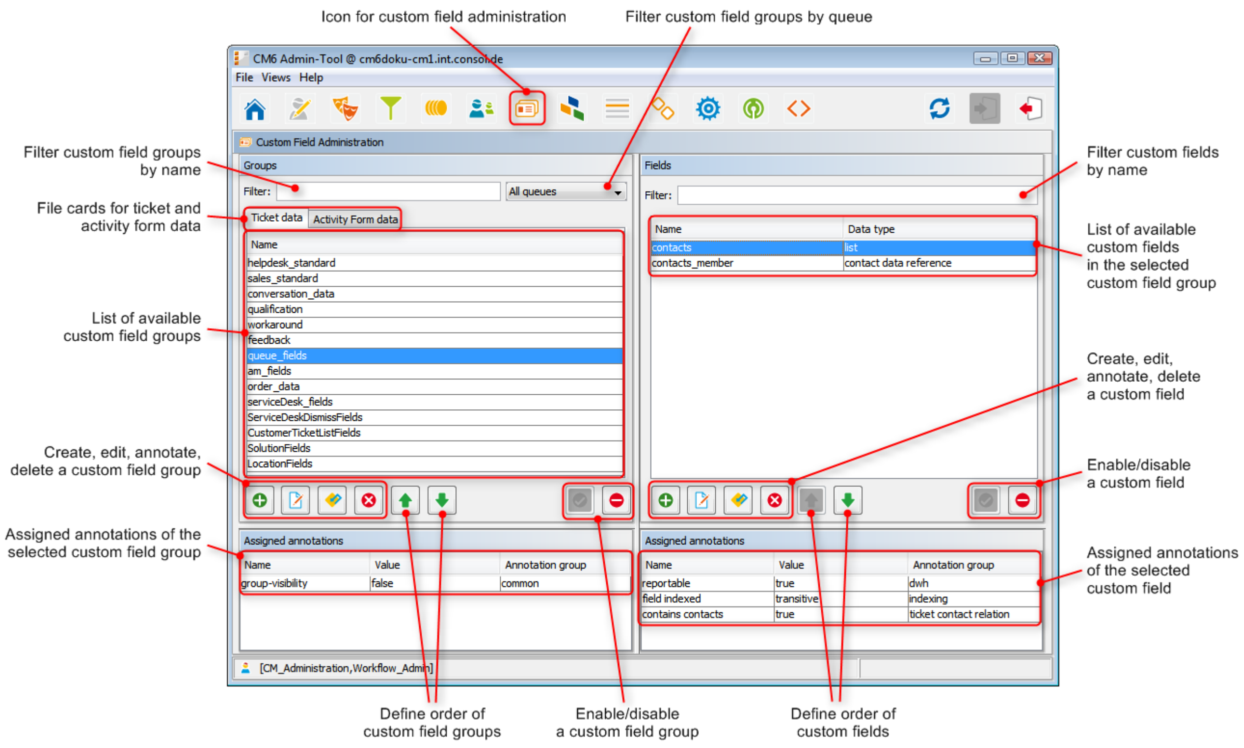
- is always defined within a custom field group.
- can be assigned to a queue only as part of its custom field group.
- can be made invisible using annotations, but cannot be faded in or out as single field during the process.
- is configured using field annotations. Annotations are used to define special parameters and characteristics for a CF, e.g. the position in the user interface. Please see section Appendix A for a list of all available annotations.
- is placed on the GUI based on the value of its *position* annotation or based on its creation time ("first CFs first in the GUI") if *position* is not set.

When you want to **define new custom fields** for a queue (e.g. you need some data about hardware details like order number, IP address, and brand), you have to perform the following steps:

1. Define a custom field group, set the respective annotations.
2. Define all sorted lists (enums) in the enum administration which you will need in the custom fields (e. g. when you need a CF brand which shold contain a sorted list, you have to define this enum first).
3. Define all custom fields within the new CF group.
4. Assign the new CF group to all queues where the 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 press F5 to refresh the page.

All those steps will be explained in detail in the following sections.

15.2 Custom Field Administration Using the Admin Tool


Icon for custom field administration
Filter custom field groups by queue
Filter custom field groups by name
File cards for ticket and activity form data
List of available custom field groups
Create, edit, annotate, delete a custom field group
Assigned annotations of the selected custom field group
Filter custom fields by name
List of available custom fields in the selected custom field group
Create, edit, annotate, delete a custom field
Enable/disable a custom field
Assigned annotations of the selected custom field
Define order of custom field groups
Enable/disable a custom field group
Define order of custom fields
CM6 Admin-Tool @ cm6doku-cm1.int.consol.de
File Views Help
Custom Field Administration
Groups
Filter:
All queues
Ticket data
Activity Form data
Fields
Assigned annotations
[CM_Administration,Workflow_Admin]

Name
helpdesk_standard
sales_standard
conversation_data
qualification
workaround
feedback
queue_fields
am_fields
order_data
serviceDesk_fields
ServiceDeskDismissFields
CustomerTicketListFields
SolutionFields
LocationFields

Name	Value	Annotation group
group-visibility	false	common

Name	Data type
contacts	list
contacts_member	contact data reference

Name	Value	Annotation group
reportable	true	dwh
field indexed	transitive	indexing
contains contacts	true	ticket contact relation

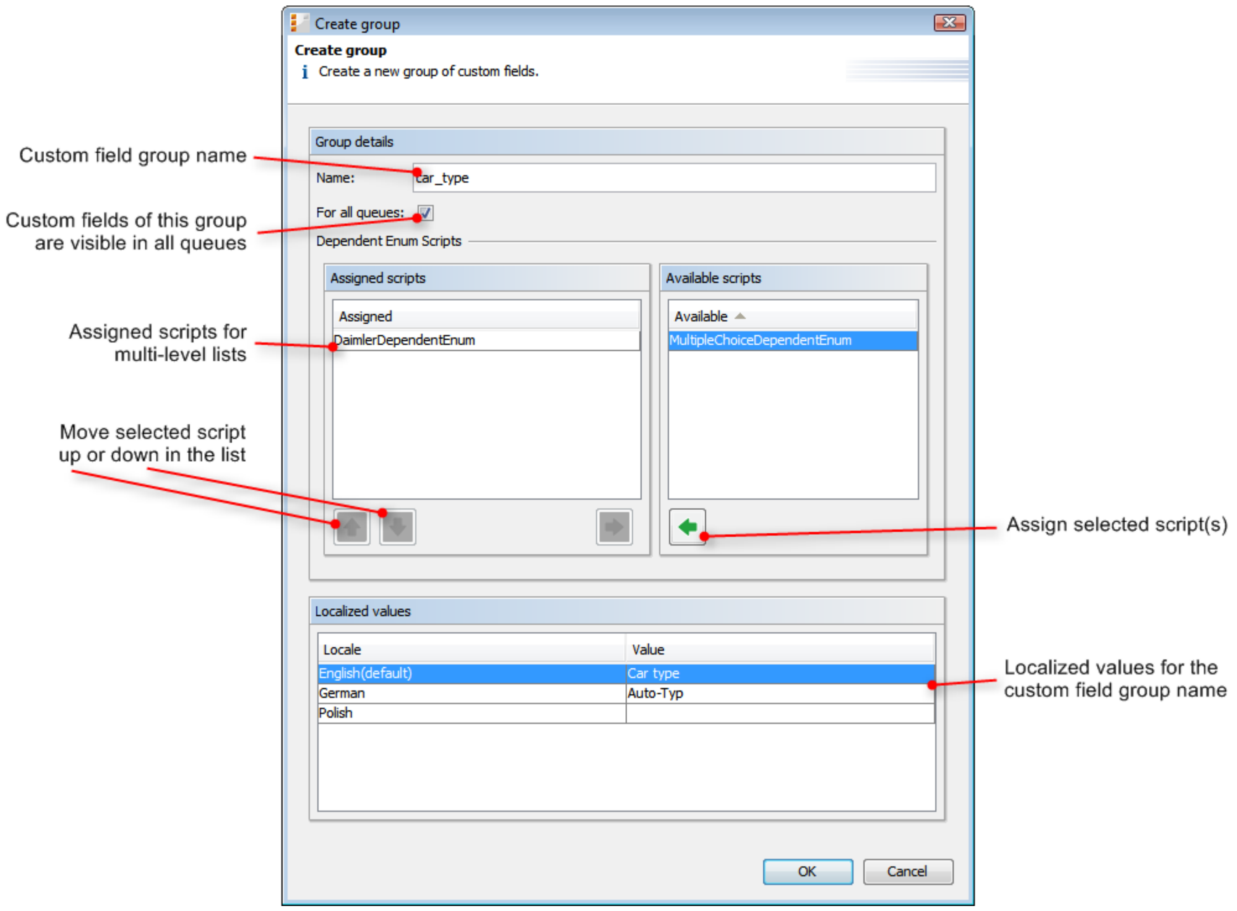
Fig. 1: ConSol*CM Admin Tool - Custom Field Administration

15.2.1 Tab Ticket Data

On this tab, you can define groups and fields for *ticket* data. The tab Activity Form data will be explained in one of the subsequent sections.

Create a Custom Field Group

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


Custom field group name
Custom fields of this group are visible in all queues
Assigned scripts for multi-level lists
Move selected script up or down in the list
Assign selected script(s)
Localized values for the custom field group name
Create group
OK
Cancel

Create group

Create a new group of custom fields.

Group details

Name: car_type

For all queues: ✓

Dependent Enum Scripts

Assigned scripts

Assigned
DaimlerDependentEnum

Available scripts

Available
MultipleChoiceDependentEnum

Localized values

Locale	Value
English(default)	Car type
German	Auto-Typ
Polish	

Fig. 2: ConSol*CM Admin Tool - Custom Field Administration: Create a Custom Field Group

- **Name**
  Enter a name for the custom field group. The name must be unique.
- **For all queues**
  If this check box is activated, this group's custom fields are visible in all queues. Usually custom field groups for ticket data are only valid in specific queues (see Queue Administration).
- **Dependent Enum Scripts**
  Dependent enum scripts define the structure of *dependent enums* (hierarchical multi-level lists) used in custom fields of this custom 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 custom fields have to be created within the Enum Administration while the scripts that couple the lists to create the dependent enum are created using an Admin Tool script, see section Admin Tool Scripts.
  To assign dependent enum scripts to a custom field group select the desired script(s) in the list *Available scripts* and move them to the list *Assigned scripts* by clicking on ⬅ .
- **Localized values**
  Enter the corresponding group name in the *Value* field for each additional language. In the Web Client's user interface the name will be displayed in the language of the engineer's web browser locale. If you do not make an entry here the object name, i.e. the content of the *Name* field, will be taken instead.

Edit a Custom Field Group

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

Annotate a Custom Field Group

Custom 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. if a group is visible in the Web Client (annotation *group-visibility*) or if it is shown in the *Groups* section of the Web Client (annotation *show-in-group-section*). To assign annotations select a group and click on . The following pop-up window appears:

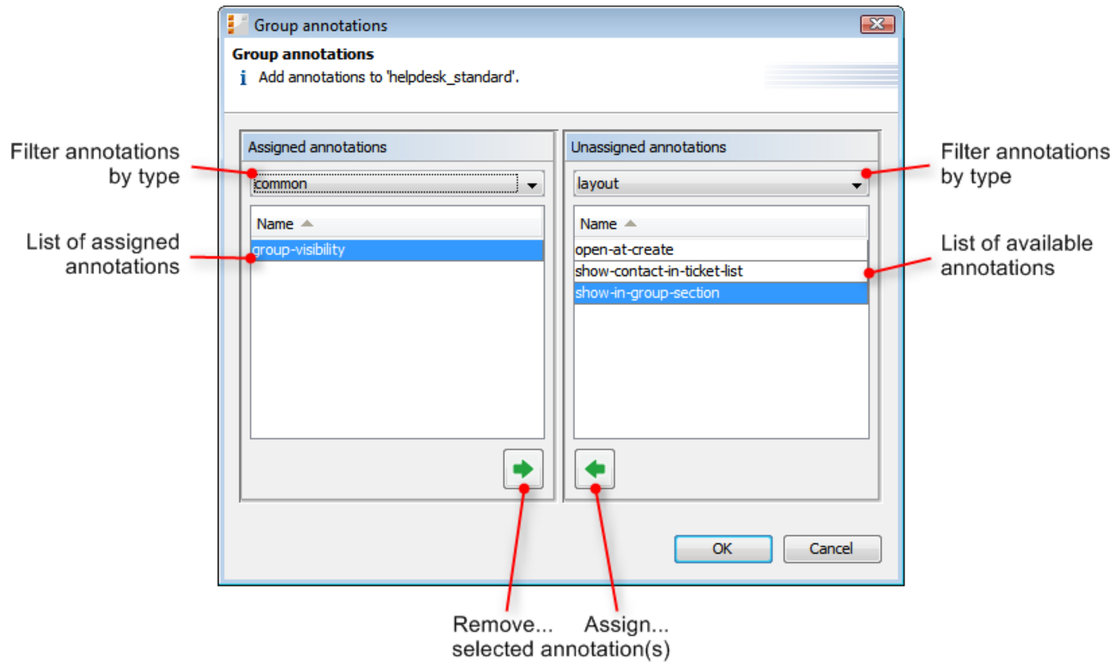


Fig. 3: ConSol*CM Admin Tool - Custom Field Administration: Assign Custom Field Group Annotations

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

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.

Custom field groups will appear in the Web Client as they are ordered in the list. Select a group and use the icons and if you want to change the position of this group in the list.

Delete a Custom Field Group

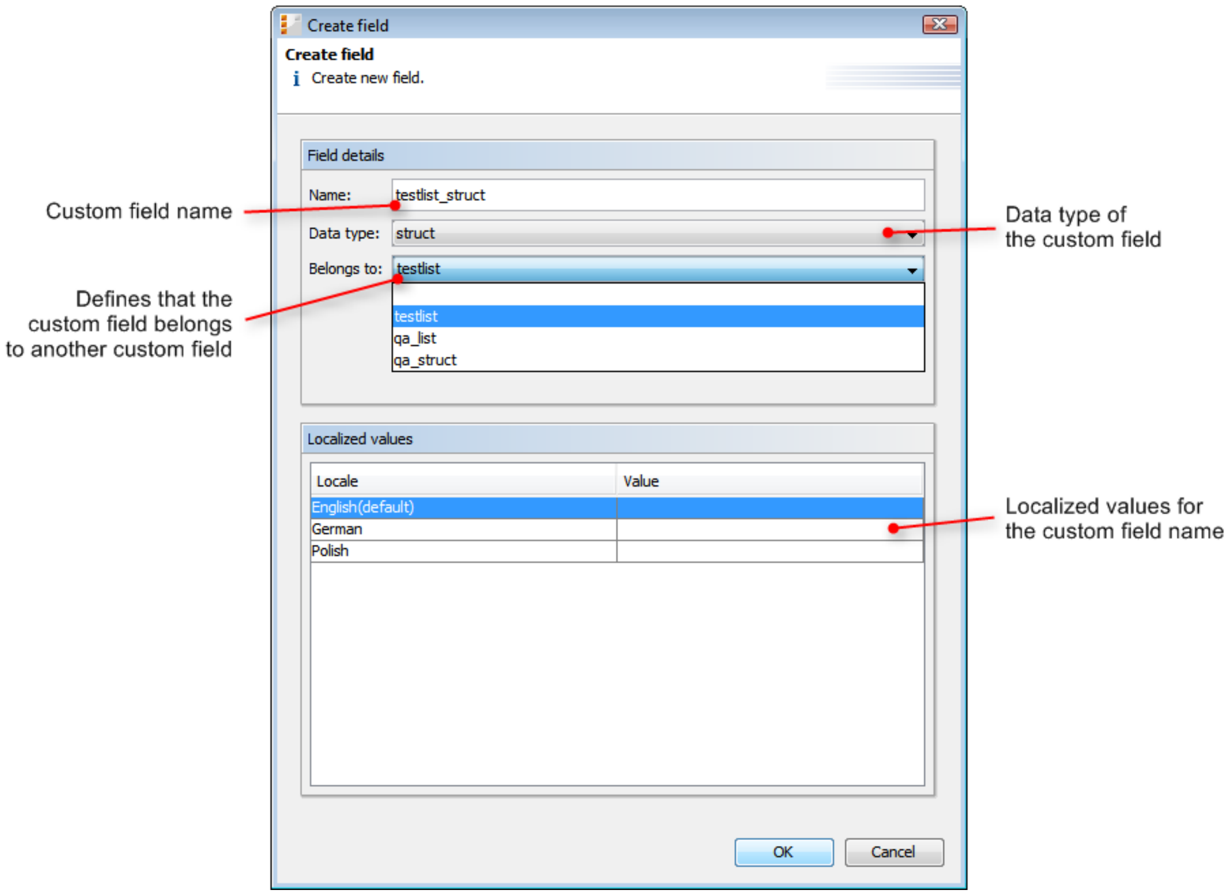
A custom field group can only be deleted if it is not assigned to a queue or a ticket. Otherwise you get a warning stating you can only disable this group (see below). In order to delete a custom field group select it in the list and click on . If you confirm the following dialog with *Yes*, the group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

Enable or Disable a Custom Field Group

If you cannot delete a custom 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 on . The entry in the list is shown in italics afterwards. Disabled custom field groups are not displayed in the Web Client. Just click on below the group list, if you want to enable the group again.

Create a Custom Field

Custom fields contain the data for tickets, e.g. priority, service level, deadline, or hardware module. The fields of a custom field group are created in the right part of the page. Select the desired group first on the left and then click on the icon below the custom field area on the right. The following pop-up window appears:


Custom field name
Defines that the custom field belongs to another custom field
Data type of the custom field
Localized values for the custom field name
Create field
OK
Cancel

Create field

Create new field.

Field details

Name: testlist_struct

Data type: struct

Belongs to: testlist

testlist
qa_list
qa_struct

Localized values

Locale	Value
English(default)	
German	
Polish	

Fig. 4: ConSol*CM Admin Tool - Custom Field Administration: Create a Custom Field for Ticket Data

- **Name:**
  Enter a name for the custom field. The name must be unique.
- **Data type:**
  Choose one of the following data types for the new custom field:
  - **boolean**
    Values: true/false. Depending on the annotation *boolean-type*, the value is displayed as check box, radio buttons, or drop-down list.

> When you work with scripts, either in CM workflows or in the Admin Tool, please note that the **behavior of boolean fields** which are represented as checkboxes, i.e. with annotation *boolean-type* = *checkbox* (default) is different depending on the CM version!
>
> - In CM versions prior to 6.9.4.0:
>   When a boolean field has not been touched, its value is *false*. If it is checked, its value is *true*, and if it is unchecked again, its value is *false* again.
>
> - In version 6.9.4.0 and up:
>   When 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*.
>
> Fields which already have been set in the database will not be changed during an update from a version prior to 6.9.4.0 to a version 6.9.4.0 and up.
>
> Boolean fields represented as radio buttons (annotation *boolean-type* = *radio*) or drop-down menu (annotation *boolean-type* = *select*) have always shown and will show in future the behavior as described for versions 6.9.4.0 and up, i.e. with *NULL* value when untouched.

-
  - **date**
    Format and accuracy can be set by annotations.
  - **enum**
    For sorted lists. The engineer can choose one of the enum values on the Web Client. Enums and values have to be created previously within the Enum Administration. Select the desired *Enum type* and *group* in the fields below.
  - **list**
    A custom field of this data type is the first step to create 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 custom 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 custom field of the type *struct* first (see below) before you can add the custom 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 custom 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 CF) always belongs to a struct, a struct always belongs to a list.

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

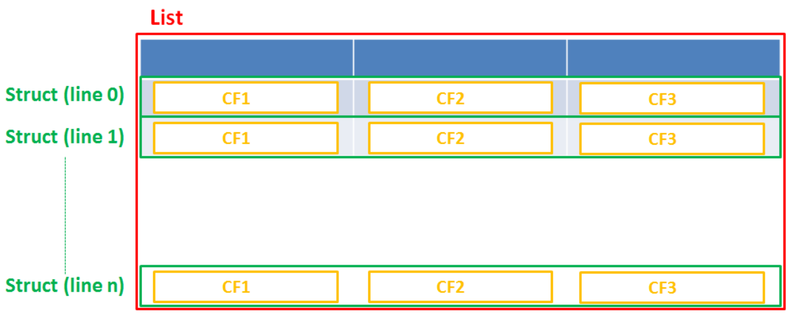
  

  Fig. 5: Scheme: List of Structs
- **number**
  For integer values.
- **fixed point number**
  For numbers with a fractional part, e.g. currencies. You have to enter the total number of digits (*Precision*) and the number of digits that fall to the right of the decimal point (*Scale*) in the respective fields below.
- **string**
  For up to 4000 alphanumeric characters.
- **long string**
  For large objects, unrestricted length.
- **short string**
  For up to 255 alphanumeric characters.
- **contact data reference**
  Special data type used internally for referencing the contacts associated with a ticket.
- **MLA field**
  This data type is used for custom fields that contain hierarchical lists with a tree structure called *MLA* (Multi Level Attributes). The name of the custom field will be the name of the new MLA that has to be defined within the MLA Administration. The group of the custom field has to be referenced when the MLA is created.

- **Belongs to:**
  This field shows the available custom 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 custom field belongs (if applicable).
- **Localized values:**
  Enter the corresponding custom field name in the *Value* field for each additional language. In the Web Client, the name will be displayed in the respective language of the user's web browser locale. If you do not make an entry here the object name, i.e. the content of the *Name* field, will be taken instead.

Attention:

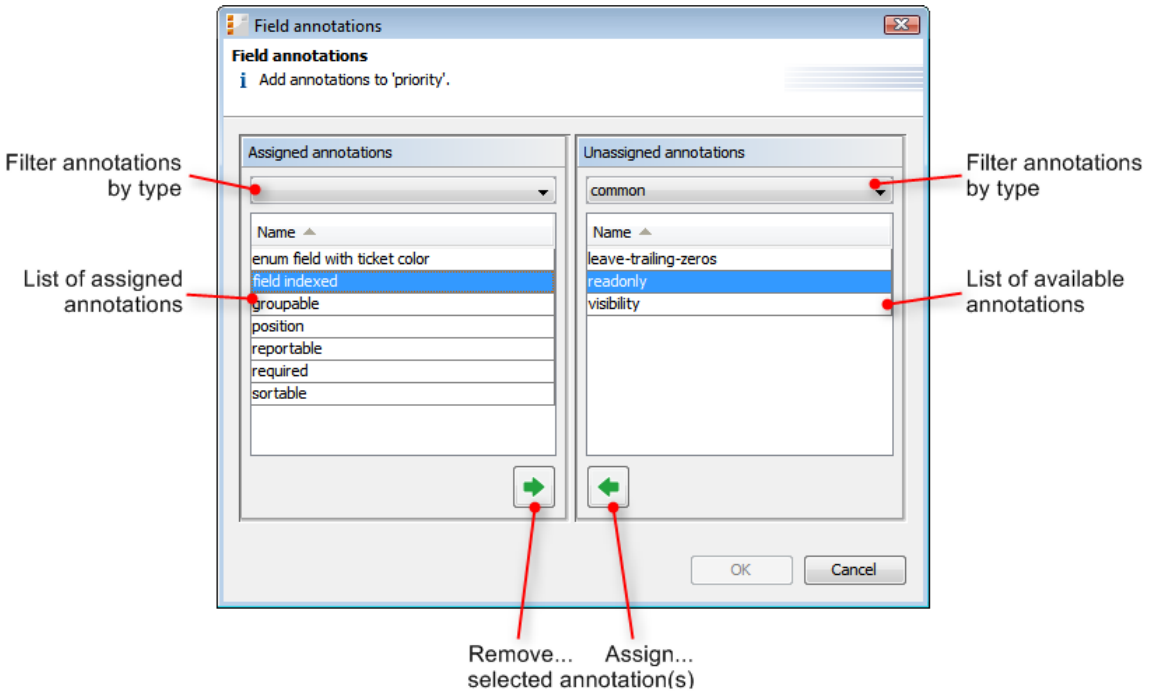
The data type you choose on creating a custom field cannot be changed afterwards!

Edit a Custom Field

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

Annotate a Custom Field

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


Field annotations
Field annotations
Add annotations to 'priority'.
Assigned annotations
Unassigned annotations
common
Name
OK
Cancel
Filter annotations by type
Filter annotations by type
List of assigned annotations
List of available annotations
Remove... Assign...
selected annotation(s)

Name
enum field with ticket color
field indexed
groupable
position
reportable
required
sortable

Name
leave-trailing-zeros
readonly
visibility

Fig. 6: ConSol*CM Admin Tool - Custom Field Administration: Assign Custom Field Annotations

The right part of the window contains the available annotations. Using the selection field above the list you can filter the display according to annotation type. Select the desired annotations and move them to the *Assigned annotations* list on the left by clicking on . This list can also be filtered according to annotation type. Click on *OK* to assign the annotations to the custom 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 into the corresponding *Value* field and typing the desired value. Press the *Enter* key afterwards.

Custom 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 icons and below.

Delete a Custom Field

A custom 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 custom field select it in the list and click on . If you confirm the following dialog with *Yes*, the custom field will be removed from the list and the system.

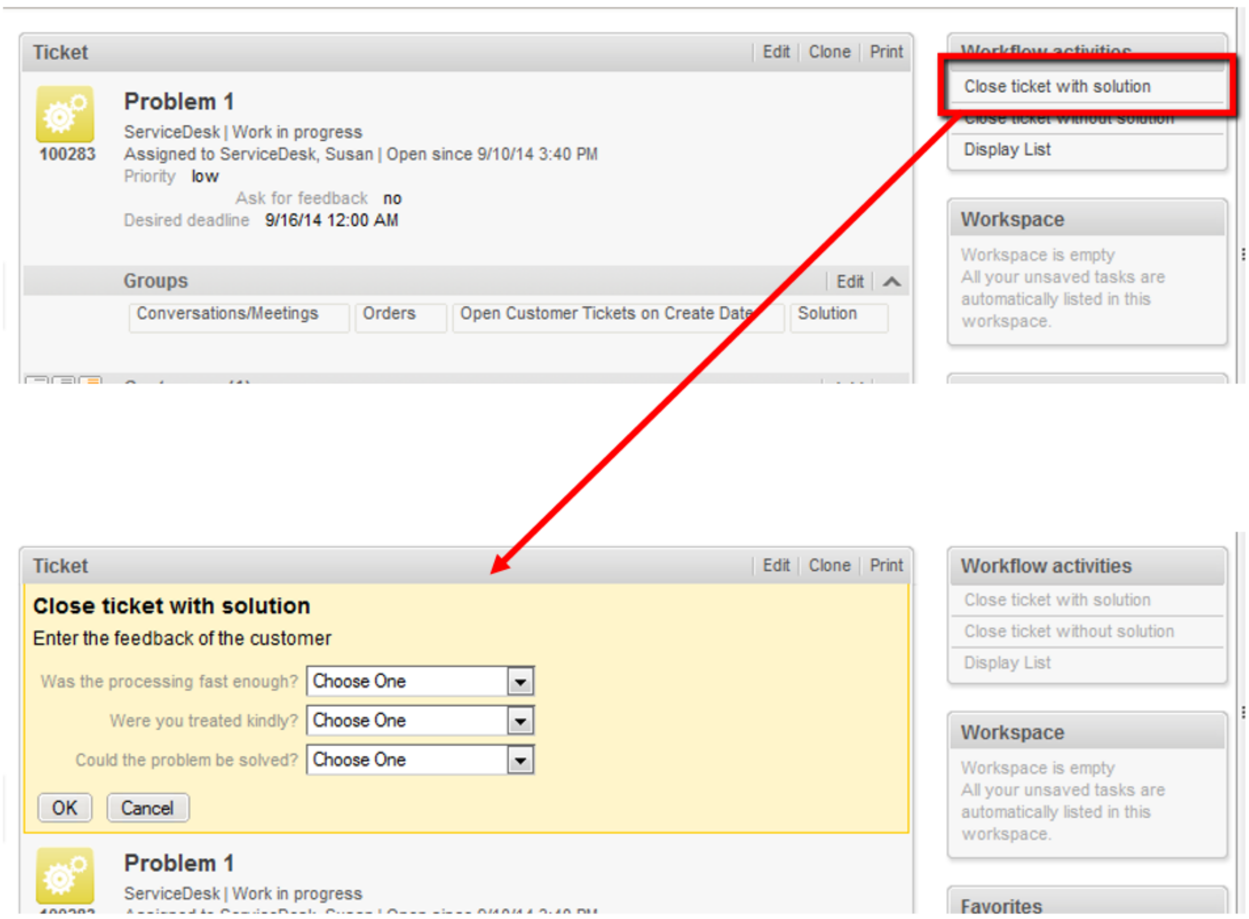
Enable or Disable a Custom Field

If you cannot delete a custom 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 on . The entry in the list is shown in italics afterwards. A disabled custom field is not displayed in the Web Client. Just click on below the custom field list, if you want to enable the field again.

15.2.2 Tab Activity Form Data

Short Introduction to Activity Control Forms (ACFs)

An Activity Control Form is a web form which is displayed during the process, i.e. when the engineer works on the ticket. When he/she clicks on a workflow activity, the ACF is displayed first. When the engineer has filled in and saved the required data, the process can continue. i.e. the workflow activity is executed.



Ticket | Edit | Clone | Print

Problem 1

100283

ServiceDesk | Work in progress

Assigned to ServiceDesk, Susan | Open since 9/10/14 3:40 PM

Priority low

Ask for feedback no

Desired deadline 9/16/14 12:00 AM

Groups | Edit

Conversations/Meetings | Orders | Open Customer Tickets on Create Date | Solution

Workflow activities: Close ticket with solution, Close ticket without solution, Display List

Workspace: Workspace is empty. All your unsaved tasks are automatically listed in this workspace.

Ticket | Edit | Clone | Print

Close ticket with solution

Enter the feedback of the customer

Was the processing fast enough?	Choose One
Were you treated kindly?	Choose One
Could the problem be solved?	Choose One

OK | Cancel

Problem 1

ServiceDesk | Work in progress

Workflow activities: Close ticket with solution, Close ticket without solution, Display List

Workspace: Workspace is empty. All your unsaved tasks are automatically listed in this workspace.

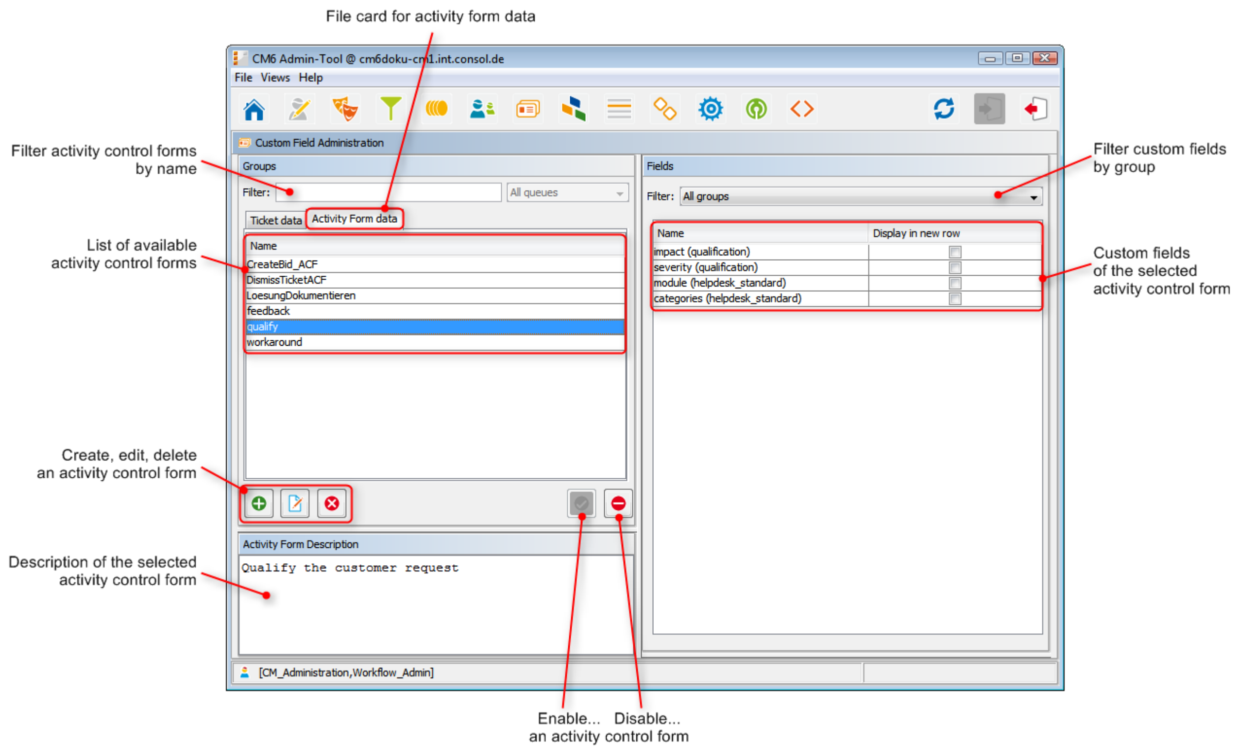
Favorites

Fig. 7: Example ACF

The custom fields which are used in an ACF have to be defined first, as regular CFs. They can belong to one or more CF groups. The ACFs is integrated into the process (workflow) using the ConSol*CM Process Designer. Here, you can also define which CFs 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 when the customer has *Gold* or *Platin* status. ACF dependencies are configured using scripts. This is explained in detail in the *ConSol*CM Process Designer Manual*.

Definition of Activity Control Forms (ACFs)

On 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 when 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 custom fields already created on the tab *Ticket data*. An ACF can contain custom fields of more than one custom field group. However, all custom fields have to be allowed in 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.


File card for activity form data
Filter activity control forms by name
List of available activity control forms
Create, edit, delete an activity control form
Description of the selected activity control form
Filter custom fields by group
Custom fields of the selected activity control form
Enable... Disable... an activity control form
CM6 Admin-Tool @ cm6doku-cm1.int.consol.de
File Views Help
Custom Field Administration
Groups
Filter:
All queues
Ticket data
Activity Form data
Fields
Filter: All groups
Activity Form Description
[CM_Administration,Workflow_Admin]

Name
CreateBid_ACF
DismissTicketACF
LoesungDokumentieren
feedback
qualify
workaround

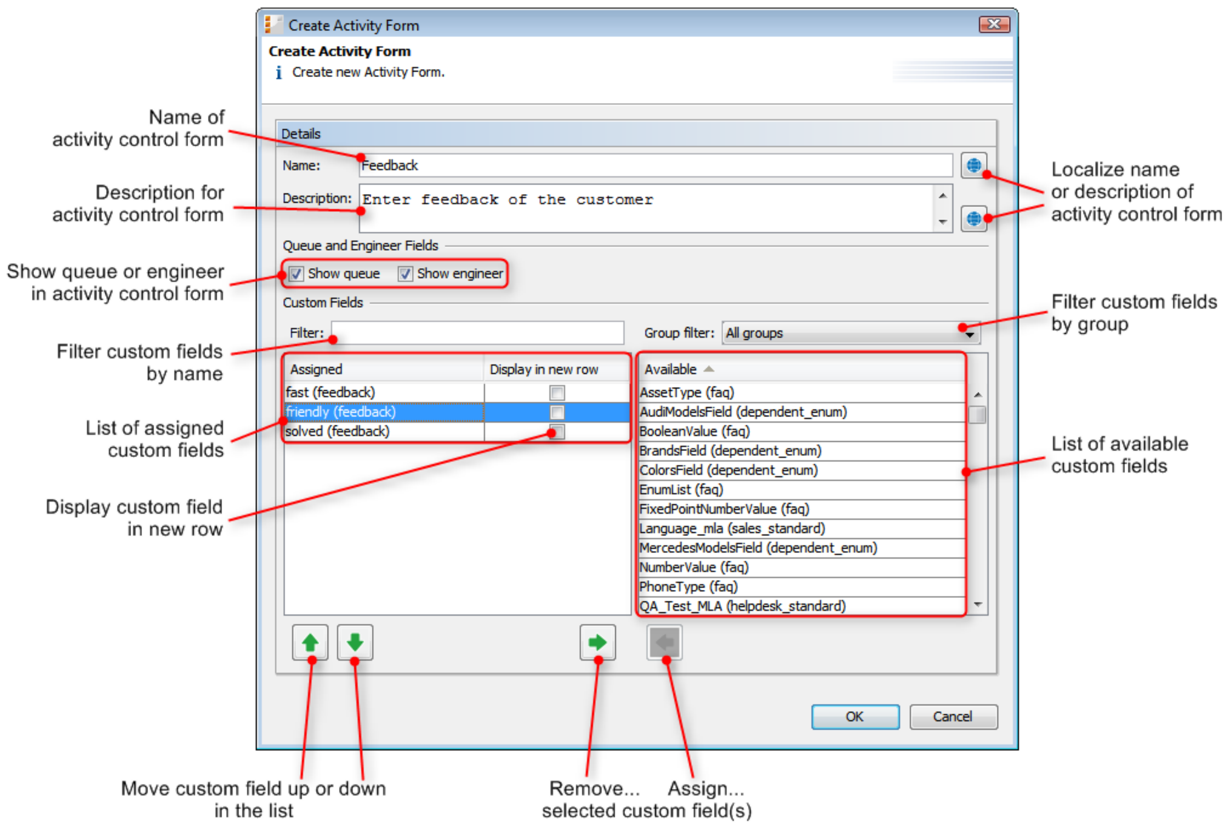
Name	Display in new row
impact (qualification)	☐
severity (qualification)	☐
module (helpdesk_standard)	☐
categories (helpdesk_standard)	☐

```
Qualify the customer request
```

Fig. 8: ConSol*CM Admin Tool - Custom Field Administration: Activity Form Data

Create an Activity Control Form

To create an ACF just click on the ⊕ icon below the list on the left side of the page. The following pop-up window appears.


Create Activity Form

Create Activity Form

Create new Activity Form.

Details

Name: Feedback

Description: Enter feedback of the customer

Queue and Engineer Fields

[✓] Show queue [✓] Show engineer

Custom Fields

Filter:

Group filter: All groups

Assigned	Display in new row
fast (feedback)	☐
friendly (feedback)	☐
solved (feedback)	☐

Available
AssetType (faq)
AudiModelsField (dependent_enum)
BooleanValue (faq)
BrandsField (dependent_enum)
ColorsField (dependent_enum)
EnumList (faq)
FixedPointNumberValue (faq)
Language_mla (sales_standard)
MercedesModelsField (dependent_enum)
NumberValue (faq)
PhoneType (faq)
QA_Test_MLA (helpdesk_standard)

OK
Cancel
Name of activity control form
Description for activity control form
Show queue or engineer in activity control form
Filter custom fields by name
List of assigned custom fields
Display custom field in new row
Localize name or description of activity control form
Filter custom fields by group
List of available custom fields
Move custom field up or down in the list
Remove... Assign... selected custom field(s)

Fig. 9: ConSol*CM Admin Tool - Custom Field Administration: Create an Activity Control Form

Please enter or select the following data:

- **Name**
  Enter the name of the ACF in this field. You can localize the name by clicking on 🌐 (see below).
- **Description**
  Enter a description for the ACF in this field. The description will be shown as the title of the ACF in the Web Client. You can localize the description by clicking on 🌐 (see below).
- **Show queue**
  Mark this check box if the queue of the ticket shall be displayed with the ACF in the Web Client.
- **Show Engineer**
  Mark this check box if the engineer of the ticket shall be displayed with the ACF in the Web Client.
- **Filter**
  You can enter a string of characters into this field to filter the assigned custom fields by name.
- **Group filter**
  Select a group of custom fields from this list if you want to display only custom fields belonging to this group in the list of available custom fields below.
- **Custom field lists**
  The list on the right shows the available custom fields with their respective custom field group. You can sort the entries in ascending or descending order by clicking into the title field of the list. The icons ▲ or ▼ show the sort order. Select the custom fields for the ACF in this list and move them to the list *Assigned* on the left by clicking on ⬅ .
  For each assigned custom field you can define if it shall be displayed in a new row by marking the corresponding check box. The assigned custom 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 on or . To remove assigned custom fields, select them and click on .

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

Edit an Activity Control Form

If you want to edit an ACF, select it in the list and click on 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*.

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 you can only disable this ACF (see below). In order to delete an ACF select it in the list and click on . If you confirm the following dialog with *Yes*, the ACF will be removed from the list and the system.

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 on . 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 on below the ACF list, if you want to enable the form again.

Localize an Activity Control Form

By clicking on the respective globe icon in the create or edit window you can localize name and description of an ACF. The pop-up window *Localize* shows the available locales on the left side. Enter the corresponding name or description in the *Value* field for each additional language on the right. After clicking *Save* the name or description will be displayed in the respective language of the engineer's locale.

15.2.3 Frequently Used Annotations

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

Annotation	Annotation Type	Description	Value	Comment
groupable	cmweb-common	Enables grouping of fields in the ticket list.		Used only with *enum* custom fields. No value is needed.
sortable	cmweb-common	Used to enable sorting of the ticket list.		Used with custom fields of type *enum* or of type *Date*.

Annotation	Annotation Type	Description	Value	Comment
				No value is needed.
readonly	common	Used to indicate that the custom field cannot be modified.	true / false	Field is read only if value is set to *tru e*. Lack of value or any value except *f alse* is also treated as *true*.
visibility	common	Defines when the field is visible.	edit	Field will be displayed in *edit* mode.
			view	Field will be displayed in *view* mode.
			none	Field is not visible.
text-type	component-type	Defines the possible types of a string field.	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	Input will be displayed as a label, i.e. the field is displayed only, no input is possible.
			url	Input will be displayed as a hyperlink in *view* mode. String has to match a specific URL pattern.

Annotation	Annotation Type	Description	Value	Comment
reportable	dwh	Indicates that the field is reportable and that it should be transferred to the DWH.	true / false	Field is reportable if value is set to *tru e*.
field indexed	indexing	Used to indicate that a database index will be created for this field.	transitive (default)	All data is displayed (ticket and customer).
			unit	Used for customer data. Only the unit and the parent unit (i.e. company) is given as a search result, no tickets are provided.
			local	Used for customer data. Only the unit is given as a search result, no company and no tickets are displayed.
			not indexed	Field is not indexed.
position	layout	Defines the position of a field within a grid layout.	<number>;<number>	Values define *row* and *column* (row; column), numbering starts at 0;0. If no values are set, the custom field will take the next free grid cell.
		Defines the position of a field within a list (struct).	0;<number>	Only the *column* value is used, the *r ow* value is ignored.

Annotation	Annotation Type	Description	Value	Comment
colspan	layout	Defines how many columns are reserved for the field in the layout.	<number>	Number of columns.
rowspan	layout	Indicates how many rows within the layout are occupied by this field.	<number>	Number of rows.
field-group	layout	Allows grouping of fields in *view* mode. Annotation is ignored in *edit* mode.	<string>	To group fields the same string value has to be set in the annotation of each field.
fieldsize	layout	Displayed field size within ticket layout.	<rows>;<cols>	For *string* custom fields with annotation *text-type* and value *text area*. <rows> defines the number of displayed rows and <cols> defines the number of characters displayed per row. Used only for layout purposes.
			<number>	For *enum* custom fields. Defines how many values are directly visible in the list box. Used only for layout purposes.
enum field with ticket color	ticket display	Defines the background color of the ticket icon for ticket list and ticket.	true / false	The field has to exist within enum administration

Annotation	Annotation Type	Description	Value	Comment
				where lists, values, and colors are defined.
accuracy	validation	Used for date fields, to define the level of detail displayed.	date (default)	Show date without time.
			date-time	Show date with time.
			only-time	Show only time, no date.
format	validation	Used to check the correct format of *d ate* fields.	<date format>	The pattern for the date is based on *Si mpleDateFormat*, e.g. dd.mm.yyyy.
maxLength	validation	Defines the maximum length of input for *string* custom fields.	<number>	May be used with *s tring* custom fields.
minLength	validation	Defines the minimum length of input for *string* custom fields.	<number>	May be used with *s tring* custom fields.
required	validation	Indicates that this is a required field.	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.

15.3 Related Topics

- Annotations
- Enums
- MLA
- Queues
- Workflow (see *ConSol*CM Process Designer Manual*)

16 Managing Sorted Lists: Enum Administration

- Introduction to Enum Administration
- Enum Administration Using the Admin Tool
    - Enum Types
        - Create an Enum Type
        - Edit an Enum Type
        - Delete an Enum Type
        - Enable or Disable an Enum Type
    - Enum Groups
        - Create an Enum Group
        - Edit an Enum Group
        - Delete an Enum Group
        - Enable or Disable an Enum Group
    - Enum Values
        - Create an Enum Value
        - Edit an Enum Value
        - Set a Background Color
        - Delete an Enum Value
        - Change the Order of the Value List
        - Enable or Disable an Enum Value
    - Placing an Enum in the Data Model
        - Enums for Ticket Data
        - Enums for Customer Data
- Related Topics

16.1 Introduction to Enum Administration

In the *Enum Administration* you can configure enums = sorted lists. They are part of the ConSol*CM6 data concept. You define an enum once and can use it multiple times:

- as a selection list (in drop-down menus) for custom fields or data object group fields of type *enum*
- as hierarchical lists for custom fields or data object group fields of type *MLA field* (Multi Level Attributes, see section MLA Administration)
- as dependent enums, i.e. as enums that form a hierarchy, a data construct implemented by Dependent Enum Scripts

Attention:

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 custom field values or data object group field values), you have to complete one of the following steps:

- create a custom field of type *enum* and assign the respective enum there
- create a data object group field of type *enum* and assign the respective enum there
- create an MLA which is linked automatically as custom field to the custom field group or as data object group field to the data object group that has been indicated during MLA set-up
- create a *dependent enum script* and assign this to a custom field group

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

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

- *enum-type* not set or *enum-type* = *select*:
  drop-down menu (default, see example in the picture below)
- *enum-type* = *radio*:
  radio buttons
- *enum-type* = *autocomplete*:
  self-completing (autocomplete) list

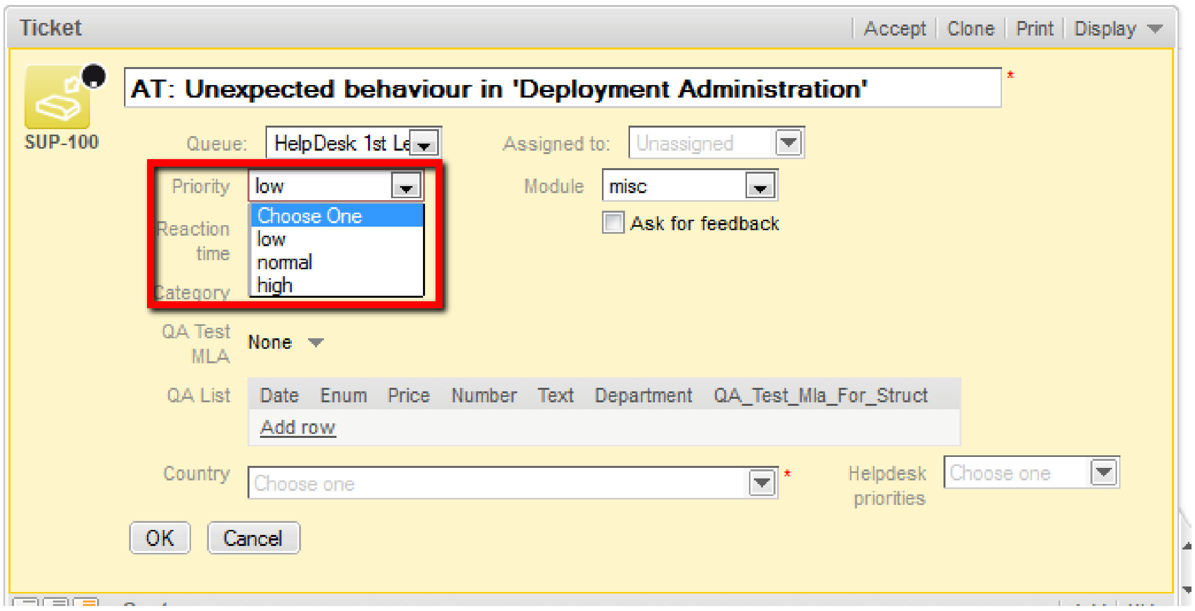
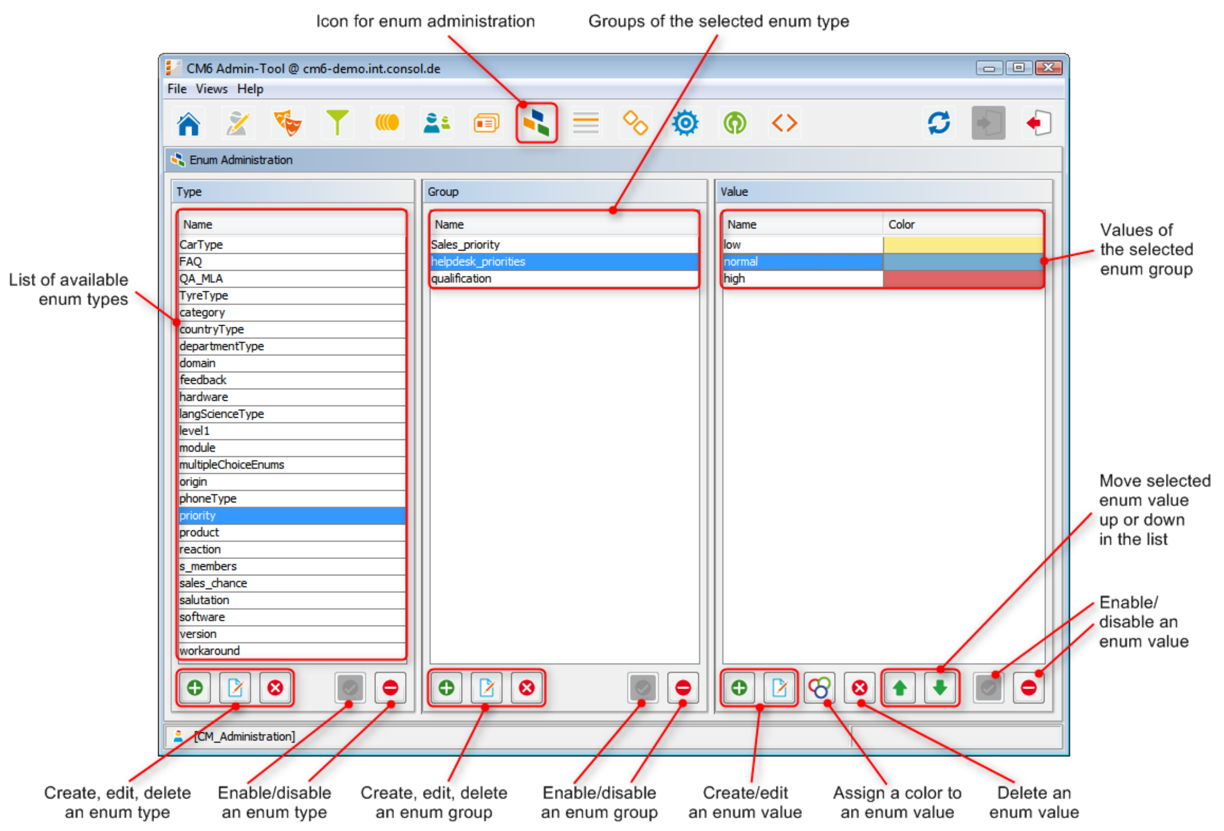


Fig. 1: ConSol*CM Web Client - Enum for Priority (Localized Enum Values Displayed as List Items)

16.2 Enum Administration Using the Admin Tool

Enums are organized in three levels:

- **Type**
  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**
  The *group* represents a group of enum values, i.e. the list.
- **Value**
  The *value* represents one value within a list.


Icon for enum administration
Groups of the selected enum type
CM6 Admin-Tool @ cm6-demo.int.consol.de
File Views Help
Enum Administration
Type
Group
Value
List of available enum types
Values of the selected enum group
Move selected enum value up or down in the list
Enable/ disable an enum value
[CM_Administration]
Create, edit, delete an enum type
Enable/disable an enum type
Create, edit, delete an enum group
Enable/disable an enum group
Create/edit an enum value
Assign a color to an enum value
Delete an enum value

Name
CarType
FAQ
QA_MLA
TyreType
category
countryType
departmentType
domain
feedback
hardware
langScienceType
level1
module
multipleChoiceEnums
origin
phoneType
priority
product
reaction
s_members
sales_chance
salutation
software
version
workaround

Name
Sales_priority
helpdesk_priorities
qualification

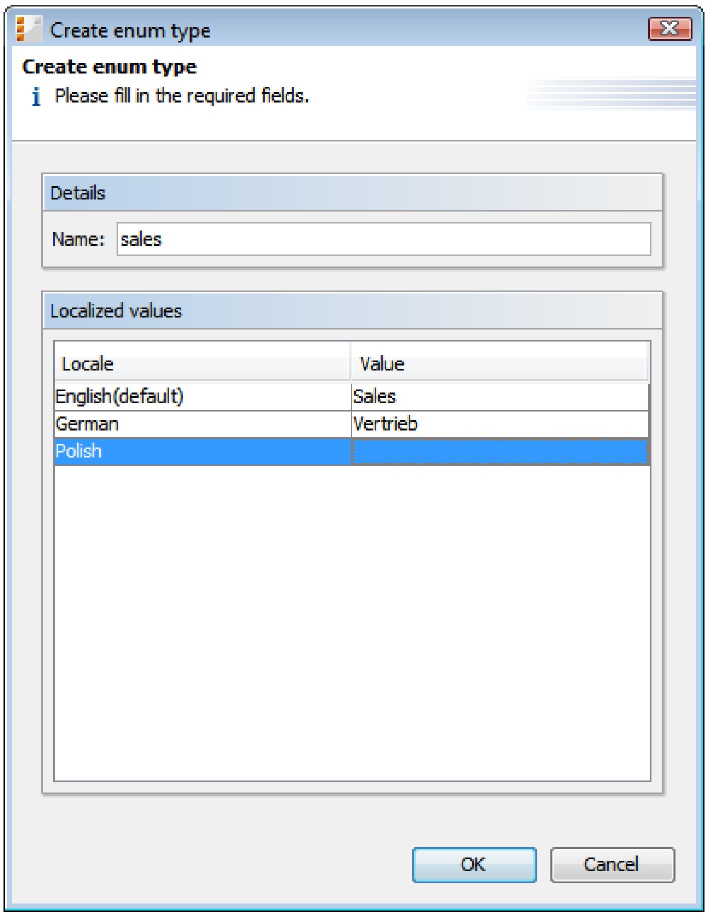
Name	Color
low	🟨
normal	🟦
high	🟥

Fig. 2: ConSol*CM Admin Tool - Enum Administration

16.2.1 Enum Types

Create an Enum Type

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



Name:	sales

Locale	Value
English(default)	Sales
German	Vertrieb
Polish	

Fig. 3: ConSol*CM Admin Tool - Enum Administration: Create 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. If you do not make an entry here the object name, i.e. the content of the *Name* field, will be taken instead.

Click on *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 on . 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 on . If you confirm the following dialog with *Yes*, the type will be removed from the list and the system.

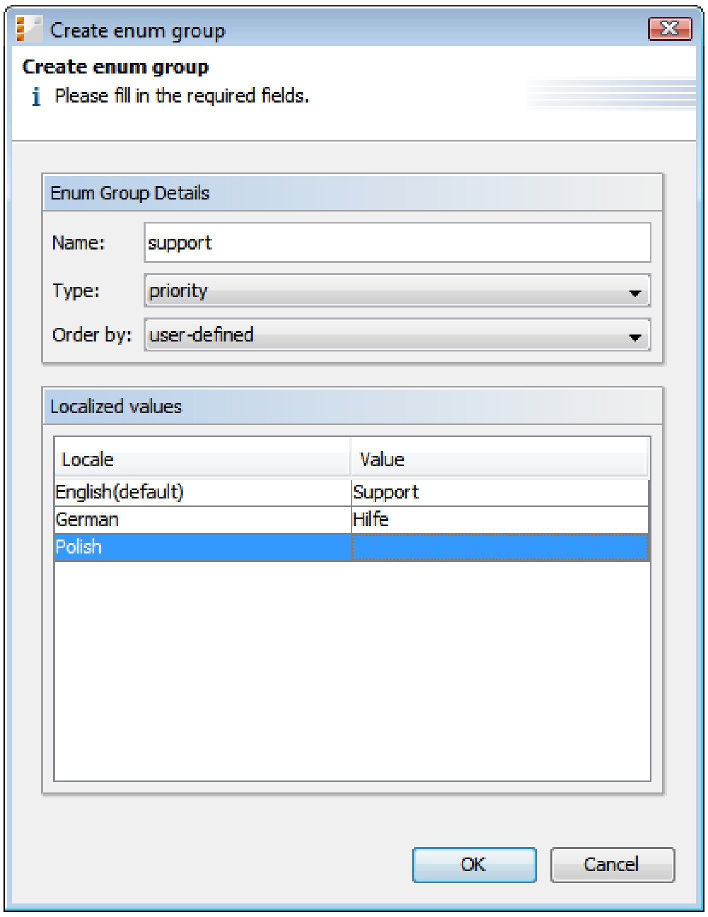
Enable or Disable an Enum Type

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 on ⛔. The entry in the list is shown in italics afterwards. Just click on ✓ below the type list, if you want to enable the type again.

16.2.2 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 on ⊕ below the *Group* area. The following pop-up window appears:



Name: support
Type: priority
Order by: user-defined

Locale	Value
English(default)	Support
German	Hilfe
Polish	

Fig. 4: ConSol*CM Admin Tool - Enum Administration: Create 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. If you select *user-defined* you can determine the sort order by means of arrow icons below the value list. If you choose *name* the values will be ordered alphabetically when you create them.
- **Localized values:**
  Enter the corresponding group name in the *Value* field for each additional language. If you do not make an entry here the object name, i.e. the content of the *Name* field, will be taken instead.

Click on *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 on . 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 on . 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

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 on . The entry in the list is shown in italics afterwards. Just click on below the group list, if you want to enable the group again.

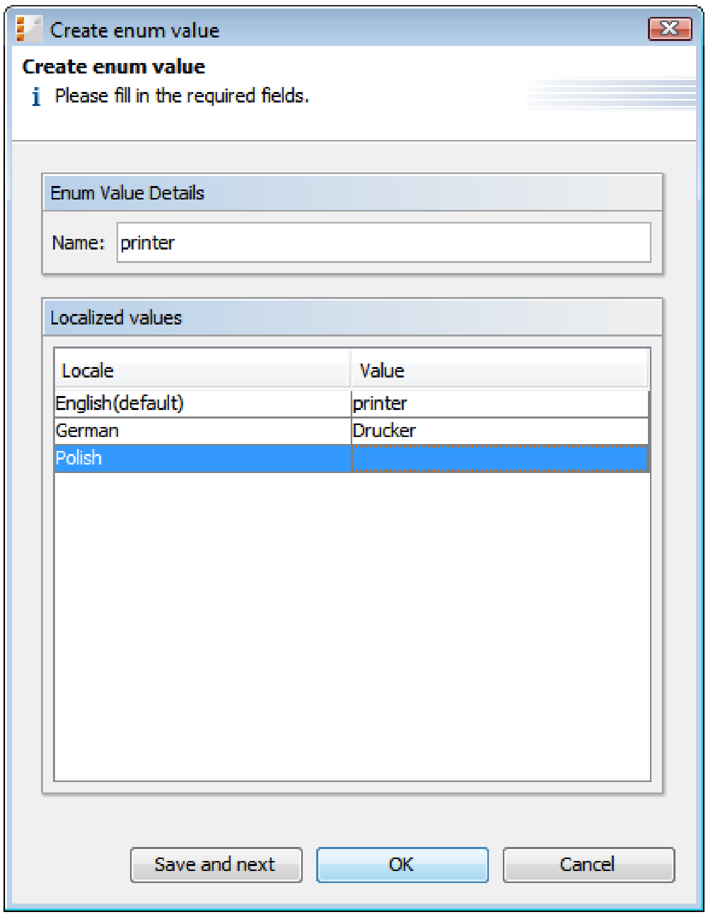
Information:

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

16.2.3 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 on below the *Value* area. The following pop-up window appears.



Create enum value

Please fill in the required fields.

Enum Value Details

Name: printer

Localized values

Locale	Value
English(default)	printer
German	Drucker
Polish	

Fig. 5: ConSol*CM Admin Tool - Enum Administration: Create an Enum Value

- **Name:**
  Enter a value which shall 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. In the Web Client's user interface the name will be displayed in the language of the user's web browser locale. If you do not make an entry here, the technical object name, i.e. the content of the *Name* field, will be displayed.

Click on *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 on . 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, if you click on . This can be used for example for priorities. The priority of a ticket in the Web Client can be recognized immediately by its background color. This will be effective when the respective annotation is set, see the following info box.

Information:

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 custom field of type *enum* in the Custom Field Administration. For example, you can use the custom field *priority* to determine the ticket icon color in the helpdesk queue and the custom 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 on the desired color to set it for the marked list value. You can check the selected color in the *Preview* area. Click on *OK* to save your choice. Click on *Reset* if you want to return to the last saved color.

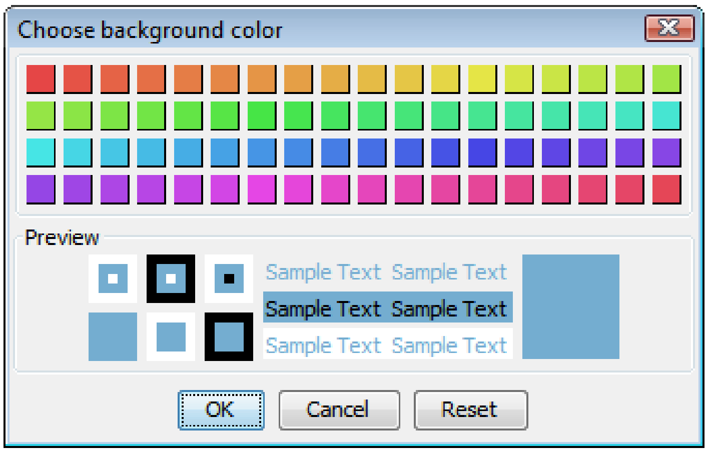


Fig. 6: ConSol*CM Admin Tool - Enum Administration: Set a Background Color

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 on ⊗ . If you confirm the following dialog with *Yes*, the value will be removed from the list and the system.

Attention:

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 on ⬆ to move the selected value one line up resp. click on ⬇ 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

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 on . The entry in the list is shown in italics afterwards and the value is not available in the Web Client any longer. Just click on below the value list, if you want to enable the value again.

Information:

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

16.2.4 Placing an Enum in the Data Model

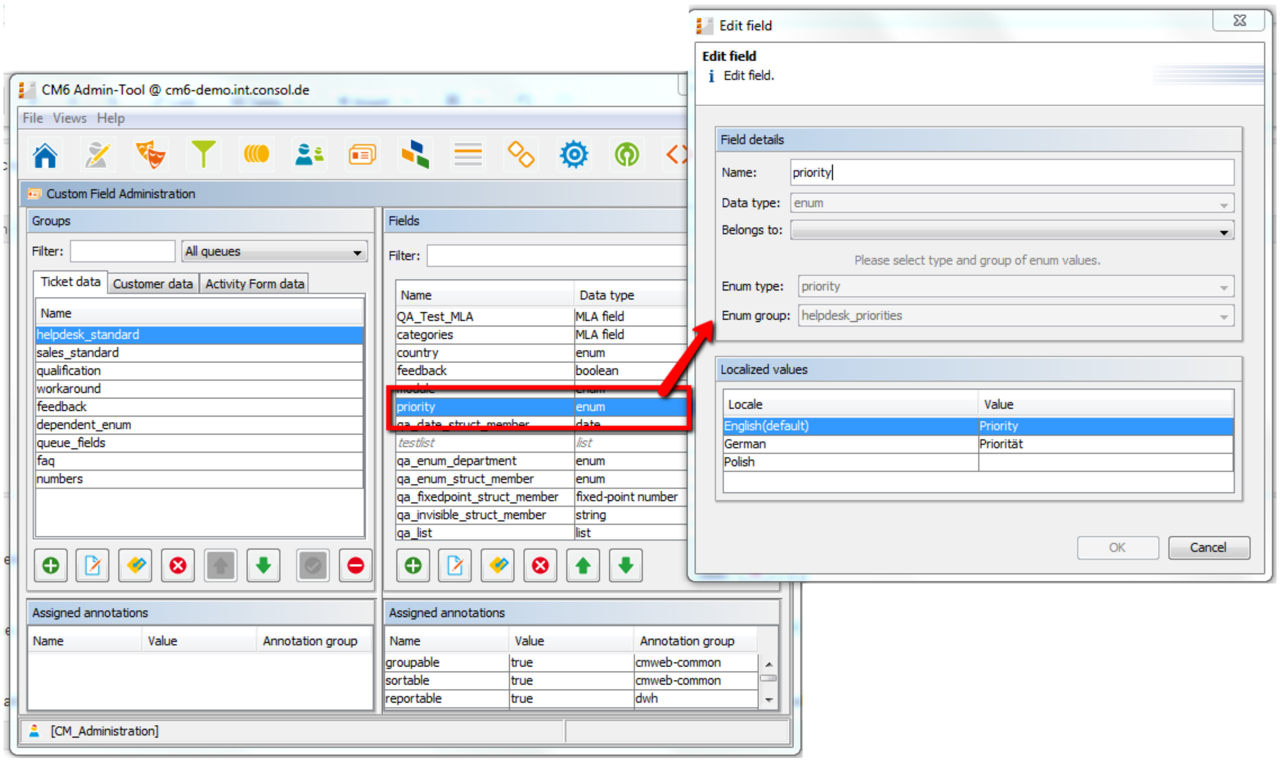
Enums can be used in custom fields (i.e. for ticket data) and for data object group fields (i.e. for customer data). The example below shows an enum for ticket data.

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 custom field of type *enum* has to be defined. Please see section Custom Field Administration for a detailed explanation of the work with custom 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 data object group field of type *enum* has to be defined. Please see section Data Object Group Field Management and GUI Design for Customer Data for a detailed explanation of the work with data object group fields.



Groups:

Name
helpdesk_standard
sales_standard
qualification
workaround
feedback
dependent_enum
queue_fields
faq
numbers

Fields:

Name	Data type
QA_Test_MLA	MLA field
categories	MLA field
country	enum
feedback	boolean
module	enum
priority	enum
qa_date_struct_member	date
testlist	list
qa_enum_department	enum
qa_enum_struct_member	enum
qa_fixedpoint_struct_member	fixed-point number
qa_invisible_struct_member	string
qa_list	list

Assigned annotations (group):

Name	Value	Annotation group

Assigned annotations (field):

Name	Value	Annotation group
groupable	true	cmweb-common
sortable	true	cmweb-common
reportable	true	dwh

Field details:

Name:	priority
Data type:	enum
Belongs to:	

Please select type and group of enum values.

Enum type:	priority
Enum group:	helpdesk_priorities

Localized values:

Locale	Value
English(default)	Priority
German	Priorität
Polish	

Fig. 7: ConSol* Admin Tool - Definition of a Custom Field of Type Enum

16.3 Related Topics

- Custom fields
- MLA

17 MLA Administration

- Introduction to MLA Administration
- MLA Administration Using the Admin Tool
    - Create an MLA
        - Create an MLA Level
        - Edit a Level Value
        - Delete a Level
        - Enable or Disable a Level
    - Edit an MLA
    - Delete an MLA
    - Enable or Disable an MLA
- Related Topics

17.1 Introduction to MLA Administration

MLA means *Multi Level Attributes*. 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 or for customer data.

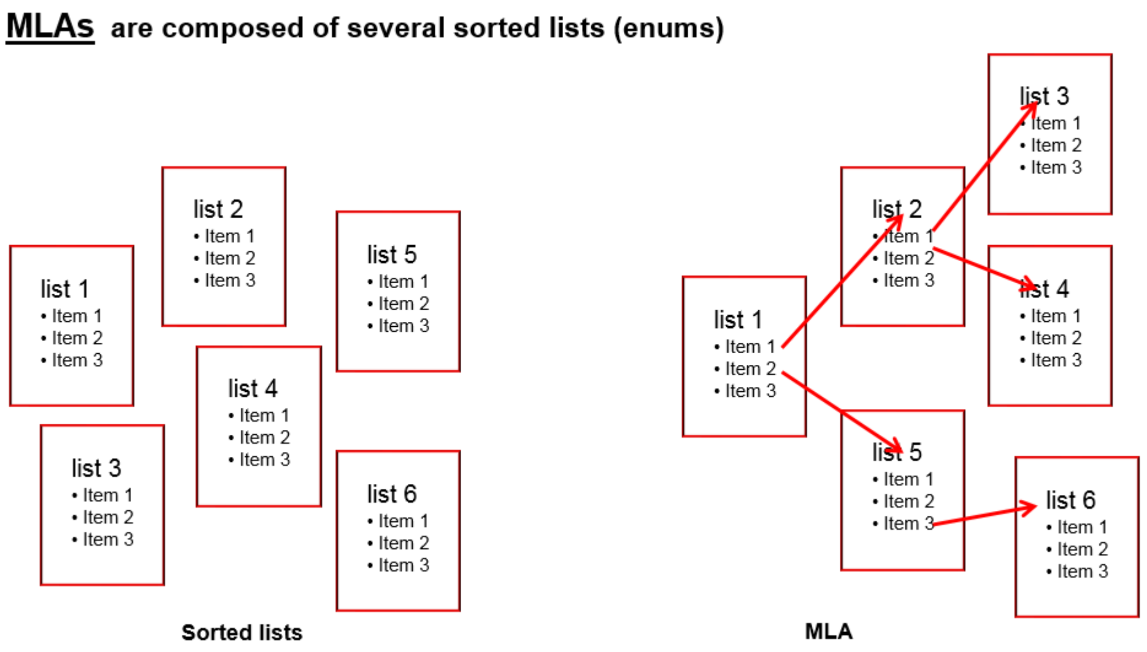
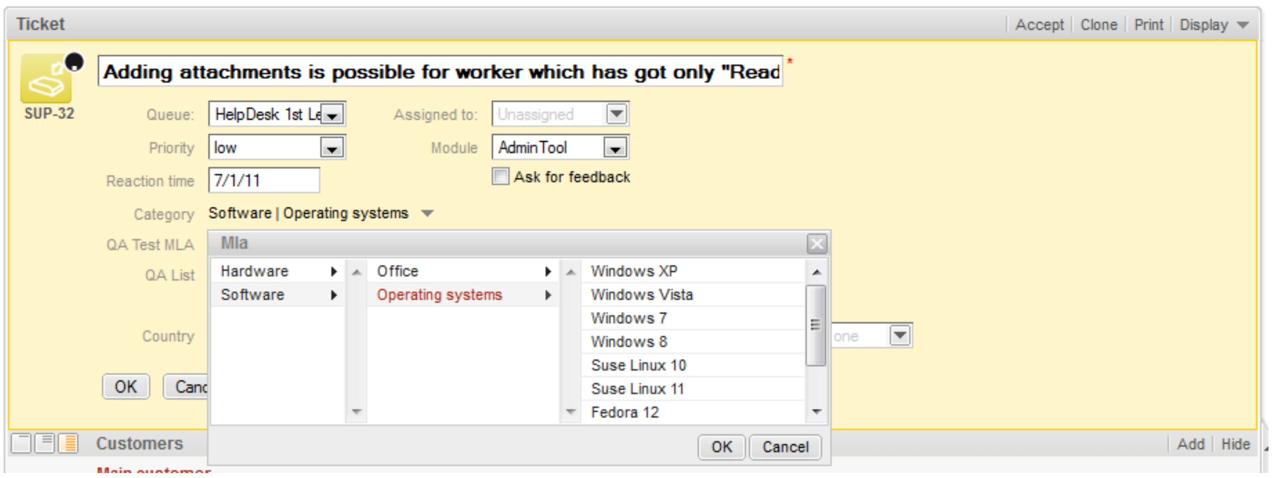

MLAs are composed of several sorted lists (enums)
list 1 • Item 1 • Item 2 • Item 3
list 2 • Item 1 • Item 2 • Item 3
list 3 • Item 1 • Item 2 • Item 3
list 4 • Item 1 • Item 2 • Item 3
list 5 • Item 1 • Item 2 • Item 3
list 6 • Item 1 • Item 2 • Item 3
Sorted lists
MLA

Fig. 1: 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 *Mainboard* for item *Hardware* and so on. The picture below shows such an MLA in the Web Client.



Adding attachments is possible for worker which has got only "Read

Queue:	HelpDesk 1st Le	Assigned to:	Unassigned
Priority	low	Module	AdminTool
Reaction time	7/1/11		Ask for feedback
Category	Software | Operating systems		
QA Test MLA			
QA List			
Country			one

Fig. 2: 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 Enum Administration and are just referenced when a new MLA level is defined.
- can be completely created within the MLA Administration during set-up of a new MLA.

17.2 MLA Administration Using the Admin Tool

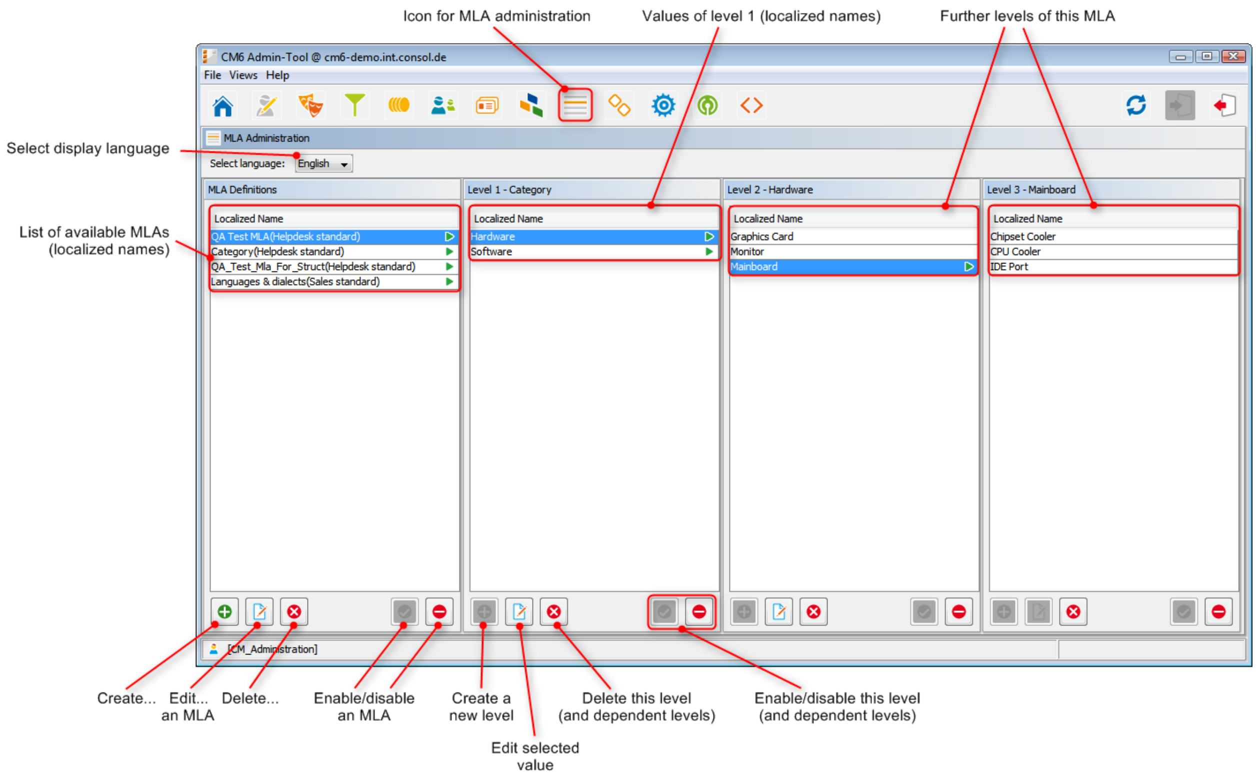
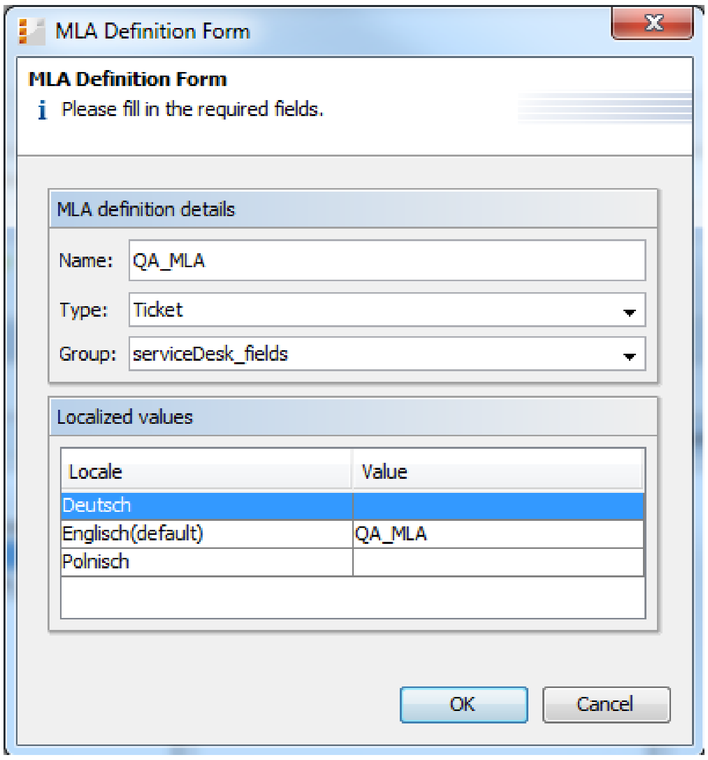

Icon for MLA administration
Values of level 1 (localized names)
Further levels of this MLA
Select display language
List of available MLAs (localized names)
CM6 Admin-Tool @ cm6-demo.int.consol.de
File Views Help
MLA Administration
Select language: English
MLA Definitions
Level 1 - Category
Level 2 - Hardware
Level 3 - Mainboard
Localized Name
QA Test MLA(Helpdesk standard)
Category(Helpdesk standard)
QA_Test_Mla_For_Struct(Helpdesk standard)
Languages & dialects(Sales standard)
Hardware
Software
Graphics Card
Monitor
Mainboard
Chipset Cooler
CPU Cooler
IDE Port
[CM_Administration]
Create... Edit... an MLA
Delete...
Enable/disable an MLA
Create a new level
Edit selected value
Delete this level (and dependent levels)
Enable/disable this level (and dependent levels)

Fig. 3: ConSol*CM Admin Tool - MLA Administration

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 above the list.

17.2.1 Create an MLA

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



Name:	QA_MLA
Type:	Ticket
Group:	serviceDesk_fields

Locale	Value
Deutsch	
Englisch(default)	QA_MLA
Polnisch	

Fig. 4: ConSol*CM Admin Tool - MLA Administration: Create 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 custom field.
  - **Data object**
    MLA will be used in customer data, i.e. in a data object group field.
- **Group:**
  Choose the required custom field group (ticket data) or data object group (customer data) in the list box. For the new MLA a custom field or data object group field of type *MLA field* will be created automatically in this group. This is necessary to display the MLA on the Web Client. The custom field or data object group field can be annotated as described in sections Custom Field Administration and Data Object Group Field Management and GUI Design for Customer Data.
- **Localized values:**
  Enter the corresponding MLA name in the *Value* field for each additional language. On the Web Client the name will be displayed in the respective language of the user's browser locale. If you do not make an entry here the object name, i.e. the content of the *Name* field, will be taken instead.

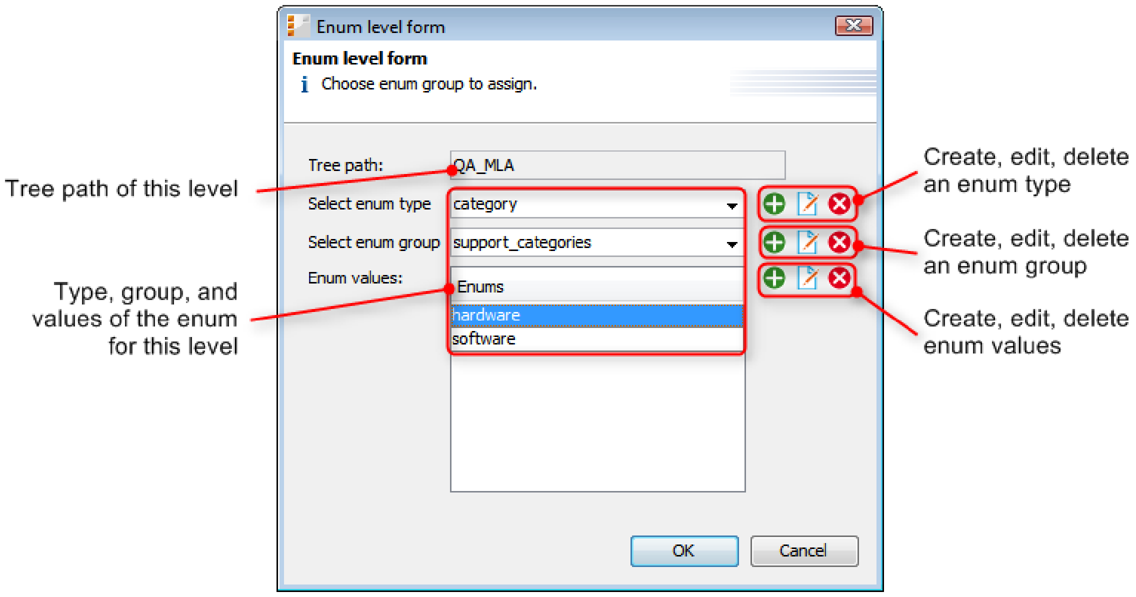
Click on *OK* to save the details of the new MLA.

Information:

You can also create the custom field or data object group field for the MLA first. In this case you will find the localized name of the custom field or data object group field already in the list of available MLAs.

Create an MLA Level

Having created a name and a custom field or data object group field for the MLA you can go on with the definition of levels. Select the MLA in the list and click on ⊕ below *Level 1*. You will get the *Enum level form* where you can specify an enum for this level:


Enum level form
Enum level form
Choose enum group to assign.
Tree path of this level
Type, group, and values of the enum for this level
Create, edit, delete an enum type
Create, edit, delete an enum group
Create, edit, delete enum values
OK
Cancel

Tree path: QA_MLA
Select enum type: category
Select enum group: support_categories
Enum values:

Enums
hardware
software

Fig. 5: ConSol*CM Admin Tool - MLA Administration: Create 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 the enum administration first) or create a new enum type directly here in the MLA administration. The new type will also be visible in the enum administration afterwards.
- **Select enum group:**
  Choose the desired enum group for this level from the lists in the 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 the 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 the respective enum values in the enum administration. If you have created a new enum group you also have to create one or more enum values for the new group.

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

You can either create all enum types, groups, and values 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 on ⊕ next to the respective fields in the window. By clicking on 📝 or ⊗ 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 already used in another MLA.

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

Information:

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 the Enum Administration.

Edit a Level Value

If you want to edit a value of the level, select it in the list and click on . 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 on 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 on below the respective level. The level values (including the values of dependent levels) are shown in italics afterwards. Click on if you want to enable the level again.

17.2.2 Edit an MLA

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

17.2.3 Delete an MLA

You can only delete an MLA if it is not in use. Otherwise 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 on . If you confirm the following dialog with *Yes*, the MLA (and the custom field within *Custom Field Administration* or the data object group field within *User attributes*) will be removed from the list and the system.

17.2.4 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 on ⛔ . The entry in the list is shown in italics afterwards. A disabled MLA will not be displayed in the Web Client. Just click on ✓ below the MLA list, if you want to enable the MLA again.

17.3 Related Topics

- Enums
- Custom fields
- Data object group fields

18 Configuration of the Web Client Dashboard

Starting with version 6.9.4, the ConSol*CM Web Client provides a *dashboard* for engineers which is displayed on the Web Client *Overview* page. The dashboard is composed of one or more so called widgets. As default configuration, the dashboard contains only one widget which displays a graphic with the the number of tickets in all groups of the selected view. Dashboards can contain interactive elements, e.g. to show or hide elements of the graphic. However, persistent engineer-specific (personal) adaption is not possible.

After an update from a previous version to CM version 6.9.4, the Web Client Dashboard will be disabled.

In a new CM 6.9.4 installation, the Web Client Dashboard will be enabled by default. A default dashboard configuration is provided (see example below).

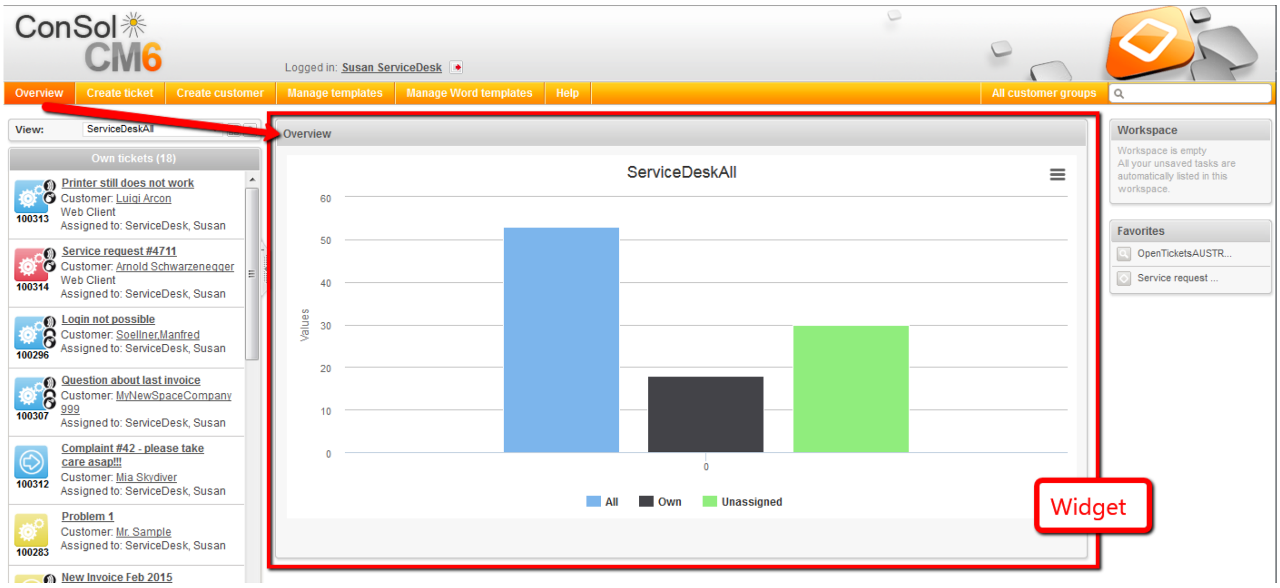


Fig. 1: Web Client Dashboard with default graphic

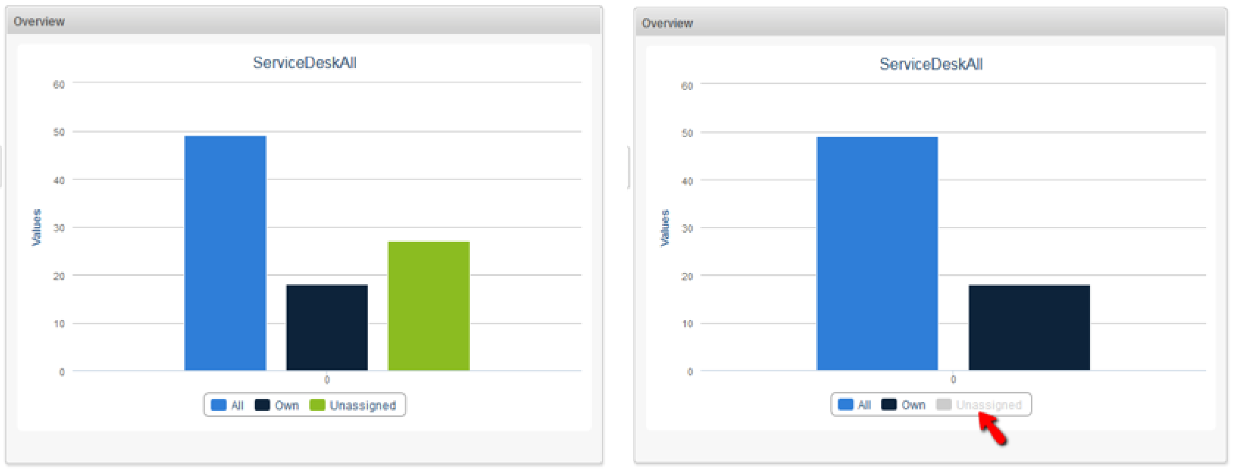


Fig. 2: Interactiv Widget (fade-in and out parts of the graphic)

Widgets which are used for the dashboard are of one of two possible types:

- **Chart**
  for graphical representation of data
- **Table**
  for table representation of data

You as an administrator can design the layout of the dashboards as required, i.e. you can place chart widgets and table widgets on the dashboard page. They are placed on the page based on a grid layout.

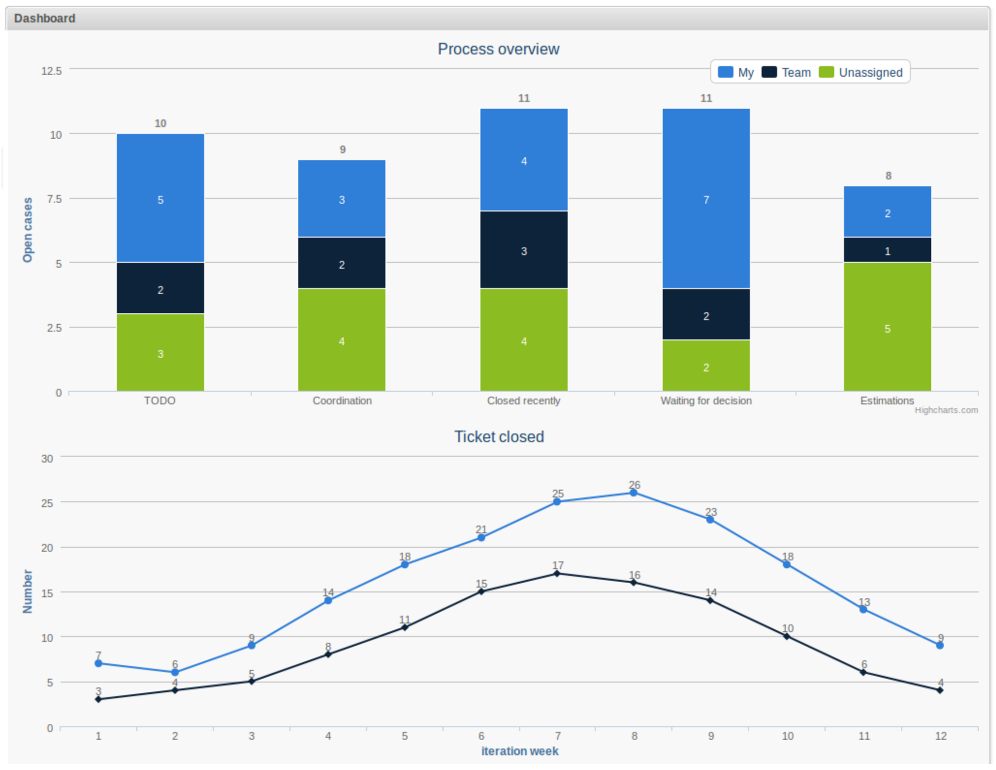
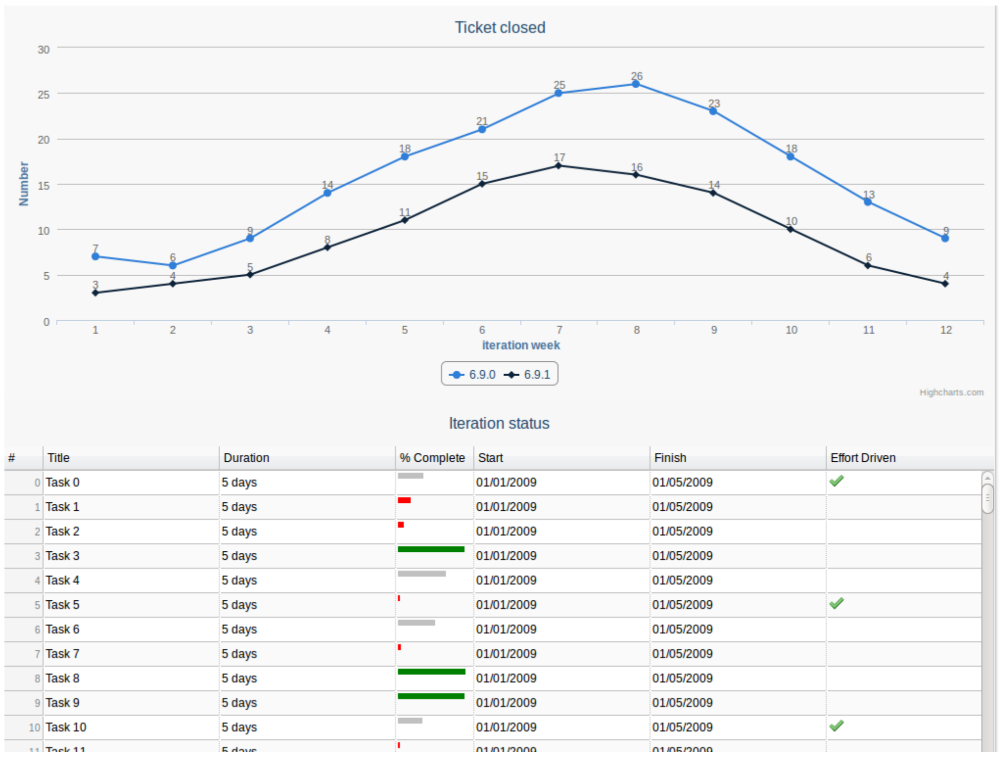


Fig. 3: Example for a dashboard with two graph widgets in one column



Iteration status

#	Title	Duration	% Complete	Start	Finish	Effort Driven
0	Task 0	5 days	▬	01/01/2009	01/05/2009	✓
1	Task 1	5 days	🟥	01/01/2009	01/05/2009	
2	Task 2	5 days	🟥	01/01/2009	01/05/2009	
3	Task 3	5 days	🟩	01/01/2009	01/05/2009	
4	Task 4	5 days	▬	01/01/2009	01/05/2009	
5	Task 5	5 days	🟥	01/01/2009	01/05/2009	✓
6	Task 6	5 days	▬	01/01/2009	01/05/2009	
7	Task 7	5 days	🟥	01/01/2009	01/05/2009	
8	Task 8	5 days	🟩	01/01/2009	01/05/2009	
9	Task 9	5 days	🟩	01/01/2009	01/05/2009	
10	Task 10	5 days	▬	01/01/2009	01/05/2009	✓
11	Task 11	5 days	🟥	01/01/2009	01/05/2009	

Fig. 4: Example for a dashboard with one chart and one table widget in one column

The data for the graphs and/or tables in the widgets are retrieved using Groovy statements which are placed in an Admin Tool script.

The dashboard is configured using Page Customization.

19 CM6 Administrator Manual 6.9.4 - Page Customization

19.1 Page Customization

- General Introduction to Page Customization
- Page Customization in the Web Client
- Page Customization Using Attributes
    - Possible Pages (Scopes) for Page Customization
    - Page Customization Attributes for Types, Scopes and Subscopes, in Alphabetical Order
        - acimSection (Type or Subscope)
        - accordionTicketList (Type)
        - attachmentSection (Type or Subscope)
        - autocomplete
        - boxContent
        - cmRichTextEditor
        - customerSectionPanel
        - detailSearch
        - engineerAutocomplete
        - enumAutocomplete
        - engineerAutocomplete
        - globalSearchField
        - mailTemplate
        - navigationLinks
        - section (Type)
        - sectionList (Type)
        - ticketList (Subscope)
        - ticketsAutocomplete
        - ticketsBookingAutocomplete
        - ticketPanel
        - timeBookingSection
        - unitAutocomplete
        - unitFormPanel
        - unitSearch
        - unitRelationSection (Type UnitSection)
        - unitSearchHeader
        - viewDiscriminatorsSection (Type)
        - welcomePage
- Order and Priorities of Page Customization

19.1.1 General Introduction to Page Customization

In addition to the design of the Web Client GUI in the process of defining custom fields (see section Custom Field Administration) and data object group fields (see section Data Object Group Field Management and GUI Design), an administrator can configure more GUI layout features and functionalities using ***page customization*** .

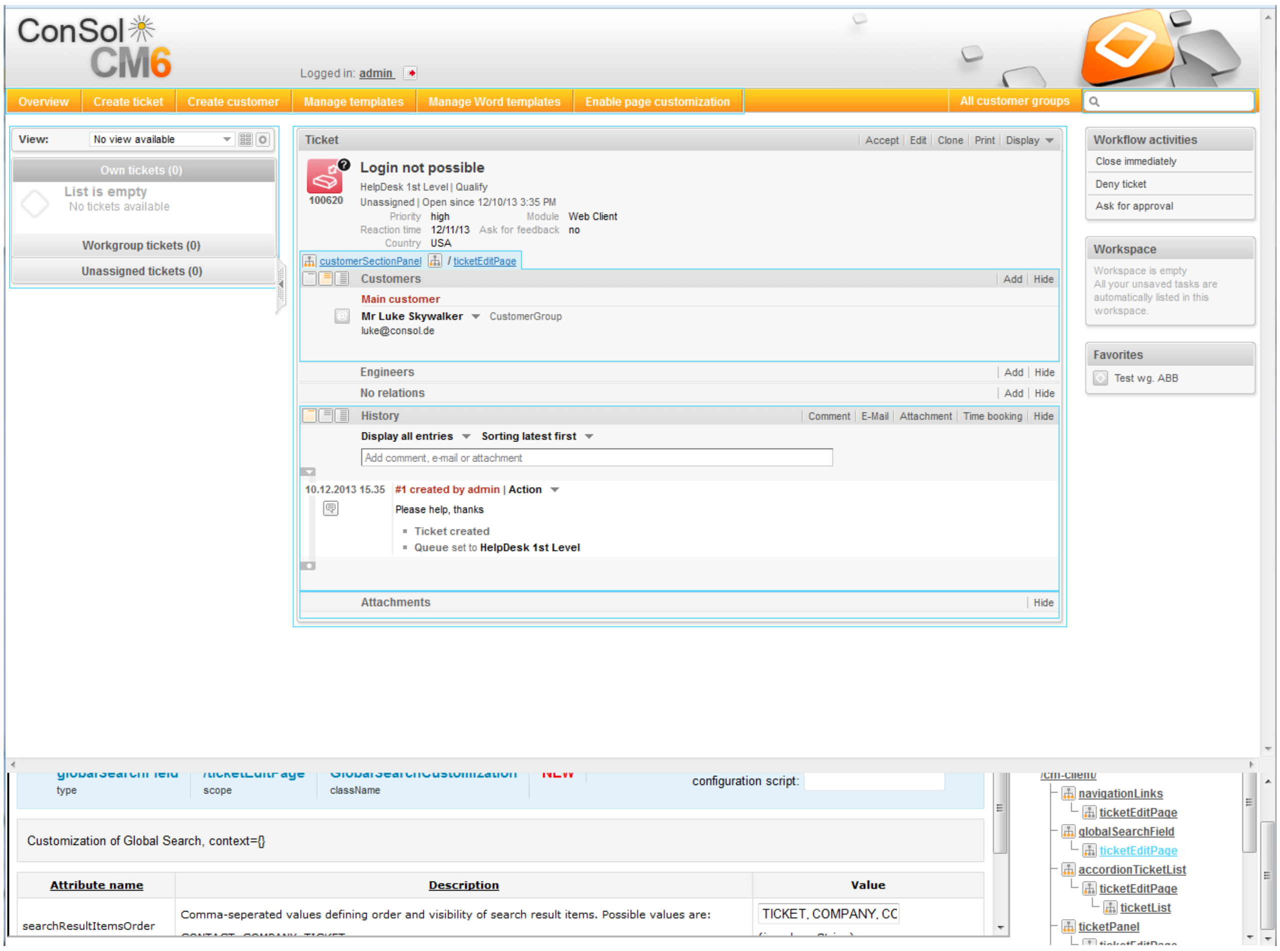
When you log in to the Web Client as an administrator, you see the item *Enable page customization* in the main menu. Depending on the context, i.e. on the page that is currently displayed, the page customization offers different, page- and context-specific functionalities.

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 E-Mail Editor is open, you can also configure e-mail 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.

19.1.2 Page Customization in the Web Client

When you start the page customization mode, the *Page Customization Definition Section* (PCDS) 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 editor section for this element in the PCDS is opened, and the element is marked in the tree. That way you can easily identify the component you would like to modify.



List is empty
No tickets available

Login not possible
100620
HelpDesk 1st Level | Qualify
Unassigned | Open since 12/10/13 3:35 PM
Priority high Module Web Client
Reaction time 12/11/13 Ask for feedback no
Country USA

Main customer
Mr Luke Skywalker CustomerGroup
luke@consol.de

Add comment, e-mail or attachment
10.12.2013 15.35 #1 created by admin | Action
Please help, thanks
- Ticket created
- Queue set to HelpDesk 1st Level

Workspace is empty
All your unsaved tasks are automatically listed in this workspace.

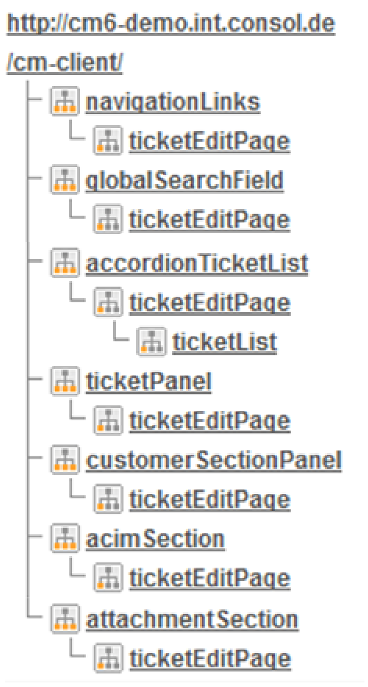
Test wg. ABB

Customization of Global Search, context={}

Attribute name	Description	Value
searchResultItemsOrder	Comma-seperated values defining order and visibility of search result items. Possible values are:	TICKET, COMPANY, CC

Fig. 1: ConSol*CM Web Client - Page Customization Definition Section

The tree might display the following elements (see next figure). Since the PCDS is rather small you might have to scroll to see all elements. In this example the administrator has opened the *ticketEditPage* (see following paragraphs for details).

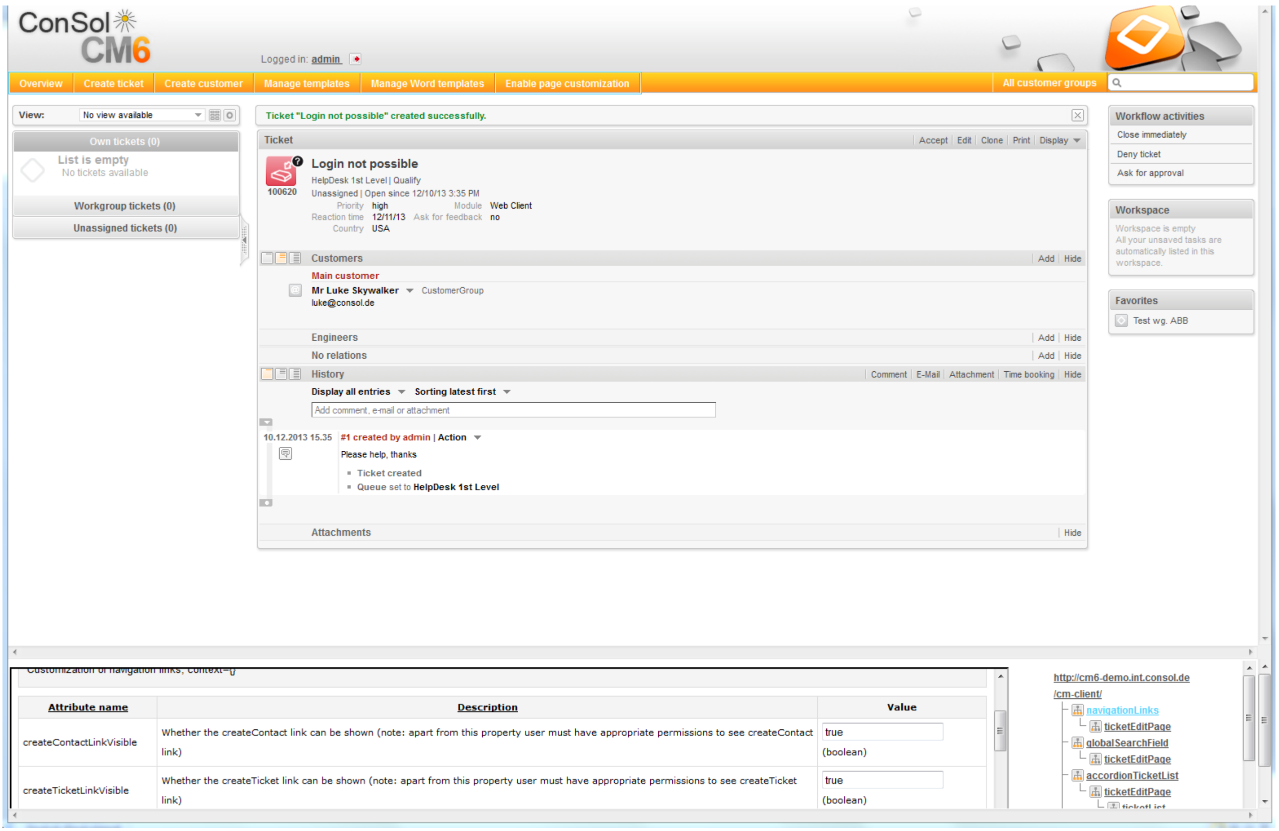


```
http://cm6-demo.int.consol.de
/cm-client/
├ navigationLinks
│  └ ticketEditPage
├ globalSearchField
│  └ ticketEditPage
├ accordionTicketList
│  └ ticketEditPage
│     └ ticketList
├ ticketPanel
│  └ ticketEditPage
├ customerSectionPanel
│  └ ticketEditPage
├ acimSection
│  └ ticketEditPage
└ attachmentSection
   └ ticketEditPage
```

Fig. 2: ConSol*CM Web Client - Page Customization Tree

Icon	Description
▦	Configuration of all components of this type
▦	Configuration of this specific component 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 PCDS, e.g. for the element *navigationLinks* (see following figure).



Ticket "Login not possible" created successfully.

Ticket

Login not possible
HelpDesk 1st Level | Qualify
100620
Unassigned | Open since 12/10/13 3:35 PM

Priority	high	Module	Web Client
Reaction time	12/11/13	Ask for feedback	no
Country	USA		

Customers
Main customer
Mr Luke Skywalker CustomerGroup
luke@consol.de

Engineers
No relations
History
Display all entries Sorting latest first
Add comment, e-mail or attachment

10.12.2013 15.35 #1 created by admin | Action
Please help, thanks
- Ticket created
- Queue set to HelpDesk 1st Level

Attachments

Customization of navigation links, context={}

Attribute name	Description	Value
createContactLinkVisible	Whether the createContact link can be shown (note: apart from this property user must have appropriate permissions to see createContact link)	true (boolean)
createTicketLinkVisible	Whether the createTicket link can be shown (note: apart from this property user must have appropriate permissions to see createTicket link)	true (boolean)

http://cm6-demo.int.consol.de/cm-client/
- navigationLinks
  - ticketEditPage
- globalSearchField
  - ticketEditPage
- accordionTicketList
  - ticketEditPage
    - ticketList

Fig. 3: ConSol*CM Web Client - Selected Tree Element in PCDS

The entire page is built according to a strictly hierarchical structure and every element is defined by a type, a scope, and a class. These are displayed in the blue header section of the editor in the PCDS 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 and classes. When you work on the *scope* level you define the attributes for all (sub-)elements of this scope. When you work on the *class* level, you define the attributes for this class only.

Please see the following example for ticket list configuration.

- **Type level:**

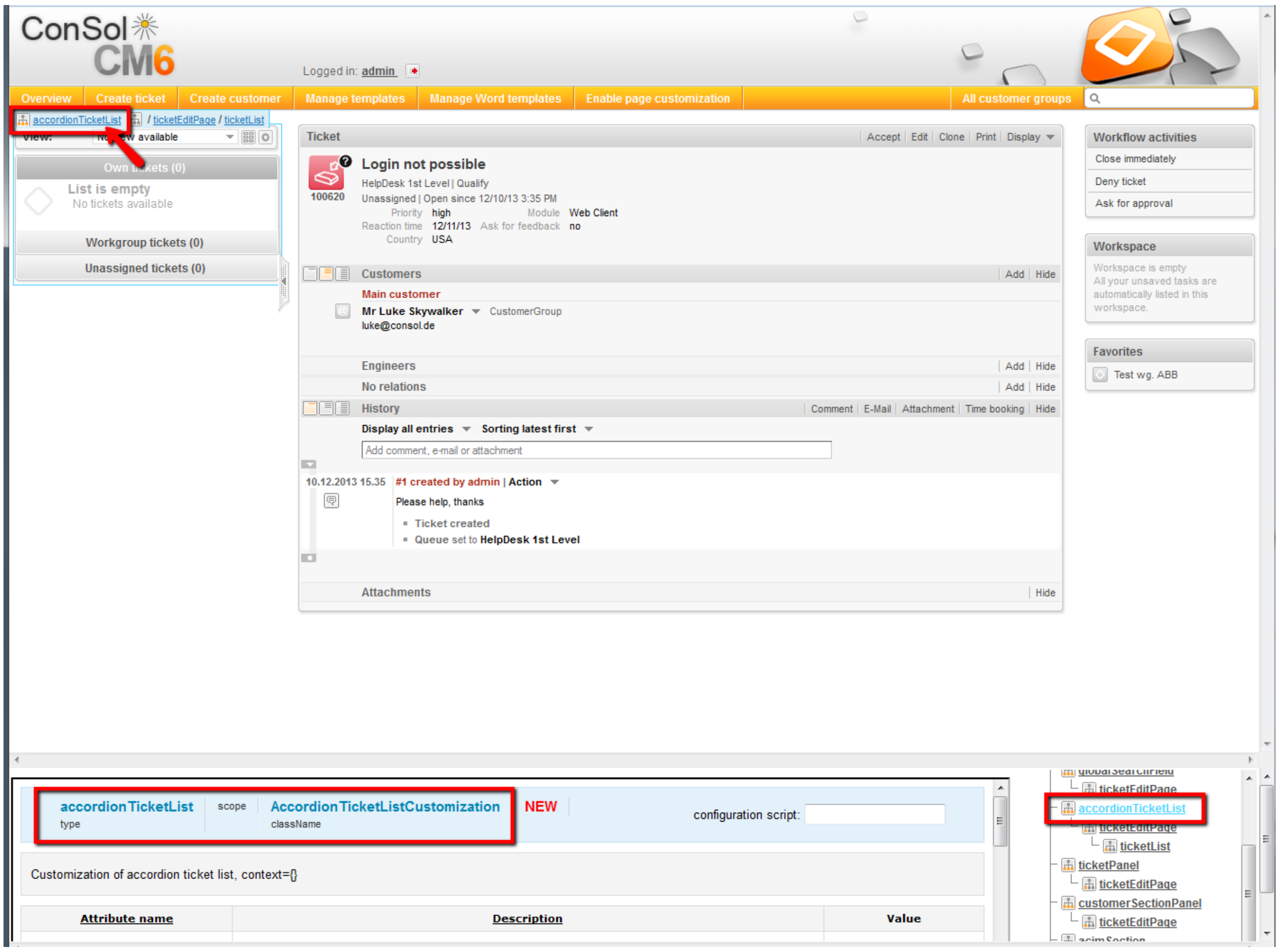
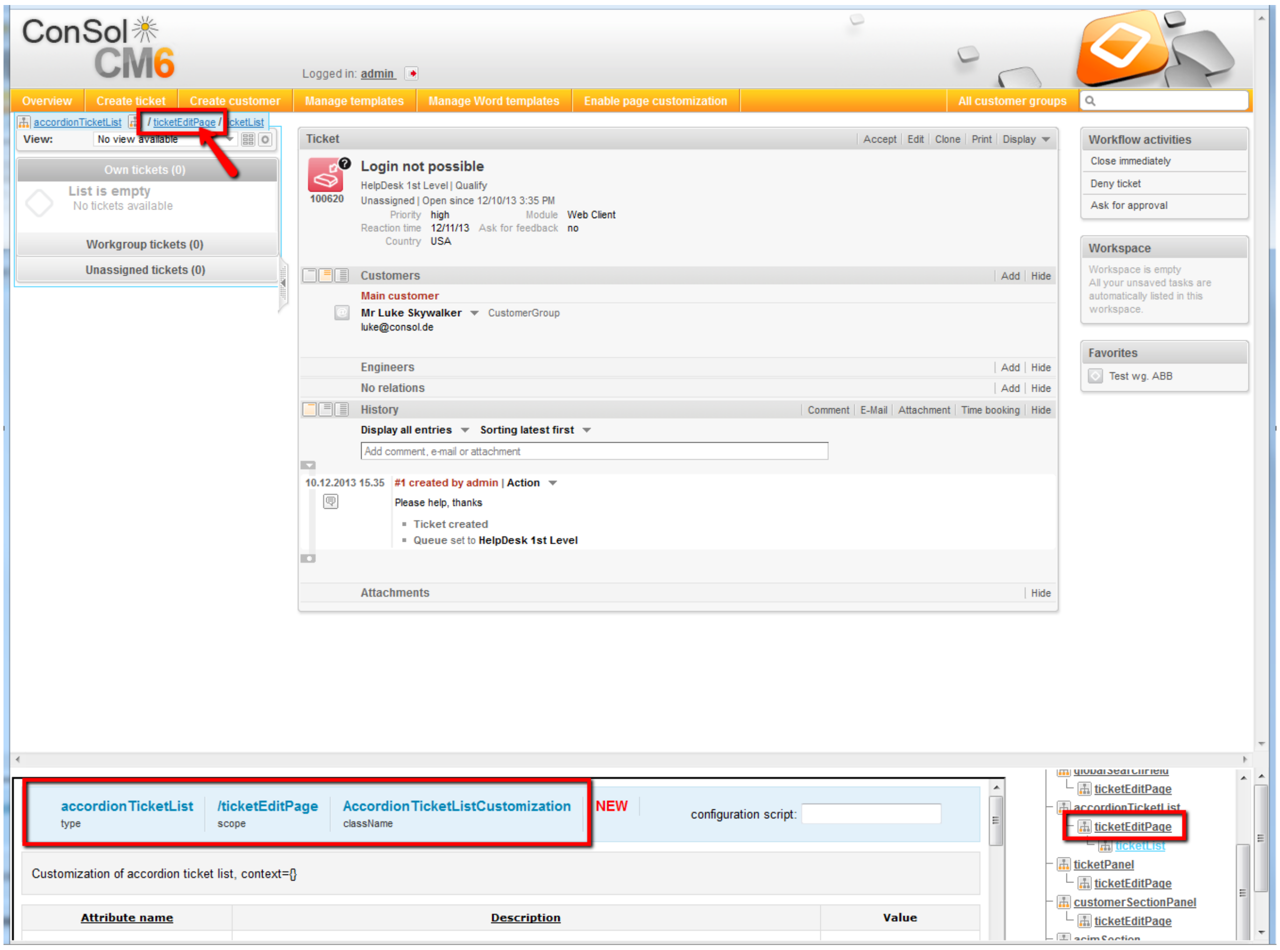


Fig. 4: ConSol*CM Web Client - Defining Parameters on Type Level

- **Scope level:**



Login not possible
100620
HelpDesk 1st Level | Qualify
Unassigned | Open since 12/10/13 3:35 PM
Priority high Module Web Client
Reaction time 12/11/13 Ask for feedback no
Country USA

Main customer
Mr Luke Skywalker CustomerGroup
luke@consol.de

10.12.2013 15.35 #1 created by admin | Action
Please help, thanks
- Ticket created
- Queue set to HelpDesk 1st Level

accordionTicketList	/ticketEditPage	AccordionTicketListCustomization	NEW	configuration script:
type	scope	className		

Customization of accordion ticket list, context={}

Attribute name	Description	Value

Fig. 5: ConSol*CM Web Client - Defining Parameters on Scope Level

- **SubScope level:**

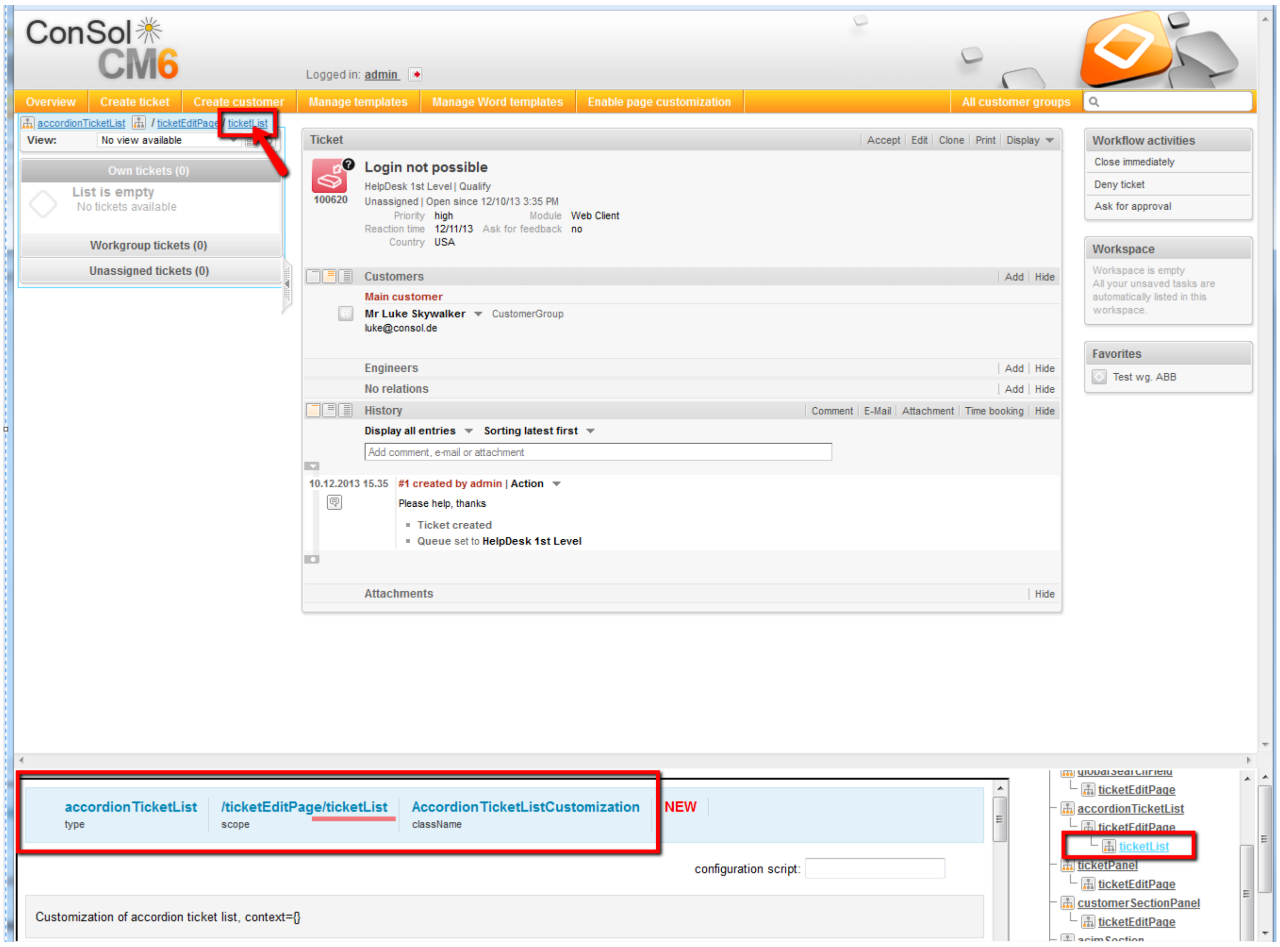
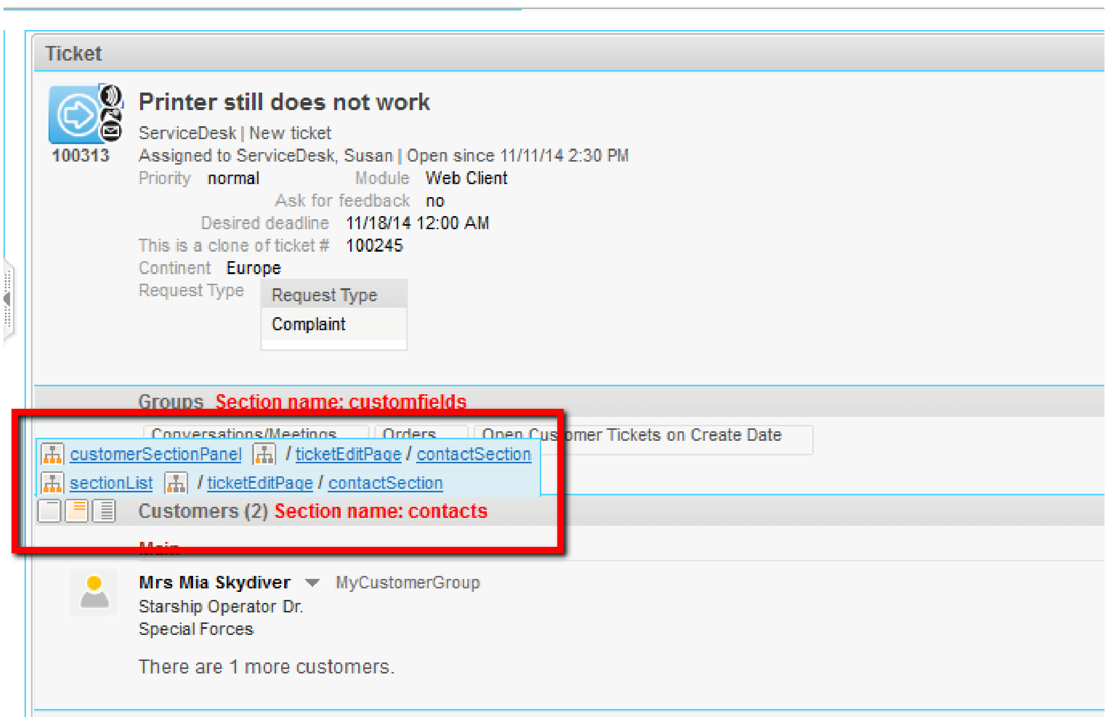

accordionTicketList
type
/ticketEditPage/ticketList
scope
AccordionTicketListCustomization
className
NEW
configuration script:
Customization of accordion ticket list, context={}
accordionTicketList / ticketEditPage / ticketList
ticketEditPage
ticketList
ticketPanel
customerSectionPanel

Fig. 6: ConSol*CM Web Client - Defining Parameters on Class 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, the ticketEditPage / contactSection subscope can be configured for two different types: *customerSectionPanel* and *sectionList*.

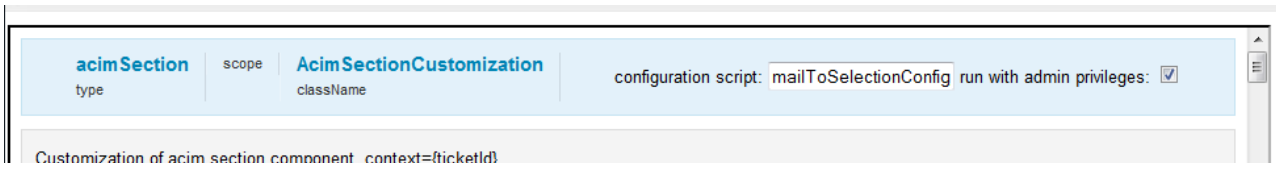

Ticket

Printer still does not work
ServiceDesk | New ticket
100313
Assigned to ServiceDesk, Susan | Open since 11/11/14 2:30 PM
Priority normal Module Web Client
Ask for feedback no
Desired deadline 11/18/14 12:00 AM
This is a clone of ticket # 100245
Continent Europe
Request Type: Request Type – Complaint

Groups Section name: customfields
Conversations/Meetings | Orders | Open Customer Tickets on Create Date
customerSectionPanel / ticketEditPage / contactSection
sectionList / ticketEditPage / contactSection
Customers (2) Section name: contacts
Main
Mrs Mia Skydiver MyCustomerGroup
Starship Operator Dr.
Special Forces
There are 1 more customers.

Fig. 7: Two Pathes 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 check box.


acimSection type | scope | AcimSectionCustomization className | configuration script: mailToSelectionConfig run with admin privileges: ☑

Customization of acim section component, context={ticketId}

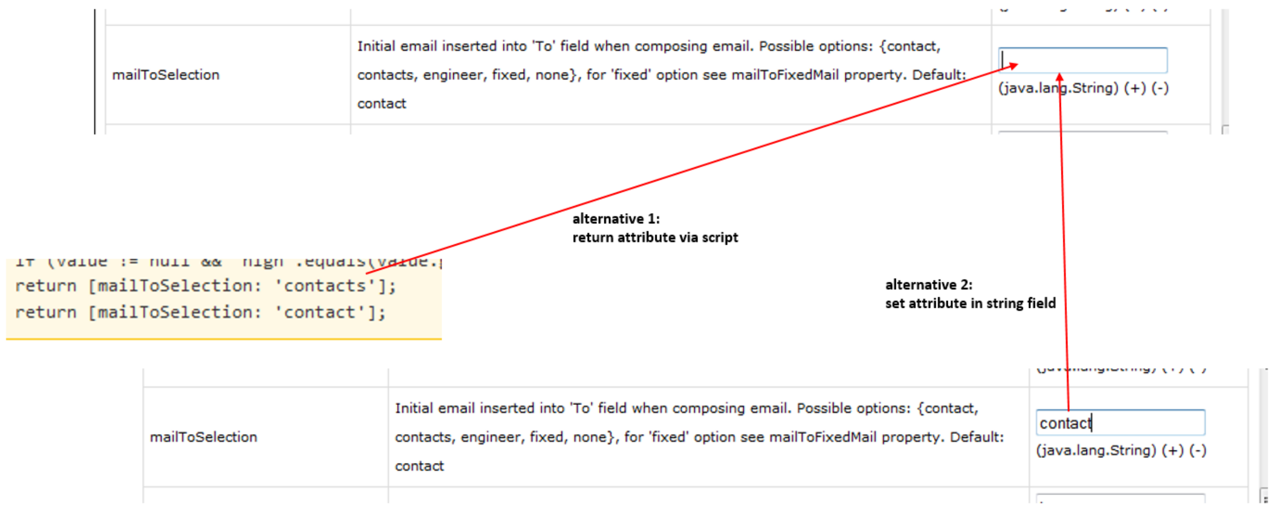
Fig. 8: ConSol*CM Web Client - Configuration Script for Defining Values

E.g. there should be only one e-mail recipient (here: the main customer) for medium and low priority tickets, but 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'];
```

It has to be stored in the *Scripts* section of the Admin Tool (see section 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.



mailToSelection	Initial email inserted into 'To' field when composing email. Possible options: {contact, contacts, engineer, fixed, none}, for 'fixed' option see mailToFixedMail property. Default: contact	(java.lang.String) (+) (-)

```
if (value != null && "high".equals(value.
return [mailToSelection: 'contacts'];
return [mailToSelection: 'contact'];
```

mailToSelection	Initial email inserted into 'To' field when composing email. Possible options: {contact, contacts, engineer, fixed, none}, for 'fixed' option see mailToFixedMail property. Default: contact	contact (java.lang.String) (+) (-)

Fig. 9: Two Alternatives for Setting Page Customization Attributes

19.1.3 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.

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:

- **welcomePage**

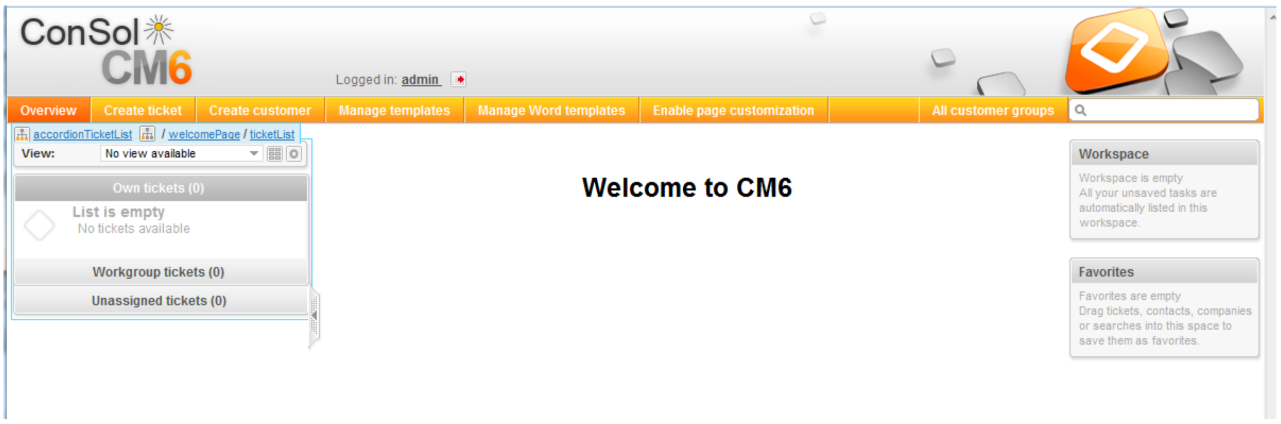


Fig. 10: ConSol*CM Web Client - Welcome Page

- **ticketEditPage**

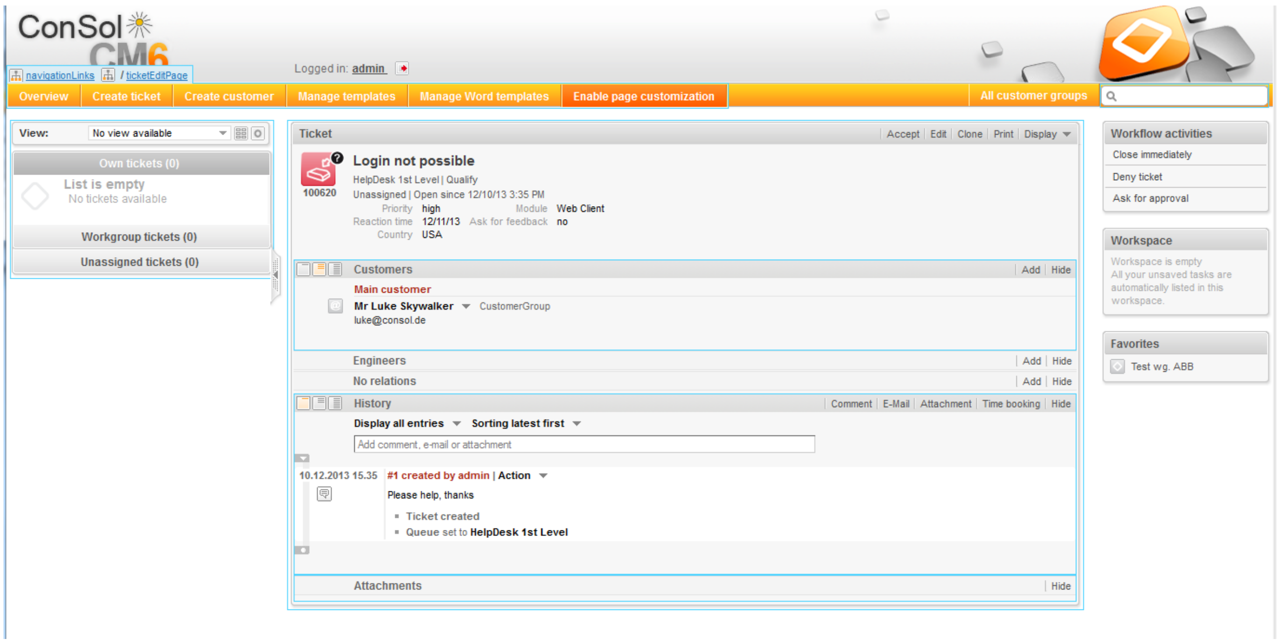


Fig. 11: ConSol*CM Web Client - Edit Ticket Page

- **searchDetailPage**

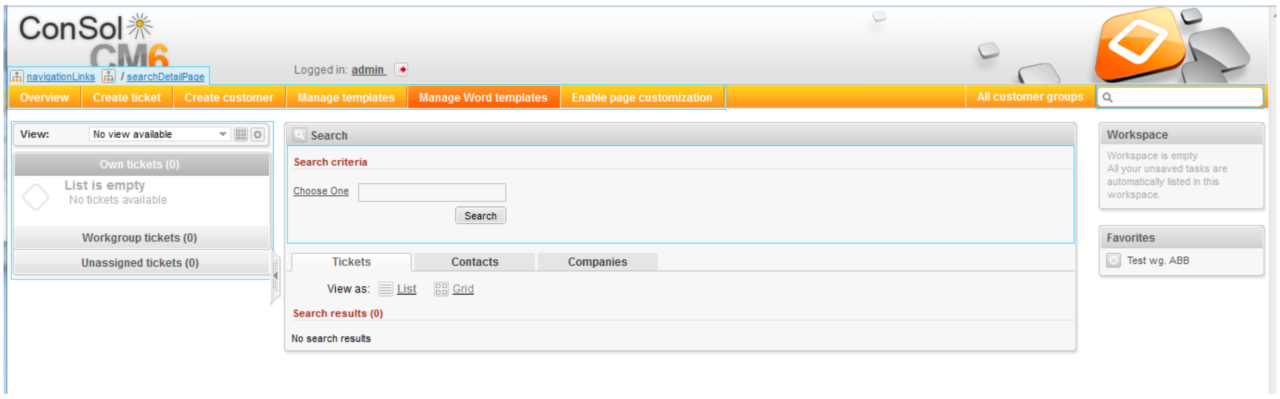


Fig. 12: ConSol*CM Web Client - Detail Search Page

- **contactCreatePage**

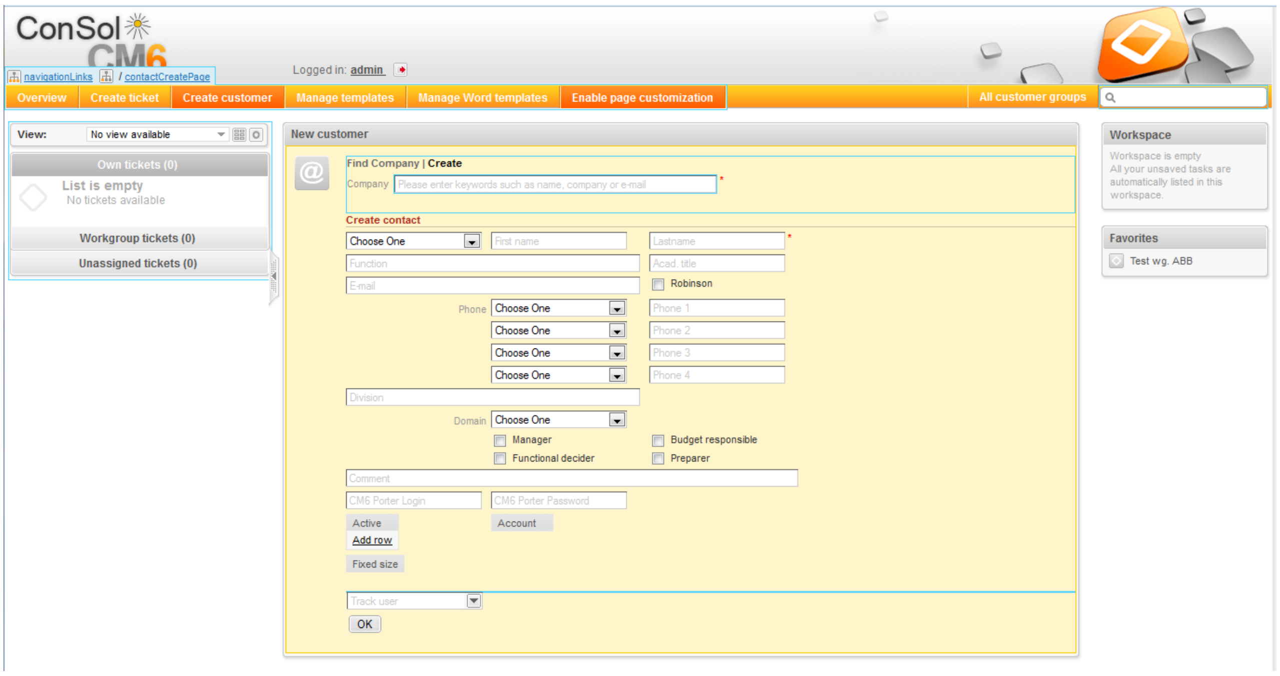
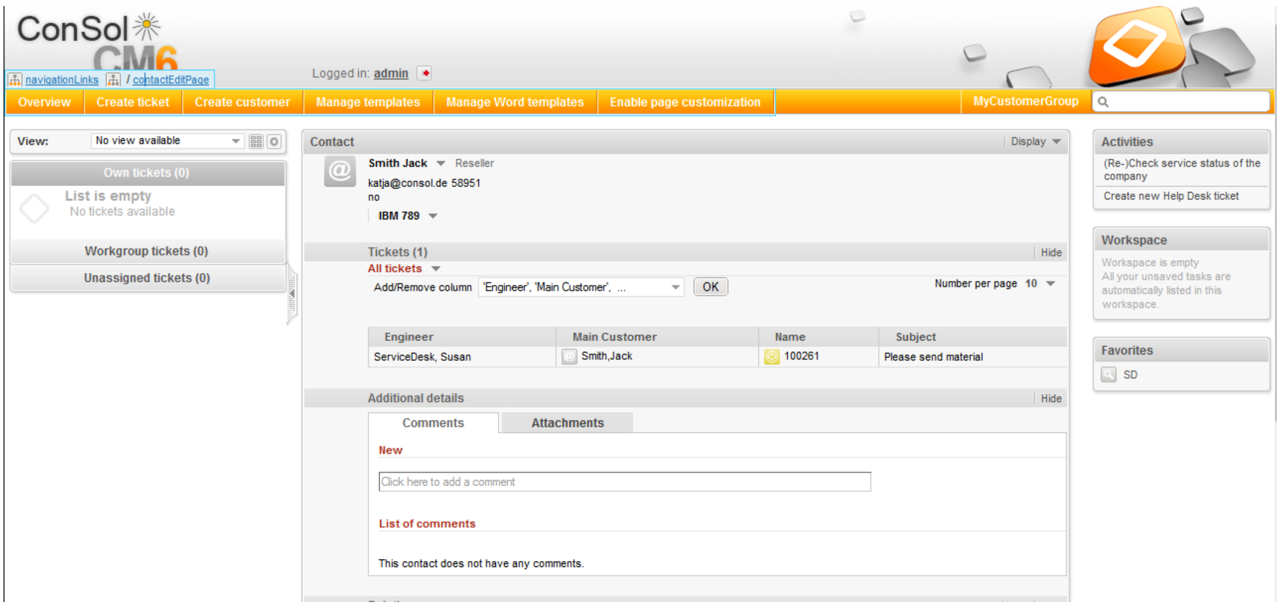


Fig. 13: ConSol*CM Web Client - Create Contact Page

- **contactEditPage**



Smith Jack Reseller
katja@consol.de 58951
no
IBM 789

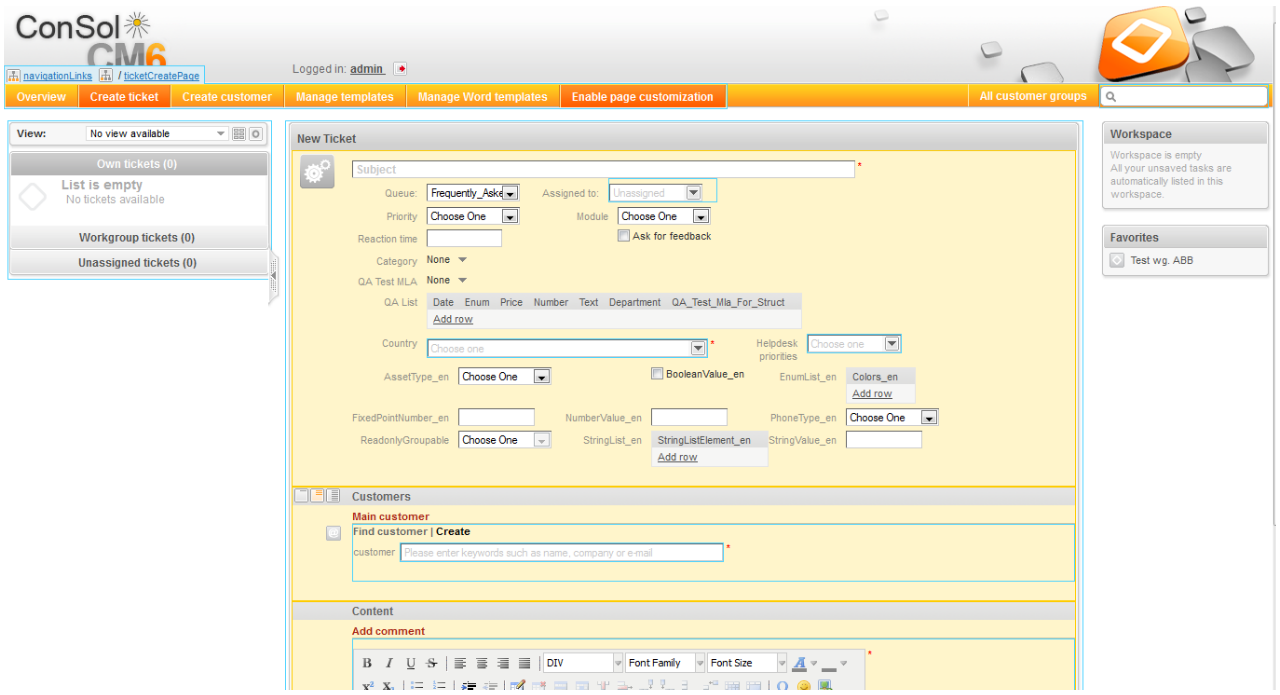
'Engineer', 'Main Customer', ...

Engineer	Main Customer	Name	Subject
ServiceDesk, Susan	Smith,Jack	100261	Please send material

This contact does not have any comments.

Fig. 14: ConSol*CM Web Client - Edit Contact Page

- **ticketCreatePage**



Subject
Queue: Frequently_Aske
Assigned to: Unassigned
Priority Choose One
Module Choose One
Reaction time
Ask for feedback
Category None
QA Test MLA None
QA List: Date Enum Price Number Text Department QA_Test_Mla_For_Struct
Add row
Country Choose one
Helpdesk priorities Choose one
AssetType_en Choose One
BooleanValue_en
EnumList_en Colors_en
Add row
FixedPointNumber_en
NumberValue_en
PhoneType_en Choose One
ReadonlyGroupable Choose One
StringList_en StringListElement_en
Add row
StringValue_en
customer Please enter keywords such as name, company or e-mail
DIV Font Family Font Size

Fig. 15: ConSol*CM Web Client - Create Ticket Page

- **userProfilePage**

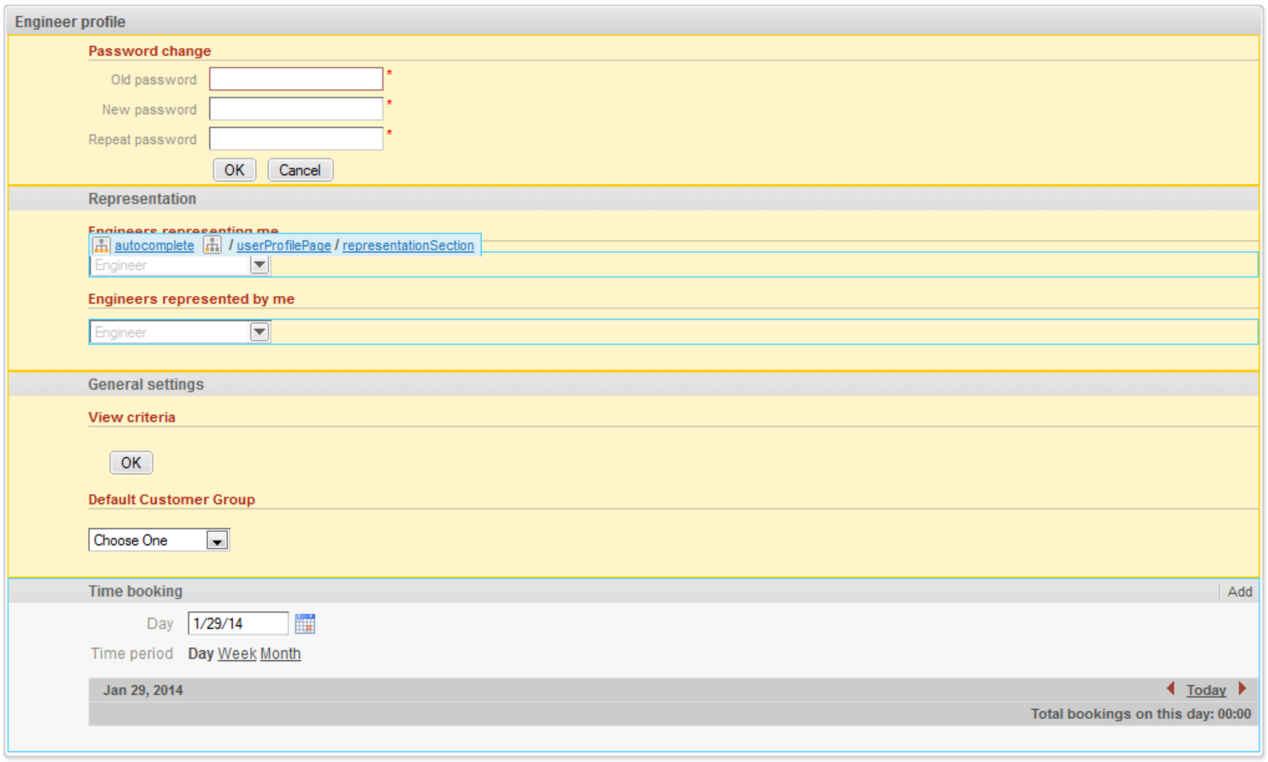
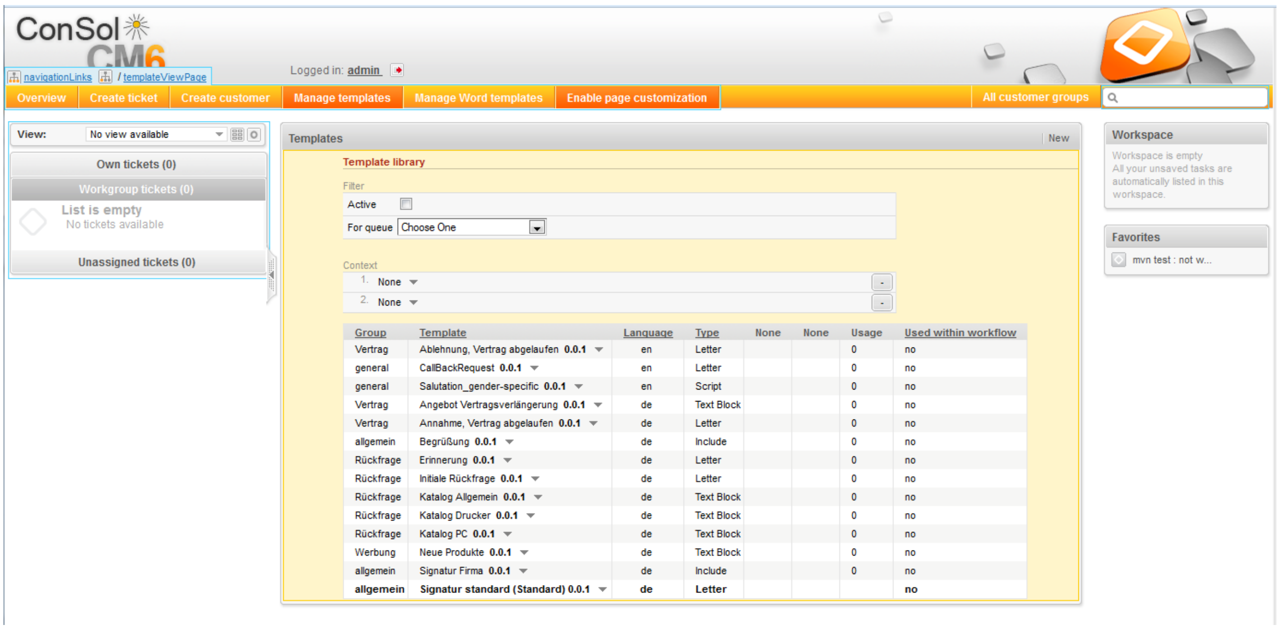


Fig. 16: ConSol*CM Web Client - User Profile Page

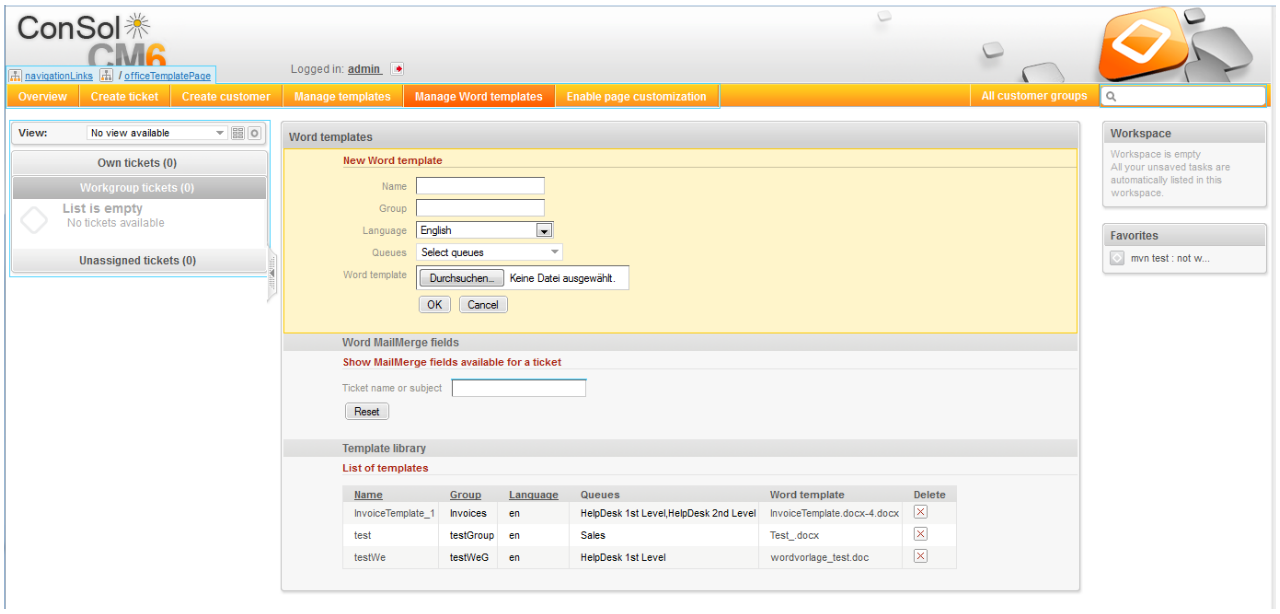
- **templateViewPage**



Group	Template	Language	Type	None	None	Usage	Used within workflow
Vertrag	Ablehnung, Vertrag abgelaufen **0.0.1**	en	Letter			0	no
general	CallBackRequest **0.0.1**	en	Letter			0	no
general	Salutation_gender-specific **0.0.1**	en	Script			0	no
Vertrag	Angebot Vertragsverlängerung **0.0.1**	de	Text Block			0	no
Vertrag	Annahme, Vertrag abgelaufen **0.0.1**	de	Letter			0	no
allgemein	Begrüßung **0.0.1**	de	Include			0	no
Rückfrage	Erinnerung **0.0.1**	de	Letter			0	no
Rückfrage	Initiale Rückfrage **0.0.1**	de	Letter			0	no
Rückfrage	Katalog Allgemein **0.0.1**	de	Text Block			0	no
Rückfrage	Katalog Drucker **0.0.1**	de	Text Block			0	no
Rückfrage	Katalog PC **0.0.1**	de	Text Block			0	no
Werbung	Neue Produkte **0.0.1**	de	Text Block			0	no
allgemein	Signatur Firma **0.0.1**	de	Include			0	no
allgemein	**Signatur standard (Standard) 0.0.1**	**de**	**Letter**				**no**

Fig. 17: ConSol*CM Web Client - Template View Page

- **officeTemplatePage** (only when CM.Doc has been activated)



Word templates

New Word template

Name	
Group	
Language	English
Queues	Select queues
Word template	Durchsuchen... Keine Datei ausgewählt.

OK Cancel

Word MailMerge fields

Show MailMerge fields available for a ticket

Ticket name or subject

Reset

Template library

List of templates

Name	Group	Language	Queues	Word template	Delete
InvoiceTemplate_1	Invoices	en	HelpDesk 1st Level,HelpDesk 2nd Level	InvoiceTemplate.docx-4.docx	☒
test	testGroup	en	Sales	Test_.docx	☒
testWe	testWeG	en	HelpDesk 1st Level	wordvorlage_test.doc	☒

Fig. 18: ConSol*CM Web Client - Office Template Page (only when CM.Doc is activated)

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/welcomePage
- globalSearchField/ticketEditPage
- globalSearchField/searchDetailPage
- globalSearchField/contactCreatePage
- globalSearchField/contactEditPage
- globalSearchField/ticketCreatePage

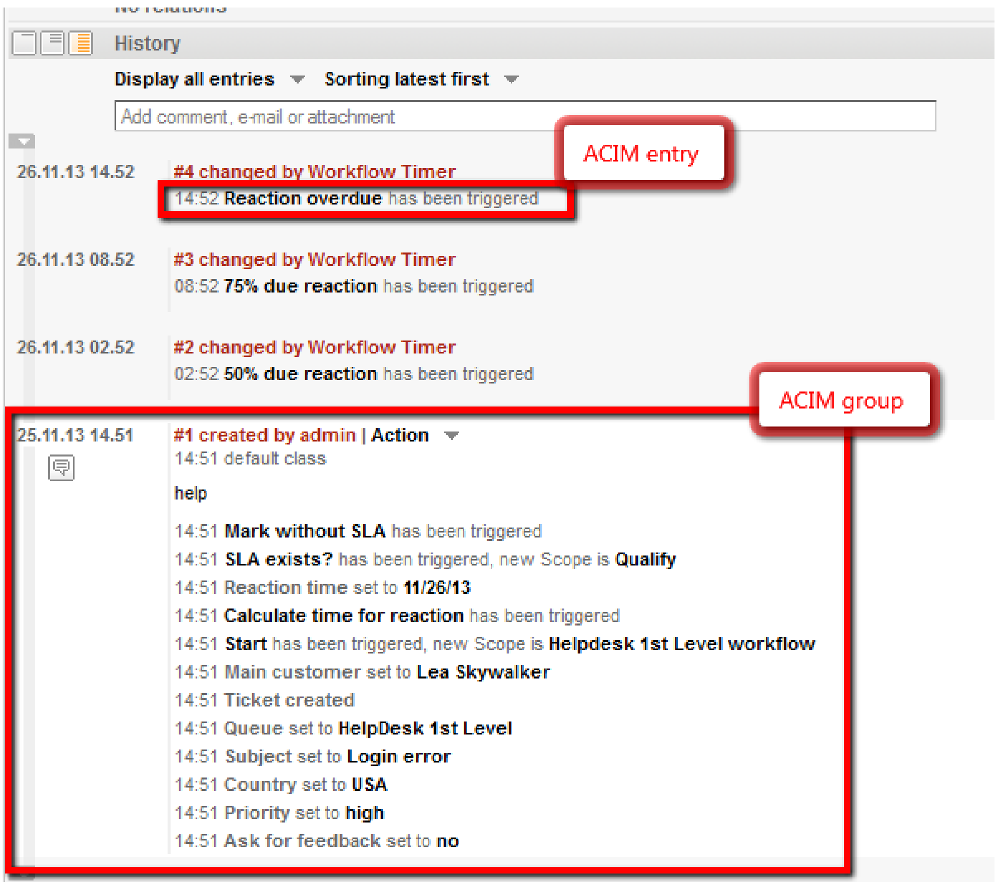
Page Customization Attributes for Types, Scopes and Subscopes, in Alphabetical Order

acimSection (Type or Subscope)

An *ACIM* (activity item) is an entry in the *History* section of a ticket. This can be ...

- a comment
- an e-mail which has been sent from the ticket
- an e-mail 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.



26.11.13 14.52 #4 changed by Workflow Timer
14:52 **Reaction overdue** has been triggered

26.11.13 08.52 #3 changed by Workflow Timer
08:52 **75% due reaction** has been triggered

26.11.13 02.52 #2 changed by Workflow Timer
02:52 **50% due reaction** has been triggered

25.11.13 14.51 #1 created by admin | Action
14:51 default class

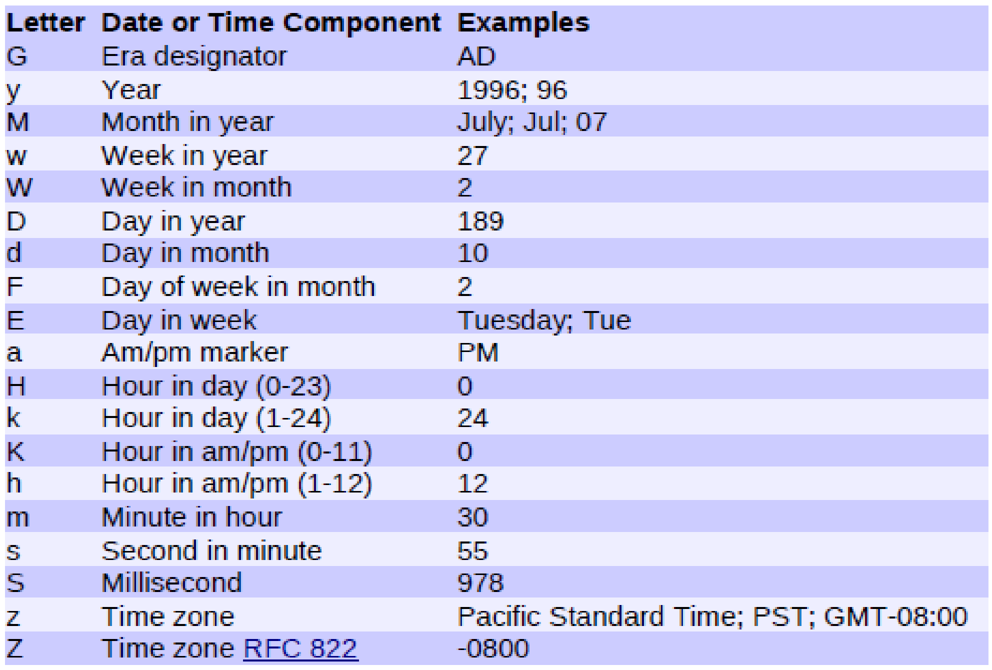
help

14:51 **Mark without SLA** has been triggered
14:51 **SLA exists?** has been triggered, new Scope is **Qualify**
14:51 Reaction time set to **11/26/13**
14:51 **Calculate time for reaction** has been triggered
14:51 **Start** has been triggered, new Scope is **Helpdesk 1st Level workflow**
14:51 Main customer set to **Lea Skywalker**
14:51 Ticket created
14:51 Queue set to **HelpDesk 1st Level**
14:51 Subject set to **Login error**
14:51 Country set to **USA**
14:51 Priority set to **high**
14:51 Ask for feedback set to **no**

Fig. 19: ConSol*CM Web Client - ACIM Group and Item

Warning:

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.

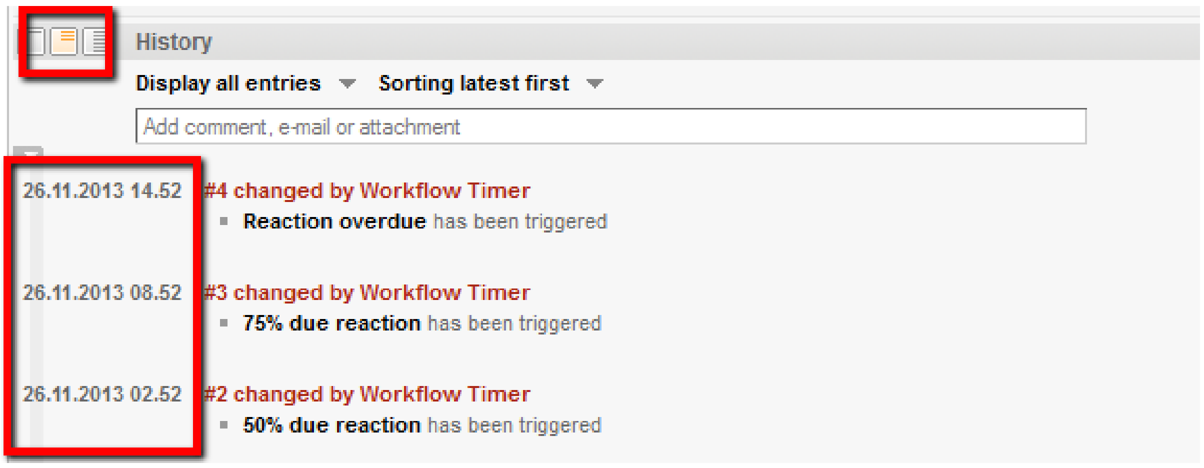


Letter	Date or Time Component	Examples
G	Era designator	AD
y	Year	1996; 96
M	Month in year	July; Jul; 07
w	Week in year	27
W	Week in month	2
D	Day in year	189
d	Day in month	10
F	Day of week in month	2
E	Day in week	Tuesday; Tue
a	Am/pm marker	PM
H	Hour in day (0-23)	0
k	Hour in day (1-24)	24
K	Hour 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

Fig. 20: ConSol*CM Web Client - Valid Date Formats for ACIM Date Configuration

Attributes:

- **acimGroupActionEntryDateFormat**
  Date format for group of ACIM without text or e-mail 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)

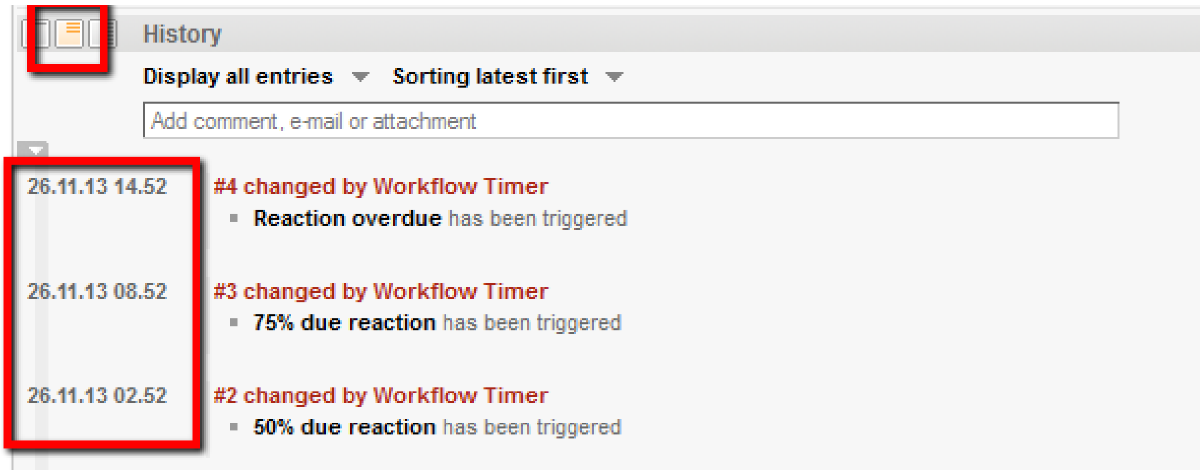


26.11.2013 14.52 #4 changed by Workflow Timer
- **Reaction overdue** has been triggered

26.11.2013 08.52 #3 changed by Workflow Timer
- **75% due reaction** has been triggered

26.11.2013 02.52 #2 changed by Workflow Timer
- **50% due reaction** has been triggered

Fig. 21: ConSol*CM Web Client - Display for Format: dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm



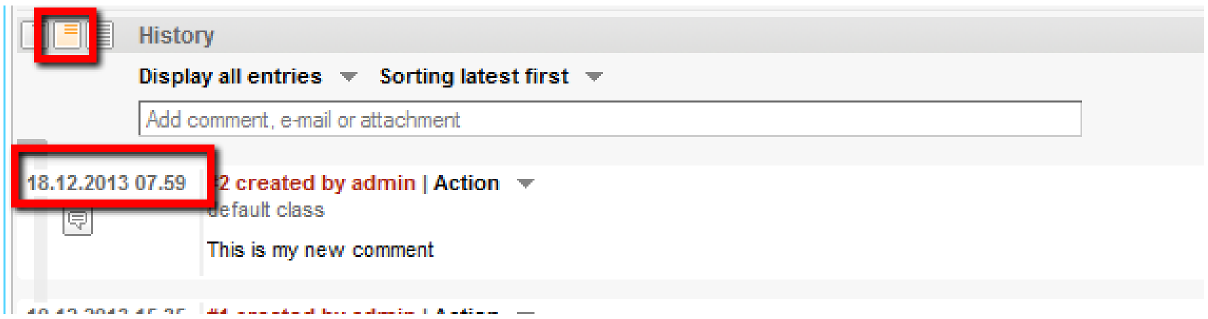
26.11.13 14.52 #4 changed by Workflow Timer
- **Reaction overdue** has been triggered

26.11.13 08.52 #3 changed by Workflow Timer
- **75% due reaction** has been triggered

26.11.13 02.52 #2 changed by Workflow Timer
- **50% due reaction** has been triggered

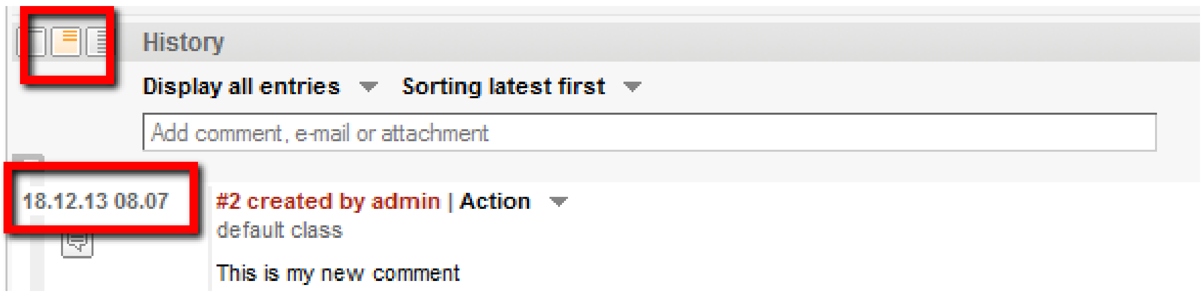
Fig. 22: ConSol*CM Web Client - Display for Format: dd.MM.yy HH.mm|dd.MM.yy HH.mm|dd.MM.yy HH.mm

- **acimGroupTextEntryDateFormat**
  Date format for group of ACIM with text, e-mail, 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)



This is my new comment

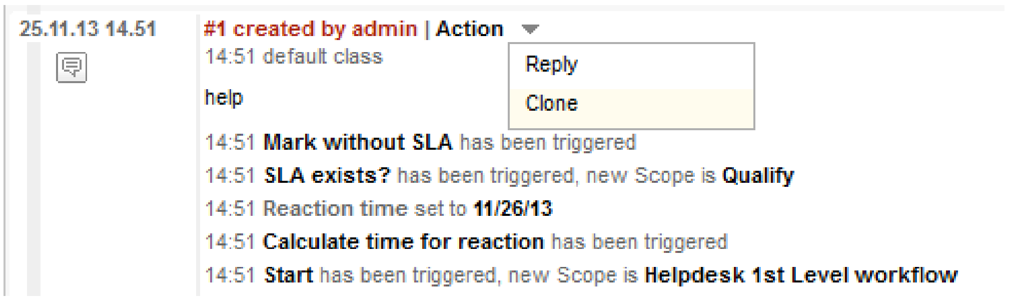
Fig. 23: ConSol*CM Web Client - Display for Format: dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm|dd.MM.yyyy HH.mm



This is my new comment

Fig. 24: 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 e-mail 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)
- **showCloneOption**
  Enables clone option for text ACIM entry (comment or e-mail entry). (boolean)



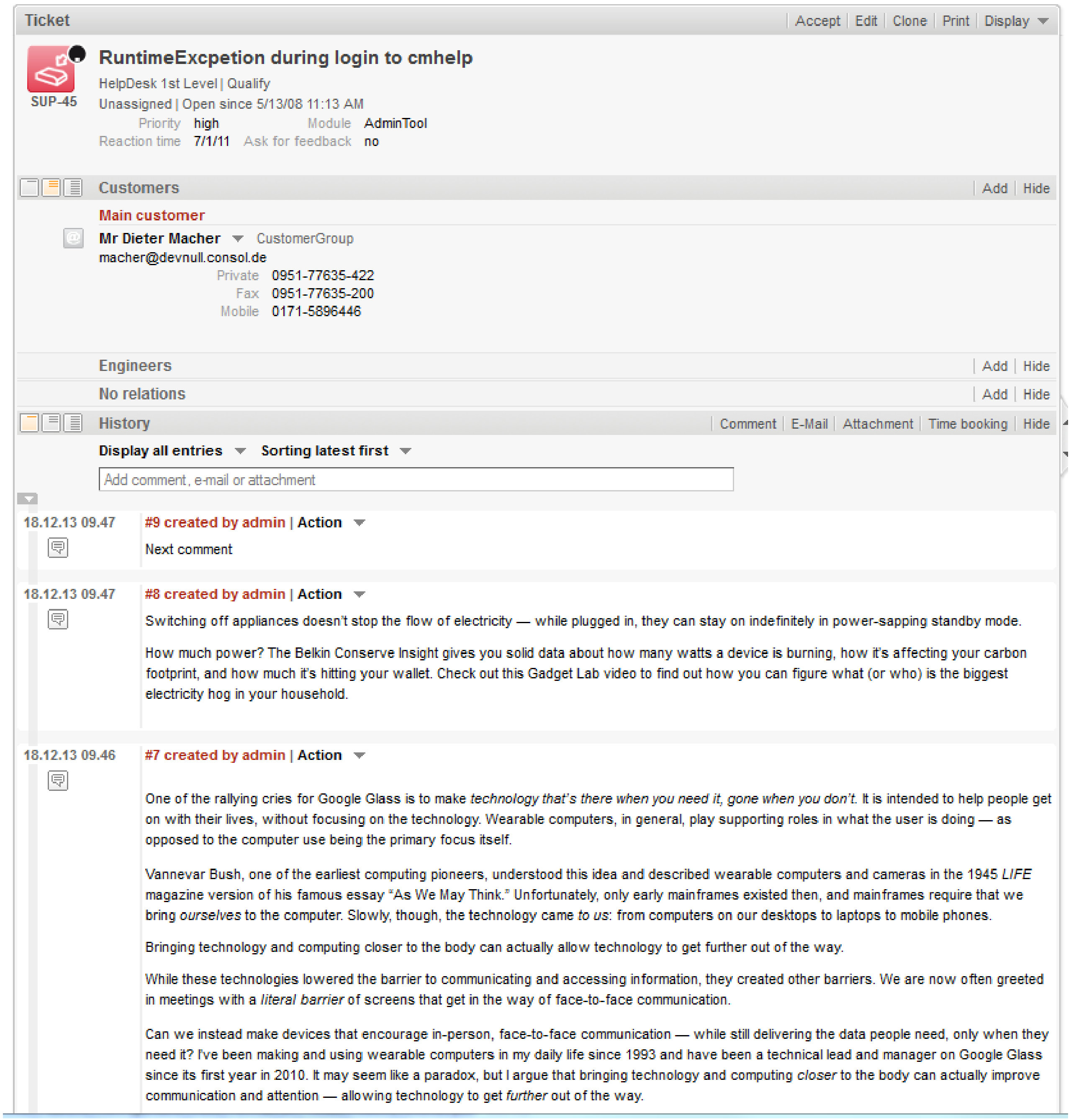
help

14:51 **Mark without SLA** has been triggered
14:51 **SLA exists?** has been triggered, new Scope is **Qualify**
14:51 Reaction time set to **11/26/13**
14:51 **Calculate time for reaction** has been triggered
14:51 **Start** has been triggered, new Scope is **Helpdesk 1st Level workflow**

Fig. 25: 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 clone for another item is selected simultaneously. Possible values are *append*, *replace*. Default is *append*. (java.lang.String)
- **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)

  Example 1: Lazy loading switched off


Ticket
Accept | Edit | Clone | Print | Display
SUP-45
Customers
Add | Hide
Engineers
Add | Hide
No relations
Add | Hide
History
Comment | E-Mail | Attachment | Time booking | Hide
Display all entries
Sorting latest first
Add comment, e-mail or attachment

RuntimeExcpetion during login to cmhelp

HelpDesk 1st Level | Qualify

Unassigned | Open since 5/13/08 11:13 AM

Priority high Module AdminTool

Reaction time 7/1/11 Ask for feedback no

Main customer

Mr Dieter Macher CustomerGroup

macher@devnull.consol.de

Private 0951-77635-422

Fax 0951-77635-200

Mobile 0171-5896446

18.12.13 09.47 #9 created by admin | Action

Next comment

18.12.13 09.47 #8 created by admin | Action

Switching off appliances doesn't stop the flow of electricity — while plugged in, they can stay on indefinitely in power-sapping standby mode.

How much power? The Belkin Conserve Insight gives you solid data about how many watts a device is burning, how it's affecting your carbon footprint, and how much it's hitting your wallet. Check out this Gadget Lab video to find out how you can figure what (or who) is the biggest electricity hog in your household.

18.12.13 09.46 #7 created by admin | Action

One of the rallying cries for Google Glass is to make *technology that's there when you need it, gone when you don't.* It is intended to help people get on with their lives, without focusing on the technology. Wearable computers, in general, play supporting roles in what the user is doing — as opposed to the computer use being the primary focus itself.

Vannevar Bush, one of the earliest computing pioneers, understood this idea and described wearable computers and cameras in the 1945 *LIFE* magazine version of his famous essay "As We May Think." Unfortunately, only early mainframes existed then, and mainframes require that we bring *ourselves* to the computer. Slowly, though, the technology came *to us*: from computers on our desktops to laptops to mobile phones.

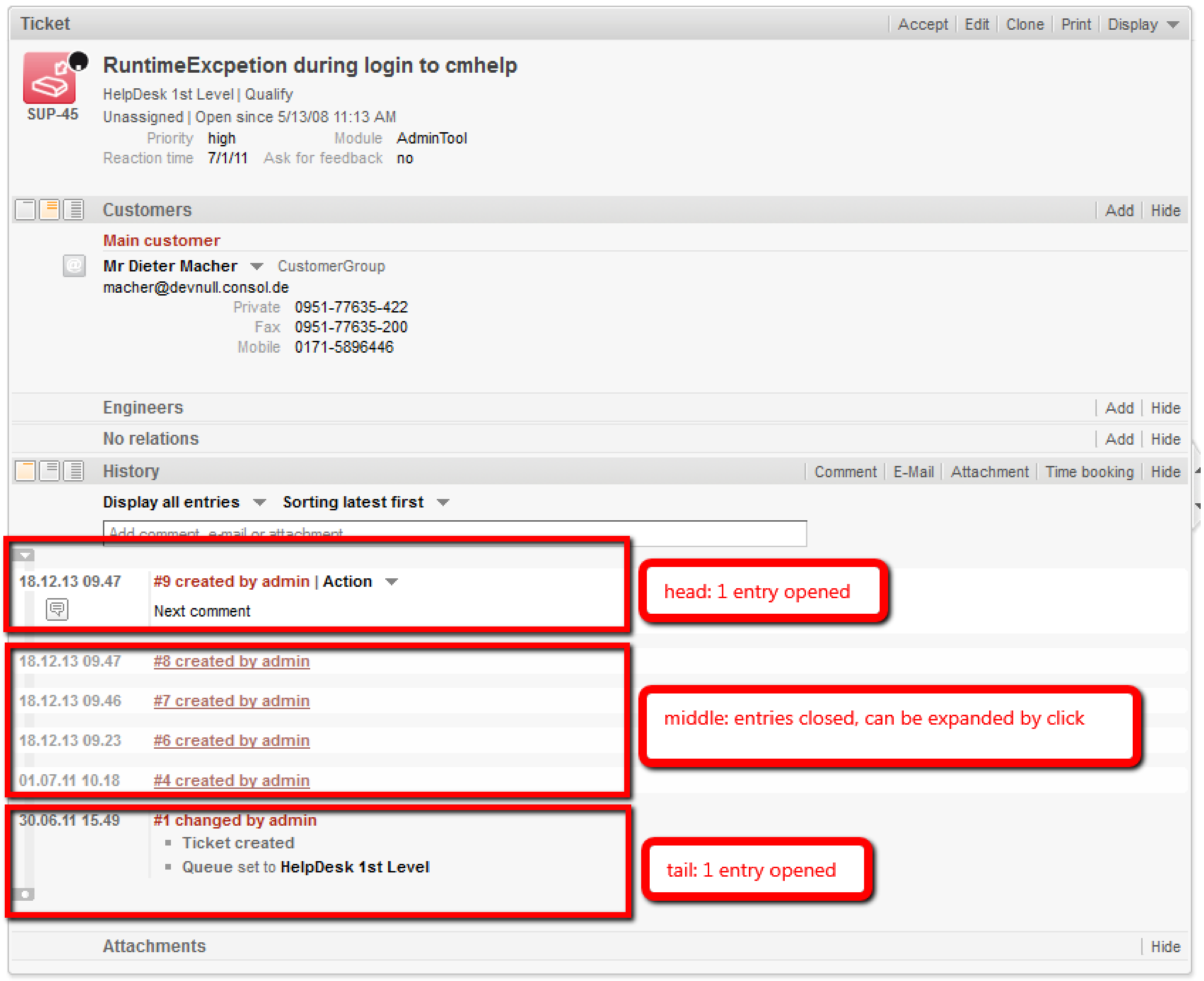
Bringing technology and computing closer to the body can actually allow technology to get further out of the way.

While these technologies lowered the barrier to communicating and accessing information, they created other barriers. We are now often greeted in meetings with a *literal barrier* of screens that get in the way of face-to-face communication.

Can we instead make devices that encourage in-person, face-to-face communication — while still delivering the data people need, only when they need it? I've been making and using wearable computers in my daily life since 1993 and have been a technical lead and manager on Google Glass since its first year in 2010. It may seem like a paradox, but I argue that bringing technology and computing *closer* to the body can actually improve communication and attention — allowing technology to get *further* out of the way.

Fig. 26: ConSol*CM Web Client - Lazy Loading Switched Off

Example 2: headHistoryElementsCount and tailHistoryElementsCount set to *1*


Ticket | Accept | Edit | Clone | Print | Display

SUP-45

RuntimeExcpetion during login to cmhelp

HelpDesk 1st Level | Qualify

Unassigned | Open since 5/13/08 11:13 AM

Priority high Module AdminTool

Reaction time 7/1/11 Ask for feedback no

Customers | Add | Hide

Main customer

Mr Dieter Macher CustomerGroup

macher@devnull.consol.de

Private 0951-77635-422

Fax 0951-77635-200

Mobile 0171-5896446

Engineers | Add | Hide

No relations | Add | Hide

History | Comment | E-Mail | Attachment | Time booking | Hide

Display all entries Sorting latest first

Add comment, e-mail or attachment

18.12.13 09.47 #9 created by admin | Action

Next comment

head: 1 entry opened

18.12.13 09.47 #8 created by admin

18.12.13 09.46 #7 created by admin

18.12.13 09.23 #6 created by admin

01.07.11 10.18 #4 created by admin

middle: entries closed, can be expanded by click

30.06.11 15.49 #1 changed by admin

- Ticket created
- Queue set to **HelpDesk 1st Level**

tail: 1 entry opened

Attachments | Hide

Fig. 27: ConSol*CM Web Client - headHistoryElementsCount and tailHistoryElementsCount Set to 1

- **mailToSelection**
  Initial e-mail address inserted into *To* field when composing e-mail.
  Possible options: {contact, contacts, engineer, fixed, none}, for *fixed* option see *mailToFixedMail* attribute. Default: *contact* (java.lang.String)
- **mailToFixedMail**
  Fixed e-mail address used when attribute *mailToSelection* is set to *fixed*. (java.lang.String)
  Example: E-mail to a fixed e-mail address, *mailToFixedMail* set to *foo@bar.de*, *mailToSelection* set to *fixed*.

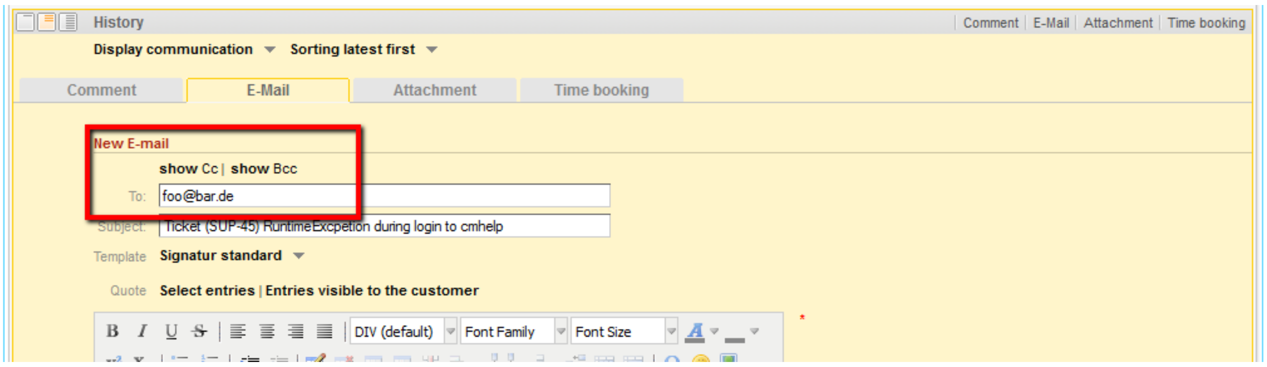
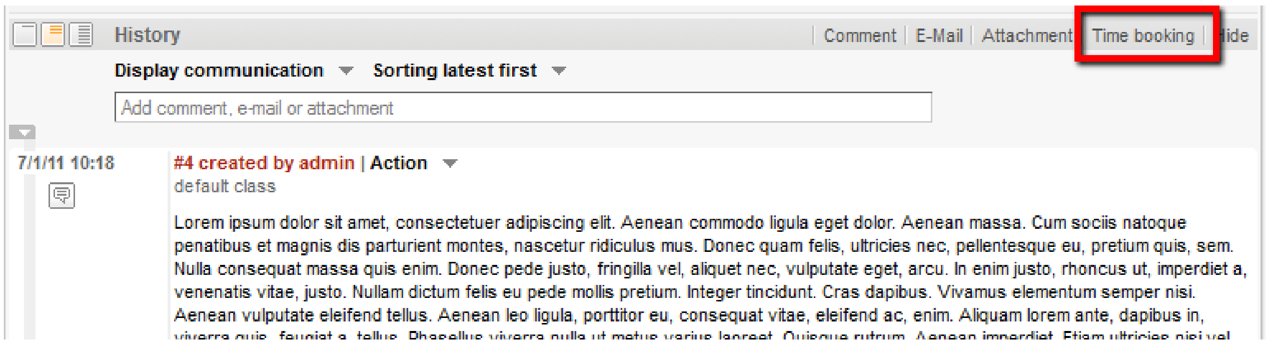

History
Display communication
Sorting latest first
Comment | E-Mail | Attachment | Time booking
Comment
E-Mail
Attachment
Time booking
New E-mail
show Cc | show Bcc
To: foo@bar.de
Subject: Ticket (SUP-45) RuntimeExcpetion during login to cmhelp
Template Signatur standard
Quote Select entries | Entries visible to the customer
DIV (default)
Font Family
Font Size

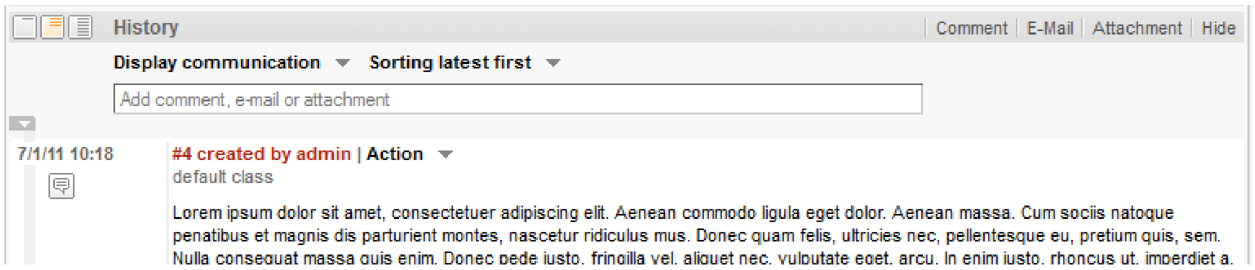
Fig. 28: ConSol*CM Web Client - E-Mail to a Fixed E-Mail Address

- **recordLastUsedAcimTab**
  Records last used ACIM tab, i.e. when you open the editor again, the tab (comment/e-mail /attachment/time booking) will be opened that was open when you closed the editor last time. (boolean)
- **reloadPageIfIE8onAcimSubmit**
  Reloads page after a new ACIM submit, only for *IE8*. This is a workaround for the problem that adding comments/e-mails may take a long time when using IE8. (boolean)
- **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 E-Mail Editor to the Comment Editor, the text you have typed will remain in the editor panel and will not be removed)
- **timeBookingFeature**
  Activates or deactivates the time booking support in acimSection (i.e. display the link to time booking and the tab in the editor). (boolean, default = *true*)


History
Comment | E-Mail | Attachment | Time booking | Hide
Display communication
Sorting latest first
Add comment, e-mail or attachment
7/1/11 10:18
#4 created by admin | Action
default class

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in,

Fig. 29: ConSol*CM Web Client - Activate Time Booking Support (timeBookingFeature = true)


History
Comment | E-Mail | Attachment | Hide
Display communication
Sorting latest first
Add comment, e-mail or attachment
7/1/11 10:18
#4 created by admin | Action
default class

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem.

Fig. 30: ConSol*CM Web Client - De-activate Time Booking Support (timeBookingFeature = false)

Information:

Please note that the value *false* in the *timeBookingFeature* hides the hyperlink from the time booking editor, see figure above. The engineer cannot show it again by using the *Display* option in the ticket's menu!

Please keep in mind that the visibility of the time booking section on the *userProfilePage* is configured via the *timeBookingSection* attribute *visible*!

accordionTicketList (Type)

Here you can define attributes for the ticket list.Only one subscope is available: *ticketList*

Attributes:

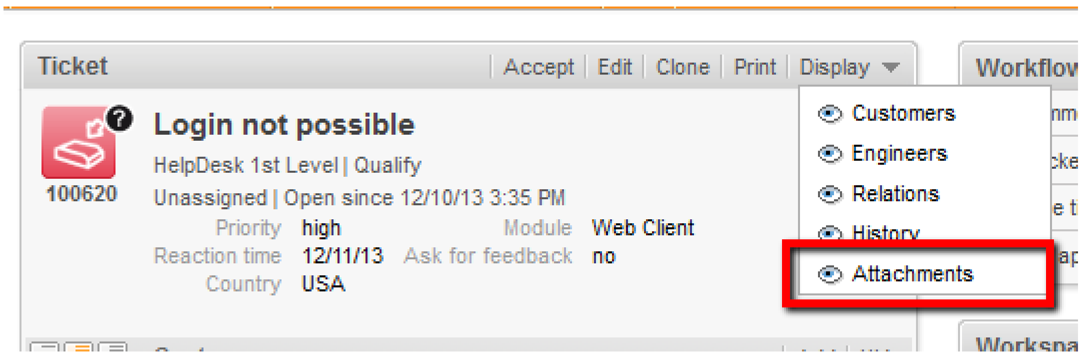
- **loadingTicketListMode**
  The mode used to render the ticket list. There are four options to select from:
  1. LAZY_SYNCH (default)
     The waiting indicator is shown in the place of the ticket list while the rest of a page is rendered. Then the ticket list is loaded. The main benefit of this approach is the possibility to show/read the main content faster.
  2. 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.
  3. INCLUDED
     The ticket list is loaded along with the rest of a page.
  4. 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 immediately after two necessary libraries are 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*.

- **quickAssignLinkShowsTicketPageFlag**
  Whether the *quick assign* link (represented by the arrow rendered for each unassigned ticket) opens the assigned ticket immediately (boolean, default is *false*).

attachmentSection (Type or Subscope)

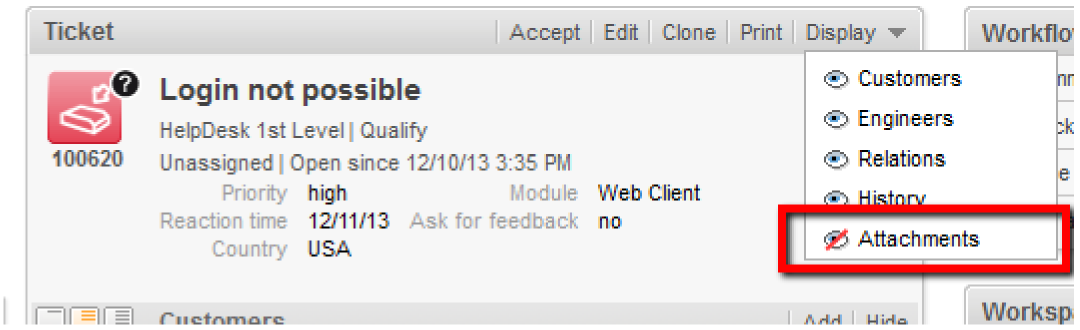
Attributes:

- **defaultVisibilityFlag**
  The visibility of the attachment section for engineers who did not change the default visibility as yet (for others the last visibility state is used, default = *false*, i.e. attachment section is not displayed). This feature defines only the initial behavior of the system. The engineer can show the attachment section using the *Display* option in the ticket's menu and the Web Client will remember this configuration for this particular engineer.



Login not possible
100620
HelpDesk 1st Level | Qualify
Unassigned | Open since 12/10/13 3:35 PM
Priority high Module Web Client
Reaction time 12/11/13 Ask for feedback no
Country USA

Fig. 31: ConSol*CM Web Client - Visibility of Attachment Section (defaultVisibilityFlag = true)



Login not possible
100620
HelpDesk 1st Level | Qualify
Unassigned | Open since 12/10/13 3:35 PM
Priority high Module Web Client
Reaction time 12/11/13 Ask for feedback no
Country USA

Fig. 32: ConSol*CM Web Client - Visibility of Attachment Section (defaultVisibilityFlag = false)

autocomplete

(available e.g. on *userProfilePage*)

Attributes:

- **suffixCharactersToRemove**
  Occurrence of any 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.

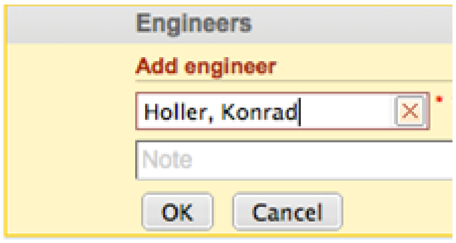


Fig. 33: Input in a search field where charactes should be ignored in the search

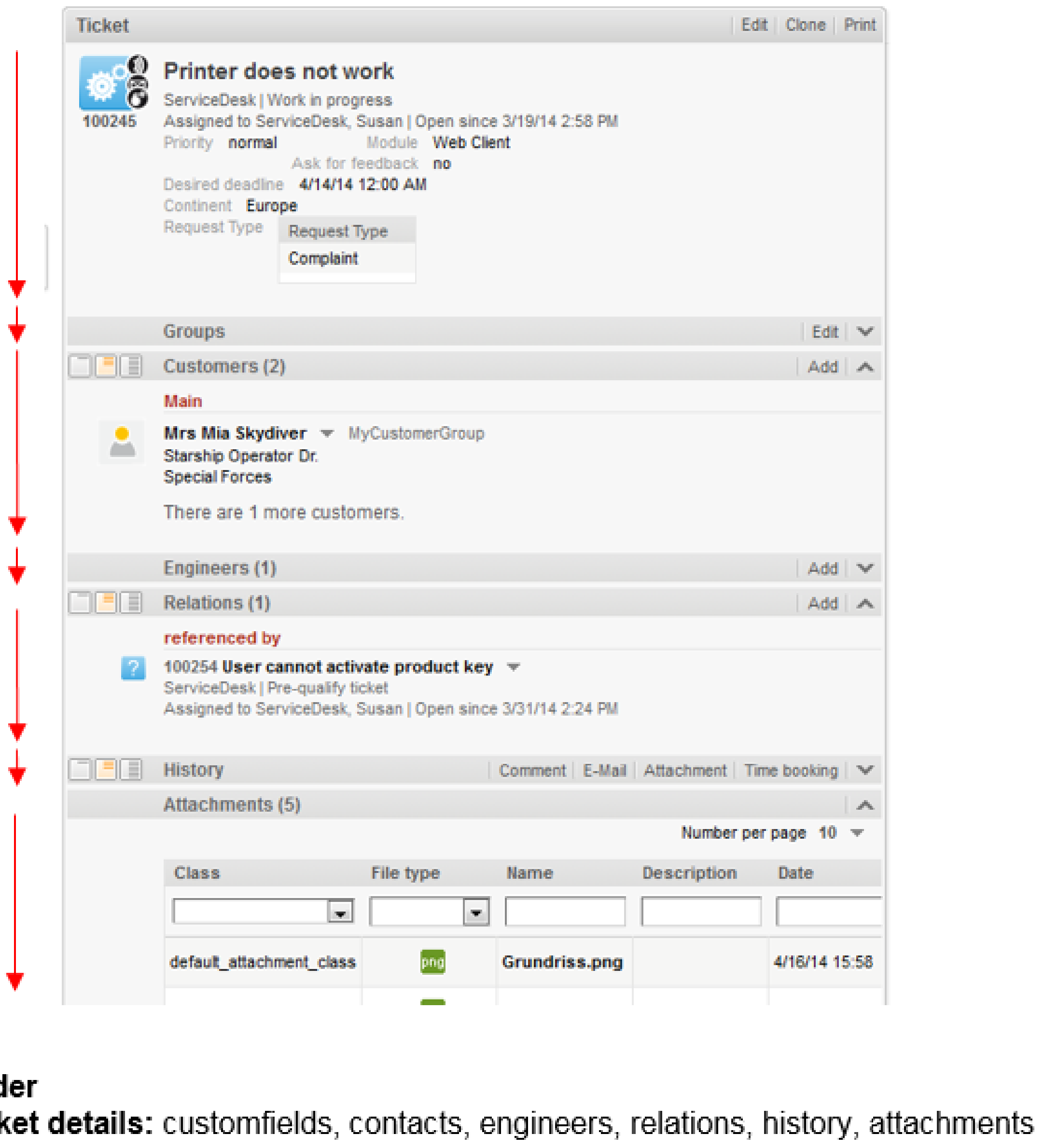
boxContent

Available for the following scopes:

- ticketEditPage
- userProfilePage

Attributes:

- **order**
  Specify the order of the sections within a ticket in cvs format (eg: 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 at 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.)


Ticket | Edit | Clone | Print

100245

Printer does not work
ServiceDesk | Work in progress
Assigned to ServiceDesk, Susan | Open since 3/19/14 2:58 PM
Priority normal Module Web Client
Ask for feedback no
Desired deadline 4/14/14 12:00 AM
Continent Europe
Request Type

Request Type
Complaint

Groups | Edit

Customers (2) | Add

Main
Mrs Mia Skydiver MyCustomerGroup
Starship Operator Dr.
Special Forces
There are 1 more customers.

Engineers (1) | Add

Relations (1) | Add

referenced by
100254 **User cannot activate product key**
ServiceDesk | Pre-qualify ticket
Assigned to ServiceDesk, Susan | Open since 3/31/14 2:24 PM

History | Comment | E-Mail | Attachment | Time booking

Attachments (5)

Number per page 10

Class	File type	Name	Description	Date
default_attachment_class	png	Grundriss.png		4/16/14 15:58

order
ticket details: customfields, contacts, engineers, relations, history, attachments

Fig. 34: Default value for attribute boxContent / order

cmRichTextEditor

Attributes:

- **editorFeatures**
  Additional editor features. As default value, all values are set, i.e. the respective menu entries are available. (java.lang.String)

  Possible values:

  - **SUB_SUP** X_2 X^2
    Allows subscript and superscript.

    - **INDENTS**
      Provides the possibility to add indents to the text.
    - **LISTS**
      Allows to insert/format lists.
    - **TABLES**
      Allows to insert tables.
    - **INSERT**
      Allows to insert text elements; for finer control:
        - **INSERT_EMOTION**
          Allows to insert emoticons.
        - **INSERT_CHAR**
          Allows to insert special characters.
        - **INSERT_IMAGE**
          Allows to insert images.
- **editorFonts**
  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. (java.lang.String) (default = empty string)
  Example: Arial=arial,helvetica,sans-serif;Courier New=courier new,courier;Verdana=verdana,geneva
- **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**
  This is the list of values offered 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”.
- **imagePasteEnabled**
  Flag, whether direct pasting of images from clipboard into editor is enabled. Note that enabling this requires a Java Applet to run (boolean, default = false). Web browsers (i.e. Internet Explorer, Firefox) might show different behavior concerning display of the images.

customerSectionPanel

Here you can define if 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 Role Administration) to edit company data.

Attributes:

- **referencedUnitEditLinkVisible** (Versions 6.9.3.3 and less. Similar, new attribute is *companyEditLinkVisible*)
  The visibility of the link for editing referenced units (boolean, default is *true*).

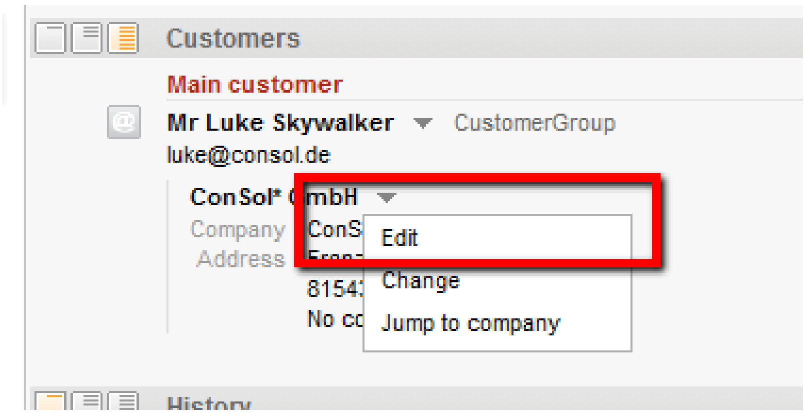


Fig. 35: ConSol*CM Web Client - Visibility of Edit Link (referencedUnitEditLinkVisible = true)

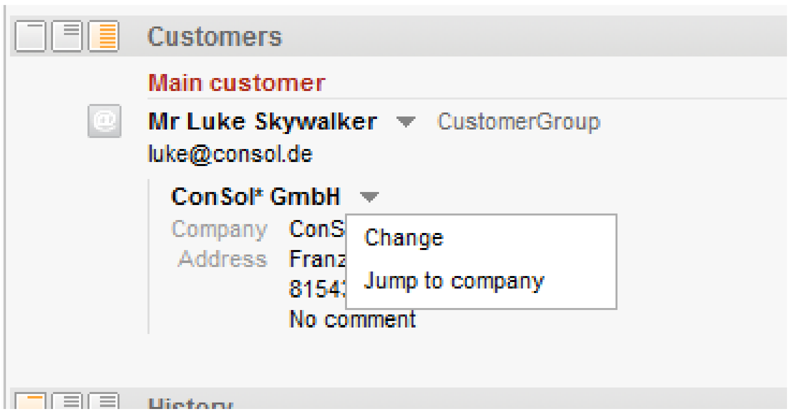


Fig. 36: ConSol*CM Web Client - Visibility of Edit Link (referencedUnitEditLinkVisible = false)

- **companyEditLinkVisible** (Versions 6.9.3.4 and up. Older attribute is *referencedUnitEditLinkVisible*) (available in *customerSectionPanel* in ticket and on *companyEditPage* and *contactEditPage*)
  The visibility of the link for editing a company, default is *true*.

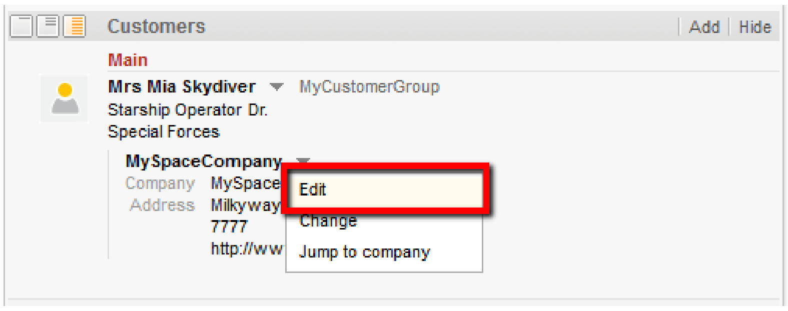


Fig. 37: ConSol*CM Web Client - Visibility of Company Edit Link (companyEditLinkVisible = true)

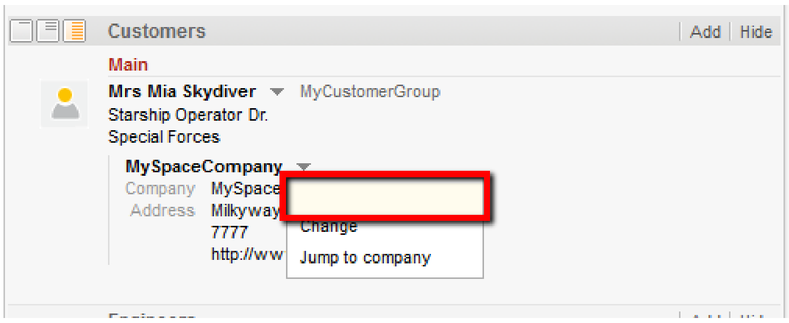
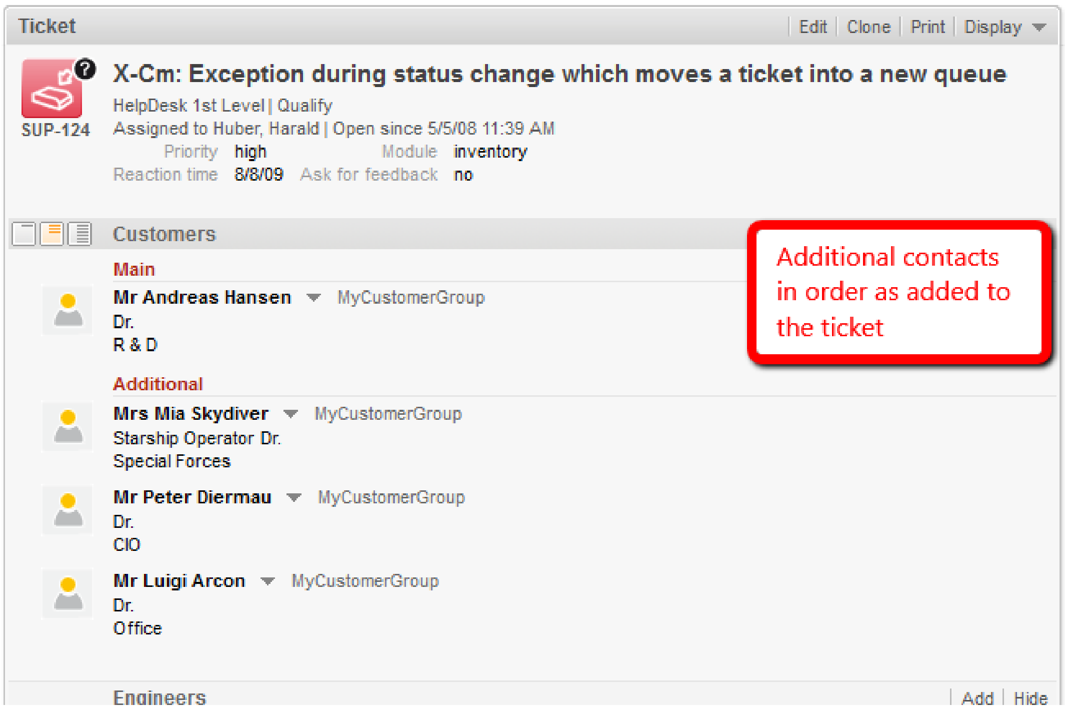


Fig. 38: ConSol*CM Web Client - Visibility of Company Edit Link (companyEditLinkVisible = false)

- **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.


Ticket | Edit | Clone | Print | Display

SUP-124 X-Cm: Exception during status change which moves a ticket into a new queue
HelpDesk 1st Level | Qualify
Assigned to Huber, Harald | Open since 5/5/08 11:39 AM
Priority high Module inventory
Reaction time 8/8/09 Ask for feedback no

Customers
Main
Mr Andreas Hansen MyCustomerGroup
Dr.
R & D
Additional
Mrs Mia Skydiver MyCustomerGroup
Starship Operator Dr.
Special Forces
Mr Peter Diermau MyCustomerGroup
Dr.
CIO
Mr Luigi Arcon MyCustomerGroup
Dr.
Office

Additional contacts in order as added to the ticket

Engineers | Add | Hide

Fig. 39: ConSol*CM Web Client - Page Customization for Sort Order of Additional Coustomers (1)

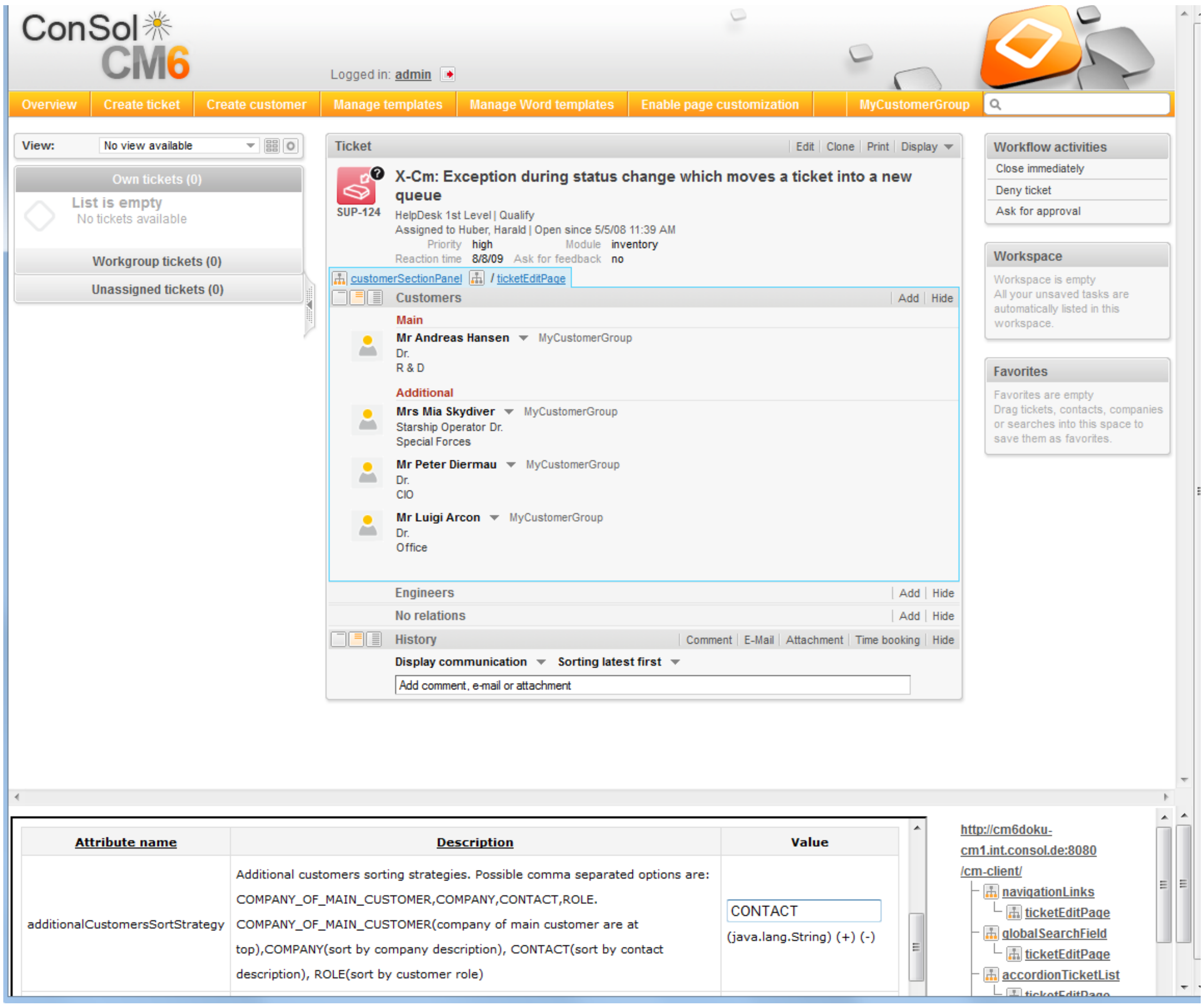
The following values are possible:

- **COMPANY_OF_MAIN_CUSTOMER**
  Contacts are sorted by the company description with the company of the main customer first.
- **COMPANY**
  Contacts are sorted by the company description.
- **CONTACT**
  Contacts are sorted by the contact description.
- **ROLE**
  Contacts 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 data object descriptions used for the display of contacts and companies are taken from the template *<Contact>.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*.



List is empty
No tickets available

SUP-124
X-Cm: Exception during status change which moves a ticket into a new queue
HelpDesk 1st Level | Qualify
Assigned to Huber, Harald | Open since 5/5/08 11:39 AM
Priority high Module inventory
Reaction time 8/8/09 Ask for feedback no

customerSectionPanel / ticketEditPage

Customers

Main
Mr Andreas Hansen MyCustomerGroup
Dr.
R & D

Additional
Mrs Mia Skydiver MyCustomerGroup
Starship Operator Dr.
Special Forces

Mr Peter Diermau MyCustomerGroup
Dr.
CIO

Mr Luigi Arcon MyCustomerGroup
Dr.
Office

Engineers
No relations
History
Display communication Sorting latest first
Add comment, e-mail or attachment

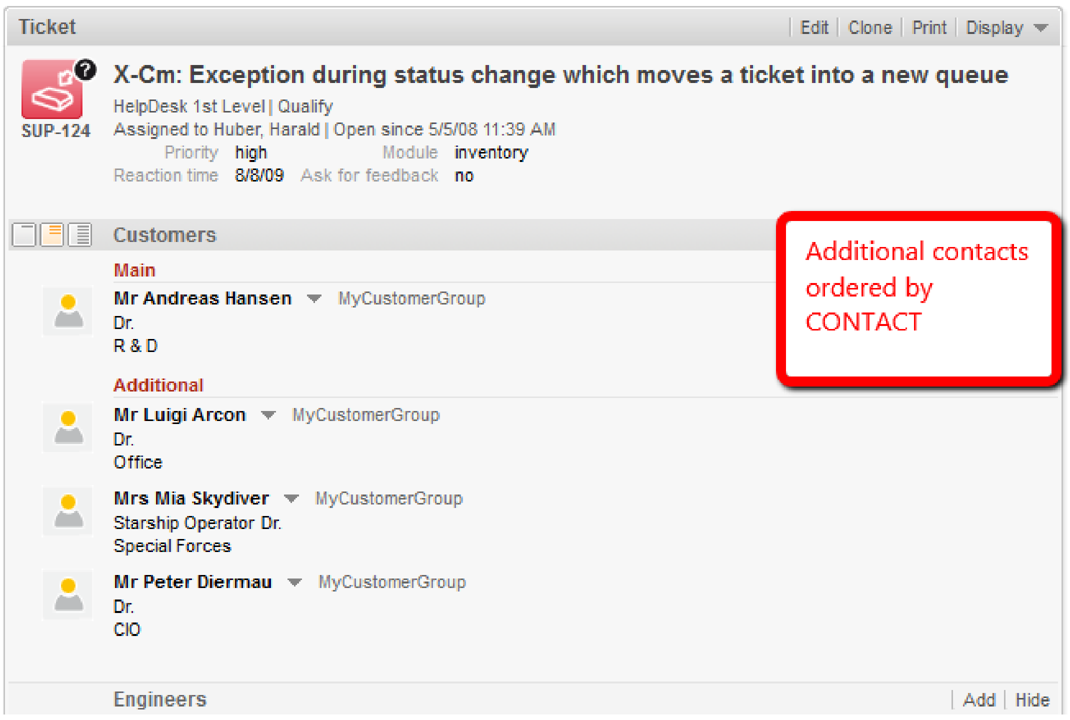
Workspace is empty
All your unsaved tasks are automatically listed in this workspace.

Favorites are empty
Drag tickets, contacts, companies or searches into this space to save them as favorites.

Attribute name	Description	Value
additionalCustomersSortStrategy	Additional customers sorting strategies. Possible comma separated options are: COMPANY_OF_MAIN_CUSTOMER,COMPANY,CONTACT,ROLE. COMPANY_OF_MAIN_CUSTOMER(company of main customer are at top),COMPANY(sort by company description), CONTACT(sort by contact description), ROLE(sort by customer role)	CONTACT (java.lang.String) (+) (-)

http://cm6doku-cm1.int.consol.de:8080/cm-client/
- navigationLinks
  - ticketEditPage
- globalSearchField
  - ticketEditPage
- accordionTicketList
  - ticketEditPage

Fig. 40: ConSol*CM Web Client - Page Customization for Sort Order of Additional Customers (2)


Ticket | Edit | Clone | Print | Display

X-Cm: Exception during status change which moves a ticket into a new queue

SUP-124

HelpDesk 1st Level | Qualify

Assigned to Huber, Harald | Open since 5/5/08 11:39 AM

Priority high Module inventory

Reaction time 8/8/09 Ask for feedback no

Customers

Main

Mr Andreas Hansen MyCustomerGroup
Dr.
R & D

Additional

Mr Luigi Arcon MyCustomerGroup
Dr.
Office

Mrs Mia Skydiver MyCustomerGroup
Starship Operator Dr.
Special Forces

Mr Peter Diermau MyCustomerGroup
Dr.
CIO

Engineers | Add | Hide

Additional contacts ordered by CONTACT

Fig. 41: ConSol*CM Web Client - Page Customization for Sort Order of Additional Customers (3)

detailSearch

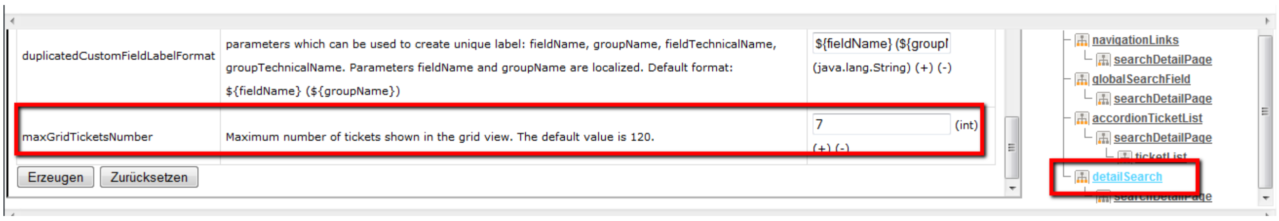
- **criteriaForAllTypeOfContacts**
  Boolean field, when set to *true*, the search will include the main customers and the additional customers of tickets. When set to *false* (default), the search will apply the search criteria only to the main customers of tickets.
- **duplicatedCustomFieldLabelFormat**

It allows to customize the label format used when several custom fields have the same localized name. There are four values which can be used to create a unique label:

-
  - fieldName
  - groupName
  - fieldTechnicalName
  - groupTechnicalName

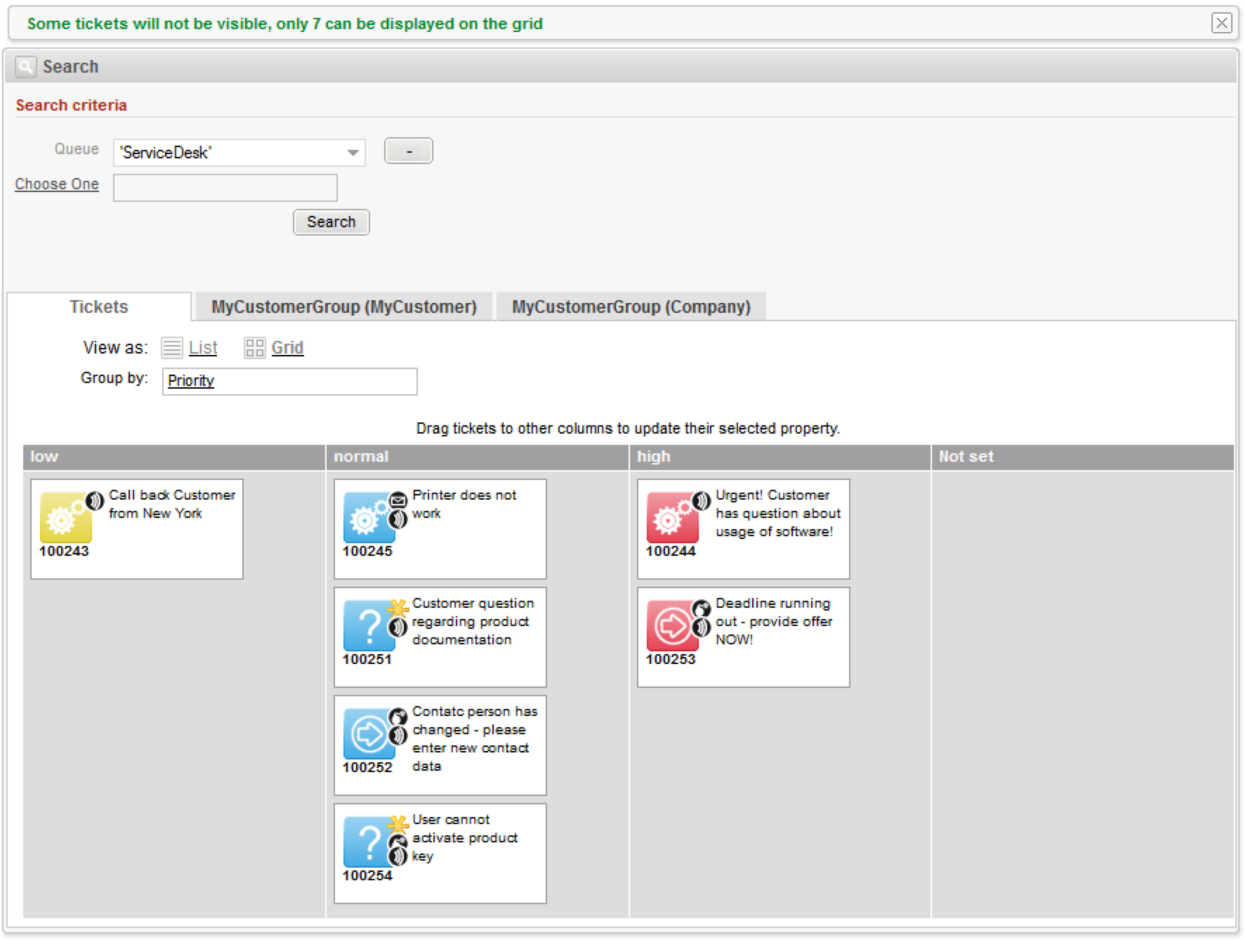
The values *fieldName* and *groupName* are localized.
Default format: ${fieldName} (${groupName}) (java.lang.String)
This helps distinguish different Custom Fields or Data Object Group Fields with the same name, e.g. when they are offered in the drop-down menu in the detail search.
Example:
SalesProcess - Priority
HelpdeskProcess - Priority

- **maxGridTicketsNumber**
  Maximum number of tickets shown in grid view, i.e. in the entire grid, not in one column. The default value is *120*.



duplicatedCustomFieldLabelFormat	parameters which can be used to create unique label: fieldName, groupName, fieldTechnicalName, groupTechnicalName. Parameters fieldName and groupName are localized. Default format: ${fieldName} (${groupName})	${fieldName} (${groupl (java.lang.String) (+) (-)
maxGridTicketsNumber	Maximum number of tickets shown in the grid view. The default value is 120.	7 (int) (+) (-)

Fig. 42: ConSol*CM Web Client - Using Page Customization to Adapt Maximum Number of Tickets Shown in Grid



Some tickets will not be visible, only 7 can be displayed on the grid

Search criteria

Queue: 'ServiceDesk'

Tickets | MyCustomerGroup (MyCustomer) | MyCustomerGroup (Company)

View as: List Grid

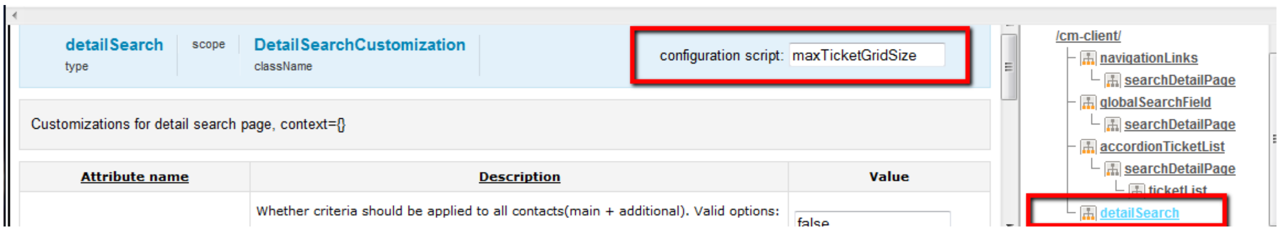
Group by: Priority

Drag tickets to other columns to update their selected property.

low	normal	high	Not set
Call back Customer from New York 100243	Printer does not work 100245	Urgent! Customer has question about usage of software! 100244	
	Customer question regarding product documentation 100251	Deadline running out - provide offer NOW! 100253	
	Contatc person has changed - please enter new contact data 100252		
	User cannot activate product key 100254		

Fig. 43: ConSol*CM Web Client - Maximum Number in Ticket Grid with Maximum Seven

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.



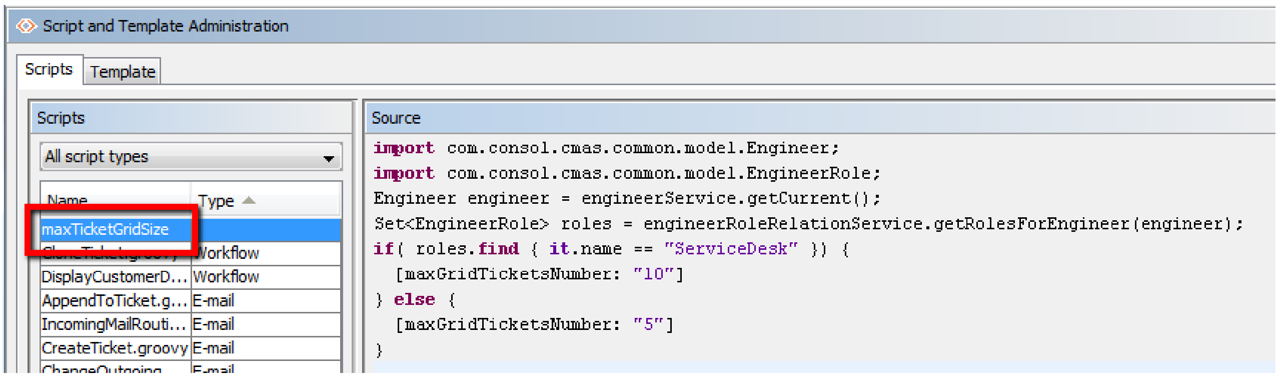
detailSearch type | scope | DetailSearchCustomization className

configuration script: maxTicketGridSize

Customizations for detail search page, context={}

Attribute name	Description	Value
	Whether criteria should be applied to all contacts(main + additional). Valid options:	false

Fig. 44: ConSol*CM Web Client - Defining a Script for Maximum Number of Tickets in Grid View Using Page Customization


Script and Template Administration
Scripts
Template
Scripts
All script types
Name
Type
maxTicketGridSize
DisplayCustomerD... Workflow
AppendToTicket.g... E-mail
IncomingMailRouti... E-mail
CreateTicket.groovy E-mail
Source

```
import com.consol.cmas.common.model.Engineer;
import com.consol.cmas.common.model.EngineerRole;
Engineer engineer = engineerService.getCurrent();
Set<EngineerRole> roles = engineerRoleRelationService.getRolesForEngineer(engineer);
if( roles.find { it.name == "ServiceDesk" }) {
  [maxGridTicketsNumber: "10"]
} else {
  [maxGridTicketsNumber: "5"]
}
```

Fig. 45: ConSol*CM Admin Tool - Script for Maximum Number of Tickets in Grid

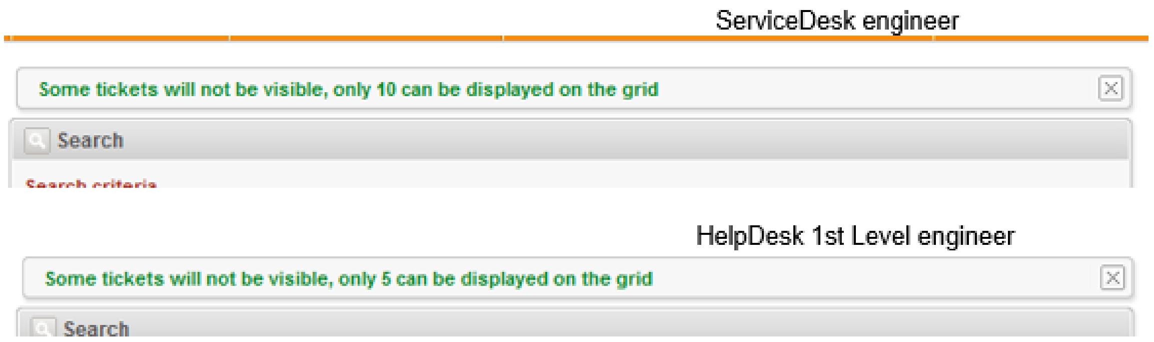

ServiceDesk engineer
Some tickets will not be visible, only 10 can be displayed on the grid
Search
HelpDesk 1st Level engineer
Some tickets will not be visible, only 5 can be displayed on the grid
Search

Fig. 46: ConSol*CM Web Client - Number of Tickets in Ticket Grid for Different Engineer Roles

engineerAutocomplete

e.g. engineerAutocomplete / ticketCreatePage / contactSection

Attributes:

- **maxSuggestions**
  This refers to the customer/contact section which is displayed when you create a new ticket. Here, suggestions for contacts are displayed if matching hits are found in the database. The number of suggestions which are displayed can be configured using this attribute.

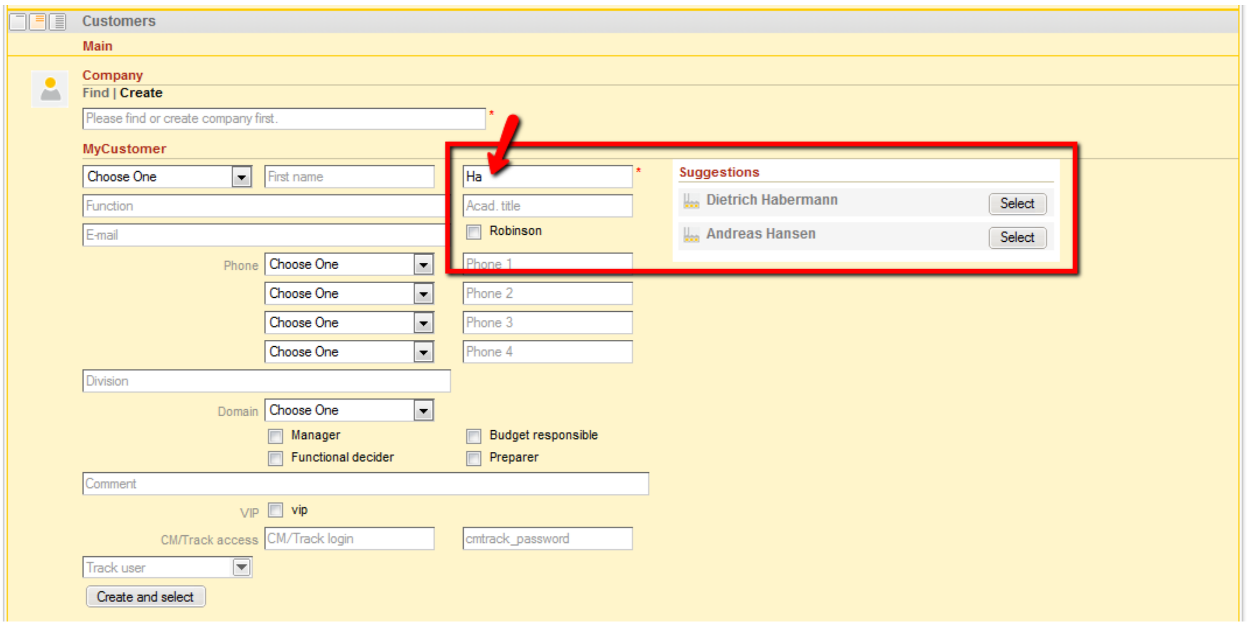


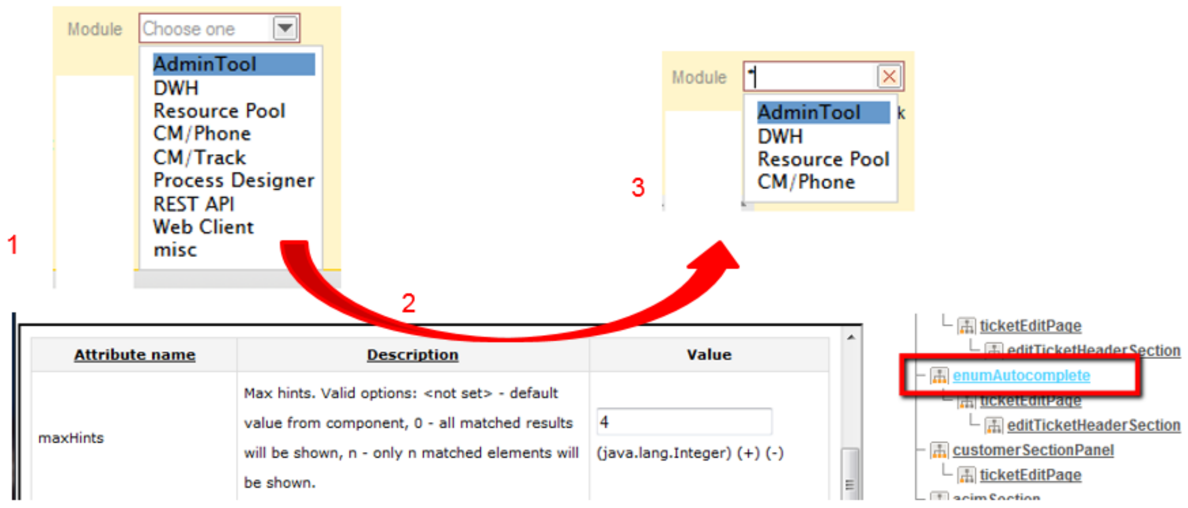
Fig. 47: ConSol*CM Web Client - Suggestions for the Ticket Contact

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, no limit.
- **suffixCharactersToRemove**
  Occurrence of any of these characters will be removed from the tail of each search pattern word. (string, default: empty)



Attribute name	Description	Value
maxHints	Max hints. Valid options: <not set> - default value from component, 0 - all matched results will be shown, n - only n matched elements will be shown.	4 (java.lang.Integer) (+) (-)

Fig. 48: ConSol*CM Web Client - Using enumAutocomplete Parameter to Reduce Number of Hints

engineerAutocomplete

(available on *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)

globalSearchField

Here you can define the layout and the behavior of the *Global Search* field, this is the quick search input field.

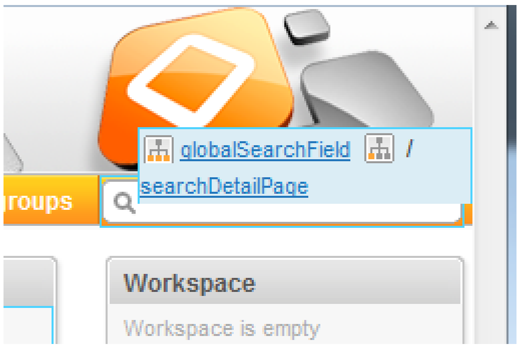


Fig. 49: ConSol*CM Web Client - Quick Search Input Field

Attributes:

- **maxHints**
  Defines the maximum number of suggestions which is displayed. When set to *0*, all suggestions are displayed, 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

Attributes:

- **addingManyTemplateEnabled**
  Makes it possible to compose an e-mail by using more than one template. (boolean)

- **engineerPersonalMailsIncluded**
  Enable appending personal e-mail feature. (boolean)
- **includeTextBlocksInEmailTemplate**
  Whether text blocks from e-mail template will be included by default. (boolean)
- **mailBodyLockedAfterTemplateSelection**
  Indicates whether e-mail body will be locked after template selection. (boolean)
- **mailEncryptionAvailable**
  Makes e-mail encryption option available. (boolean)
- **mailSelectionComponentWidth**
  The width of the e-mail selection component (in pixels). (java.lang.Integer)
- **mailTemplateSortStrategy**
  E-mail 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)
- **quotingFeature**
  Activate the quoting function in e-mail content. (boolean)
- **showUniqueEmails**
  Results in autocomplete e-mail fields will be compared by e-mail, only first e-mail will be used in results. If set to *true* then results will be compared by whole e-mail description. (boolean)

navigationLinks

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* link can be shown (boolean, i.e. possible values are *true* or *false*, default value is *true*).

  Information:

  In addition to this attribute being set to *true*, engineers must have appropriate permissions to see the *createContact* link.

- **createTicketLinkVisible**
  Whether the *createTicket* link can be shown (boolean, i.e. possible values are *true* or *false*, default value is *true*).

  Information:

In addition to this attribute being set to *true*, engineers must have appropriate permissions to see the *createTicket* link.

- **externalLinks**
  External links which will be appended to navigation bar. This attribute may configure more than one external link (the order matters).
  Format (compatible with wiki): [http://link description]
  This attribute might 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 on the Internet and/or intranet.
  Example: [http://www.consol.com ConSol*][http://www.somewhere.com Somewhere]
- **manageTemplateLinkVisible**
  Whether the *Manage Templates* link (for the start of the *ConSol*CM Template Manager*) can be shown (boolean, i.e. possible values are *true* or *false*, default value is *true*). See also section The ConSol*CM Template Manager.

Information:

In addition to this attribute being set to *true*, engineers must have appropriate permissions (*Global Permissions* - *Write template*) to see the *Manage Template*s link.

- **officeTemplateLinkVisible**
  Whether the *officeTemplate* link (for the start of the *Doc Template Manager*) can be shown (boolean, i.e. possible values are *true* or *false*, default value is *true*).

Information:

In addition to this attribute being set to *true*, engineers must have appropriate permissions (*Global Permissions* - *Write template*) to see the *officeTemplate* link. CM.Doc has to be enabled in the system (see section CM.Doc).

- **overviewLinkVisible**
  Whether the *Overview* link can be shown.

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)

Avaiable in the following scopes.

- ticketEditPage
- contactEditPage
- companyEditPage

Attributes:

- **counterVisible**
  Whether the counter should be shown in the section header. The default value is true.

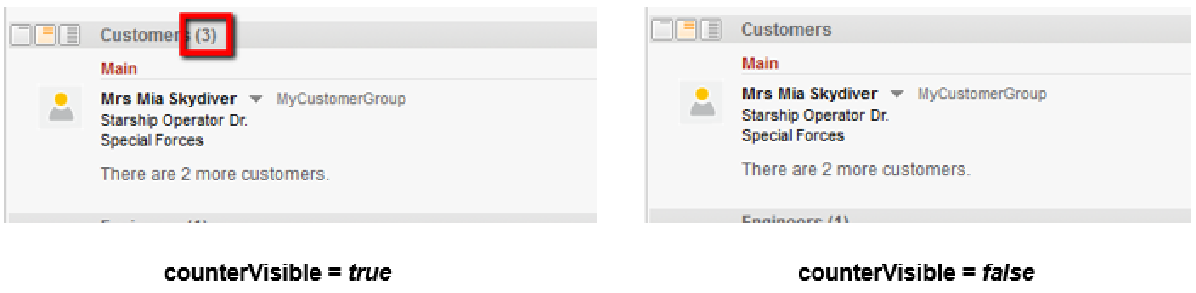


Fig. 50: ConSol*CM Web Client - Attribute counterVisible

- **state**
  The visibility mode of the section, possible values are [expanded, collapsed, collapsed_and_preload, hidden], default: 'expanded'. 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 (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 by database or JBoss Admin Console access ⚠

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 by database access or using the JBoss Admin Console ⚠ So please think twice before applying this value ⚠

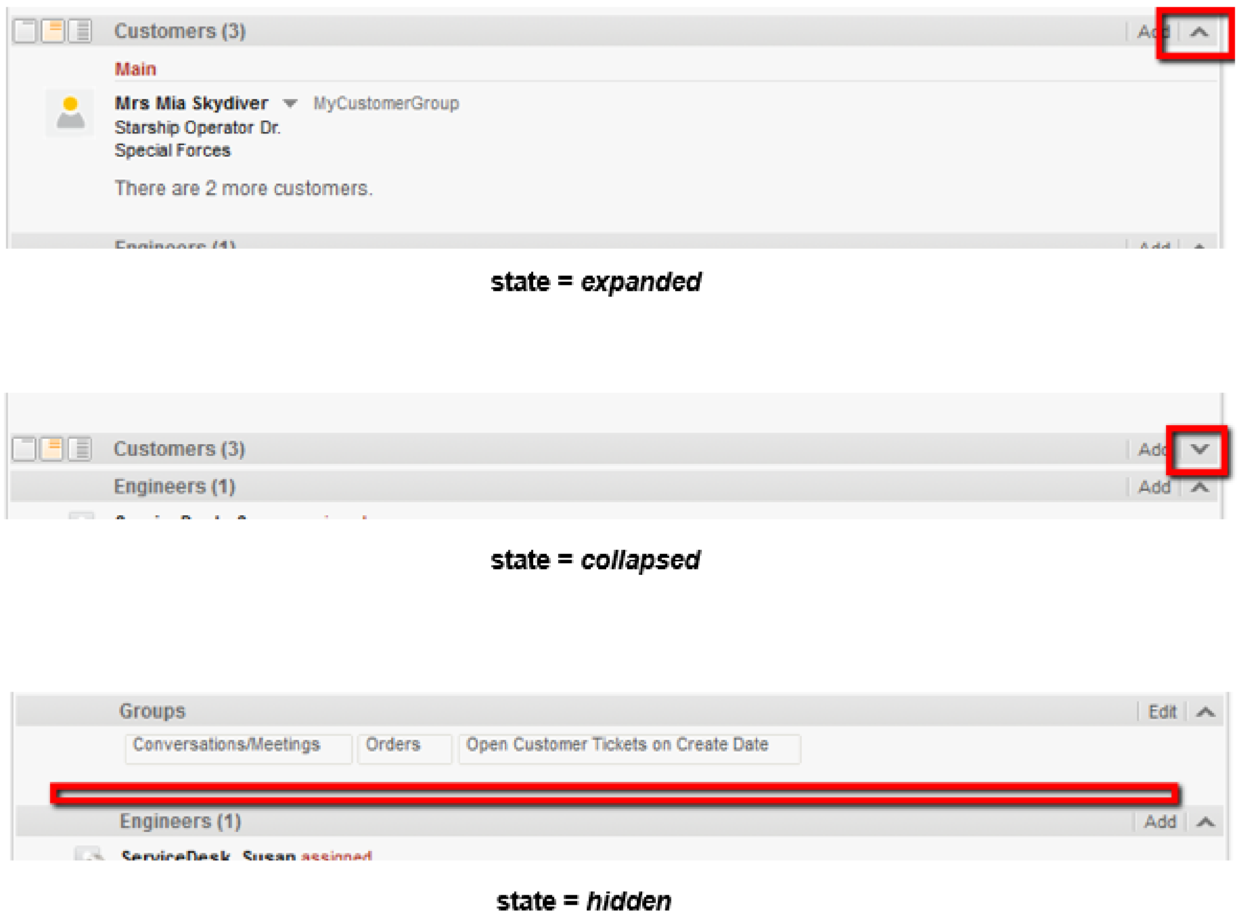


Fig. 51: ConSol*CM Web Client - attribute state

ticketList (Subscope)

see *accordionTicketList* (type)

ticketsAutocomplete

(available on the relations addition form)

Attributes:

- **maxHints**
  Defines the maximum number of suggestions which is displayed. When set to *0*, all suggestions are displayed, no limit.

ticketsBookingAutocomplete

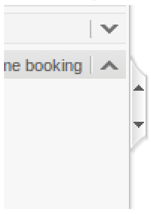
(available on the time booking addition form of the *userProfilePage*)

Attributes:

- **maxHints**
  Defines the maximum number of suggestions which is displayed. When set to *0*, all suggestions are displayed, no limit.

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. Typical values: 200, 600, 1000 ... (higher value means slower) (java.lang.String, default = 200). E.g., 200 means that the scroll to the bottom /top of the page will take 200ms.
- **topBottomPageButtonVisible**
  Whether *go to top and bottom page* button is visible (boolean, default is *false*).
  - **topBottomPageButtonVisible = false:**

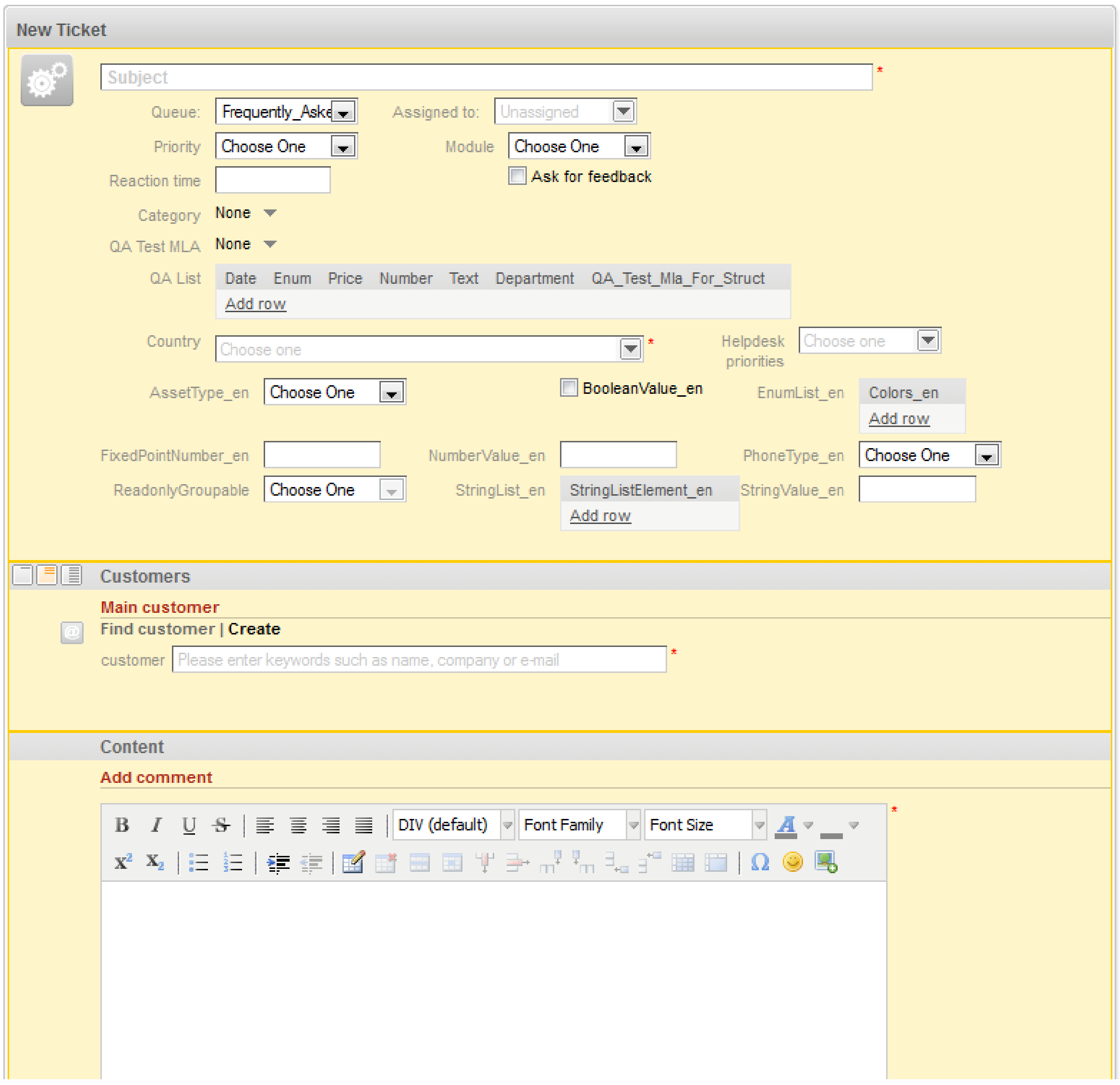
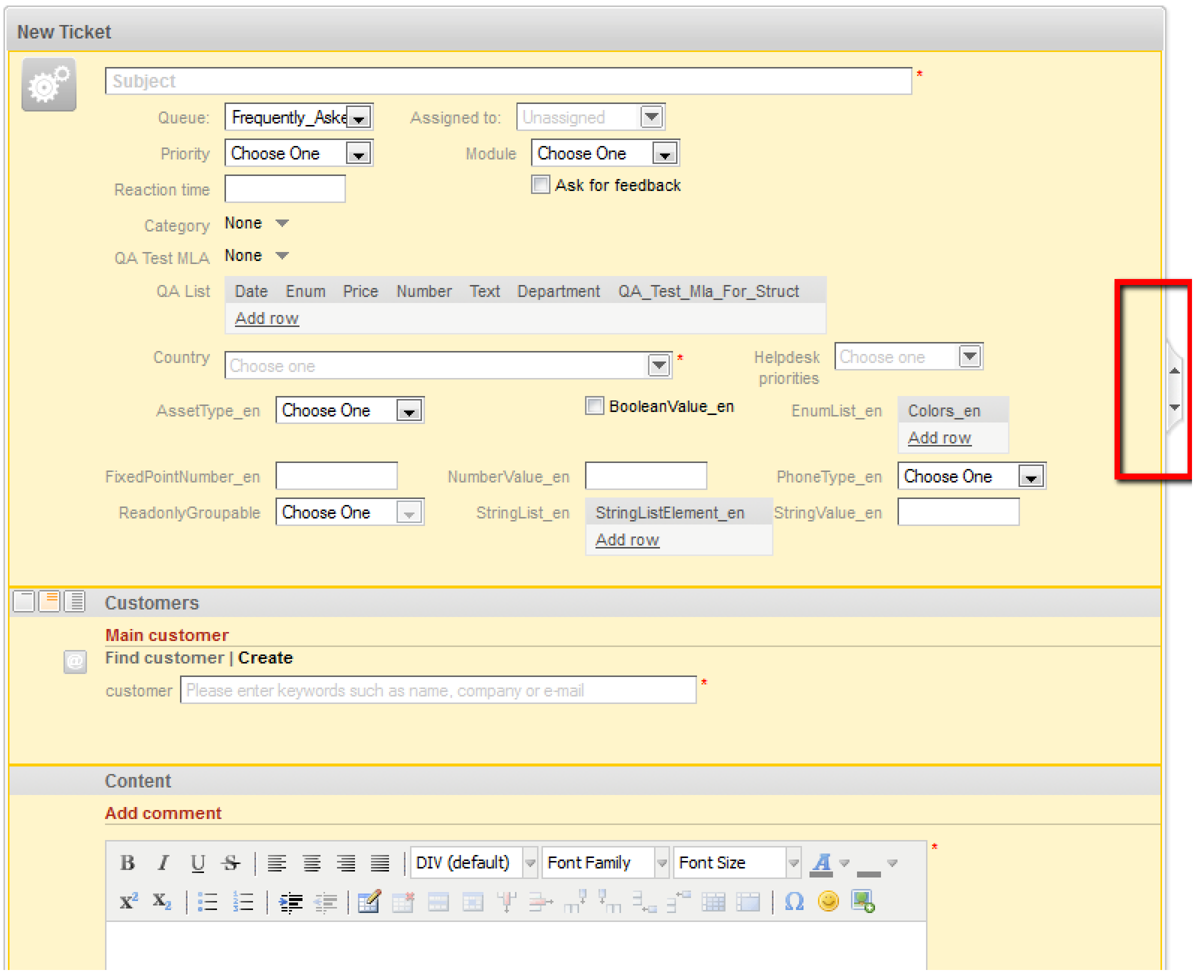


Fig. 52: ConSol*CM Web Client - Button "go to top and bottom page" Not Visible

- **topBottomPageButtonVisible = true:**


New Ticket

Subject

Queue: Frequently_Aske
Assigned to: Unassigned
Priority Choose One
Module Choose One
Reaction time
Ask for feedback
Category None
QA Test MLA None
QA List: Date Enum Price Number Text Department QA_Test_Mla_For_Struct
Add row
Country Choose one
Helpdesk priorities Choose one
AssetType_en Choose One
BooleanValue_en
EnumList_en Colors_en
Add row
FixedPointNumber_en
NumberValue_en
PhoneType_en Choose One
ReadonlyGroupable Choose One
StringList_en StringListElement_en
Add row
StringValue_en

Customers
Main customer
Find customer | Create
customer Please enter keywords such as name, company or e-mail

Content
Add comment

Fig. 53: ConSol*CM Web Client - Button "go to top and bottom page" Visible

timeBookingSection

(available e.g. on *userProfilePage*)

Attributes:

- **visible**
  The visibility of the time booking section on the *userProfilePage*. (boolean, default value = *true*)
  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, no limit.

unitFormPanel

(available on *contactCreatePage*, *ticketEditPage*, *contactEditPage*)

Attributes:

- **maxSuggestions**
  The limit of unit suggestions. Must be greater than zero.

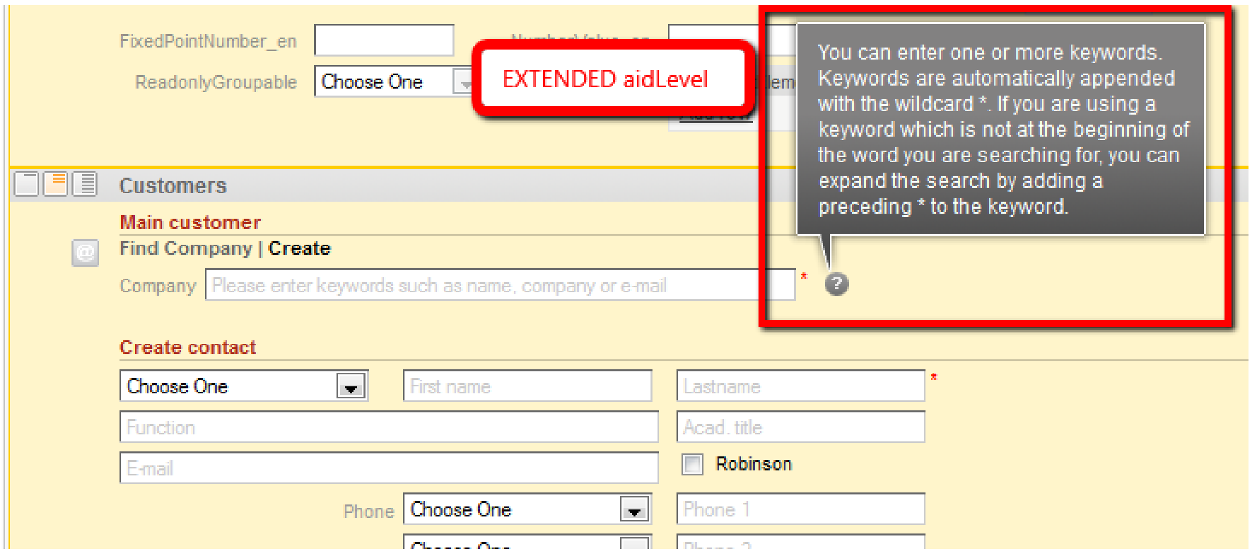
unitSearch

(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*)



You can enter one or more keywords. Keywords are automatically appended with the wildcard *. If you are using a keyword which is not at the beginning of the word you are searching for, you can expand the search by adding a preceding * to the keyword.

Please enter keywords such as name, company or e-mail

Fig. 54: ConSol*CM Web Client - EXTENDED aidLevel in the Unit Search

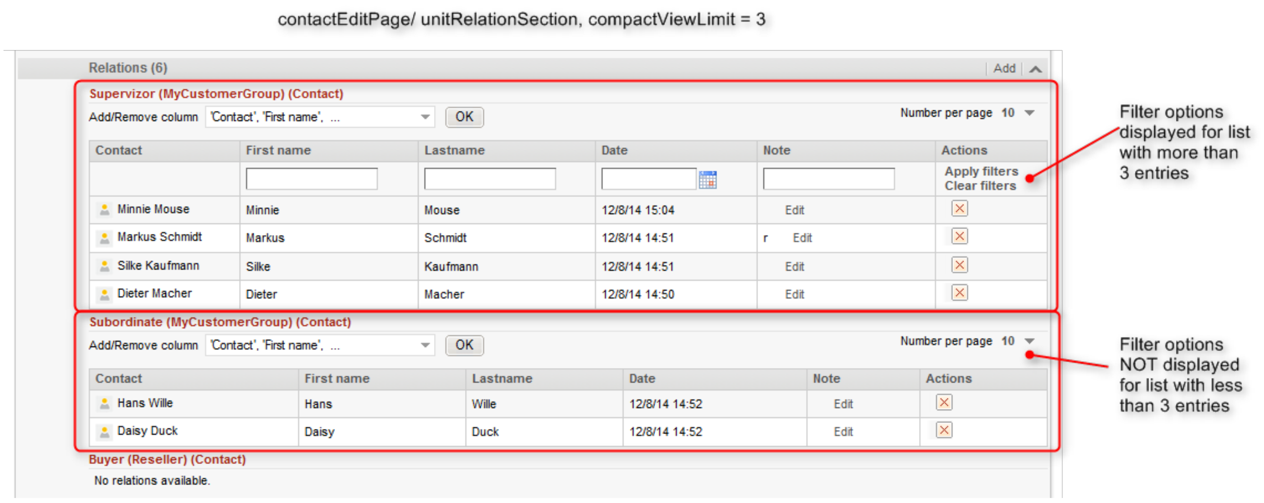
unitRelationSection (Type UnitSection)

Available on the

- contactEditPage
- companyEditPage

Attributes:

- **compactViewLimit**
  If the limit is exceed the relations are presented along with filtering feature. The default limit is 10.


contactEditPage/ unitRelationSection, compactViewLimit = 3
Filter options displayed for list with more than 3 entries
Filter options NOT displayed for list with less than 3 entries

Relations (6) Add

Supervizor (MyCustomerGroup) (Contact)

Add/Remove column 'Contact', 'First name', ... OK Number per page 10

Contact	First name	Lastname	Date	Note	Actions
					Apply filters Clear filters
Minnie Mouse	Minnie	Mouse	12/8/14 15:04	Edit	☒
Markus Schmidt	Markus	Schmidt	12/8/14 14:51	r Edit	☒
Silke Kaufmann	Silke	Kaufmann	12/8/14 14:51	Edit	☒
Dieter Macher	Dieter	Macher	12/8/14 14:50	Edit	☒

Subordinate (MyCustomerGroup) (Contact)

Add/Remove column 'Contact', 'First name', ... OK Number per page 10

Contact	First name	Lastname	Date	Note	Actions
Hans Wille	Hans	Wille	12/8/14 14:52	Edit	☒
Daisy Duck	Daisy	Duck	12/8/14 14:52	Edit	☒

Buyer (Reseller) (Contact)

No relations available.

Fig. 55: ConSol*CM Web Client - Effect of attribute compactViewLimit

- **numberOfRelations**
  The default number of the presented relations. 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, see next code example.

JSon object for customer data format in box preview

```
{    "layout": [                ["firstname", "lastname", "lastname"],               ["email",
"email",     null]             ]}
```

For different Unit definitions in the FlexCDM use the following syntax:

Example value for customer data preview box layout

```
{''unit_definition_1": <json description>, "unit_definition_2": "component_name_used_to_display_
preview"}.
```

In Json we set the configuration for a particular unit definition. Json document can contain unit preview or just the name of a component (spring bean) used to render the preview.

unitSearchHeader

(available on *ticketCreatePage* in the company section)

Attributes:

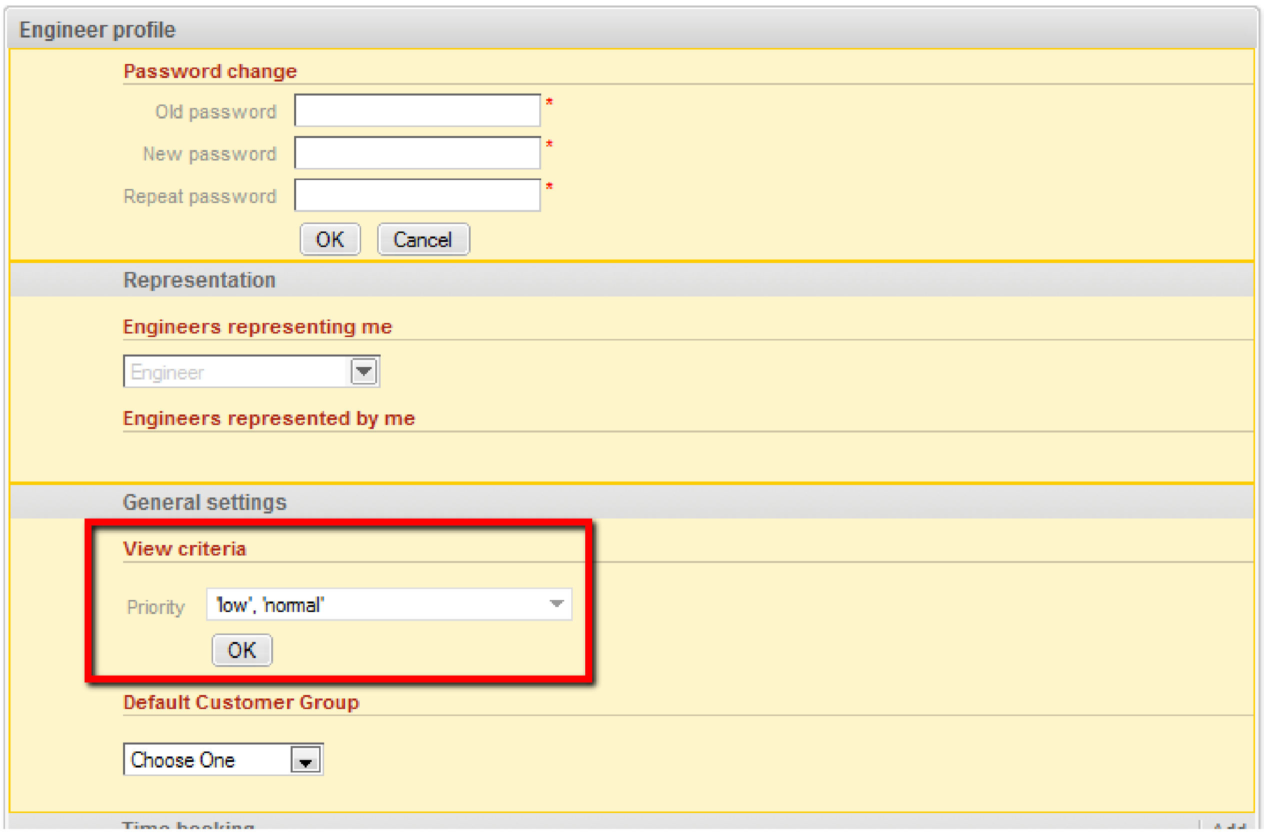
- **companyCreateLinkVisible**
  The visibility of the link for referenced company creation. (boolean)

viewDiscriminatorsSection (Type)

(available e.g. on *userProfilePage*)

Attributes:

- **visibilityFlag**
  The visibility of the *View criteria* section (section for attribute settings for dynamic views on *userProfilePage*). (boolean, default value = *true*)


Engineer profile

Password change

Old password

New password

Repeat password

Representation

Engineers representing me

Engineer

Engineers represented by me

General settings

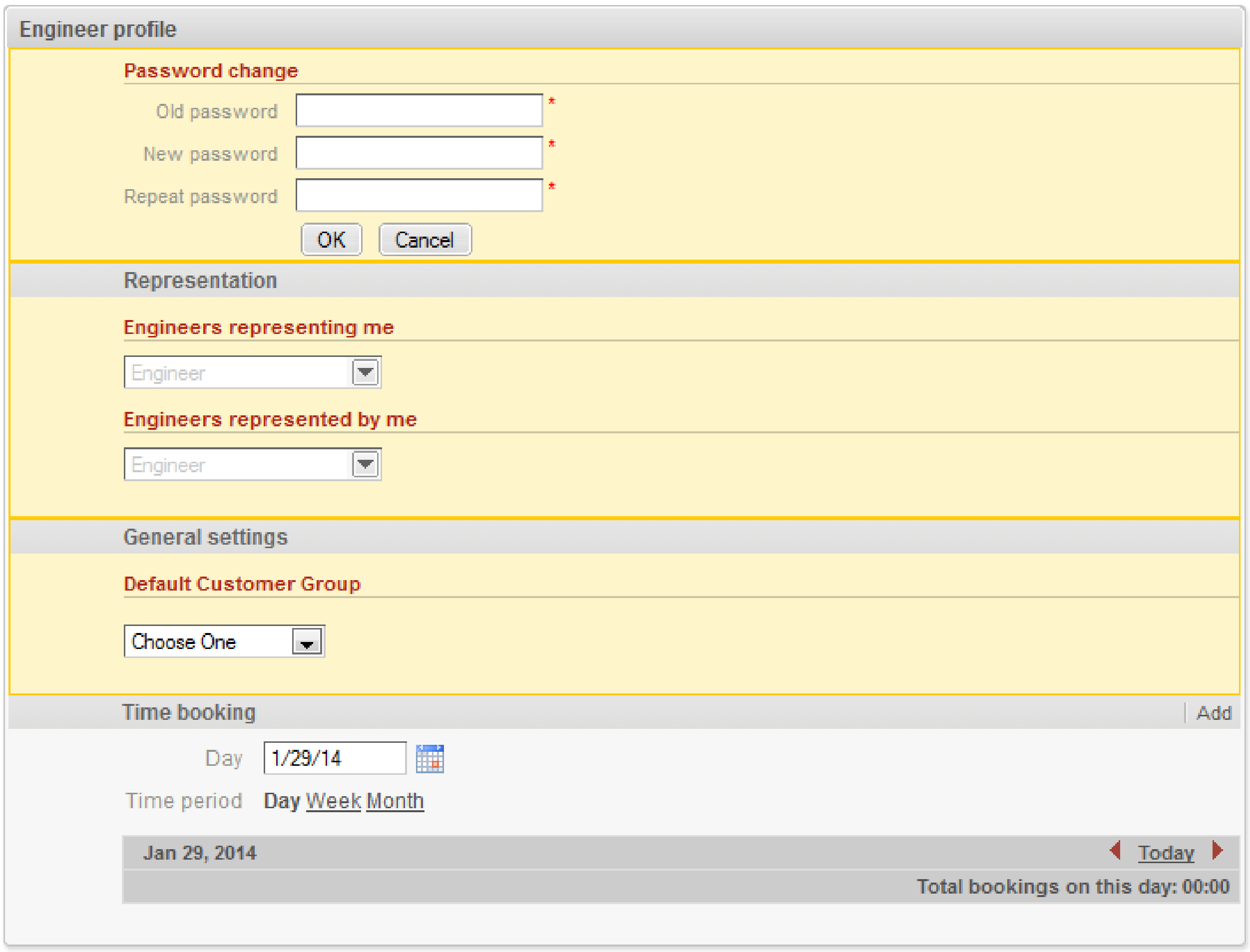
View criteria

Priority 'low', 'normal'

Default Customer Group

Choose One

Fig. 56: ConSol*CM Web Client - Visibility of View Criteria Section (visibilityFlag=true)


Engineer profile

Password change

Old password *

New password *

Repeat password *

Representation

Engineers representing me

Engineer

Engineers represented by me

Engineer

General settings

Default Customer Group

Choose One

Time booking

Day 1/29/14

Time period Day Week Month

Jan 29, 2014 Today

Total bookings on this day: 00:00

Fig. 57: ConSol*CM Web Client - Visibility of View Criteria Section (visibilityFlag=false)

welcomePage

The most important configuration on the *welcomePage* is the Page Customization for the Web Client Dashboard.

19.1.4 Order and Priorities of Page Customization

In case there are more than one values which are 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 *Global Search* field 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 definitoin (GlobalSearchField): maxHints = 5
    - => maxHints will be 7
- **Variant C:**
    - script: no value
    - scope definition (GlobalSearchField/ticketEditPage): no value
    - type definitoin (GlobalSearchField): maxHints = 5
    - => maxHints will be 5

19.2 Page Customization for the Web Client Dashboard

- Introduction
- Definition of the Overall Dashboard Layout
- Configuration of Widgets
  - Configuration Script for Widgets
  - Configuration Attributes for Widgets
    - General Attributes
      - Examples for Visibility Configuration
    - Attributes for Chart Widgets
      - Example for a Chart Widget
    - Attributes for Table Widgets
      - Example for a Table Widget
    - Example for a Composed Dashboard
- Print Functionality in the Web Client Dashboard
- 3D-Rendering for Graphics (Chart Widgets)
- Drilldown Functionality for Graphics (Chart Widgets)

19.2.1 Introduction

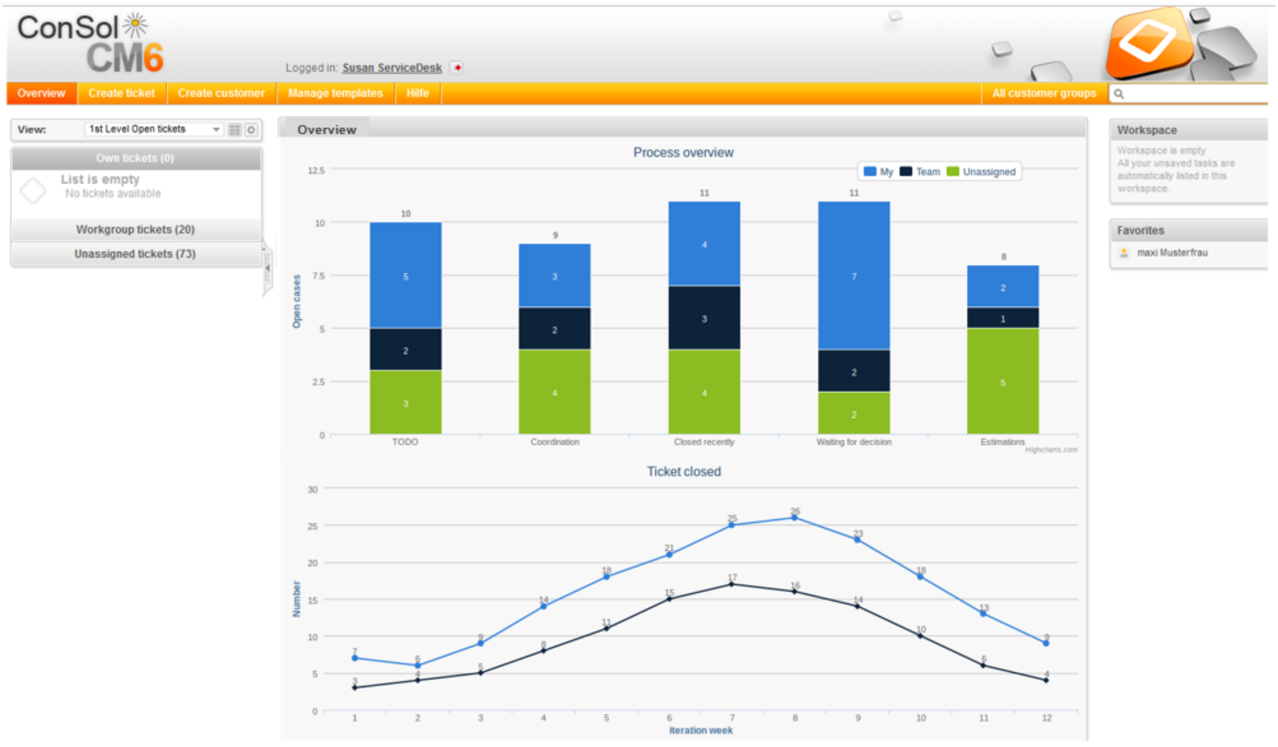


Fig. 1: Web Client Dashboard with two Widgets

The **Web Client Dashboard** is configured using *page customization*.

Log in as *admin*, open the *Overview* page and select *Enable page customization* in the main menu. Besides other attributes which are not relevant for the Dashboard and are explained on in the section Page Customization, the following (Dashboard-relevant) elements can be configured (i.e. attributes can be set). Each of the three elements is represented by a subtree in the page customization tree.

1. ***widgetsGrid / welcomePage***
   Here, the Web Client Dashboard can be switched on or off. If a correct value is entered in the layout field, the Dashboard is displayed. If the field is empty, no Dashboard is shown.
   The following configuration can be made here:
   a. the Dashboard layout, i.e. the layout of the grid on which the Dashboard is based (see section Definition of the overall Dashboard Layout), this comprises:
      i. the widgets which should be displayed
      ii. the order and organization of those widgets on the Dashboard page
2. ***chartWidget / welcomePage*** (only available if chart widgets are present)
   a. the definition/layout for all chart widgets in the chartWidget subtree
   b. each widget is represented by one node which has the name of the widget, e.g. *chartWidget / welcomePage / ticketsInView*
   c. a new chart widget is added when its name has been added in the layout value
   d. attributes can be defined for all chartWidgets on the level *chartWidget* or *chartWidget / welcomePage* or they can be configured for one chart widget individually using the values of the attributes for the chartWidget, e.g. *chartWidget / welcomePage / ticketsInView*.
3. ***tableWidget / welcomePage*** (only available if table widgets are present)
   a. the definition of all table widgets in the tableWidget subtree
   b. each widget is represented by one node which has the name of the widget, e.g. tableWidget / welcomePage / ticketsOverview
   c. a new table widget is added when its name has been added in the layout value
   d. attributes can be defined for all tableWidgets on the level *tableWidget* or *tableWidget / welcomePage* or they can be configured for one table widget individually using the values of the attributes for the tableWidget, e.g. *tableWidget / welcomePage / ticketsOverview*.

The following figure provides an example page customization tree with subtrees relevant for the Web Client Dashboard. A detailed explanation is provided in the following sections.

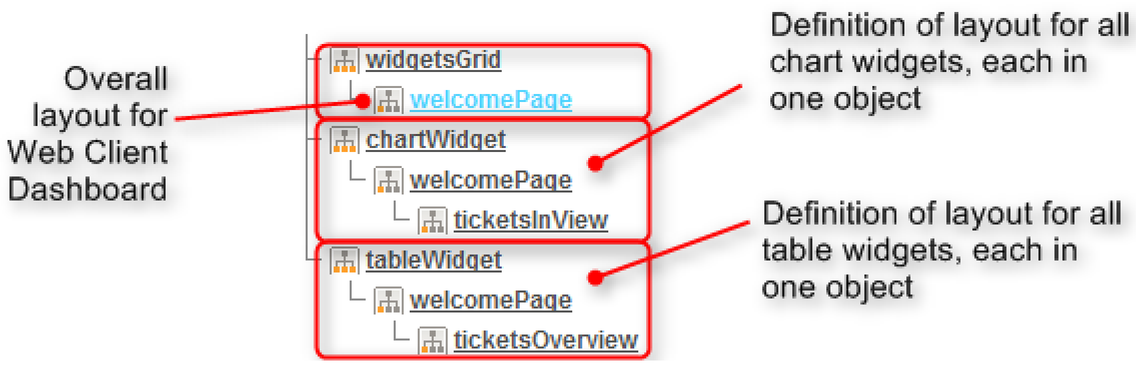

Overall layout for Web Client Dashboard
widgetsGrid
welcomePage
chartWidget
welcomePage
ticketsInView
tableWidget
welcomePage
ticketsOverview
Definition of layout for all chart widgets, each in one object
Definition of layout for all table widgets, each in one object

Fig. 2: Page Customization subtrees for Web Client Dashboard Layout

19.2.2 Definition of the Overall Dashboard Layout

The overall layout of the entire Web Client Dashboard is defined using the page customization attribute *widgetsGrid / welcomePage*.

Attributes:

- **layout**
  This defines the layout of the entire Dashboard on the welcomePage based on the following principles:
  - Each row of the Dashboard grid is represented as an array of elements: [x,y,z]. A new widget object will be added to the page customization tree automatically when it is added as value in the layout attribute, e.g. when the value *has been*[ticketsInView:Chart, ticketsInView] and the value is now [ticketsInView:Chart, ticketsInView, myTickets:Table], a new table widget named *myTickets* will appear in the page customization tree (see figure above). In the same way, widgets can be removed from the Dashboard - just remove the name and type of the widget in the layout value. After saving and reloading the page all layout changes are available in the tree for further configuration.
  - A widget is described by its name and its type, separated by a colon, i.e 'ticketsInView:Chart'. The name for a specific widget must be unique.
  - The grid starts with the upper-left corner (0,0) and it is built up row after row, e.g. a layout value with two pairs of [ ] brackets will represent two rows as shown in the figure and code shown below.
  - *null* is a reserved key word for an empty cell.
  - The type of a widget is *Chart* or *Table*. The type has to be indicated only at the first appearance of the widgets name, afterwards, it can be omitted, e.g. [ticketsInView:Chart, ticketsInView, ticketsInView].
  - The widget can occupy multiple adjacent rows and columns.
  - The Dashboard can be completely disabled in removing the value from the attribute *layout*.

The following figures show the organization of an example grid and its representation in the page customization.

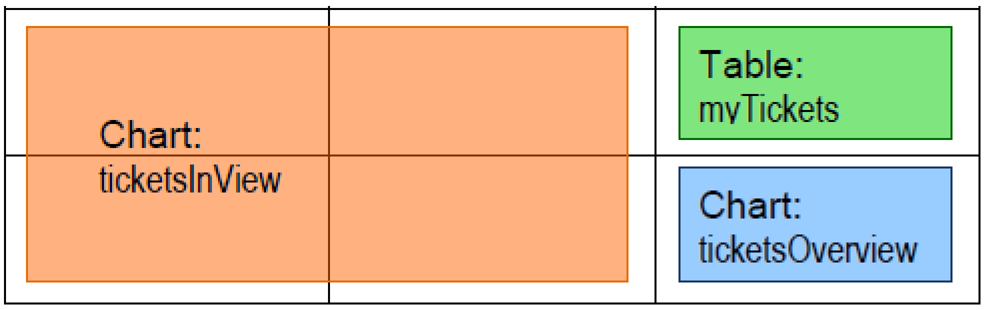


Fig. 3: Organization of an example grid for a welcomePage

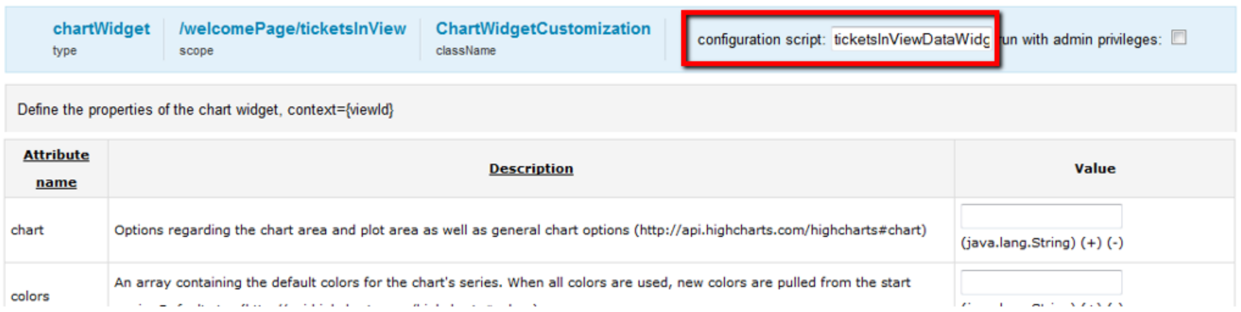
The value of the respective *layout* attribute would be:

```
[ticketsInView:Chart, ticketsInView, myTickets:Table], [ticketsInView, ticketsInView,
ticketsOverview:Chart]
```

19.2.3 Configuration of Widgets

Configuration Script for Widgets

Each chart widget and each table widget has a configuration script. This script is a Groovy script which is stored in the Admin Tool script section ad is referenced by its name. The scripts has to be of type *Page customization*. Select the widget in the PCTS and enter the script name:



chartWidget
type

/welcomePage/ticketsInView
scope

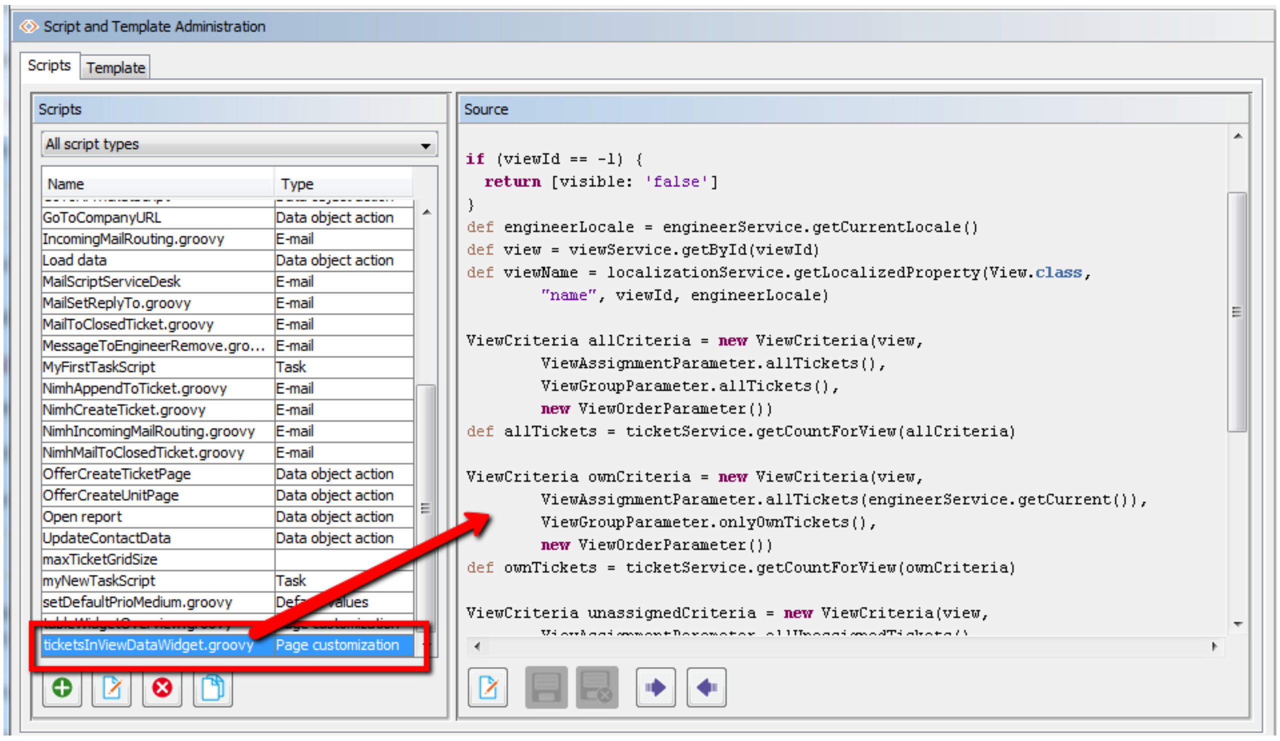
ChartWidgetCustomization
className

configuration script: ticketsInViewDataWidg
un with admin privileges: ☐

Define the properties of the chart widget, context={viewId}

Attribute name	Description	Value
chart	Options regarding the chart area and plot area as well as general chart options (http://api.highcharts.com/highcharts#chart)	(java.lang.String) (+) (-)
colors	An array containing the default colors for the chart's series. When all colors are used, new colors are pulled from the start	

Fig. 4: Script Definition for a Chart Widget



Name	Type
GoToCompanyURL	Data object action
IncomingMailRouting.groovy	E-mail
Load data	Data object action
MailScriptServiceDesk	E-mail
MailSetReplyTo.groovy	E-mail
MailToClosedTicket.groovy	E-mail
MessageToEngineerRemove.gro...	E-mail
MyFirstTaskScript	Task
NimhAppendToTicket.groovy	E-mail
NimhCreateTicket.groovy	E-mail
NimhIncomingMailRouting.groovy	E-mail
NimhMailToClosedTicket.groovy	E-mail
OfferCreateTicketPage	Data object action
OfferCreateUnitPage	Data object action
Open report	Data object action
UpdateContactData	Data object action
maxTicketGridSize	
myNewTaskScript	Task
setDefaultPrioMedium.groovy	Defa[illegible]values
ticketsInViewDataWidget.groovy	Page customization

```
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,
```

Fig. 5: Admin Tool Script for a widget of the Web Client Dashboard

The configuration script of a widget is the place where the statements are defined which retrieve the required data from the CM system and where the widget layout is defined. The execution of this groovy script is a core part of the customization. The script must return a map of variables which correspond to the defined widget properties.

An **incorrect script** does not provide a data structure which can be displayed by the Web Client Dashboard. Since the Dashboard is displayed on the Overview page which is the start page, the **Web Client will not start** in those cases! Disable or comment out the script to start the Web Client again.

The script overwrites the configuration data provided in the page customization The values are *not* merged! The script thus will overrides any widget attribute value set in the customization, so please make sure that the desired property is not set within the script, if you want to use the page customization for property setting.

A **script** which is associated with a widget is usually executed with user (= engineer) permissions, e.g. the standard chart widget shows a graphical representation of the selected view. However, sometimes values have to be used which are not available in the engineer context, e.g. escalated tickets (of all engineers) in a certain queue. In order to execute a script with admin permissions, select the check box *run with admin privileges*. Please keep in mind that the results of the Java or Groovy methods which retrieve the data will vary depending on the context. For example, the method *ticketService.getAll()* will return only tickets for which the the current engineer has at least read permissions, but will return absolutely all tickets in the system when executed as admin.

The **chart** representation in the Web Client Dashboard is based on the Highcharts library. Thus, for chart widgets, the Admin Tool script has to return the attributes which should be set, as a HashMap (see return statement in the code example above which uses the attributes *series*, *visible*, *chart*, *title*, *tooltip*, and *localization*). A detailed explanation of all attributes and the respective hyperlinks are provided in the section Attributes for Chart Widgets.

The **table** representation in the Web Client Dashboard is based on the Datatables library. Thus, for table widgets, the Admin Tool script has to return the attributes which should be set, as a HashMap. Please see section Attributes for Table Widgets.

Very complex scripts can decrease system performance!!!

The following example shows the script ticketsInViewDataWidget.groovy which is provided with a standard ConSol*CM distribution.

Standard ConSol*CM Table Widget Script ticketsInViewDataWidget.groovy

```
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'}"];
```

The following chart is defined by the script above. For a detailed explanation, please refer to the section Example for a Chart Widget.

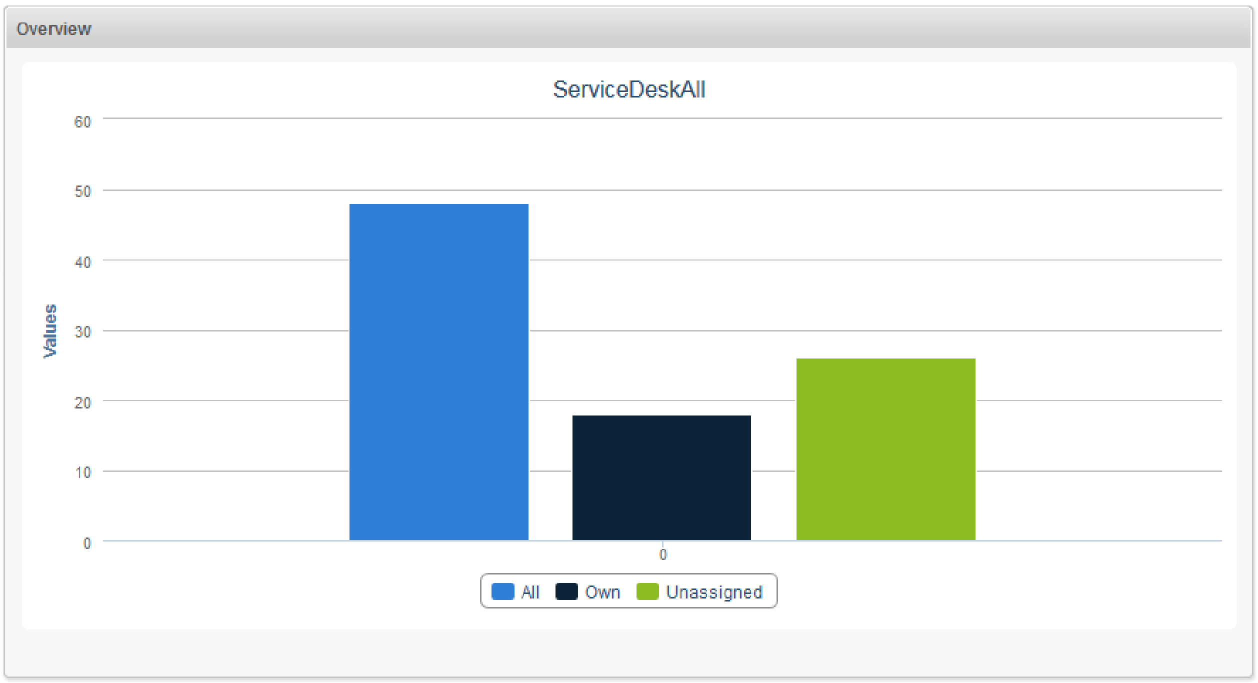


Fig. 6: Example Chart Widget

Configuration Attributes for Widgets

All definitions which can be used for a widget can be set using the attributes and values of the page customization. There are three types of attributes:

- general attributes (available for each widget type)
- attributes for chart widgets
- attributes for table widgets

Please keep in mind that a attribute which is set within the Admin Tool script of a widget always overwrites the respective attribute which has been set as attribute!

Example: For the chart widget *ticketsInView*, the attribute *visibility* has been set to *true*. The Admin Tool script which is associated with the widget (*ticketsInViewDataWidget.groovy*) contains the statement

```
return [visible: 'false']
```

In this case the widget will not be displayed!

General Attributes

Those attributes are valid for all widgets and can be set in two locations of the page customization tree

- widgetsGrid
- widgetsGrid/welcomePage

Attributes:

- **layout**
  see the section about layout
- **refreshOnViewChange**
  Indicates whether the Dashboard should be refreshed when the engineer changes the view, i.e. uses the drop-down view list to select another view for the ticket list, default value is *true*

Examples for Visibility Configuration

Example 1: The chart should not be displayed.

Visibility switched off (set in Admin Tool script)

```
return [visible: 'false']
```

Example 2: The chart should only be displayed if the selected view is *service_customer and the* engineer has *the consultant* role.

Visibility depending on engineer role (set within Admin Tool script)

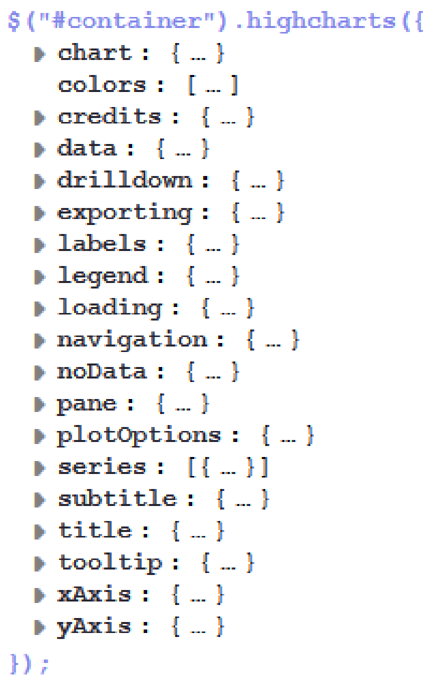
```
view = viewService.getById(view_id) // view_id is passed in context
if (!view.getName().equals("service_customer"))
{
  return {"visible": false}
}
def role =  roleService.getById('consultant');
def engineer = engineerService.getById(engineer_id);
if
(!getRolesForEngineer(engineer).contains(role)) {
  return {"visible": false}
}
```

Attributes for Chart Widgets

Chart widgets use the Highcharts library. All attributes are JSON objects.

The attributes which can be set comprise

- general attributes like visibility
- the basic configuration options of the Highcharts library. Their values ...
  - can be set using the page customization attributes
  - can be set using the Admin Tool script which is associated with the chart widget, see section above. The attributes have to be returned as a HashMap.
  - can be left empty



```
$("#container").highcharts({
  ▶ chart : { … }
    colors : [ … ]
  ▶ credits : { … }
  ▶ data : { … }
  ▶ drilldown : { … }
  ▶ exporting : { … }
  ▶ labels : { … }
  ▶ legend : { … }
  ▶ loading : { … }
  ▶ navigation : { … }
  ▶ noData : { … }
  ▶ pane : { … }
  ▶ plotOptions : { … }
  ▶ series : [{ … }]
  ▶ subtitle : { … }
  ▶ title : { … }
  ▶ tooltip : { … }
  ▶ xAxis : { … }
  ▶ yAxis : { … }
});
```

Fig. 7: Highcharts configuration options

General attributes:

- **localization**
  localized values, i.e -> "de: {subject:'Thema', yes:'Ja'}, en: {subject:'Subject', yes:'Yes'}"
- **visible**
  defines if the widget is displayed, *true* or *false*.

Highchart-specific attributes:

- **chart**
  Options regarding the chart area and plot area as well as general chart options (http://api.highcharts.com/highcharts#chart). Example:

Example for chart object

```
chart =  "type:'column', pltShadow:false, backgroundColor:'#4dc245', height:
300";"items: [{html:'sometext', style: { left: '100px'; }}]"
```

- **colors**
  An array containing the default colors for the chart's series. When all colors are used, new colors are pulled from the start again. Defaults to: (http://api.highcharts.com/highcharts#colors)
- **credits**
  Highchart by default puts a credits label in the lower right corner of the chart. This can be changed using these options (http://api.highcharts.com/highcharts#credits)
- **drilldown**
  Options for drill down, the concept of inspecting increasingly high resolution data through clicking on chart items like columns or pie slices (http://api.highcharts.com/highcharts#drilldown)

- **exporting**
  Options for the exporting module (http://api.highcharts.com/highcharts#exporting)
- **global**
  Global options that don't apply to each chart (http://api.highcharts.com/highcharts#global). Can only be set for a type, i.e. for chartWidget or tableWidget, not for scopes or single widgets!
- **labels**
  HTML labels that can be positioned anywhere in the chart area (http://api.highcharts.com /highcharts#labels).

Example for lables object

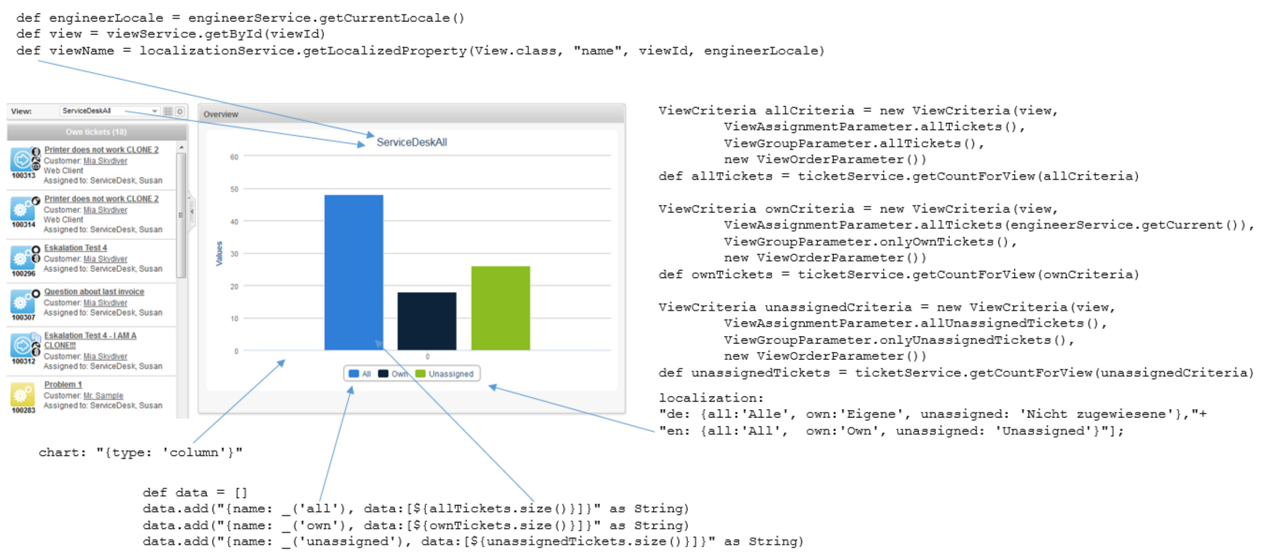
```
labels = "items: [{html:'sometext', style: { left: '100px'; }}]"
```

- **lang**
  Language object. The language object is global and it can't be set on each chart initiation (http://api.highcharts.com/highcharts#lang). Can only be set for a type, i.e. for chartWidget or tableWidget, not for scopes or single widgets!
- **legend**
  The legend is a box containing a symbol and name for each series item or point item in the chart (http://api.highcharts.com/highcharts#legend)
- **loading**
  The loading options control the appearance of the loading screen that covers the plot area on chart operations (http://api.highcharts.com/highcharts#loading)
- **localization**
  localized values, i.e -> "de: {subject:'Thema', yes:'Ja'}, en: {subject:'Subject', yes:'Yes'}"
- **navigation**
  A collection of options for buttons and menus appearing in the exporting module (http://api.highcharts.com/highcharts#navigation)
- **noData**
  Options for displaying a message like "No data to display" (http://api.highcharts.com /highcharts#noData)
- **pane**
  Applies only to polar charts and angular gauges. This configuration object holds general options for the combined X and Y axes set (http://api.highcharts.com/highcharts#pane)
- **plotOptions**
  The plotOptions is a wrapper object for config objects for each series type (http://api.highcharts.com /highcharts#plotOptions)
- **series**
  The actual series to append to the chart (http://api.highcharts.com/highcharts#series)
- **subtitle**
  The chart's subtitle (http://api.highcharts.com/highcharts#subtitle)
- **title**
  The chart's main title (http://api.highcharts.com/highcharts#title)
- **tooltip**
  Options for the tool tip that appears when the user hovers over a series or point (http://api.highcharts.com/highcharts#tooltip)

- **visible**
  Indicates whether the widget is shown
- **xAxis**
  The X axis or category axis (http://api.highcharts.com/highcharts#xAxis)
- **yAxis**
  The Y axis or value axis (http://api.highcharts.com/highcharts#yAxis)

Example for a Chart Widget

The following example shows the widget *TicketsInView* and explains the logic of the associated Admin Tool script *ticketsInViewDataWidget.groovy*. For the entire script, please see the code block above. Here, the lines of code are set in relation to the GUI elements which they configure.



```
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)

localization:
"de: {all:'Alle', own:'Eigene', unassigned: 'Nicht zugewiesene'},"+
"en: {all:'All',  own:'Own', unassigned: 'Unassigned'}"];
```

```
chart: "{type: 'column'}"
```

```
def data = []
data.add("{name: _('all'), data:[${allTickets.size()}]}" as String)
data.add("{name: _('own'), data:[${ownTickets.size()}]}" as String)
data.add("{name: _('unassigned'), data:[${unassignedTickets.size()}]}" as String)
```

View:
ServiceDeskAll
Own tickets (18)
Printer does not work CLONE 2
Customer: Mia Skydiver
Web Client
Assigned to: ServiceDesk, Susan
100313
Printer does not work CLONE 2
Customer: Mia Skydiver
Web Client
Assigned to: ServiceDesk, Susan
100314
Eskalation Test 4
Customer: Mia Skydiver
Assigned to: ServiceDesk, Susan
100296
Question about last invoice
Customer: Mia Skydiver
Assigned to: ServiceDesk, Susan
100307
Eskalation Test 4 - I AM A CLONE!!!
Customer: Mia Skydiver
Assigned to: ServiceDesk, Susan
100312
Problem 1
Customer: Mr. Sample
Assigned to: ServiceDesk, Susan
100283
Overview
ServiceDeskAll
Values
60
50
40
30
20
10
0
0
All
Own
Unassigned

Fig. 8: Chart Widget Example with Script Code

Attributes for Table Widgets

Table widgets use the Datatables library.

The attributes can be set in the page customization or they can be set in the associated Admin Tool script. Please keep in mind that the script parameters always overwrite the respective attributes.

Attributes comprise

- general attributes
- Datatables-specific attributes

General attributes:

- **localization**
  localized values, i.e -> "de: {subject:'Thema', yes:'Ja'}, en: {subject:'Subject', yes:'Yes'}"
- **visible**
  defines if the widget is displayed, *true* or *false*.

Datatables-specific attributes:

- **columns**
  Options which you can apply to the columns objects (http://datatables.net/reference/option/#Columns)
- **data**
  Data (http://datatables.net/reference/option/#Data)
- **localization**
  localized values, i.e -> "de: {subject:'Thema', yes:'Ja'}, en: {subject:'Subject', yes:'Yes'}"
- **options**
  Options (http://datatables.net/reference/option/)
- **visible**
  Indicates whether the widget is shown

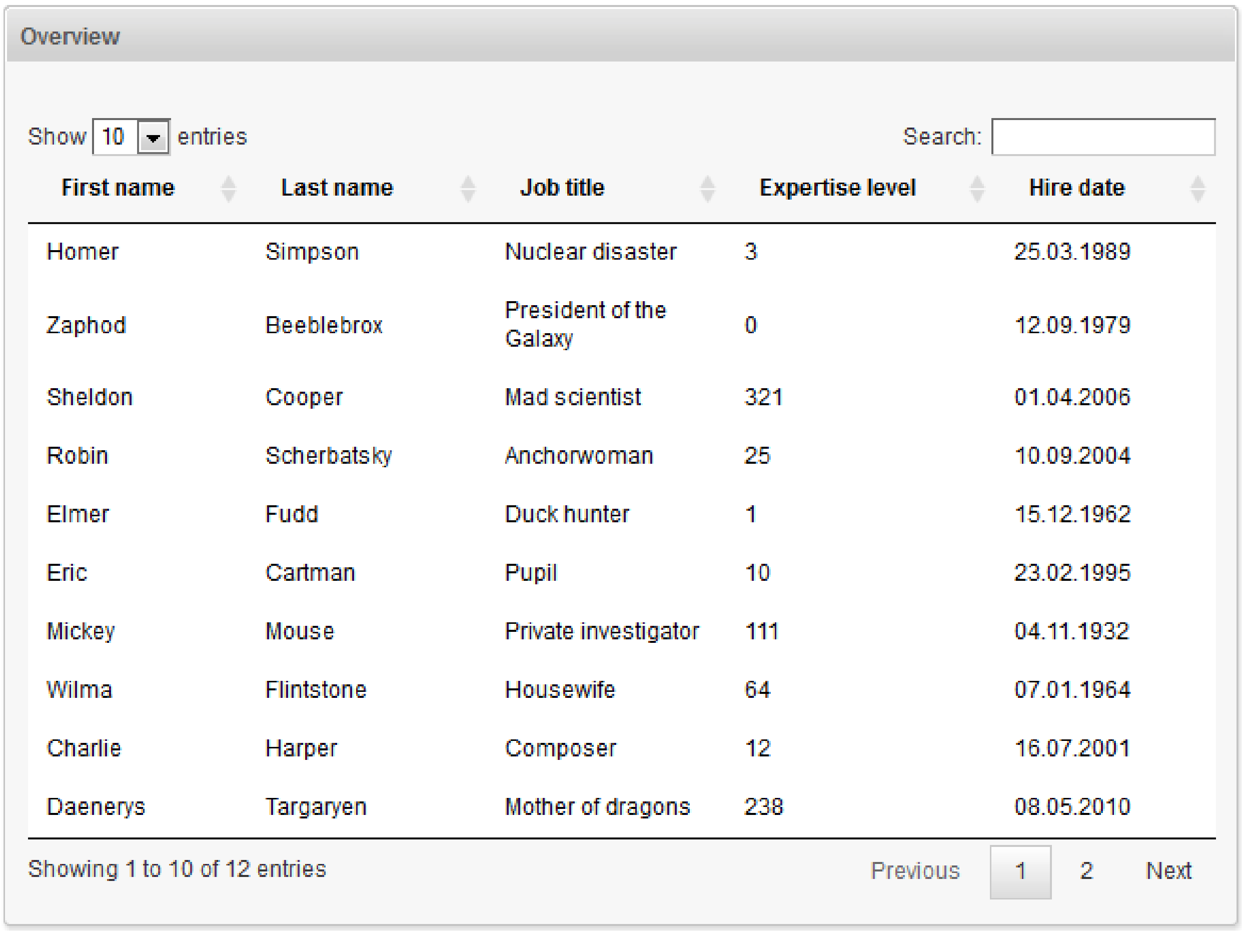
Example for a Table Widget

The following demonstrates the basic principle of the implementation of a table widget based on the Datatables library.

Example Adem Tool script for a table widget

```
// provide some dummy data for display
def rawdata = [
[firstname:'Homer'   , lastname:'Simpson'    , title:'Nuclear disaster'       , level:'3'  ,
hired:'25.03.1989'],
[firstname:'Zaphod'  , lastname:'Beeblebrox' , title:'President of the Galaxy', level:'0'  ,
hired:'12.09.1979'],
[firstname:'Sheldon' , lastname:'Cooper'     , title:'Mad scientist'          , level:'321',
hired:'01.04.2006'],
[firstname:'Robin'   , lastname:'Scherbatsky', title:'Anchorwoman'            , level:'25' ,
hired:'10.09.2004'],
[firstname:'Elmer'   , lastname:'Fudd'       , title:'Duck hunter'            , level:'1'  ,
hired:'15.12.1962'],
[firstname:'Eric'    , lastname:'Cartman'    , title:'Pupil'                  , level:'10' ,
hired:'23.02.1995'],
[firstname:'Mickey'  , lastname:'Mouse'      , title:'Private investigator'   , level:'111',
hired:'04.11.1932'],
[firstname:'Wilma'   , lastname:'Flintstone' , title:'Housewife'              , level:'64' ,
hired:'07.01.1964'],
[firstname:'Charlie' , lastname:'Harper'     , title:'Composer'               , level:'12' ,
hired:'16.07.2001'],
[firstname:'Daenerys', lastname:'Targaryen'  , title:'Mother of dragons'      , level:'238',
hired:'08.05.2010'],
[firstname:'Lara'    , lastname:'Croft'      , title:'Tomb Raider'            , level:'239',
hired:'10.12.1991'],
[firstname:'Henry'   , lastname:'Jones'      , title:'Archeologist'           , level:'109',
hired:'08.06.1942']
]
// prepare the data for display
def tabledata = []
rawdata.each { element ->
    tabledata.add("""
      {'firstname': '${element['firstname']}',
       'lastname' : '${element['lastname']}' ,
       'jobtitle' : '${element['title']}'    ,
       'expertise': '${element['level']}'    ,
        'hiredate' : '${element['hired']}'    }
    """)
}
// return the table information including the data
return [
"columns": """[
    {title: 'First name'     , data: 'firstname'},
    {title: 'Last name'      , data: 'lastname' },
    {title: 'Job title'      , data: 'jobtitle' },
    {title: 'Expertise level', data: 'expertise'},
    {title: 'Hire date'      , data: 'hiredate' }
    ]""",
"options": """{
    'order': []
    }""",
"data": "[${tabledata.join(",")}]" as String
]
```

In the Web Client, the table is displayed as follows (all other widgets has been set to invisible).


Overview

Show 10 entries Search:

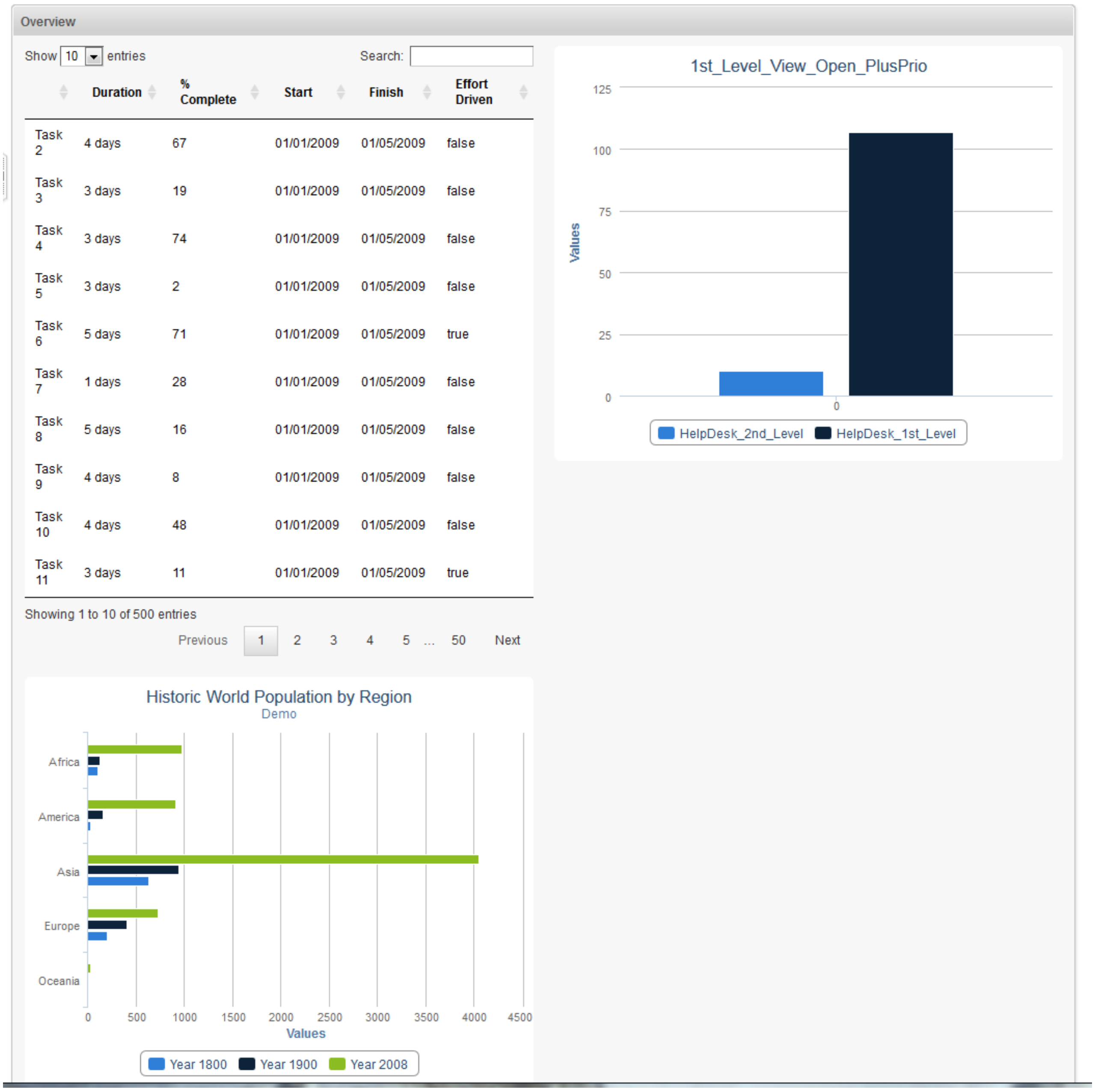
First name	Last name	Job title	Expertise level	Hire date
Homer	Simpson	Nuclear disaster	3	25.03.1989
Zaphod	Beeblebrox	President of the Galaxy	0	12.09.1979
Sheldon	Cooper	Mad scientist	321	01.04.2006
Robin	Scherbatsky	Anchorwoman	25	10.09.2004
Elmer	Fudd	Duck hunter	1	15.12.1962
Eric	Cartman	Pupil	10	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

Fig. 9: Web Client Dashboard with one Example Table Widget

Example for a Composed Dashboard

The Dashboards looks like this:



	Duration	% Complete	Start	Finish	Effort Driven
Task 2	4 days	67	01/01/2009	01/05/2009	false
Task 3	3 days	19	01/01/2009	01/05/2009	false
Task 4	3 days	74	01/01/2009	01/05/2009	false
Task 5	3 days	2	01/01/2009	01/05/2009	false
Task 6	5 days	71	01/01/2009	01/05/2009	true
Task 7	1 days	28	01/01/2009	01/05/2009	false
Task 8	5 days	16	01/01/2009	01/05/2009	false
Task 9	4 days	8	01/01/2009	01/05/2009	false
Task 10	4 days	48	01/01/2009	01/05/2009	false
Task 11	3 days	11	01/01/2009	01/05/2009	true

Showing 1 to 10 of 500 entries

Fig. 10: Example of Composed Dashboard

The page customization attribute *widgetsGrid / welcomePage / layout* is set as follows:

Layout attribute for Composed Dashboard

```
[process:Table, escalation:Chart], [process:Table, null], [bar:Chart, null]
```

The page customization attributes subtree looks like this:

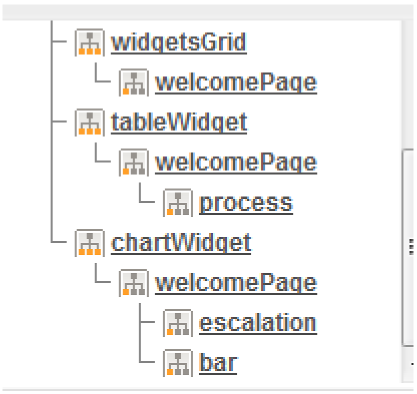
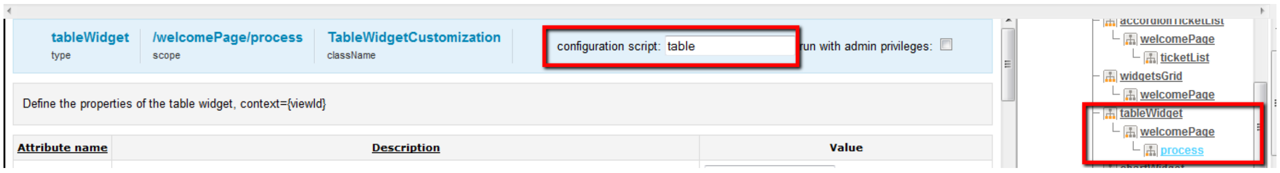


Fig. 11: Example Page Customization Attributes Subtree

The **table widget** (named *process*) has an associated Admin Tool script:



tableWidget
type

/welcomePage/process
scope

TableWidgetCustomization
className

configuration script: table

run with admin privileges: ☐

Define the properties of the table widget, context={viewId}

Attribute name	Description	Value

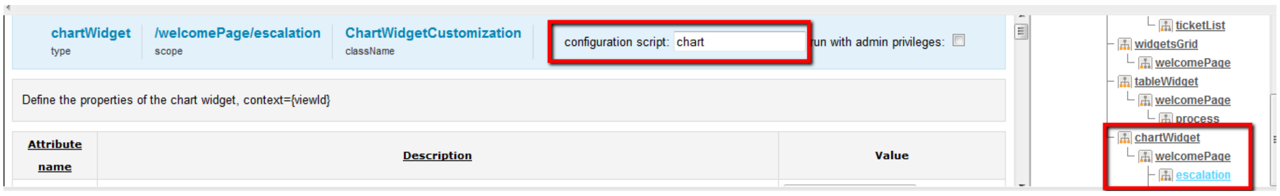
Fig. 12: Configuration of Exaple Table Widget

The Admin Tool script (named *table*) for the table widget (named *process*):

Example Admin Tool script associated with Table Widget

```
def data = []
(1..500).each { i ->
  data.add("""
  {'title': 'Task ${i+1}',
  'duration': '${Math.round(Math.random() * 5) + 1} days',
  'percentComplete': '${Math.round(Math.random() * 100)}',
  'start': '01/01/2009',
  'finish': '01/05/2009',
  'effortDriven': '${i % 5 == 0}'}
  """)
}
return [
"columns": """[
    {data: 'title'},
    {title: 'Duration', data: 'duration'},
    {title: '% Complete', data: 'percentComplete'},
    {title: 'Start', data: 'start'},
    {title: 'Finish', data: 'finish'},
    {title: 'Effort Driven', data: 'effortDriven'}]""",
"options": """{
    'order': []
    }""",
"data": "[${data.join(",")}]" as String
];
```

The first chart widget (named *escalation*) also has an associated Admin Tool script (named *chart*):



chartWidget
type

/welcomePage/escalation
scope

ChartWidgetCustomization
className

configuration script: chart

run with admin privileges: ☐

Define the properties of the chart widget, context={viewId}

Attribute name	Description	Value

Fig. 13: Configuration of Example Chart Widget

The Admin Tool script for the first chart widget (upper right, named *escalation*):

Example Admin Tool script associated with Chart Widget

```
import com.consol.cmas.common.model.ticket.*;
import java.util.*;
import java.util.Map.Entry;
if (viewId == -1) {
  return [visible: 'false']
}
def view = viewService.getById(viewId);
TicketCriteria crt = new TicketCriteria();
crt.setStatus(view.name.toLowerCase() =~ 'close' ?
    TicketCriteria.Status.CLOSED : TicketCriteria.Status.OPEN);
tickets = ticketService.getByCriteria(crt);
def queues = [:]
tickets.each { ticket ->
  def qname = ticket.queue.name
  if (queues.containsKey(qname)) {
    queues[qname] = queues[qname] + 1
  } else {
    queues[qname] = 1
  }
}
def data = []
queues.each { entry ->
  data.add("{name: '$entry.key', data:[$entry.value]}")
}
return [series: "[${data.join(',')}]" as String, visible: 'true',
    chart: "{type: 'column'}", title: "{text: '${view.name}'}" as String];
```

The second chart widget (lower line, named *bar*) does not have an associated script. It is configured entirely by attributes using the page customization. The following values have been set:

attribute	value
chart	type:'bar'
credits	enabled: false
series	[{ name: 'Year 1800', data: [107, 31, 635, 203, 2] }, {name: 'Year 1900', data: [133, 156, 947, 408, 6]}, {name: 'Year 2008', data: [973, 914, 4054, 732, 34] }]
subtitle	text:'Demo'

attribute	value
title	text: 'Historic World Population by Region'
visible	true
xAxis	categories: ['Africa', 'America', 'Asia', 'Europe', 'Oceania'], title: {text: null}
yAxis	min: 0, title: {text: 'Population (millions)', align: 'high'}, labels: { overflow: 'justify'}

19.2.4 Print Functionality in the Web Client Dashboard

Starting with version 6.9.4.2, ConSol*CM offers print functionality for Chart Widgets in the Web Client Dashboard. The print button opens the print dialog of the operation system.

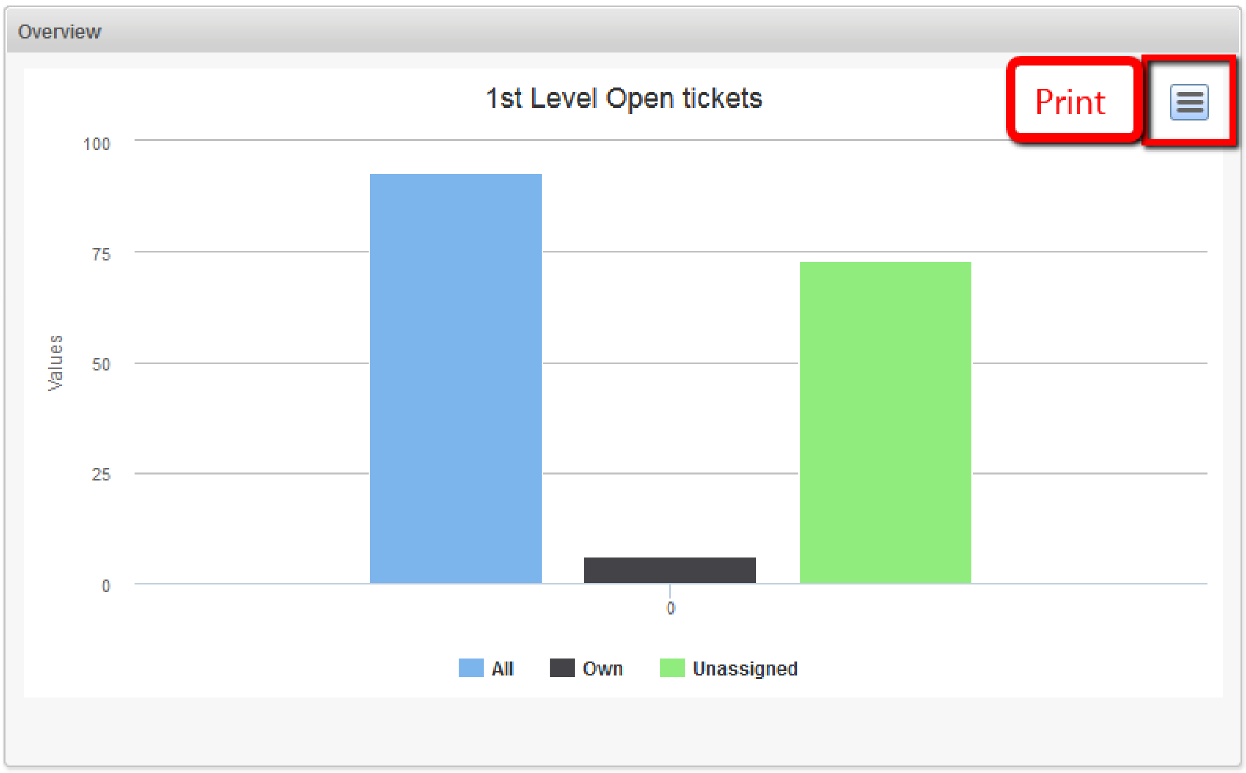


Fig. 14: Print button in Web Client dashboard

In order to disable the print functionality ,i.e. to hide the button, set the page customization attribute *exporting* to the value *enabled:false*.

19.2.5 3D-Rendering for Graphics (Chart Widgets)

You can use 3D-rendering for chart widgets. The following library is used: http://api.highcharts.com/highcharts#chart.options3d.

For more information on the 3D implementation and concepts of the solution see http://www.highcharts.com/docs/chart-concepts/3d-charts.

Caution!
Usage of this kind of rendering puts a rather heavy performance load on the browser. Please be sure that the client environments are capable of displaying these 3D charts when implementing them!

An example script for showing the default chart in a 3D view is listed below. It sets the 3D options in the return value of the script. The additional options in comparison to the standard script are highlighted.

```
import com.consol.cmas.common.model.ticket.*;
import com.consol.cmas.common.model.ticket.view.*;
import java.util.*;
import java.util.Map.Entry;
if (viewId == -1) {
    return [visible: 'false']
}
def engineerLocale = engineerService.getCurrentLocale()
def view = viewService.getById(viewId)
def viewName = localizationService.getLocalizedProperty(View.class, "name", viewId,
engineerLocale)
ViewCriteria allCriteria = new ViewCriteria(view,
         ViewAssignmentParameter.allTickets(),
         ViewGroupParameter.allTickets(),
         new ViewOrderParameter())
def allTickets = ticketService.getIdsByView(allCriteria)
ViewCriteria ownCriteria = new ViewCriteria(view,
         ViewAssignmentParameter.allTickets(engineerService.getCurrent()),
         ViewGroupParameter.onlyOwnTickets(),
         new ViewOrderParameter())
def ownTickets = ticketService.getIdsByView(ownCriteria)
ViewCriteria unassignedCriteria = new ViewCriteria(view,
         ViewAssignmentParameter.allUnassignedTickets(),
         ViewGroupParameter.onlyUnassignedTickets(),
         new ViewOrderParameter())
def unassignedTickets = ticketService.getIdsByView(unassignedCriteria)
def data = []
data.add("{name: _('all'), data:[${allTickets.size()}]}" as String)
data.add("{name: _('own'), data:[${ownTickets.size()}]}" as String)
data.add("{name: _('unassigned'), data:[${unassignedTickets.size()}]}" as String)
return [series: "[${data.join(',')}]" as String,
     visible: 'true',
    chart: "{type: 'column',
              options3d: {enabled: 'true', alpha: '15', beta: '15', depth: '50',
                          viewDistance: '25'}}",
     plotOptions: "{column: {depth: '25'}}",
     title: "{text: '${viewName}'}" as String,
    tooltip:"{headerFormat:''}" ,
    localization: "de: {all:'Alle', own:'Eigene', unassigned:'Nicht zugewiesene'},"
                + "en: {all:'All',  own:'Own', unassigned: 'Unassigned'}"];
```

19.2.6 Drilldown Functionality for Graphics (Chart Widgets)

ConSol*CM offers basic drilldown functionality for charts in the dashboard. The page customization attribute *drilldown* can be used for general settings. The following library is used: http://api.highcharts.com /highcharts#drilldown

However, to be reasonably used by this functionality the data script should be extended, too. The data in the return script needs to be extended with the data shown in the drilldown. These additional data must be referenced with the data they extend. For a description of the concepts see the highcharts documentation: http://www.highcharts.com/docs/chart-concepts/drilldown

The effect is that a second level of data can be shown in the same chart when clicking on a subset. In the left screenshot the columns are clickable and the column name labels are links. After clicking on either a detail view of this subset is shown, illustrated by the right screenshot.

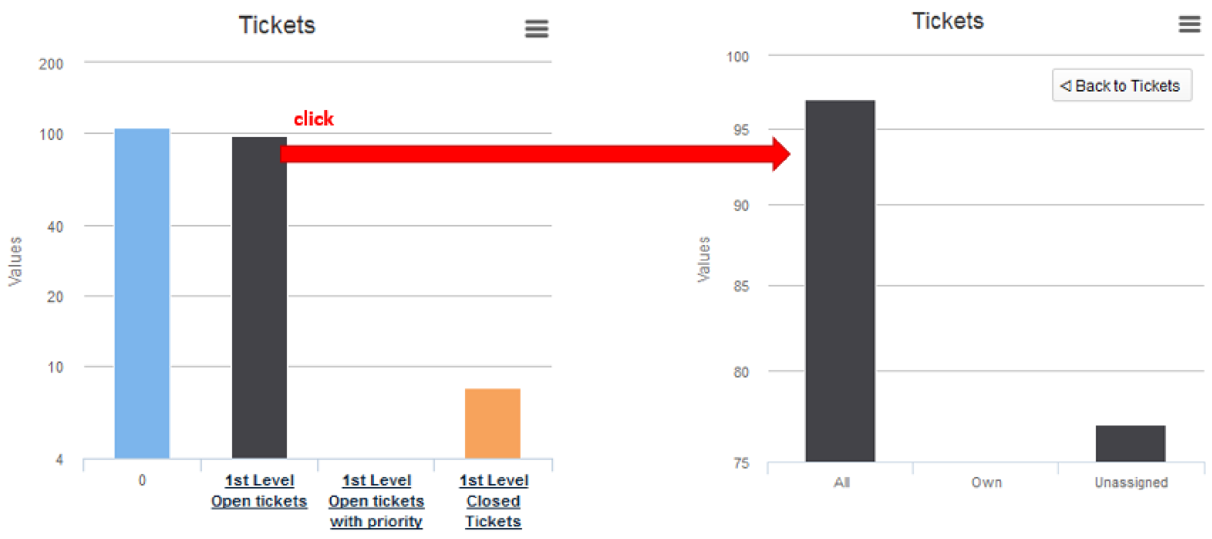
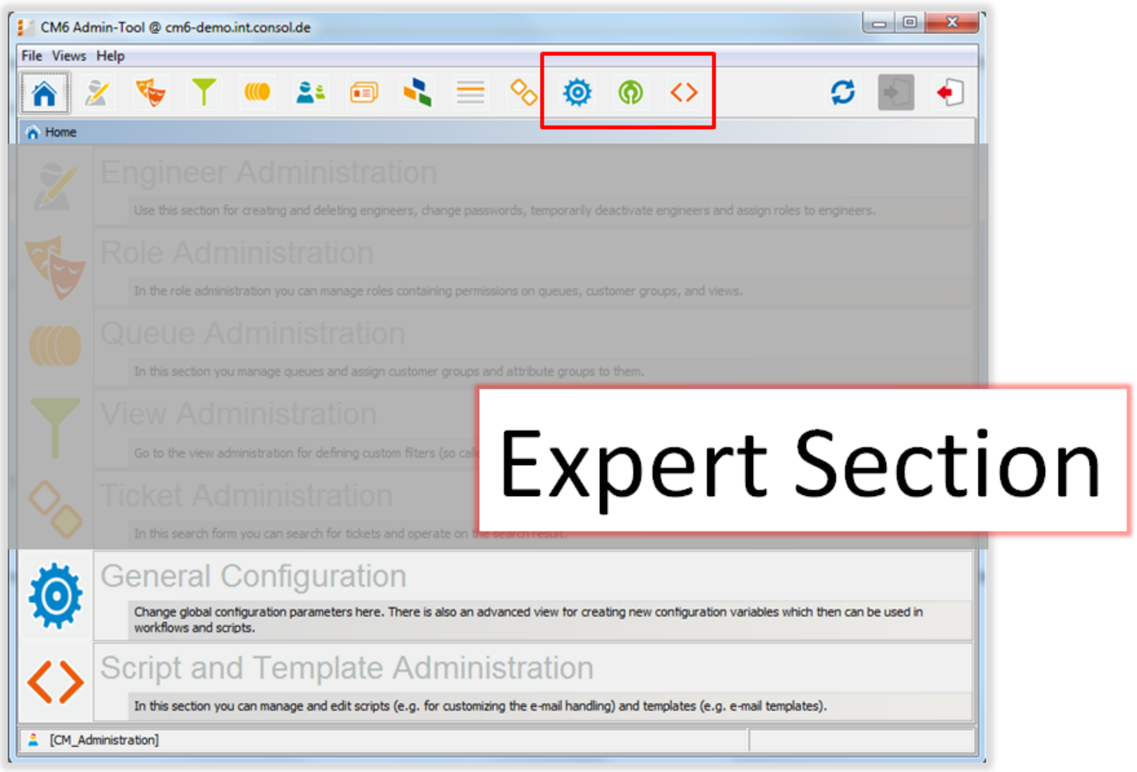


Fig. 15: Drilldown Functionality in Chart Widgets

The color of the columns in the detail view is the color of the selected subset in the main view. The detail view also displays a back button on the upper left side of the chart. The following data script dynamically provides the data for this drilldown.

```
import com.consol.cmas.common.model.ticket.*;
import com.consol.cmas.common.model.ticket.view.*;
import java.util.*;
import java.util.Map.Entry;
if (viewId == -1) {
    return [visible: 'false']
}
def engineerLocale = engineerService.getCurrentLocale()
def views = []
def seriesdata = []
def drilldownseries = []
def allTicketsCounter = 0
views = viewService.getByEngineer(engineerService.getCurrent())
for (view in views){
    def viewName = localizationService.getLocalizedProperty(View.class,
           "name",
           view.getId(),
           engineerLocale)
    ViewCriteria allCriteria = new ViewCriteria(view,
           ViewAssignmentParameter.allTickets(),
          ViewGroupParameter.allTickets(),
           new ViewOrderParameter())
     def allTickets = ticketService.getIdsByView(allCriteria)
    ViewCriteria ownCriteria = new ViewCriteria(view,
           ViewAssignmentParameter.allTickets(engineerService.getCurrent()),
           ViewGroupParameter.onlyOwnTickets(),
           new ViewOrderParameter())
    def ownTickets = ticketService.getIdsByView(ownCriteria)
    ViewCriteria unassignedCriteria = new ViewCriteria(view,
           ViewAssignmentParameter.allUnassignedTickets(),
           ViewGroupParameter.onlyUnassignedTickets(),
           new ViewOrderParameter())
    def unassignedTickets = ticketService.getIdsByView(unassignedCriteria)
    seriesdata.add("{name: '${viewName}',
           y: ${allTickets.size()},
           drilldown: '${view.getName()}'}")
    def data = []
    data.add("['All', ${allTickets.size()}]")
    data.add("['Own', ${ownTickets.size()}]")
    data.add("['Unassigned', ${unassignedTickets.size()}]")
    drilldownseries.add("{id: '${view.getName()}',
           data:[${data.join(',')}]}" as String)
    allTicketsCounter += allTickets.size()
}
return [series: "[{name: 'Tickets', colorByPoint: true,
          data: [${allTicketsCounter}, ${seriesdata.join(',')}]}]" as String,
        drilldown: "{series: [${drilldownseries.join(',')}]}" as String,
        visible: 'true',
        chart: "{type: 'column'}",
        title: "{text: 'Tickets'}",
        xAxis: "{type: 'category'}",
        yAxis: "{type: 'linear'}",
        legend: "{enabled: false}"
];
```

20 Expert Section



Engineer Administration

Use this section for creating and deleting engineers, change passwords, temporarily deactivate engineers and assign roles to engineers.

Role Administration

In the role administration you can manage roles containing permissions on queues, customer groups, and views.

Queue Administration

In this section you manage queues and assign customer groups and attribute groups to them.

View Administration

Go to the view administration for defining custom filters (so cal

Ticket Administration

In this search form you can search for tickets and operate on the search result.

General Configuration

Change global configuration parameters here. There is also an advanced view for creating new configuration variables which then can be used in workflows and scripts.

Script and Template Administration

In this section you can manage and edit scripts (e.g. for customizing the e-mail handling) and templates (e.g. e-mail templates).

[CM_Administration]

Only experienced system administrators and ConSol*CM consultants are allowed to perform the operations explained in the following sections!!!

When wrong configuration parameters are applied, the system will not work properly or will not work at all!!!

21 Ticket Administration

- Introduction to Ticket Administration
- Ticket Administration Using the Admin Tool
    - Delete or Reopen Tickets
    - Switching off the Delete Functionality Using a System Property
    - Search Tickets
- Related Topics

21.1 Introduction to Ticket Administration

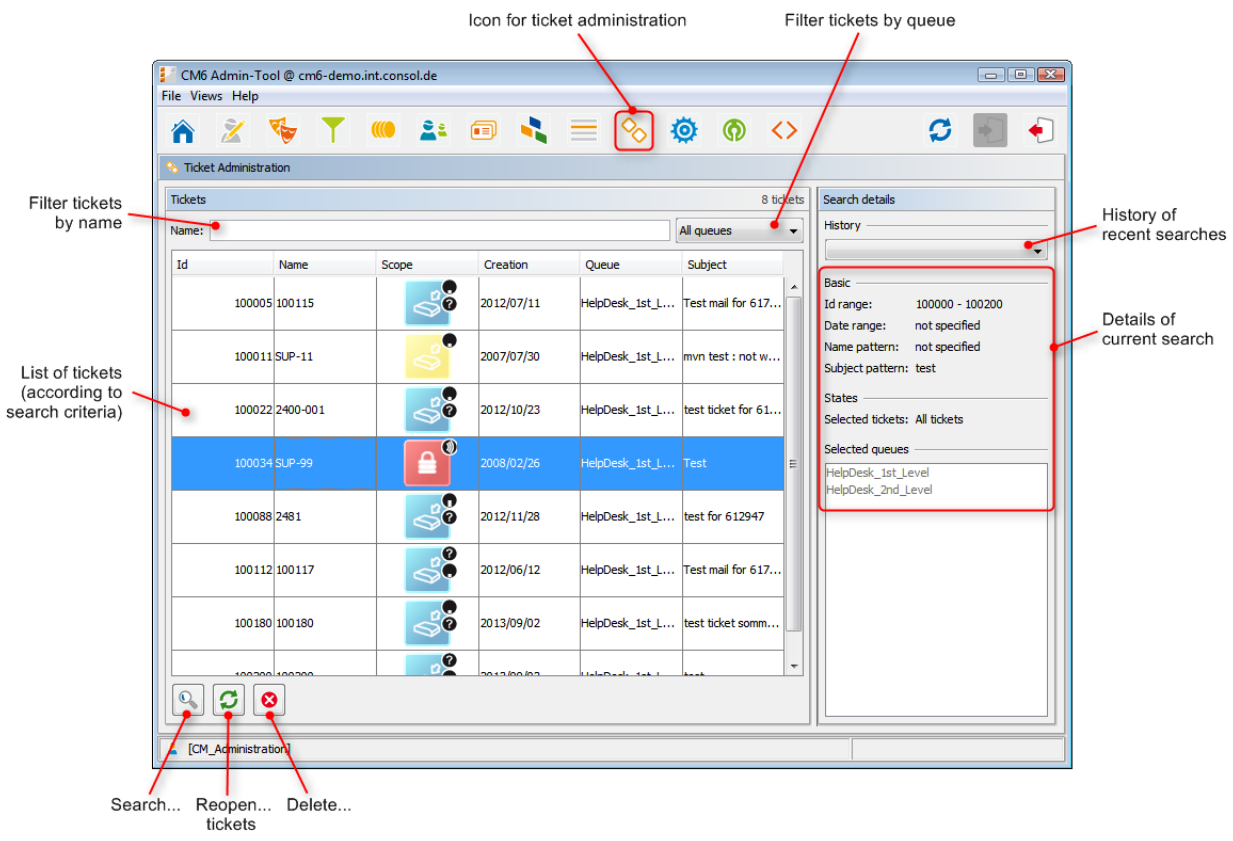
In the ticket administration you can:

- **Delete tickets**
  e.g. if a ticket was created by mistake.
- **Reopen tickets**
  e.g. if a ticket has been closed too early.

Attention:

Please keep in mind that a ticket, which is reopened, starts in the process at the start node of the respective workflow. So when a ticket has passed 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 way is to modify the workflow to contain a shortcut.

21.2 Ticket Administration Using the Admin Tool



Id	Name	Scope	Creation	Queue	Subject
100005	100115		2012/07/11	HelpDesk_1st_L...	Test mail for 617...
100011	SUP-11		2007/07/30	HelpDesk_1st_L...	mvn test : not w...
100022	2400-001		2012/10/23	HelpDesk_1st_L...	test ticket for 61...
100034	SUP-99		2008/02/26	HelpDesk_1st_L...	Test
100088	2481		2012/11/28	HelpDesk_1st_L...	test for 612947
100112	100117		2012/06/12	HelpDesk_1st_L...	Test mail for 617...
100180	100180		2013/09/02	HelpDesk_1st_L...	test ticket somm...

Basic

Id range:	100000 - 100200
Date range:	not specified
Name pattern:	not specified
Subject pattern:	test

States

Selected tickets: All tickets

Selected queues

HelpDesk_1st_Level
HelpDesk_2nd_Level

Fig. 1: ConSol*CM Admin Tool - Ticket Administration after Ticket Search

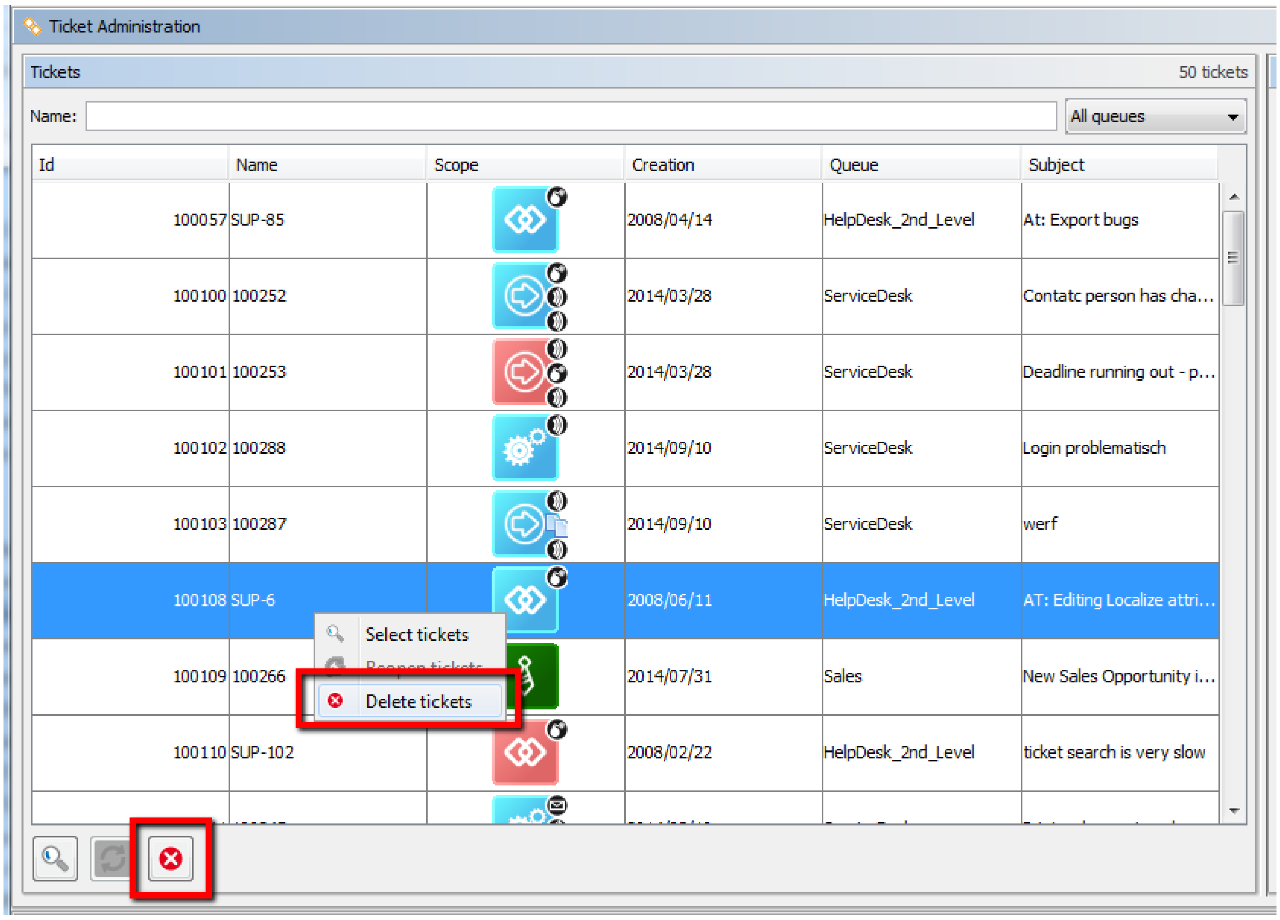
21.2.1 Delete or Reopen Tickets

For those operations you can either use the buttons below the list, or you can use the context menu.

1. buttons:
   Select the desired tickets in the list and click on ⊗ to delete tickets resp. click on ↻ to reopen tickets. If you confirm the following dialog with *Yes*, the corresponding action will be executed.
2. context menu:
   Select the desired tickets in the list and use the right mouse button to open the context menu. Select the desired operation.

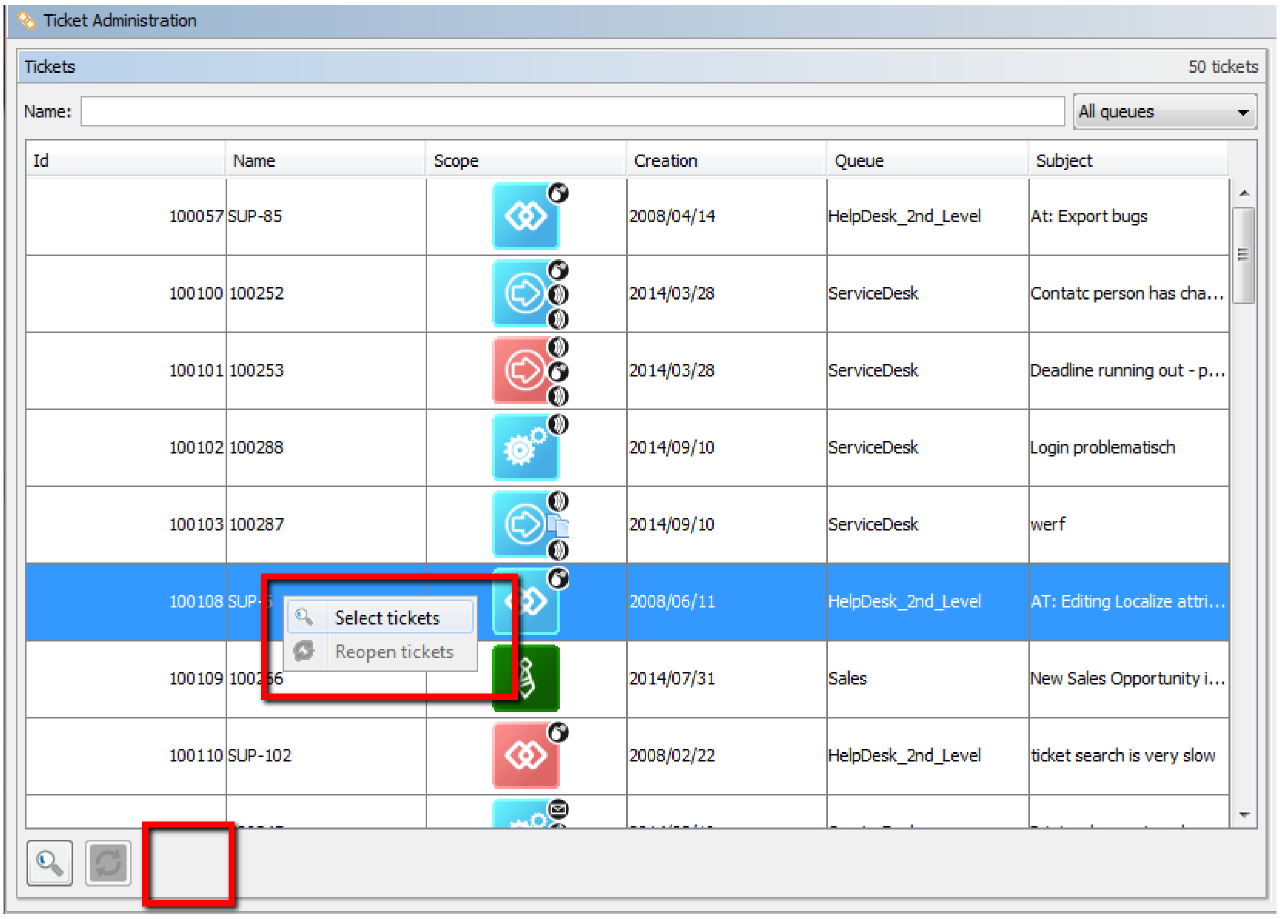
21.2.2 Switching off the Delete Functionality Using a System Property

The delete functionality can be switched off 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.



Id	Name	Scope	Creation	Queue	Subject
100057	SUP-85		2008/04/14	HelpDesk_2nd_Level	At: Export bugs
100100	100252		2014/03/28	ServiceDesk	Contatc person has cha...
100101	100253		2014/03/28	ServiceDesk	Deadline running out - p...
100102	100288		2014/09/10	ServiceDesk	Login problematisch
100103	100287		2014/09/10	ServiceDesk	werf
100108	SUP-6		2008/06/11	HelpDesk_2nd_Level	AT: Editing Localize attri...
100109	100266		2014/07/31	Sales	New Sales Opportunity i...
100110	SUP-102		2008/02/22	HelpDesk_2nd_Level	ticket search is very slow

Fig. 2: Ticket administration page with cmas-app-admin-tool,delete.ticket.enabled = true

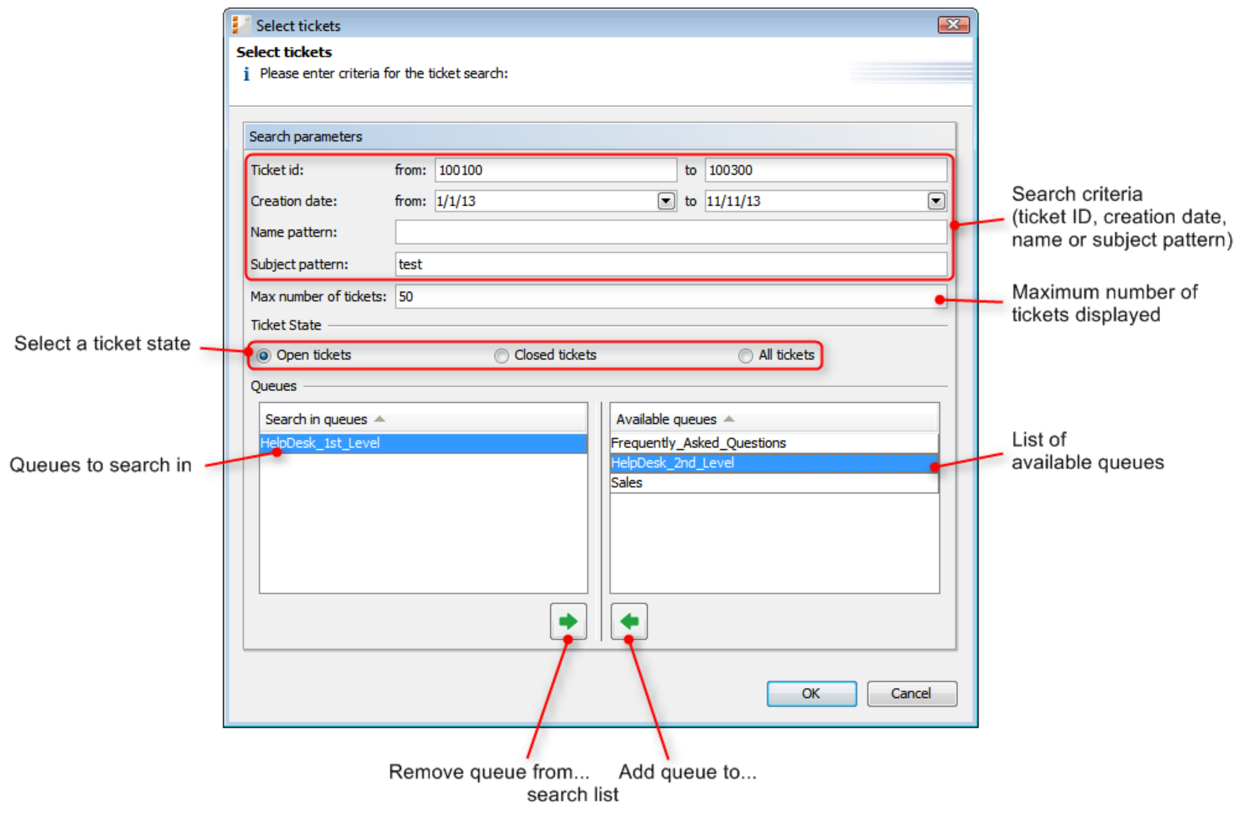


Id	Name	Scope	Creation	Queue	Subject
100057	SUP-85		2008/04/14	HelpDesk_2nd_Level	At: Export bugs
100100	100252		2014/03/28	ServiceDesk	Contatc person has cha...
100101	100253		2014/03/28	ServiceDesk	Deadline running out - p...
100102	100288		2014/09/10	ServiceDesk	Login problematisch
100103	100287		2014/09/10	ServiceDesk	werf
100108	SUP-5		2008/06/11	HelpDesk_2nd_Level	AT: Editing Localize attri...
100109	100256		2014/07/31	Sales	New Sales Opportunity i...
100110	SUP-102		2008/02/22	HelpDesk_2nd_Level	ticket search is very slow

Fig. 3: Ticket administration page with cmas-app-admin-tool,delete.ticket.enabled = false

21.2.3 Search Tickets

To search for tickets you want to delete or reopen click on 🔍 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.


Select tickets

Select tickets

Please enter criteria for the ticket search:

Search parameters

Ticket id:	from:	100100	to	100300
Creation date:	from:	1/1/13	to	11/11/13
Name pattern:				
Subject pattern:		test		
Max number of tickets:		50		

Ticket State

[✓ Open tickets] Closed tickets All tickets

Queues

Search in queues	Available queues
HelpDesk_1st_Level	Frequently_Asked_Questions
	HelpDesk_2nd_Level
	Sales

OK Cancel

Fig. 4: ConSol*CM Admin Tool - Ticket Administration: Ticket Search

The following parameters can be used for searching:

- **Ticket id:**
  You can enter an ID range for the tickets here.
- **Creation date:**
  Via calendars you can define a time period within which the tickets have been opened.
- **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:**
  Here you can specify the maximum number of tickets displayed in the list.
- **Ticket State:**
  Using the radio buttons you can determine if you want to search for *open*, *closed*, or *all* tickets.
- **Queues:**
  The list on the right shows the available queues. Select the queues to search in here and click on 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.

Click on *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 last 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 just start it again.

21.3 Related Topics

- Queue

22 CM6 Administrator Manual 6.9.4 - Configuration

22.1 Configuration

- Introduction to the Configuration Page
- Perform Configuration Operations Using the Admin Tool
- Related Topics

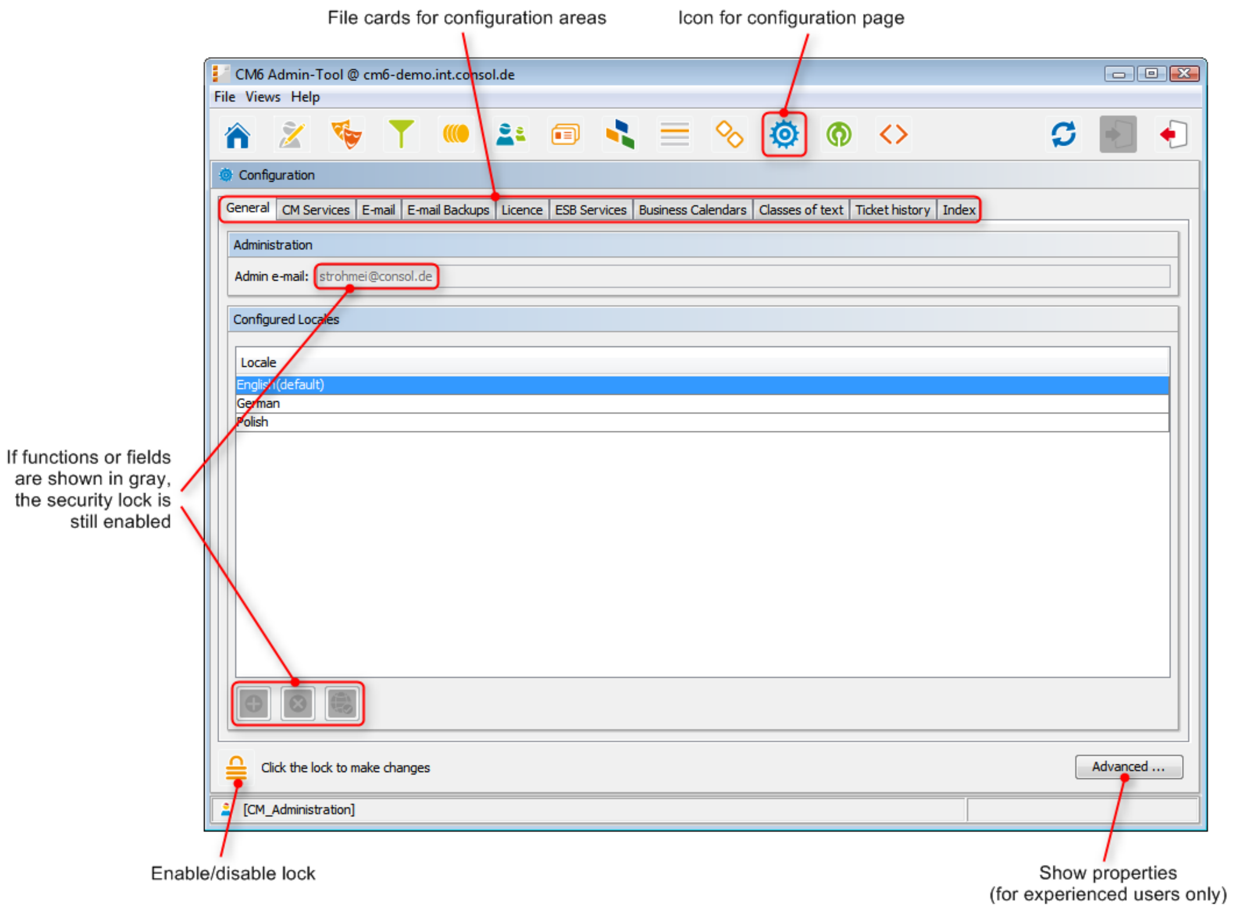
22.1.1 Introduction to the Configuration Page

On the *Configuration* page the general settings of the ConSol*CM server can be configured. Because changes done here affect the central functionality of the server significantly or even deactivate it, there is a security lock (see *enable/disable lock* in the following figure) on the bottom left corner of the page which has to be disabled to change the settings.

Attention:

If you are not sure about the effects of your changes please contact the ConSol*CM6 support team first and ask for help.

22.1.2 Perform Configuration Operations Using the Admin Tool


File cards for configuration areas
Icon for configuration page
CM6 Admin-Tool @ cm6-demo.int.consol.de
File Views Help
Configuration
General
CM Services
E-mail
E-mail Backups
Licence
ESB Services
Business Calendars
Classes of text
Ticket history
Index
Administration
Configured Locales
If functions or fields are shown in gray, the security lock is still enabled
Click the lock to make changes
Advanced ...
[CM_Administration]
Enable/disable lock
Show properties (for experienced users only)

Admin e-mail: strohmei@consol.de

Locale
English (default)
German
Polish

Fig. 1: ConSol*CM Admin Tool - Configuration Page

Using the tabs on top of the page you can switch to the tabs of the different configuration areas:

- Tab General
- Tab CM Services
- Tab E-Mail
- Tab E-Mail Backups
- Tab Licence
- Tab ESB Services
- Tab Business Calendars
- Tab Classes of Text
- Tab Ticket History
- Tab Index (Search and Indexer Configuration)

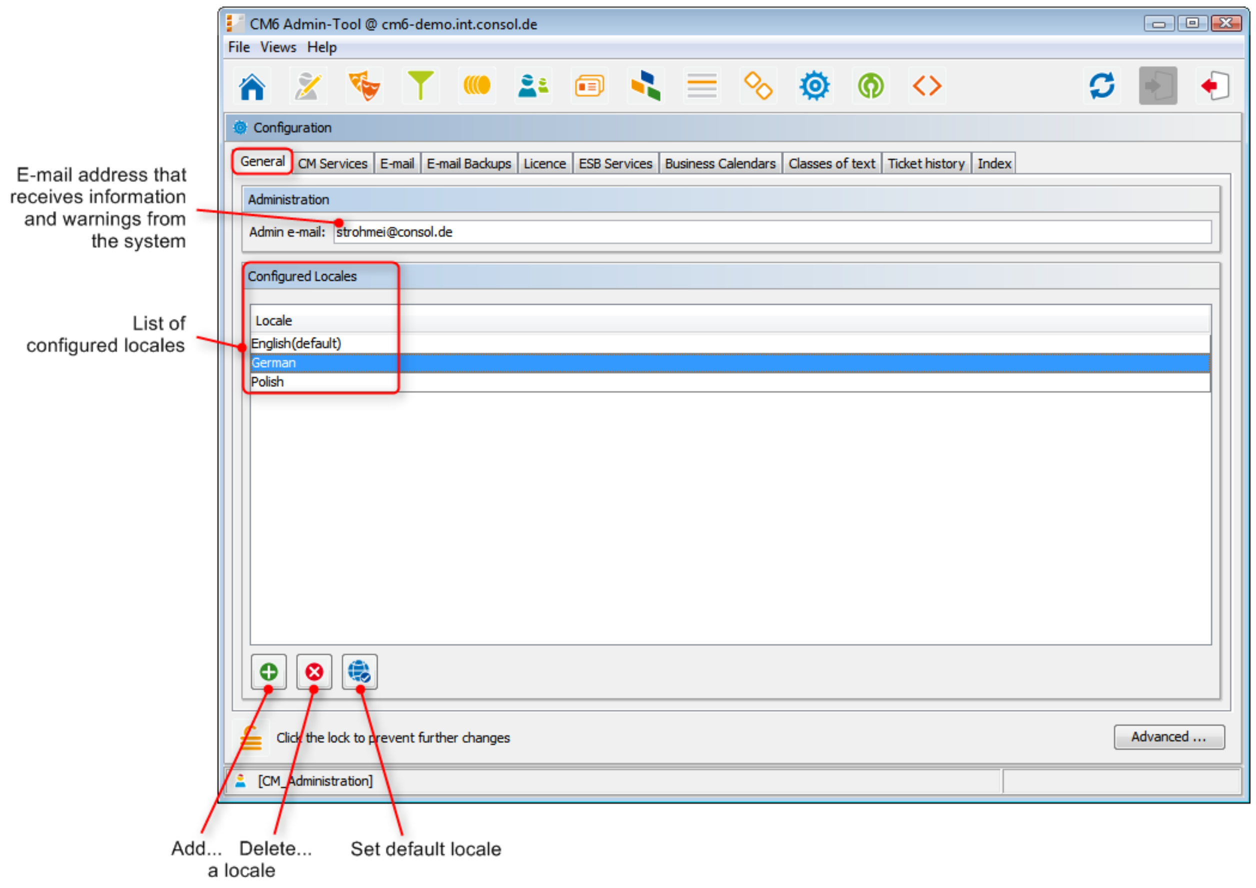
The *Advanced* button on the bottom right corner leads to a special page that contains all settings which are stored as system properties. This page should only be used by trained staff or upon request by ConSol*CM support or consultants. See section Appendix C (System Properties) for a detailed list with explanations of all system properties.

22.1.3 Related Topics

- Workflow (see *ConSol*CM Process Designer Manual*)

22.2 Tab General

On this tab you can set the administrator e-mail address and the locales for the administration interface, i.e. for the Admin-Tool and the Process Designer.



Admin e-mail: strohmei@consol.de

Locale
English(default)
German
Polish

Fig. 1: ConSol*CM Admin-Tool - Configuration: General

- **Admin e-mail:**
  Enter the e-mail address which shall receive general messages or warnings from the system. Multiple addresses separated by commas are possible, the total number of characters should not exceed 72. If there are many recipients we recommend using a mailing list on the mail server system.
- **Configured Locales:**
  In this list, the locales 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 custom field values). The displayed values for those activities or fields then depend on the locale of the web browser which is used to display the ConSol*CM Web Client.
  - Click on ⊕ to add more locales.
  - Click on ⊗ to remove the selected locale from the list.
  - Click on 🌐 to set the selected locale as default locale. The default locale will be used if the browser locale is not present in CM, e.g. when the engineer has set the browser locale to FR and in the CM administration only English (default), German, and Polish are available, the English values will be displayed.

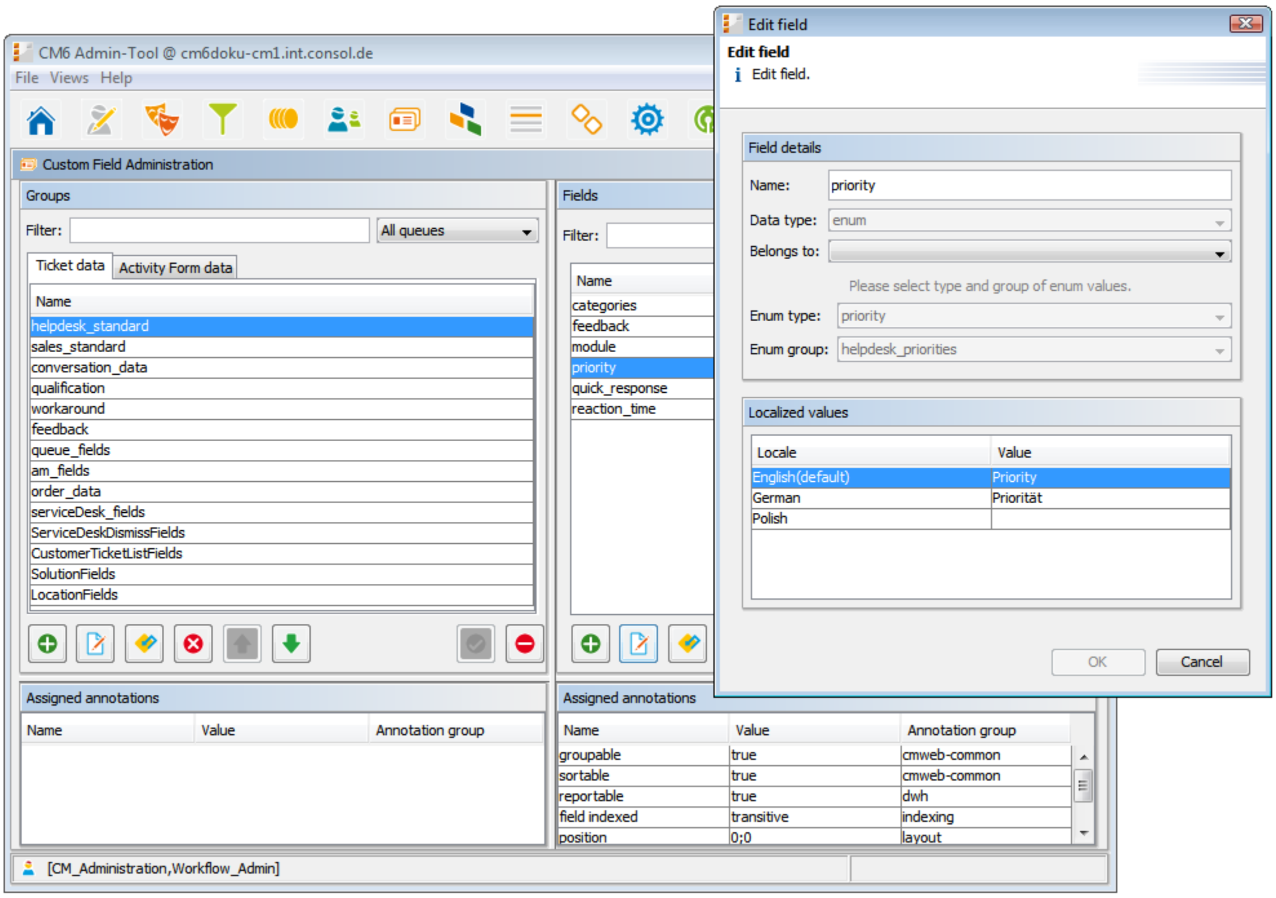
Attention:

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.

22.2.1 The Use of Locales

For an engineer who works with the ConSol*CM Web Client, the GUI 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 locale can be found, the default locale which has been set in the Admin-Tool is used.

When an administrator configures custom fields or data object group fields he/she can always indicate a translation for each configured language/locale, see following figure.



Name
helpdesk_standard
sales_standard
conversation_data
qualification
workaround
feedback
queue_fields
am_fields
order_data
serviceDesk_fields
ServiceDeskDismissFields
CustomerTicketListFields
SolutionFields
LocationFields

Name
categories
feedback
module
priority
quick_response
reaction_time

Name: priority
Data type: enum
Belongs to:
Please select type and group of enum values.
Enum type: priority
Enum group: helpdesk_priorities

Locale	Value
English(default)	Priority
German	Priorität
Polish	

Name	Value	Annotation group

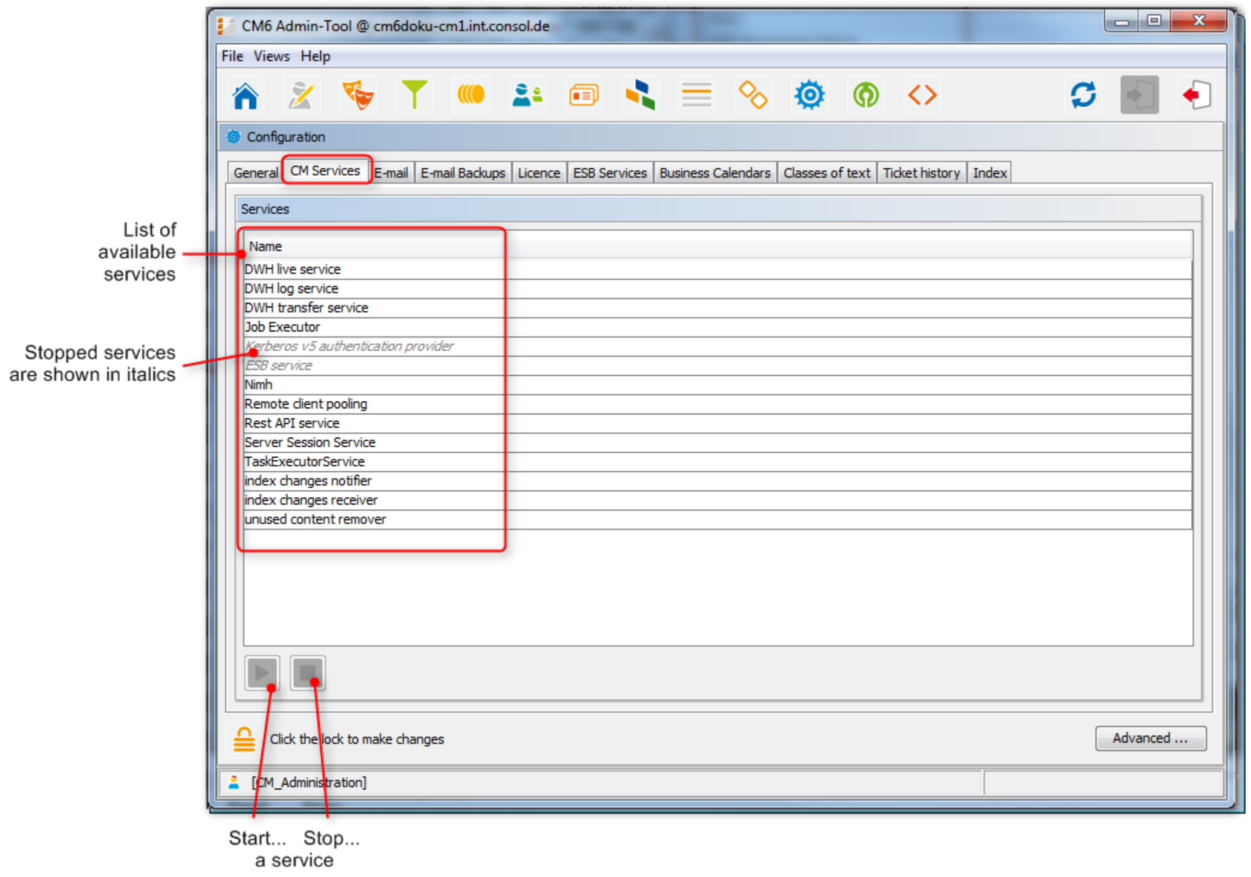
Name	Value	Annotation group
groupable	true	cmweb-common
sortable	true	cmweb-common
reportable	true	dwh
field indexed	transitive	indexing
position	0;0	layout

Fig. 2: ConSol*CM Admin-Tool - Custom Field Administration: Localized Values for a Custom Field

In the Process Designer, the locales which have been configured in the Admin-Tool are offered. However, you can also delete locales in the Process Designer. Please refer to the *ConSol*CM Process Designer Manual* for details.

22.3 Tab CM Services

On this tab you can start ▶ or stop ■ the individual services of the CM system, e.g. data indexing or mail connectivity.


List of available services
Stopped services are shown in italics
Start... Stop... a service

CM6 Admin-Tool @ cm6doku-cm1.int.consol.de

File Views Help

Configuration

General | CM Services | E-mail | E-mail Backups | Licence | ESB Services | Business Calendars | Classes of text | Ticket history | Index

Services

Name
DWH live service
DWH log service
DWH transfer service
Job Executor
Kerberos v5 authentication provider
ESB service
Nimh
Remote client pooling
Rest API service
Server Session Service
TaskExecutorService
index changes notifier
index changes receiver
unused content remover

Click the lock to make changes

Advanced ...

[CM_Administration]

Fig. 1: ConSol*CM Admin Tool - Configuration: CM Services

Attention:

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:

- **DWH live service**
  Controls just-in-time DWH update in LIVE mode.
- **DWH log service**
  Reads and processes CMRF/DWH log messages for Admin Tool and stores them in CM DB. The entries are used for the log protocol in the Admin Tool. See section Data Warehouse (DWH) Management.

- **DWH transfer service**
  Controls DWH transfer.
- **Job Executor**
  Controls the escalations for processes resp. workflows.
- **Kerberos v5 authentication provider**
  Required if Kerberos authentication is in operation.
- **ESB service**
  Retrieves incoming e-mails when the servcer is running in ESB/Mule Mail mode. This service is deactivated when NIMH is used as incoming mail module. Please see also section Tab ESB Services
  .
- **NIMH**
  Retrieves incoming e-mails when the server is running in NIMH mode. This service is deactivated when ESB/Mule Mail is used as incoming mail module.
- **Remote client pooling**
  Controls that Web Clients get changes from Admin Tool.
- **Rest API service**
  Activates or deactivates REST (*Representation State Transfer* interface).
- **Server Session Service**
  Checks sessions and stops session when end of client or Admin Tool session end has been reached. 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 threads pool used for tasks execution.
- **Index changes notifier**
  Creates JMS (*Java Message Service*) messages with notifications that there has been one or more change(s) that concern(s) the index.
- **Index changes receiver**
  Reads JMS queue and starts update in Indexer.
- **Unused content remover**
  Removes attachments and comments which have been marked as *deleted* in the Web Client (in the protocol section of a ticket).

22.4 Tab E-Mail

- Introduction to E-Mails in ConSol*CM
    - Sending E-Mails from ConSol*CM
        - Manual E-Mails
        - Automatic E-Mails
    - Receiving E-Mails with ConSol*CM
- E-Mail Configuration Using the Admin Tool
    - General E-Mail Configuration (Tab E-Mail Configurations)
        - Incoming E-Mail
        - Outgoing E-Mail
    - E-Mail Modes: ESB/Mule Mail or NIMH
        - E-Mail Configuration Using ESB/Mule Mail
        - E-Mail Configuration Using NIMH
        - Switching from ESB/Mule Mail to NIMH
        - Running NIMH in a Clustered Environment
    - E-Mail Encryption
        - General Explanation about E-Mail Encryption in ConSol*CM
            - Requirements
            - Certificate Import from LDAP
        - Certificates Management in the Admin Tool
            - Server Certificates
            - Client Certificates
        - Use Cases
        - Sending Encrypted E-Mails
            - Choosing if E-Mails Should be Sent Encrypted from the Web Client
            - Sending an Encrypted E-Mail from the Workflow
            - Sending Encrypted E-Mails by Default
- E-Mail Duplication in the ConSol*CM Web Client
- Related Topics

In this section, the tab *E-mail* in the Admin Tool will be explained, including e-mail encryption. Furthermore, the e-mail-related system properties and the configuration for e-mail duplication will be shown.

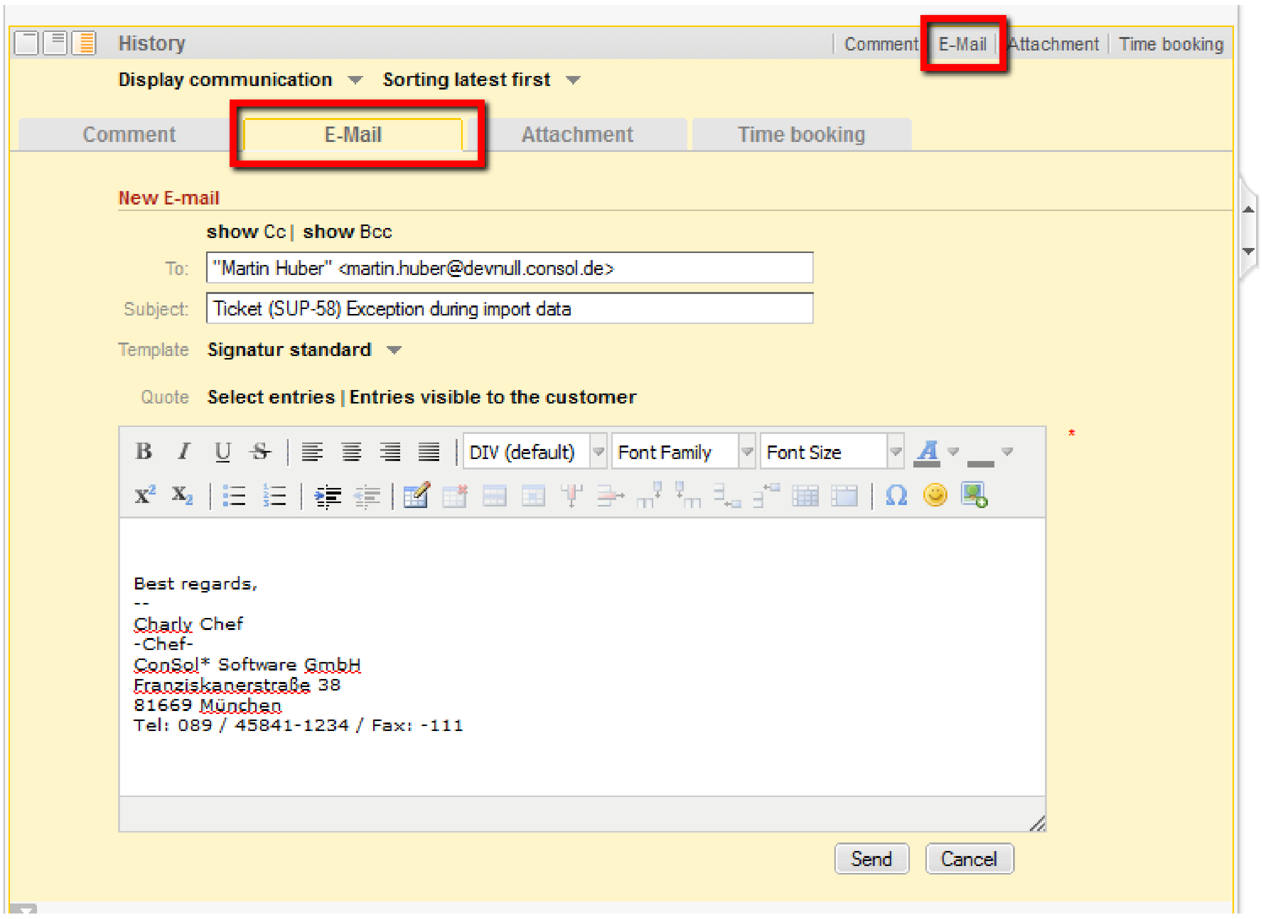
22.4.1 Introduction to E-Mails in ConSol*CM

Before we explain the administration of e-mail accounts using the ConSol*CM Admin Tool, we will give you a short introduction on the subject *e-mail with ConSol*CM*, because this represents a core functionality of the application. ConSol*CM can send and receive e-mails.

Sending E-Mails from ConSol*CM

Manual E-Mails

E-mails can be sent manually by an engineer or automatically by the system. *Manual* e-mails are sent using the *Ticket E-Mail Editor*. As default, the ticket's main contact is the receiver of the e-mail, but the engineer can select or type any other e-mail address. Furthermore, he can use e-mail templates and/or quote ticket text. Please see the *ConSol**CM User Manual* for a detailed introduction about working with the Ticket E-Mail Editor. The default setting can also be modified by using page customization, see section Page Customization.



To: "Martin Huber" <martin.huber@devnull.consol.de>
Subject: Ticket (SUP-58) Exception during import data
Template Signatur standard
Quote Select entries | Entries visible to the customer

Best regards,
--
Charly Chef
-Chef-
ConSol* Software GmbH
Franziskanerstraße 38
81669 München
Tel: 089 / 45841-1234 / Fax: -111

Fig. 1: ConSol*CM Web Client - Ticket E-Mail Editor

Automatic E-Mails

Automatic e-mails might be sent by ConSol*CM in situations like the following:

1. Initiated by the workflow engine, e.g.
   a. when the engineer to whom the ticket is assigned should be reminded to take care of the ticket.
   b. when the customer should receive an automatic confirmation that a ticket has been opened for him/her.

   c. when the customer should receive an automatic confirmation that his/her ticket has been closed.
   d. when a supervisor or approver should receive a message that a new case has to be approved.

   In any workflow activity, an e-mail can be sent to every valid e-mail address. Please see the *ConSol*CM Process Designer Manual* for a detailed explanation of the methods to use.
2. Initiated by the system in case of an error or for a success message, e.g.
   a. system error
   b. e-mail error
   c. DWH synchronization (error or success)
      Usually, those e-mails are sent to the ConSol*CM administrator. However, for most special error cases a special receiver e-mail address can be configured using system properties. Please see section Appendix C (System Properties) for details.
3. Initiated by the CM system to remind engineers
   a. When an engineer receives a ticket or a ticket is retrieved from the engineer, an e-mail can be sent to this engineer. This can be configured for every queue, see section Queue Administration.

Receiving E-Mails with ConSol*CM

The ConSol*CM system can fetch e-mails from one or more mailboxes (= e-mail accounts) on one or more e-mail server(s). The mailboxes are configured in the Admin Tool (E-mail configuration). Please keep in mind that ConSol*CM works with mailboxes here. Each of the mailboxes can be reached by at least one e-mail address. In certain cases, one mailbox might be used for more than one e-mail address. This can be of importance for writing e-mail scripts.

ConSol*CM acts towards the e-mail server like a regular e-mail client by fetching the e-mails using a standard mail protocol: IMAP(s), POP3(s). Depending on the mail server configuration and on the ConSol*CM system property *cmas-esb-mail*, *mail.delete.read*, the e-mails are deleted from the mailbox on the e-mail server after ConSol*CM has picked them up. The default setting is *mails are not deleted after pick-up*.

⚠ In case you do not want ConSol*CM to delete e-mails from the e-mail server, please make sure to control the mailbox(es) manually to avoid a data overflow and server or performance problems.

All incoming e-mails are first stored in an incoming mail pool in ConSol*CM and are then processed in a chain of mail scripts. Please see section E-Mail Scripts for a detailed explanation of those scripts. When an e-mail cannot be processed, the administrator will receive a notification e-mail. The unprocessed e-mail is listed under E-mail Backups.

There are different possibilities concerning the default system behavior concerning an incoming e-mail:

- The subject of the e-mail 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 e-mail does contain a ticket number with a valid syntax (RegEx) and the ticket is still open:
  The e-mail is attached to the existing ticket.
- The subject of the e-mail 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 e-mail scripts (see section E-Mail Scripts), 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!

22.4.2 E-Mail Configuration Using the Admin Tool

IMPORTANT INFORMATION

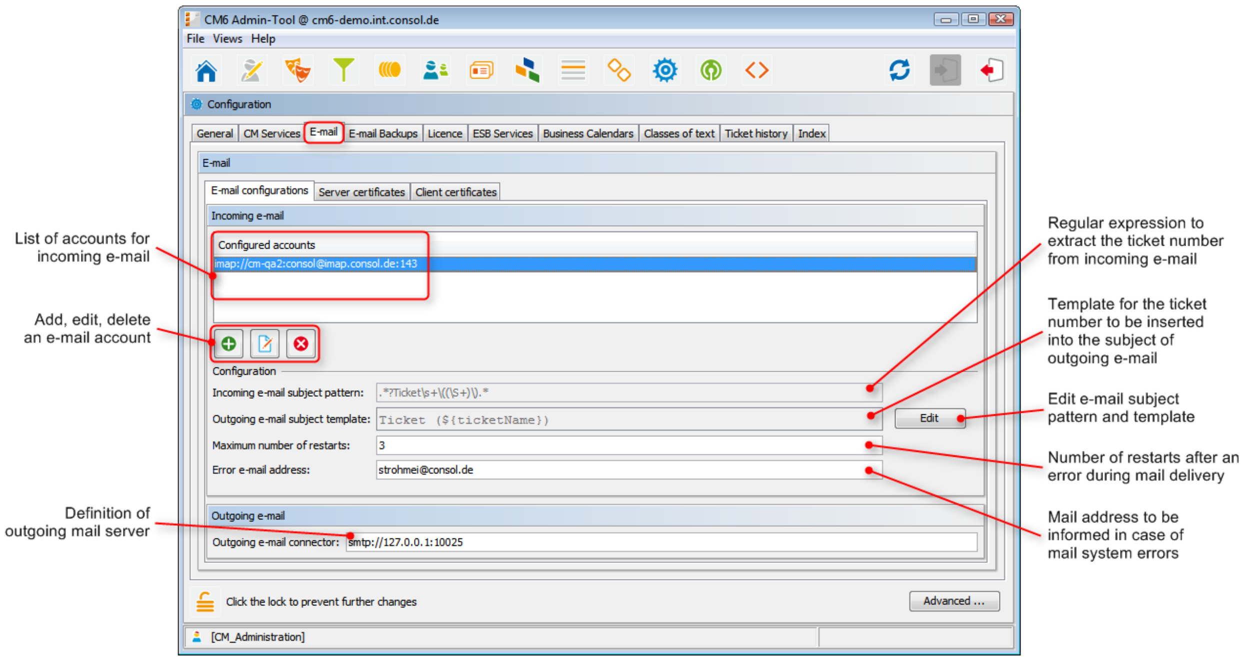
In ConSol*CM version 6.9.4, there are two modes to receive incoming e-mails:

- **Mule/ESB** - this has also been available in all previous CM versions
- **NIMH** (New Incoming Mail Handler) - new in version 6.9.4

For all configurations/settings which are valid for both modes, no further notes are added. For all settings which vary depending on the mode, this will be explained in separate (i.e. Mule/ESB or NIMH specific) sections.

General E-Mail Configuration (Tab E-Mail Configurations)

On this tab you can set parameters for the e-mail connection.


CM6 Admin-Tool @ cm6-demo.int.consol.de
File Views Help
Configuration
General
CM Services
E-mail
E-mail Backups
Licence
ESB Services
Business Calendars
Classes of text
Ticket history
Index
E-mail
E-mail configurations
Server certificates
Client certificates
Incoming e-mail
Configured accounts
Configuration
Edit
Outgoing e-mail
Click the lock to prevent further changes
Advanced ...
[CM_Administration]
List of accounts for incoming e-mail
Add, edit, delete an e-mail account
Definition of outgoing mail server
Regular expression to extract the ticket number from incoming e-mail
Template for the ticket number to be inserted into the subject of outgoing e-mail
Edit e-mail subject pattern and template
Number of restarts after an error during mail delivery
Mail address to be informed in case of mail system errors

Configured accounts: imap://cm-qa2:consol@imap.consol.de:143

Incoming e-mail subject pattern:	.*?Ticket\s+\((\S+)\).*
Outgoing e-mail subject template:	Ticket (${ticketName})
Maximum number of restarts:	3
Error e-mail address:	strohmei@consol.de
Outgoing e-mail connector:	smtp://127.0.0.1:10025

Fig. 2: ConSol*CM Admin Tool - Configuration: E-Mail

Incoming E-Mail

The configuration of incoming e-mail is divided into two areas:

- **Configured accounts:**
  Here you can use a pop-up window to add or edit accounts from which e-mails 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 as system property *cmas-esb-mail*, *mail.incoming.uri*. Please see the e-mail properties section in Appendix C for detailed information. Required values are:
  - **Protocol**
    The protocol used to retrieve e-mails from the server. Supported protocols are IMAP4, IMAP4s, POP3, and POP3s. Please keep in mind that ConSol*CM acts towards the e-mail server like a regular e-mail 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**
    The name or IP address of the e-mail server.
  - **Port**
    The port on the e-mail server where the mail daemon/service is listening.
  - **User name**
    The user name of the e-mail account.
  - **Password**
    The password of the e-mail account.

    Attention:

    Please keep in mind that one e-mail account can have more than one e-mail 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 e-mails, it might be required to use the e-mail address. The e-mail address is also required when you configure the REPLY-TO-address, the FROM-address, and queue-specific e-mail addresses! So be sure to use the correct parameter: mailbox or e-mail address!

- **Configuration:**
  - **Incoming e-mail subject pattern:**
    Describes the elements that the subject of an incoming e-mail has to contain in order to assign this e-mail 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 (<Ticketnumber>)*.
  - **Outgoing e-mail subject template:**
    Describes the pattern which is used to create the ticket ID in the subject of an outgoing e-mail. The template should be matched by the incoming e-mail subject pattern. Via the *Edit* button on the right you can modify the incoming e-mail subject pattern and outgoing e-mail subject

template and verify if they match.
Example: *Ticket (${ticketName})* would match the example RegEx above.

Attention:

You can check if the pattern for the incoming e-mail subject pattern and for the outgoing e-mail subject template match by using the *Edit* button and the editor that is opened. Please make sure that the e-mail 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 e-mails. Valid for all e-mail pollers.
- **Error e-mail address:**
  E-mail 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 e-mail you might also want to check the e-mail-related system properties, see Appendix C (System Properties). Particularly, the polling interval (the time interval for fetching e-mails from the mail server, system property *cmas-esb-mail*, *mail.polling.interval*) might be of interest.

Outgoing E-Mail

The connection data for outgoing e-mails are set here:

- **Outgoing e-mail connector**
  Use the following format:

```
smtp://<IP address of mail server>:<port>
```

Example:

```
smtp://10.0.1.151:25
```

E-Mail Modes: ESB/Mule Mail or NIMH

Starting with version 6.9.4, there are two modules for the retrieval of incoming e-mails:

- **ESB/Mule Mail**
  Which has been available since the start of ConSol*CM version 6.
- **NIMH**
  The New Incoming Mail Handler, which is a new module in version 6.9.4. This will be the only available incoming e-mail module in future CM versions.

In CM version 6.9.4, you as an administrator can decide which module to use: you can run CM in ESB/Mule mode **or** in NIMH mode. ESB/Mule and NIMH use different system settings which are stored as system properties. They are explained in the following two sections. The "switch" which changes the incoming e-mail mode is the system property *cmas-core-server*, *nimh.enabled* which can be set to *true* or *false.*

The sending of e-mails, i.e. the SMTP server configuration is not influenced by the incoming e-mail mode.

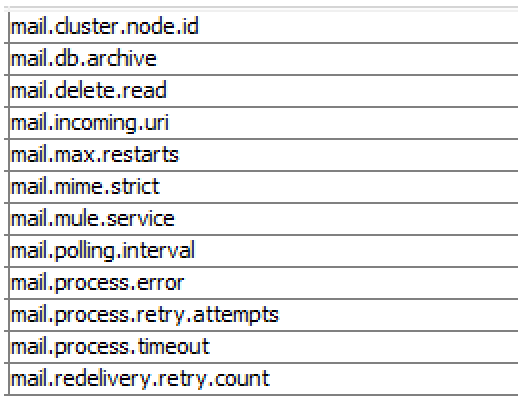
E-Mail Configuration Using ESB/Mule Mail

Mule is the internal ESB (Enterprise Service Bus) which is - besides other functions - used to retrieve incoming e-mails.

When ESB/Mule Mail is enabled, the following mechanisms apply:

- ESB/Mule Mail runs as services, see sections Tab CM Services and Tab ESB Services.
- E-mails are retrieved from the configured mailboxes.
- It is not possible to retrieve e-mails from the file system.
- E-mails which could not be processed are stored in a separate directory (*unparsable*) in the main ConSol*CM data directory and can be re-sent to the CM system, see section Tab E-Mail Backups, ESB/Mule Mail.
- ESB/Mule Mail system properties have to be set. Please keep in mind that you can set most of the properties using the graphical user interface of the Admin Tool, tab E-Mail (see sections above).

The following figure provides an example of ESB/Mule Mail properties. Please also refer to the detailed explanation of ESB/Mule Mail system properties in the properties section in Appendix D (Important System Properties, ordered by area of application)



mail.cluster.node.id
mail.db.archive
mail.delete.read
mail.incoming.uri
mail.max.restarts
mail.mime.strict
mail.mule.service
mail.polling.interval
mail.process.error
mail.process.retry.attempts
mail.process.timeout
mail.redelivery.retry.count

Fig. 3: ESB/Mule Mail system properties

E-Mail 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.

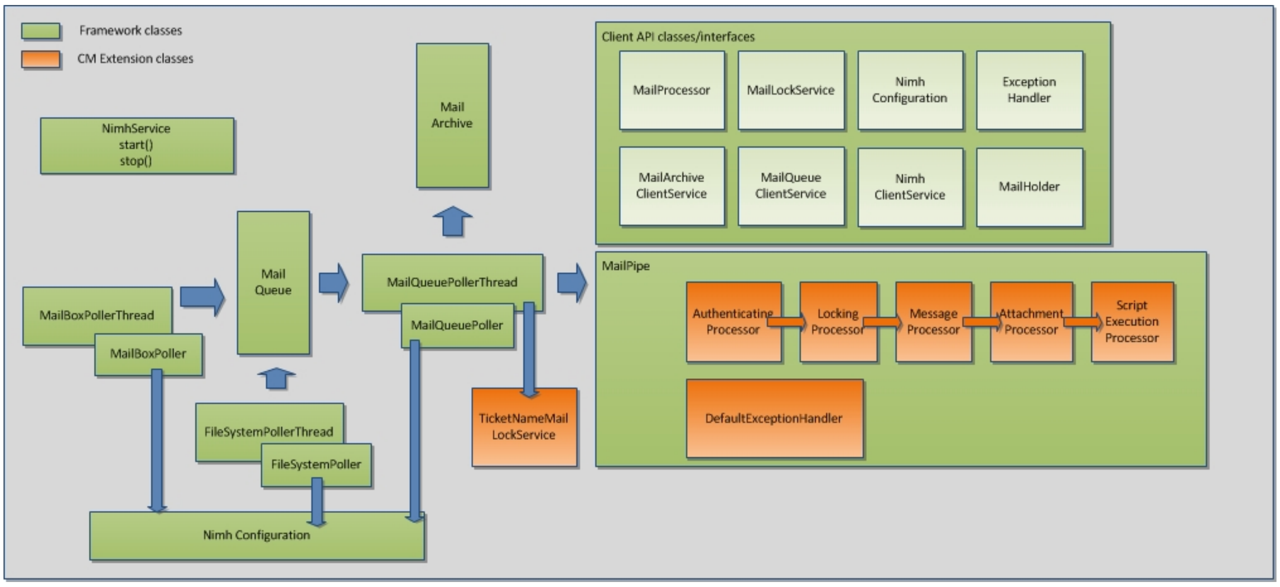
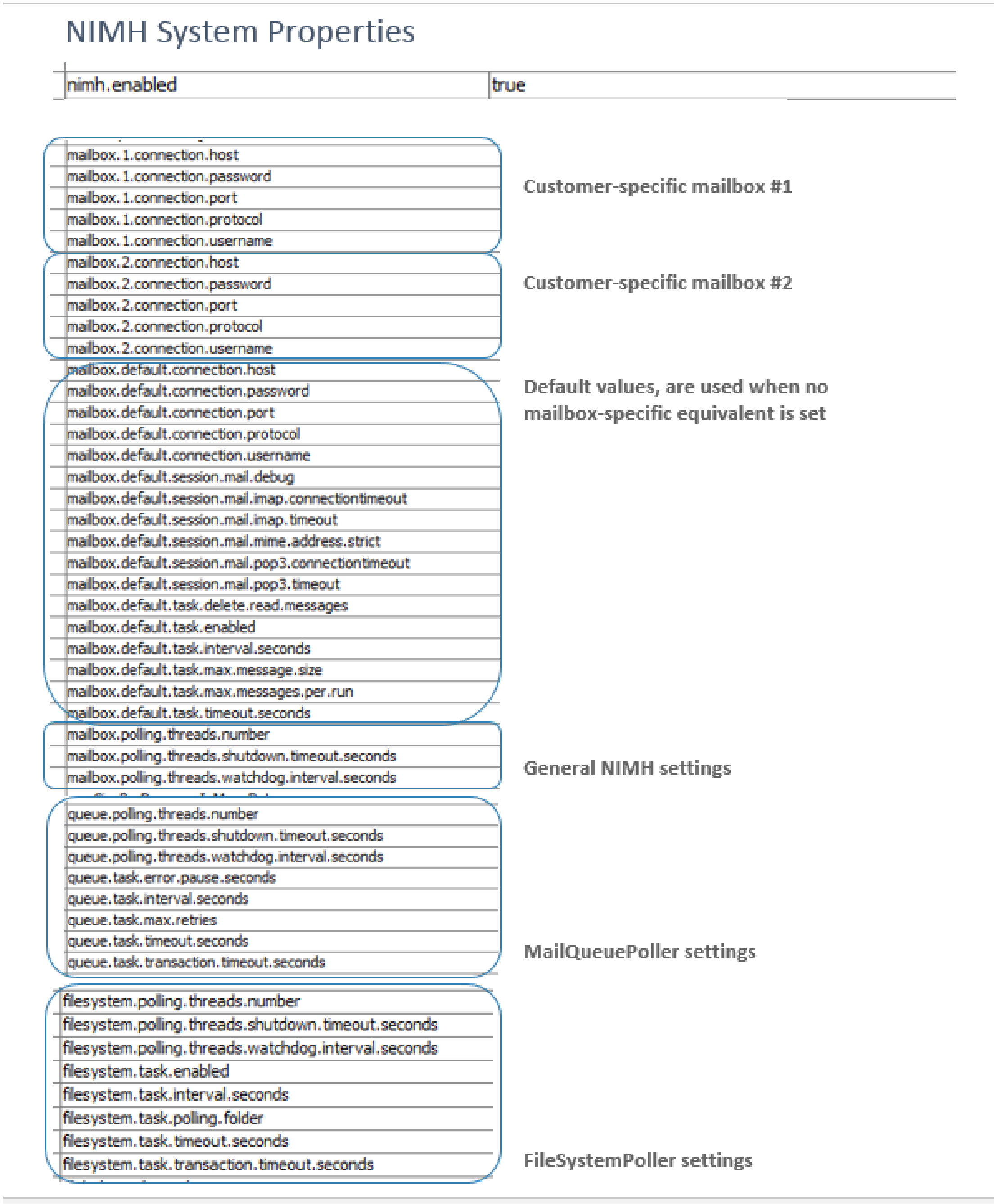


Fig. 4: NIMH components

When NIMH is enabled, the following mechanisms apply:

- NIMH runs as a (single) CM service, see section Tab CM Services. When NIMH is active, the ESB /Mule Mail services are disabled.
- E-mails from mailboxes on a mail server are retrieved by CM using the MailBoxPoller.
- E-mails can also be fetched from a data directory using the FileSystemPoller.
- All CM e-mails are stored in the CM database (nothing is stored in the file system).
- The MailQueuePoller retrieves the e-mails from the database and forwards them to the core CM system where the mail runs through the e-Mail script pipeline.
- NIMH uses e-mail scripts very similar to the ones used by ESB/Mule Mail, see section Admin Tool Scripts, Scripts of type e-mail (NIMH).
- E-mails which could not be processed are stored in a separate database table and can be re-sent to the CM system, see section Tab E-Mail Backups, NIMH.
- NIMH-specific system properties are used (they are added automatically to the system configuration during an update to version 6.9.4):
  - General NIMH properties
  - Default mailbox properties which are used when the property is not set as 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 the properties section in Appendix D (Important System Properties, ordered by area of application)



NIMH System Properties

nimh.enabled	true

mailbox.1.connection.host
mailbox.1.connection.password
mailbox.1.connection.port
mailbox.1.connection.protocol
mailbox.1.connection.username

Customer-specific mailbox #1

mailbox.2.connection.host
mailbox.2.connection.password
mailbox.2.connection.port
mailbox.2.connection.protocol
mailbox.2.connection.username

Customer-specific mailbox #2

mailbox.default.connection.host
mailbox.default.connection.password
mailbox.default.connection.port
mailbox.default.connection.protocol
mailbox.default.connection.username
mailbox.default.session.mail.debug
mailbox.default.session.mail.imap.connectiontimeout
mailbox.default.session.mail.imap.timeout
mailbox.default.session.mail.mime.address.strict
mailbox.default.session.mail.pop3.connectiontimeout
mailbox.default.session.mail.pop3.timeout
mailbox.default.task.delete.read.messages
mailbox.default.task.enabled
mailbox.default.task.interval.seconds
mailbox.default.task.max.message.size
mailbox.default.task.max.messages.per.run
mailbox.default.task.timeout.seconds

Default values, are used when no mailbox-specific equivalent is set

mailbox.polling.threads.number
mailbox.polling.threads.shutdown.timeout.seconds
mailbox.polling.threads.watchdog.interval.seconds

General NIMH settings

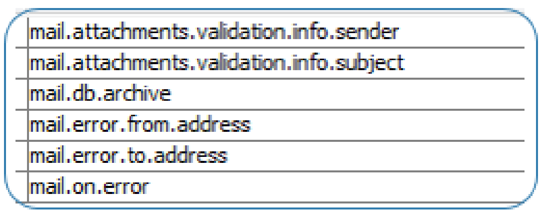
queue.polling.threads.number
queue.polling.threads.shutdown.timeout.seconds
queue.polling.threads.watchdog.interval.seconds
queue.task.error.pause.seconds
queue.task.interval.seconds
queue.task.max.retries
queue.task.timeout.seconds
queue.task.transaction.timeout.seconds

MailQueuePoller settings

filesystem.polling.threads.number
filesystem.polling.threads.shutdown.timeout.seconds
filesystem.polling.threads.watchdog.interval.seconds
filesystem.task.enabled
filesystem.task.interval.seconds
filesystem.task.polling.folder
filesystem.task.timeout.seconds
filesystem.task.transaction.timeout.seconds

FileSystemPoller settings

Fig. 5: NIMH System Properties, 1



mail.attachments.validation.info.sender
mail.attachments.validation.info.subject
mail.db.archive
mail.error.from.address
mail.error.to.address
mail.on.error

Extended NIMH settings

Fig. 6: NIMH System Properties, 2

Information about System Property mail.on.error

In ConSol*CM versions up to 6.9.4.1, the default value for the property *mail.on.error* (module *cmas-nimh-extension*) is *false*.

In ConSol*CM versions 6.9.4.2 and up, the default value for the property *mail.on.error* (module *cmas-nimh-extension*) is *true*.

The value will be automatically set to *true* during an update from versions 6.9.4.1 and below to a version 6.9.4.2 and up. Please be aware that you may have to disable these mail notifications after an update in case you intentionally have set the property value to *false*.

Switching from ESB/Mule Mail to NIMH

If you want to switch from ESB/Mule ("old") mode to NIMH ("new") incoming mail mode, you have to perform the following steps, please also note the info box *Step-By-Step Guide for a Switch From ESB/Mule Mail to NIMH*.

- Copy the Admin Tool mail scripts and rename the new scripts with the same with Nimh as Prefix, see also section Admin Tool Scripts, Scripts of type e-mail (NIMH).
  - AppendToTicket.groovy -> NimhAppendToTicket.groovy
  - CreateTicket.groovy -> NimhCreateTicket.groovy
  - IncomingMailRouting.groovy -> NimhIncomingMailRouting.groovy
  - MailToClosedTicket.groovy -> NimhMailToClosedTicket.groovy
- Adapt the new Admin Tool scripts, see also section Admin Tool Scripts, NIMH. There are new Groovy classes which have to be used to replace the old ones
  - import required NIMH classes
  - change methods according to the table displayed in the section Admin Tool Scripts, NIMH
  - change the section in mailRouting script for target handlers
  - control / set NIMH-specific system properties for mailboxes, see section above and properties section in Appendix D (Important System Properties, ordered by area of application)
  - switch system property to activate NIMH: *cmas-core-server.nimh.enabled* = *true*

Step-By-Step Guide for a Switch From ESB/Mule Mail to NIMH

If you have to switch your CM system from ESB/Mule Mail to NIMH, perform the following steps in the order listed here:

1. Prepare the NIMH environment:
   a. Create the Admin Tool scripts as described in the section above
   b. Adjust the system properties. The NIMH system properties will be automatically added to the system configuration during an update from a previous version to CM version 6.9.4. Only the required values have to be set. Mailbox properties are added automatically for each mailbox which is added using the Admin Tool.
2. Shut down ESB/Mule Mail (stop all ESB services using the Tab ESB Services and stop the ESB service using the Tab CM Services. Make sure CM finishes processing e-mails, review

e-mail backups and re-processing or delete the remaining e-mail backups. "Old" ESB/Mule e-mail backups will not be displayed when you have switched to NIMH (but will be displayed when you switch back to ESB/Mule Mail).

3. Start NIMH:
   a. Set the system property *cmas-core-server.nimh.enabled* = *true*
   b. Start the NIMH service

For a mapping of ESB Mail system properties to NIMH system properties, please refer to section Appendix D (Important System Properties, ordered by area of application).

Running NIMH in a Clustered Environment

In a cluster, NIMH can run only on one node. Here, the according system property has to be set: *cmas-core-server*, *nimh.enabled.CLUSTER_NODE_ID* = true
for example: cmas-core-server.nimh.enabled.1 = trueThis property replaces the general property *cmas-core-server.nimh.enabled* = *true*. On all other nodes NIMH and Mule (and ESB services) have to be disabled. Furthermore, set the system property *cmas-nimh*, *mailbox.polling.threads.mail.log.enabled* = *true.* Without this setting there is the chance that incoming e-mails may be processed several times by different cluster nodes

E-Mail Encryption

Due to increasing security policies, it might be required to encrypt the e-mail traffic (including the e-mails which are sent and received by the ConSol*CM installation) using standard S/MIME encryption.

In order to enable ConSol*CM to work with encrypted e-mails, you first have to enable the e-mail encryption in the system:

1. **Mandatory:**
   Set the system property *cmas-esb-mail*, *mail.encryption* to *true*. It is set to *false* as default value. This is the basic configuration for the entire system to enable e-mail encryption.
2. **Optional:**
   Set the page customization property *mailEncryptionAvailable* to *true.* This activates the possibility in the Web Client to choose if the e-mail should be encrypted.

General Explanation about E-Mail Encryption in ConSol*CM

There are two types of certificates:

- **Personal Information Exchange Certificates**
  For incoming e-mails (here, server certificates are relevant).
  - The *Personal Information Exchange* certificate can be manually imported to the system from the *PKCS12* (.p12) files. This file contains the public and the private key for the corresponding e-mail address. If the certificate file is protected with a password, the administrator must enter it during the import process.

- **Security Certificates**
  For outgoing e-mails (here, client certificates are relevant).
  Security certificates can be imported into the system in two ways:
  - **Manually**
    By selecting the *X.509* (.cer or .crt) file.
  - **Automatically**
    From the LDAP repository which holds it in the same format as for the file import. This can be done on demand during the e-mail sending.

Attention:

The certificates treated here, are used for e-mail encryption only and **not** for the access of ConSol*CM (as e-mail client) to the e-mail server! This has to be managed by certificates which are stored in the security store of the application server.

Requirements

- The client certificate must contain the e-mail address of the customer in the attribute *SubjectDN* (*E=* or *EMAILADDRESS=*) or the *X509v3 Subject Alternative Name* element from the *Extensions* section of the certificate must contain the e-mail address.
- *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

If LDAP is configured, ConSol*CM will lookup the certificate for the requested contact in the LDAP repository. This is done in the following way:

1. Someone tries to send an encrypted e-mail.
2. The cryptography service is looking for a client certificate for the recipient.
3. If a client certificate is found, the e-mail is encrypted 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 e-mail is encrypted and sent.
6. If it is not found, the e-mail is sent unencrypted.

The following configuration properties have to be set for the certificate lookup via LDAP:

- ldap.certificate.basedn
- ldap.certificate.searchattr
- ldap.certificate.content.attribute

Please see section LDAP certificate parameters in Appendix C (System Properties) for details.

Certificates Management in the Admin Tool

Server Certificates

Server certificates are used to decipher incoming e-mail messages and also to encrypt outgoing e-mail messages. They each contain the public and the private key for the given e-mail address. If you define an incoming e-mail account (see section above), you have to upload a server certificate for that e-mail address (or for all e-mail addresses covered by this mailbox) to be able to receive encrypted messages. 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 e-mail addresses.

When you open the tab *Server certificates*, a list of all existing server certificates is displayed. To add a new server certificate, click on ⊕ and use the file browser to find the required certificate. The certificate is checked 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

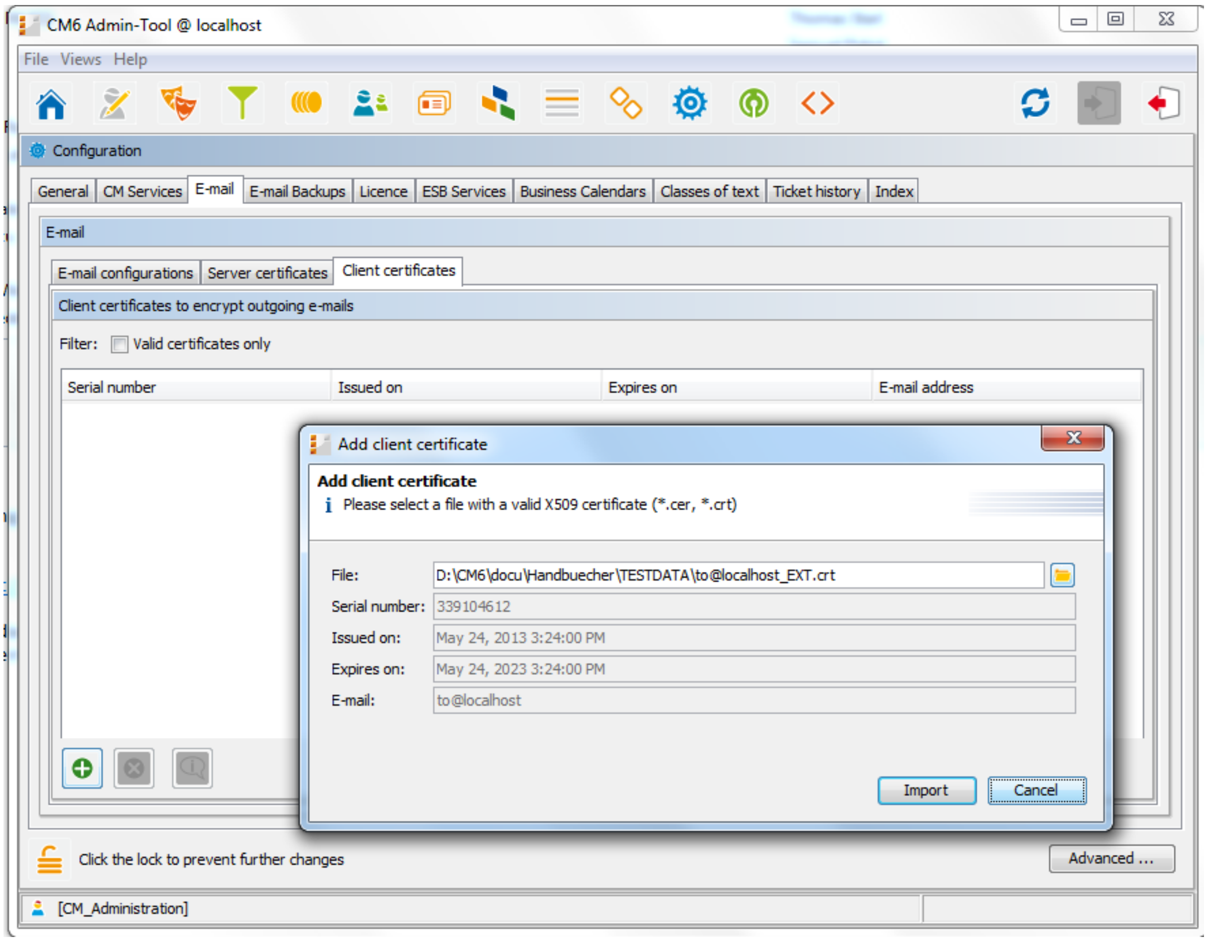
Client Certificates

A client certificate contains only the public key of a user. It allows encrypting messages going to that user.

When you open the tab *Client certificates*, a list of all existing client certificates is displayed. To add a new client certificate, click on ⊕ and use the file browser to find the required certificate. The certificate is checked 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:

- *X509* standard format.
  Supported filename extensions for *X.509* certificates are:
  - .cer
  - .crt
  - .der
  - .pem



File:	D:\CM6\docu\Handbuecher\TESTDATA\to@localhost_EXT.crt
Serial number:	339104612
Issued on:	May 24, 2013 3:24:00 PM
Expires on:	May 24, 2023 3:24:00 PM
E-mail:	to@localhost

Fig. 7: ConSol*CM Admin Tool - Pop-up Window for Adding a Client Certificate

Use Cases

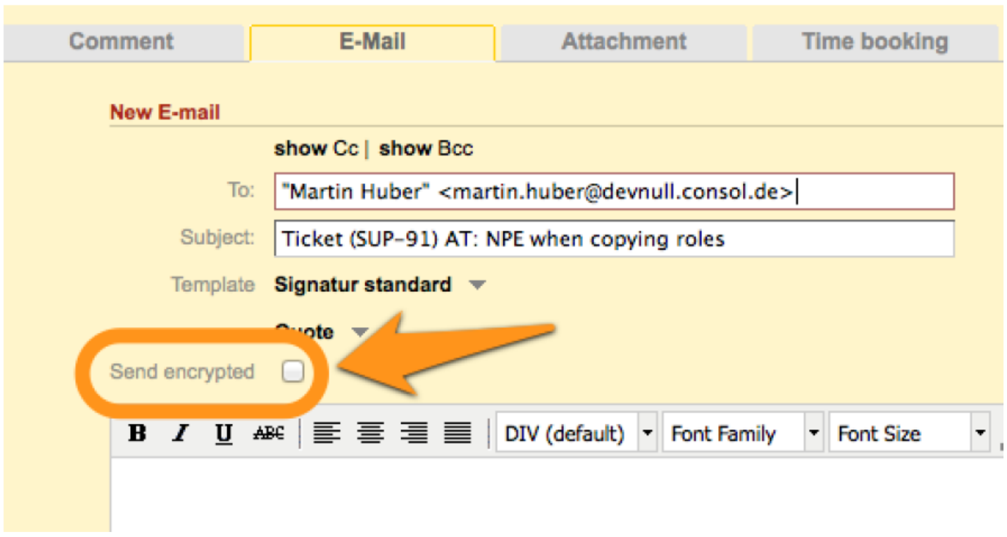
Here are some example use cases:

1. An engineer uses the ConSol*CM Web Client and writes an encrypted e-mail using the Ticket E-Mail Editor. When he/she presses the *Send* button, the ConSol*CM system looks up the receiver address in the list of mail addresses under *Client certificates* and uses the public key of the recipient to encrypt the outgoing e-mail. If ConSol*CM cannot find a matching certificate (the e-mail address is not mentioned in the list), the e-mail is loaded from LDAP. If this does not work either, the e-mail is sent unencrypted. If one of the recipients is the same as one of the incoming e-mail accounts, then also the server certificate will be used to encrypt that message.
2. ConSol*CM receives an e-mail and checks the TO-address. If this is found in the list under *Client 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 E-Mails

Choosing if E-Mails Should be Sent Encrypted from the Web Client

If the page customization property mailEncryptionAvailable has been set to *true*, a check box *Send encrypted* is available in the Ticket E-Mail Editor in the Web Client. Thus, the user can choose if the e-mail should be sent encrypted.



To: "Martin Huber" <martin.huber@devnull.consol.de>

Subject: Ticket (SUP-91) AT: NPE when copying roles

Template Signatur standard

Fig. 8: ConSol*CM Web Client - Send Encrypted E-Mail

Sending an Encrypted E-Mail from the Workflow

An encrypted e-mail can be sent by using the method *enableEncryption()*. Please see the *ConSol*CM Process Designer Manual* for a detailed explanation.

Sending Encrypted E-Mails by Default

If the system property *cmas-esb-mail*, *mail.encryption* is set to *true*, all outgoing e-mails from the workflow and Web Client are encrypted by default.

If users would like to send selected e-mails unencrypted, they can uncheck the check box *Send encrypted* in the Web Client. For e-mails sent by the workflow the method *disableEncryption()* has to be used.

22.4.3 E-Mail Duplication in the ConSol*CM Web Client

Please see explanations on the *Page Customization* page at showCloneOption and appendOrReplaceOnClone.

22.4.4 Related Topics

- E-mail properties - see section Appendix C (System Properties)
- E-mail scripts - see section E-Mail Scripts

22.5 Tab E-Mail Backups

- Introduction
- E-Mail Backups in the Admin Tool
- Management of E-Mail Backups with ESB/Mule
- Management of E-Mail Backups with NIMH

22.5.1 Introduction

Incoming e-mails which could not be processed are stored in a special store in the CM system (the location varies depending on the mode of the incoming mail configuration, see ESB and NIMH sections below).

You as an administrator can then try to re-send the e-mails to the system manually. The e-mails stored here can also be deleted, e.g. spam e-mails.

IMPORTANT INFORMATION

In ConSol*CM version 6.9.4, there are two modes to receive incoming e-mails:

- **Mule/ESB** - this has also been available in all previous CM versions
- **NIMH** (New Incoming Mail Handler) - new in version 6.9.4

For all configurations/settings which are valid for both modes, no further notes are added. For all settings which vary depending on the mode, this will be explained in separate (i.e. Mule/ESB or NIMH specific) sections.

22.5.2 E-Mail Backups in the Admin Tool

All e-mails which could not be processed are listed in the *Backup e-mails* panel. If your system is running in ESB/Mule Mail mode, the ESB/Mule backup e-mails are displayed, if the system is running in NIMH mode, the NIMH backup e-mails are displayed. Backups of both types are never ⚠ displayed at the same time!

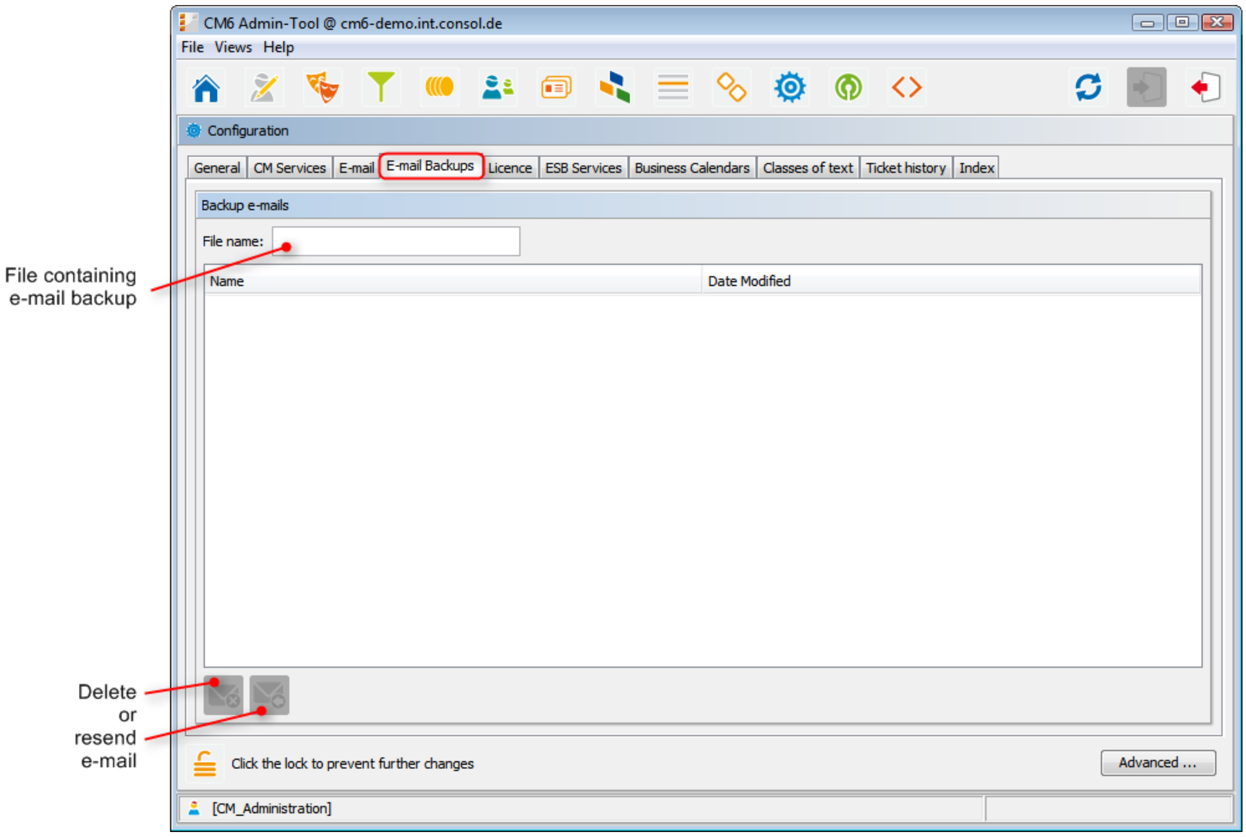


Fig. 1: ConSol* Admin Tool - Configuration: E-mail Backups

The list panel for unparsable e-mails contains the following elements:

- **File name**
  This field provides a filter. When you enter the name or part of the name of e-mail files, only the matching file names will be displayed in the list.
- **Name**
  The name of the e-mail file (usually with an *.eml* extension).
- **Date modified**
  The last date when the file was modified. Usually the date when the e-mail has been stored on the CM server.

In an error-free ConSol*CM6 system, the list panel for unparsable e-mails should be empty. In case there are e-mails listed, an error has occurred concerning the processing of the incoming e-mail(s) in the system. Please see section E-Mail Scripts for a detailed explanation of the processing pipeline.

To delete an e-mail from the list, select the list entry and press the *Delete* button . Please keep in mind that the information will be lost! It will not be saved or transferred to CM in any way!

You can also try to re-send the e-mail to the processing pipeline (e.g. when a script was not working correctly and has been fixed now) by selecting it in the list and by pressing the *Resend* button .

22.5.3 Management of E-Mail Backups with ESB/Mule

Incoming e-mails which could not be processed are stored in the file system, in the following directory (as *.eml* files):

```
 <CMAS_DATADIR>/mail/unparsable
```

E-Mails which were stored in the *unparsable* directory and were re-sent successfully are transferred from the *unparsable* directory to the following directory (as *.eml* file):

```
<CMAS_DATADIR>/mail/reimported
```

22.5.4 Management of E-Mail Backups with NIMH

Incoming e-mails which could not be processed are stored in the CM database, in the following table (as *.eml* files):

```
cmas_nimh_archived_mail
```

22.6 Tab Licence

- General Information about Licenses in ConSol*CM
- Managing the ConSol*CM License Using the Admin Tool

22.6.1 General Information about Licenses in ConSol*CM

A ConSol*CM license file 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 Web Client, REST section. Ten licenses have been purchased.

```
[CONCURRENT_USERS]
contractParty = Demo-Licence ConSol
products = WEB_CLIENT,REST
version = 6.9
expirationDate = 31.12.2014
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 Engineer Administration) 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*, Appendix C (System Properties)).

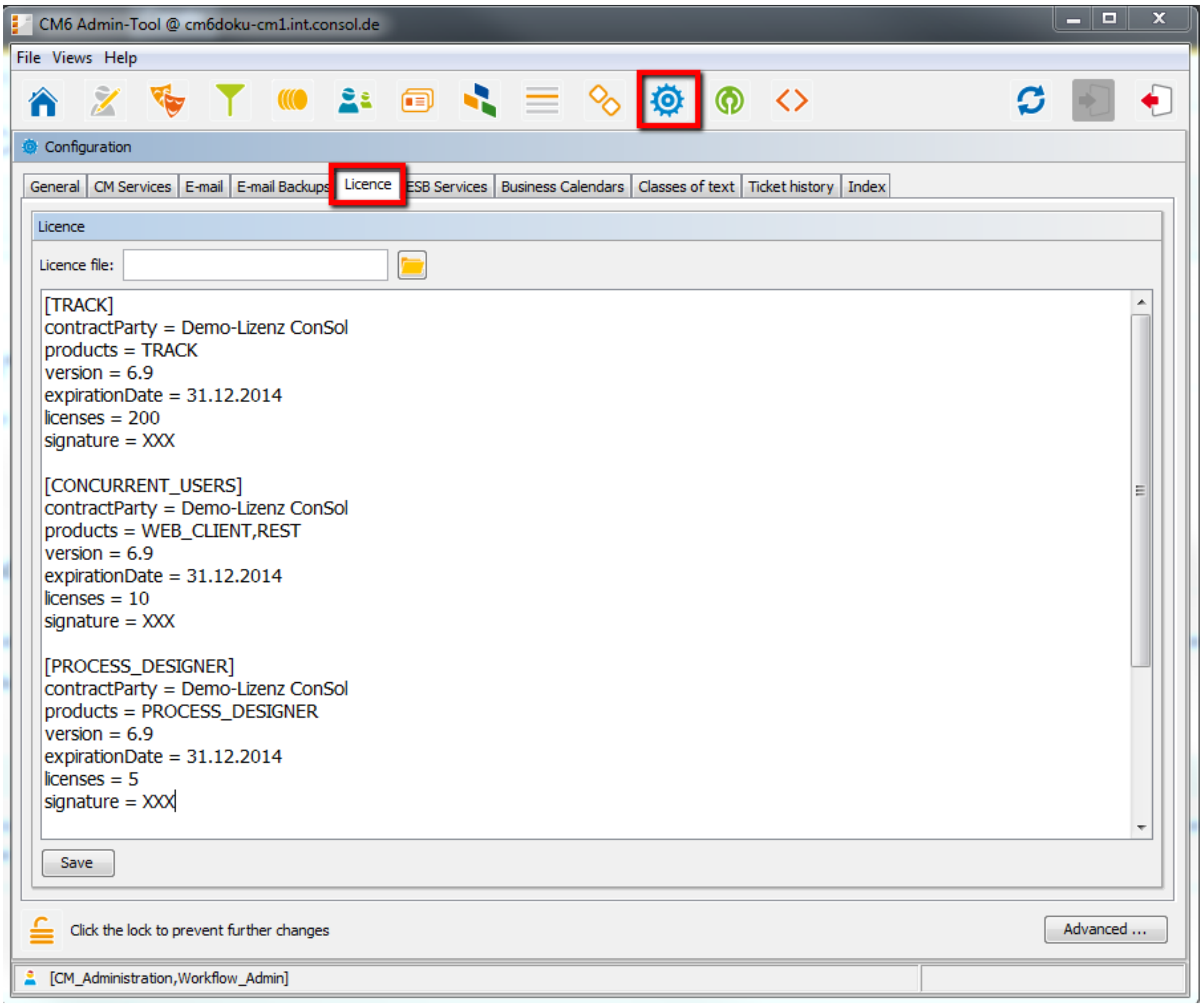
22.6.2 Managing the ConSol*CM License Using the Admin Tool

Here you have to import a valid license for your ConSol*CM system. You will receive a license for a test and /or a productive system when you have signed the software contracts with ConSol*.

Please ask your consultant for details. The license is a plain text file.

Attention:

There is no *Back* button to undo changes with one click when you enter or delete text in the *Licence* field. If you accidentally changed 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.



```
[TRACK]
contractParty = Demo-Lizenz ConSol
products = TRACK
version = 6.9
expirationDate = 31.12.2014
licenses = 200
signature = XXX

[CONCURRENT_USERS]
contractParty = Demo-Lizenz ConSol
products = WEB_CLIENT,REST
version = 6.9
expirationDate = 31.12.2014
licenses = 10
signature = XXX

[PROCESS_DESIGNER]
contractParty = Demo-Lizenz ConSol
products = PROCESS_DESIGNER
version = 6.9
expirationDate = 31.12.2014
licenses = 5
signature = XXX
```

Fig. 1: ConSol*CM Admin Tool - Configuration: Licence

Choose one of the two ways to import your ConSol*CM license file. In either case you have to unlock the editor panel first.

- Insert the entire text of the license file by copy and paste. In case an old license is present, just replace the entire text. Click on *Save*.
- Load the license using the file browser next to the field *Licence file*. Click on *Save*.

You should receive a message that the license has been imported into the system successfully. It is in operation at once, without further action.

22.7 ConSol*CM ESB Services

- Introduction to ESB Services
- Starting and Stopping ESB Services Using the Admin Tool

IMPORTANT INFORMATION

In ConSol*CM version 6.9.4, there are two modes to receive incoming e-mails:

- **Mule/ESB** - this has also been available in all previous CM versions
- **NIMH** (New Incoming Mail Handler) - new in version 6.9.4

For all configurations/settings which are valid for both modes, no further notes are added. For all settings which vary depending on the mode, this will be explained in separate (i.e. Mule/ESB or NIMH specific) sections.

The ESB services are only active if ESB/Mule Mail is active! If your systems runs in NIMH mode, the tab *ESB Services* is disabled!

22.7.1 Introduction to ESB Services

The ESB services are in operation for incoming e-mails. Please see the following figure for the functions of the ESB services and also section E-Mail Scripts for a detailed description of the general principle of ConSol*CM mailing.

ESB stands for *Enterprise Service Bus* and ConSol*CM has integrated an ESB (Mule ESB^{TM}) as one of the application modules.

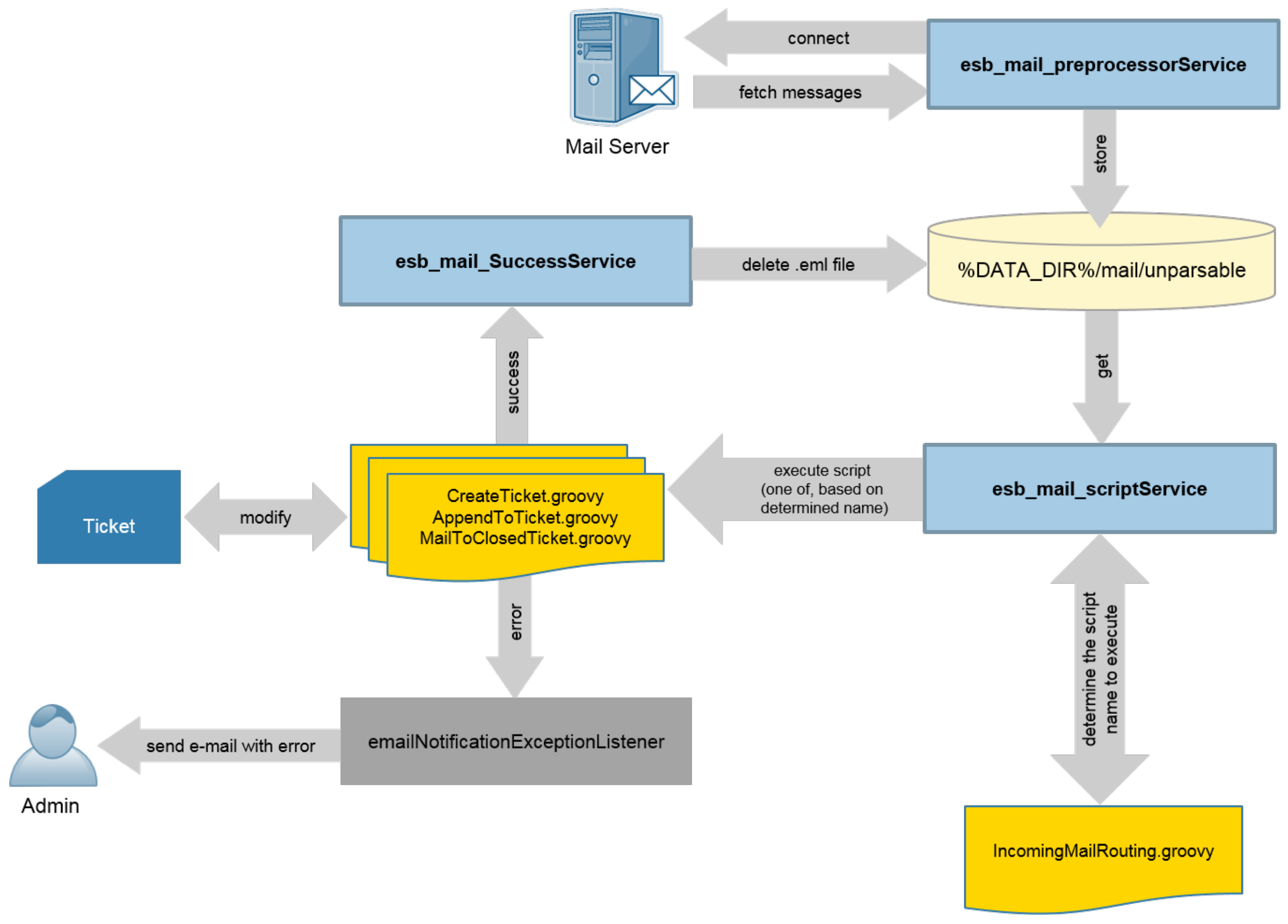

Mail Server
connect
fetch messages
esb_mail_preprocessorService
store
%DATA_DIR%/mail/unparsable
delete .eml file
esb_mail_SuccessService
get
esb_mail_scriptService
execute script (one of, based on determined name)
success
CreateTicket.groovy
AppendToTicket.groovy
MailToClosedTicket.groovy
modify
Ticket
error
emailNotificationExceptionListener
send e-mail with error
Admin
determine the script name to execute
IncomingMailRouting.groovy

Fig. 1: ConSol*CM ESB Services

ESB services:

- **esb_mail_preprocessorService**
  Responsible for fetching e-mail messages from the configured incoming mailboxes. Retrieved messages are stored in the directory *%DATA_DIR%/mail/unparsable* as *.eml* files. Stopping this service will cause the ConSol*CM server to disconnect from configured e-mail servers. This means that e-mails will not be fetched and initially processed. After starting this service again, ConSol*CM will connect to configured e-mail servers and process all queued messages.

- **esb_mail_scriptService**
  This service calls the *IncomingMailRouting.groovy* script to determine the script name to execute. It can be one of *CreateTicket.groovy*, *AppendToTicket.groovy*, or *MailToClosedTicket.groovy*. Then the determined script is executed. On success, the *esb_mail_SuccessService* is called. On error, an e-mail with detailed cause is sent to the administrator. When this service is stopped e-mail messages will be retrieved from the mailboxes and stored in the directory *%DATA_DIR%/mail/unparsable*. Then the processing will stop. After the service is started again, the messages will be picked up from the *unparsable* directory and processed.

- **esb_mail_SuccessService**
  Responsible for deleting e-mail files that were processed correctly from the backup folder. Stopping this service will cause e-mail copies to remain in the backup folder (*%DATA_DIR%/mail/unparsable*) after processing.

  Warning:

  When this service (*esb_mail_SuccessService*) is started again, it will delete all messages which were not removed when it was stopped.

22.7.2 Starting and Stopping ESB Services Using the Admin Tool

In this tab you can start and stop the sub-services of the *Enterprise Service Bus* (ESB). You should only change the service status upon request of CM consulting or CM support!

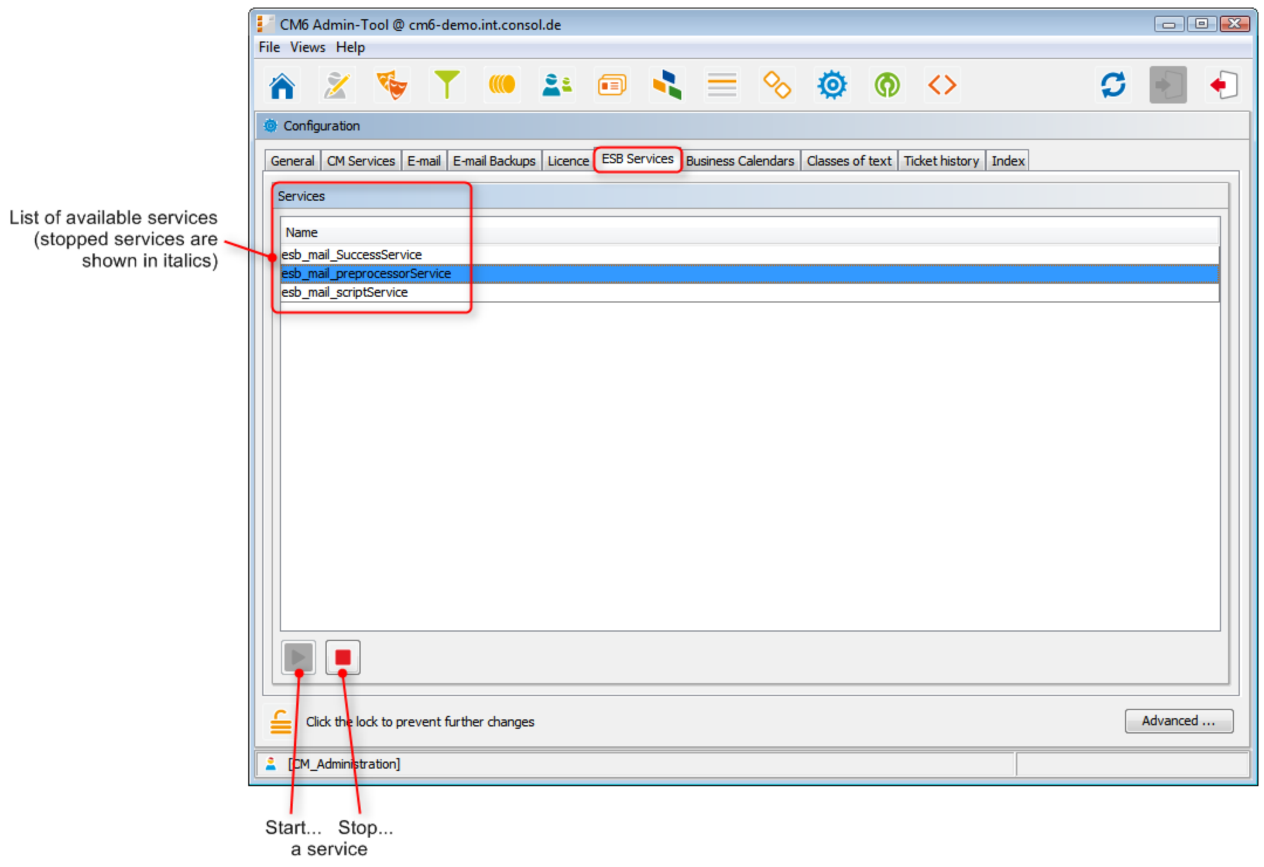


Fig. 2: ConSol*CM Admin Tool - Configuration: ESB Services

22.8 Tab Business Calendars

- Configuration of Business Calendars in the Admin Tool
    - Creating a New Calendar
    - Defining the Working Hours for a Calendar
    - Defining the Holidays for a Calendar
        - Defining the Holidays for a Calendar Manually
        - Importing the Holidays for a Calendar from an Excel File

On this tab you can create and manage business calendars. These define times when automatic workflow actions shall be active.

Example:

Tickets which have not been assigned to an engineer more than one hour after opening shall be 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.

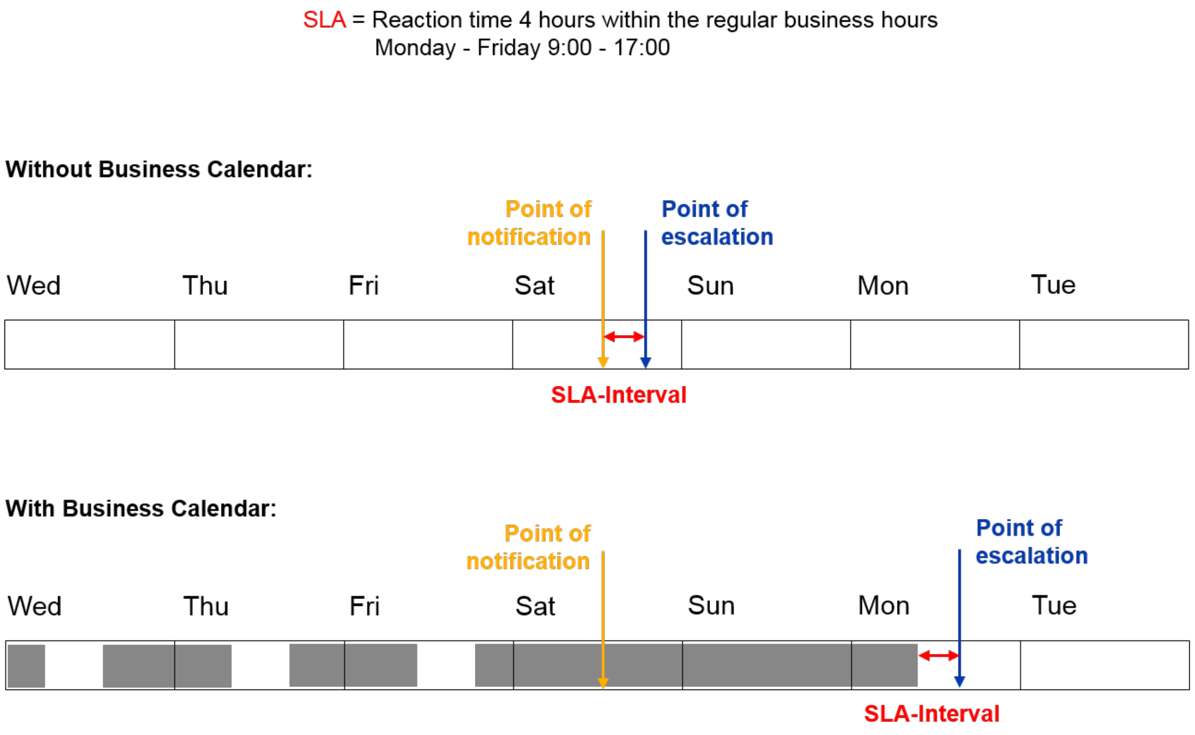


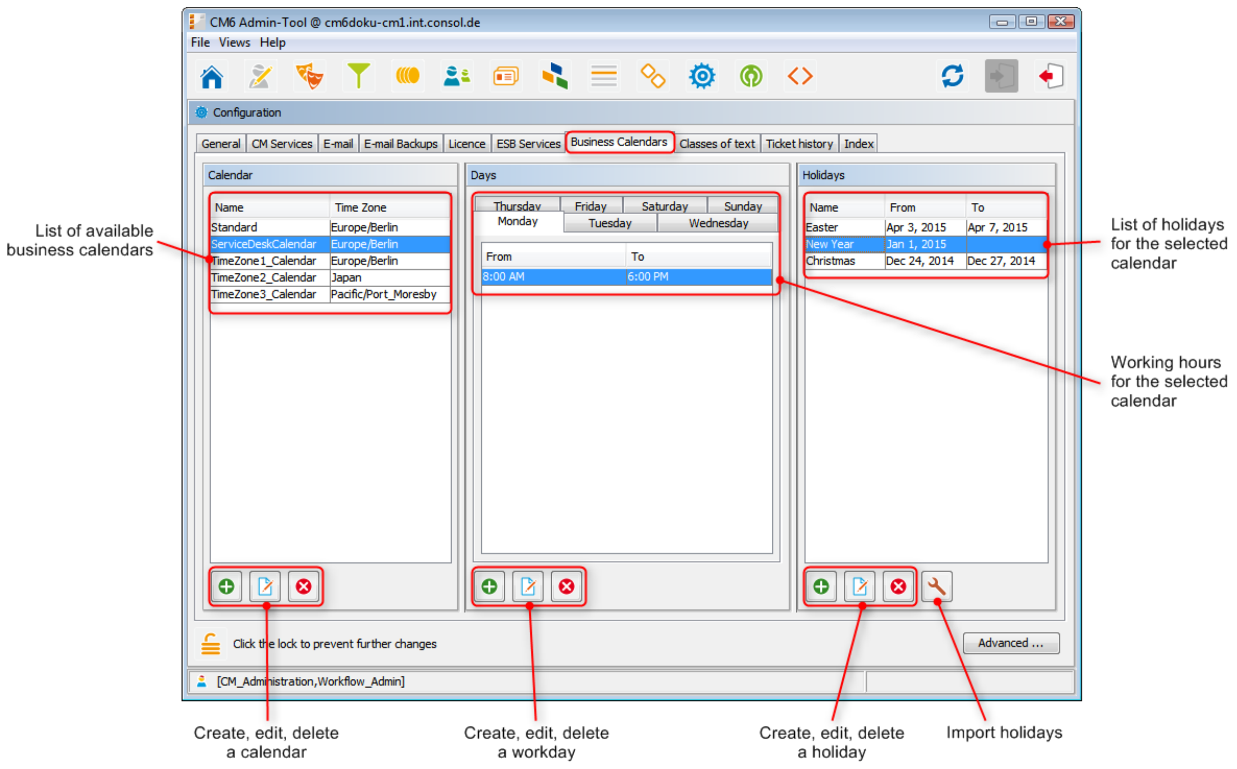
Fig. 1: ConSol*CM Principle - Business Calendar

Besides 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 Queue Administration (Admin Tool)
- assign the use of a calendar to every single workflow element where the calendar should be the basis for time calculations (Process Designer), this is explained in the *ConSol*CM Process Designer Manual*.

22.8.1 Configuration of Business Calendars in the Admin Tool


CM6 Admin-Tool @ cm6doku-cm1.int.consol.de
File Views Help
Configuration
General | CM Services | E-mail | E-mail Backups | Licence | ESB Services | Business Calendars | Classes of text | Ticket history | Index
Calendar
Days
Holidays
Thursday Friday Saturday Sunday Monday Tuesday Wednesday
Click the lock to prevent further changes
Advanced ...
[CM_Administration,Workflow_Admin]
List of available business calendars
List of holidays for the selected calendar
Working hours for the selected calendar
Create, edit, delete a calendar
Create, edit, delete a workday
Create, edit, delete a holiday
Import holidays

Name	Time Zone
Standard	Europe/Berlin
ServiceDeskCalendar	Europe/Berlin
TimeZone1_Calendar	Europe/Berlin
TimeZone2_Calendar	Japan
TimeZone3_Calendar	Pacific/Port_Moresby

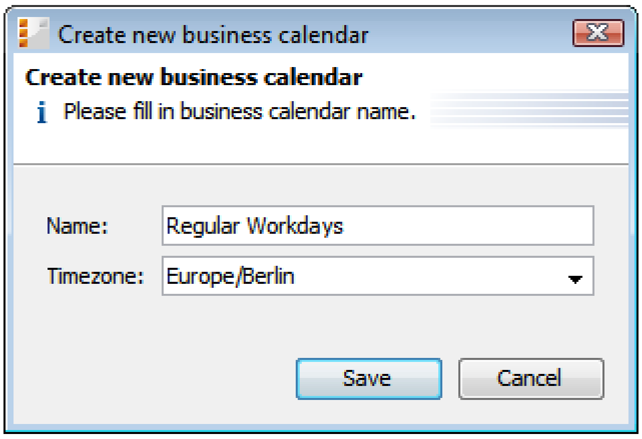
From	To
8:00 AM	6:00 PM

Name	From	To
Easter	Apr 3, 2015	Apr 7, 2015
New Year	Jan 1, 2015	
Christmas	Dec 24, 2014	Dec 27, 2014

Fig. 2: ConSol*CM Admin Tool - Configuration: Business Calendars

Creating a New Calendar

Click on ⊕ in the left part of the page to create a new calendar. The following window appears:


Create new business calendar
Create new business calendar
Please fill in business calendar name.
Save
Cancel

Name: Regular Workdays
Timezone: Europe/Berlin

Fig. 3: ConSol*CM Admin Tool - Configuration: New Calendar

- **Name:**
  Enter a unique name for the calendar.
- **Timezone:**
  Choose the time zone to be used for the calendar.

> **Attention:**
>
> 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 timezone. A time trigger which uses the business trigger would fire according to the Berlin time and not the Detroit time.

Click on *Save* afterwards to create the calendar.

Clicking on you can modify a selected calendar in the same way. Click on , if you want to delete the selected calendar.

Defining the Working Hours for a Calendar

Select a calendar on the left and click on in the middle part of the page to create the days and hours for this calendar. The following window appears:

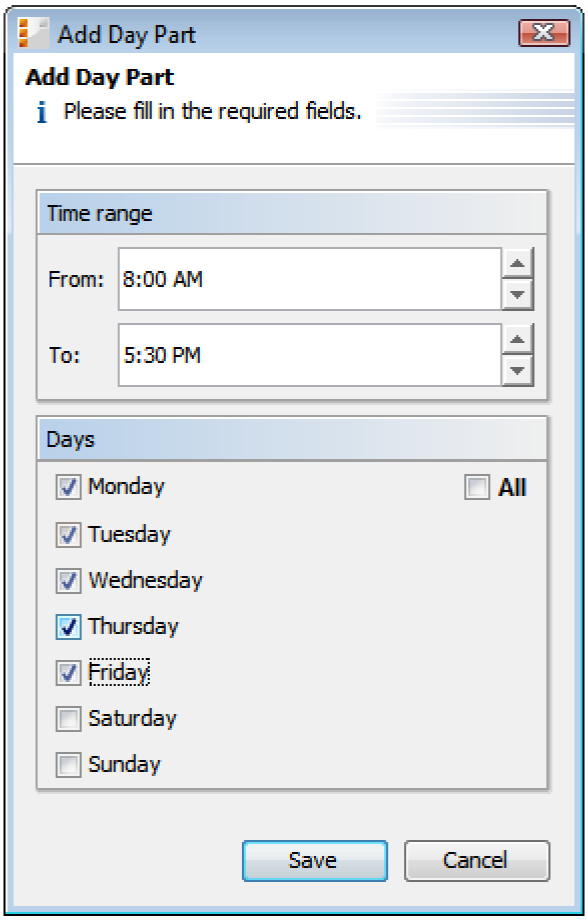


Fig. 4: ConSol*CM Admin Tool - Configuration: Working Hours of a Calendar

- **Time range**
  Enter the time range for which the automatic workflow escalations shall be active.
- **Days**
  Mark the check boxes of the days for which the time range shall be valid. It is possible to choose individual or all days (check box *All*).

> **Information:**
>
> In case, the system detects an inconsistency of the time you define here with an already existing time, you will get a corresponding message.

Click on *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 on and change the time range in the window that appears. Or click on if you want 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.

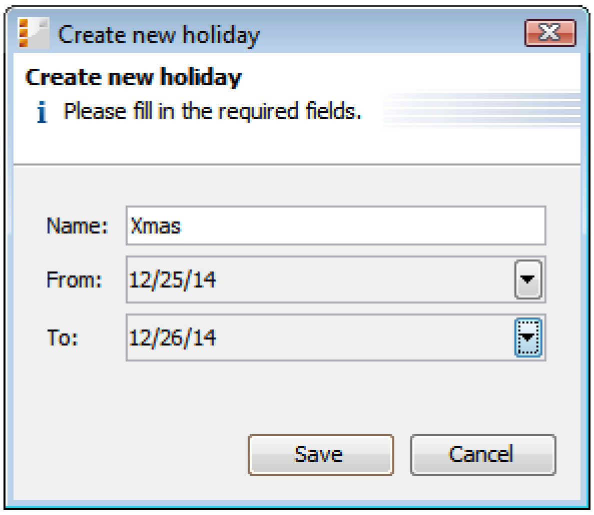
Defining the Holidays for a Calendar

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 on in the right part of the page to create a new holiday entry. The following window appears:



Name: Xmas
From: 12/25/14
To: 12/26/14

Fig. 5: ConSol*CM Admin Tool - Configuration: Holidays of a Calendar

- **Name:**
  Enter the name of the holiday here.
- **From:**
  Enter the date of the holiday in this field.
- **To:**
  If it is a multi-day holiday (e.g. Christmas), you can enter the last date of the holiday here.

> **Information:**
>
> It is not possible to define holidays that last only half a day.

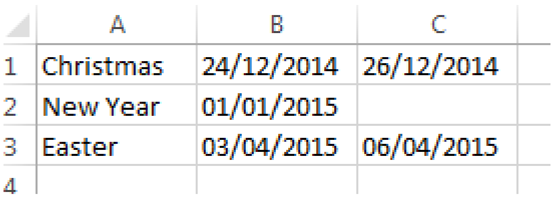
Click on *Save* afterwards to create the holiday.

If you want to edit a selected holiday entry just click on . Clicking on deletes an entry.

Importing the Holidays for a Calendar from an Excel File

Holiday data can be imported from an Microsoft Excel file which is based on the following format:

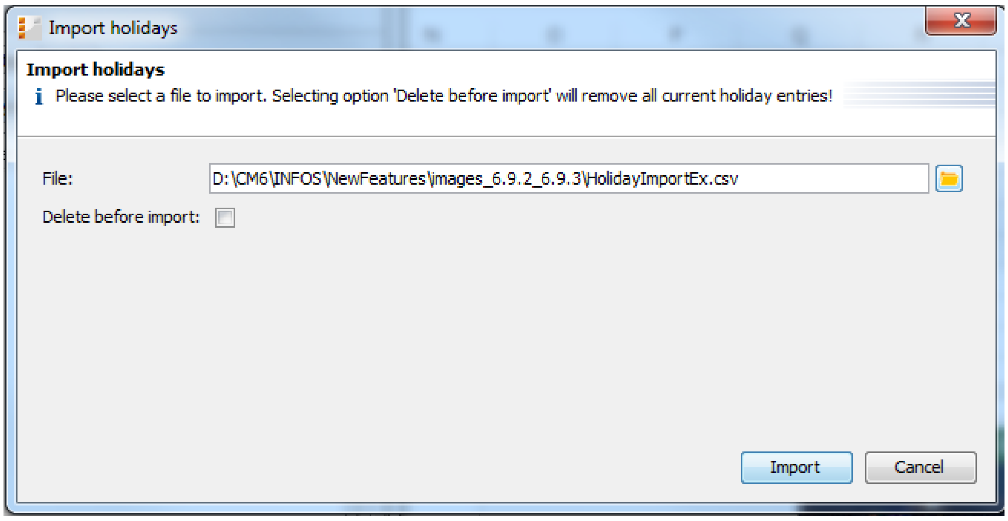
- First column: title/name of the holiday
- Second column: start date
- Third column: end date (or empty when it's only one day)



	A	B	C
1	Christmas	24/12/2014	26/12/2014
2	New Year	01/01/2015	
3	Easter	03/04/2015	06/04/2015
4			

Fig. 6: Excel File for Holiday Import

In the Admin Tool, tab *Configuration* - *Business Calendars*, select a calendar and click on *Import holidays* and enter the path for the Excel import file.



Import holidays

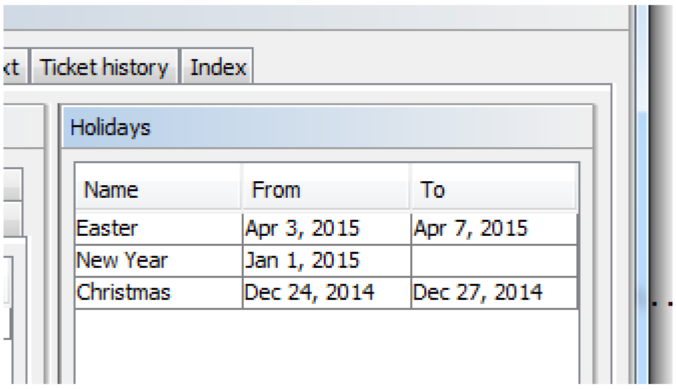
Please select a file to import. Selecting option 'Delete before import' will remove all current holiday entries!

File: D:\CM6\INFOS\NewFeatures\images_6.9.2_6.9.3\HolidayImportEx.csv

Delete before import: ☐

Fig. 7: Consol*CM Admin Tool - Configuration: Importing Holidays

The new holidays will be imported in the holiday list of the selected business calendar.



Holidays

Name	From	To
Easter	Apr 3, 2015	Apr 7, 2015
New Year	Jan 1, 2015	
Christmas	Dec 24, 2014	Dec 27, 2014

Fig. 8: ConSol*CM Admin Tool - Configuration: Newly Imported Holidays

22.9 Tab Classes of Text

- Installing a New Class of Text
    - Defining a Class of Text
    - Assigning the Class of Text to a Queue
- Edit a Class of Text
- Delete a Class of Text
- Setting the Default Class of Text
- Working With Classes of Text in Scripts

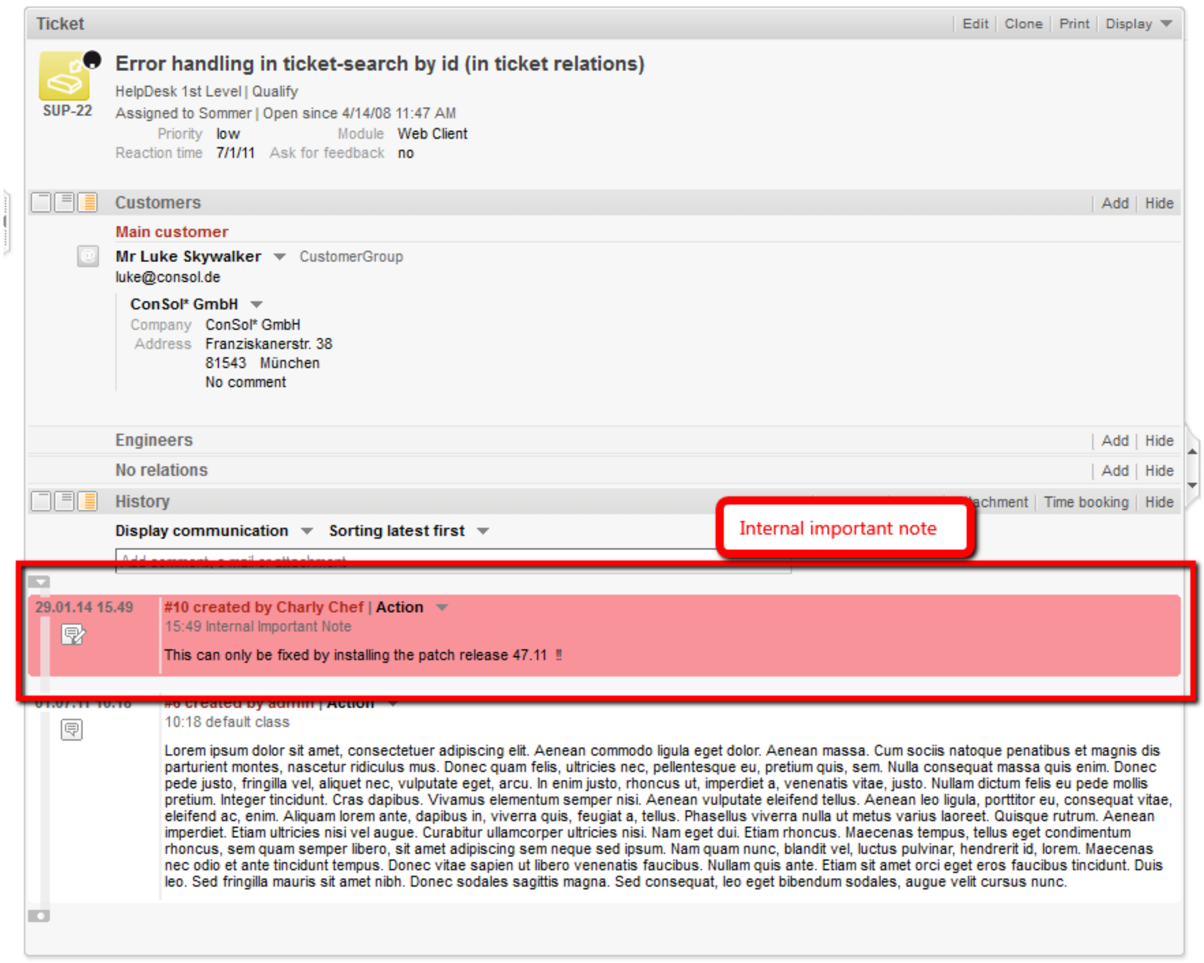
A class of text is a class that you assign to a ticket entry. This entry can be:

- a comment
- an e-mail that was sent from the ticket
- an e-mail that was received in the ticket
- an attachment

Assigning a class of text can have one or more of the following purposes:

- Highlighting the text in the ticket with a special color to make it easier to find it (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 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-over 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.



Error handling in ticket-search by id (in ticket relations)

SUP-22

HelpDesk 1st Level | Qualify

Assigned to Sommer | Open since 4/14/08 11:47 AM

Priority low Module Web Client

Reaction time 7/1/11 Ask for feedback no

Main customer

Mr Luke Skywalker CustomerGroup

luke@consol.de

ConSol* GmbH

Company ConSol* GmbH

Address Franziskanerstr. 38

81543 München

No comment

29.01.14 15.49

#10 created by Charly Chef | Action

15:49 Internal Important Note

This can only be fixed by installing the patch release 47.11 ‼

01.07.11 10:18

#6 created by admin | Action

10:18 default class

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies nisi. Nam eget dui. Etiam rhoncus. Maecenas tempus, tellus eget condimentum rhoncus, sem quam semper libero, sit amet adipiscing sem neque sed ipsum. Nam quam nunc, blandit vel, luctus pulvinar, hendrerit id, lorem. Maecenas nec odio et ante tincidunt tempus. Donec vitae sapien ut libero venenatis faucibus. Nullam quis ante. Etiam sit amet orci eget eros faucibus tincidunt. Duis leo. Sed fringilla mauris sit amet nibh. Donec sodales sagittis magna. Sed consequat, leo eget bibendum sodales, augue velit cursus nunc.

Fig. 1: ConSol*CM Web Client - Using a Class of Text for an Internal Important Note

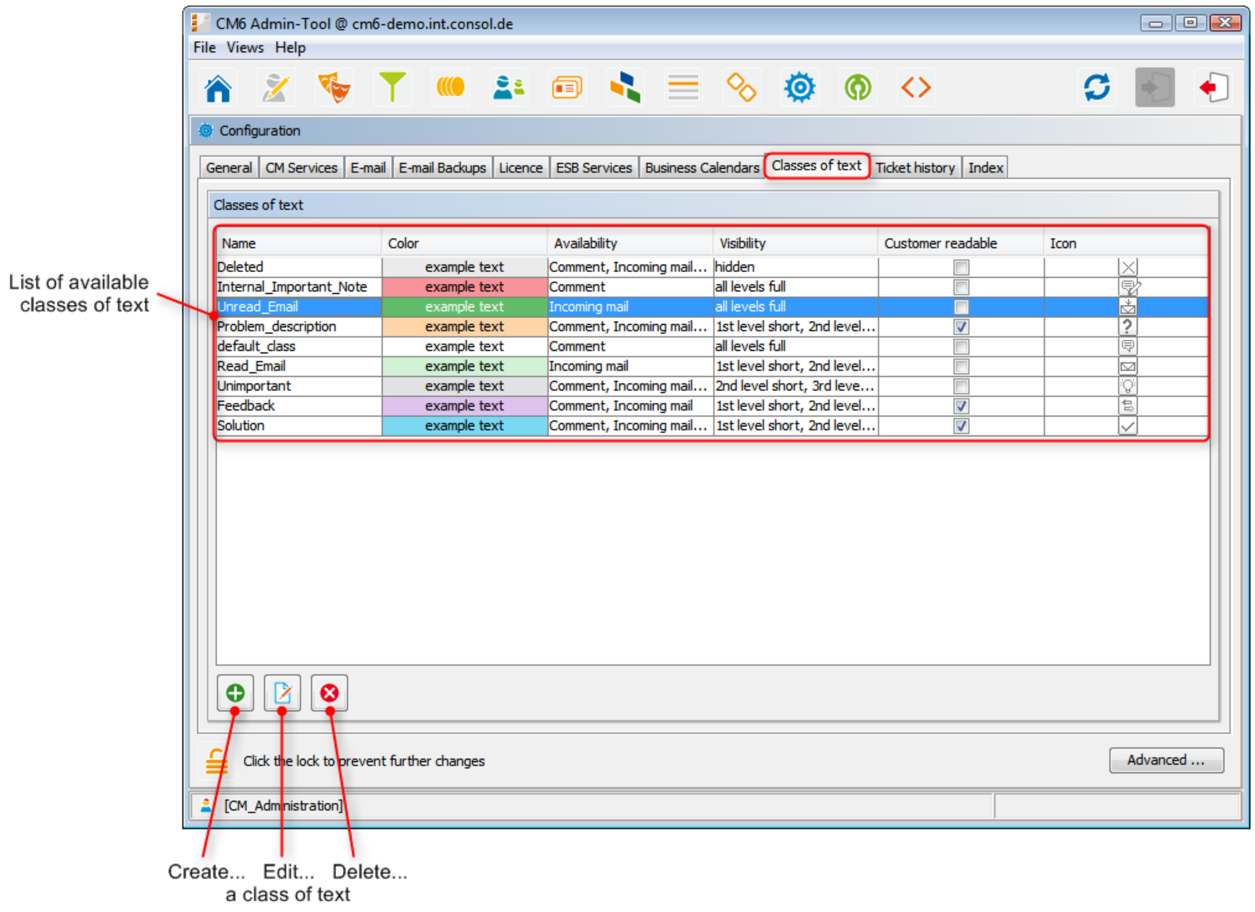
22.9.1 Installing a New Class of Text

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 *Classes of text* tab.
2. Assigning the class of text to the queue where it should be available for tickets (see section Queue Administration for more information).

Defining a Class of Text

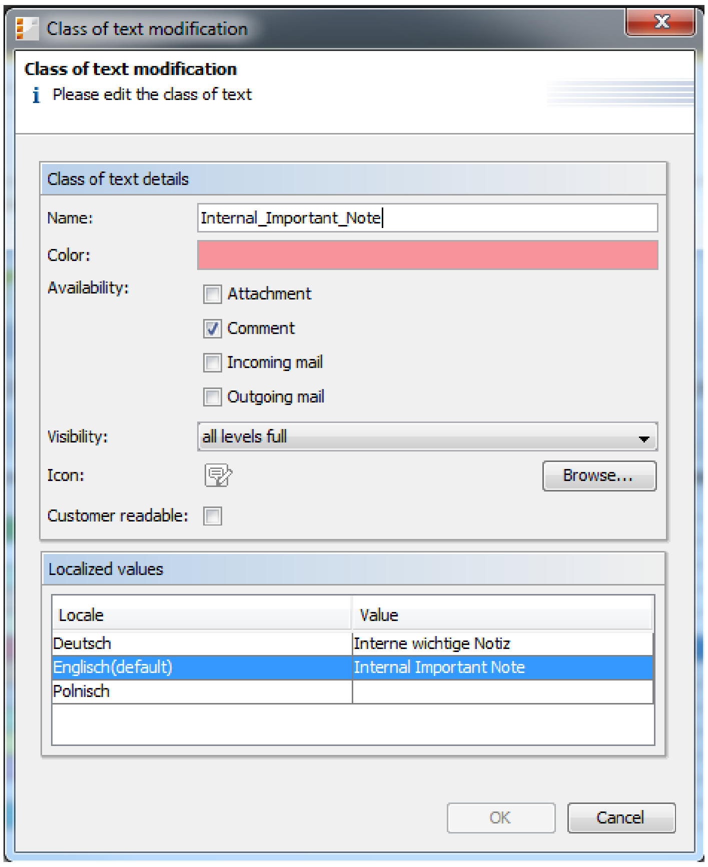
Classes of text are defined and managed in the corresponding tab in the Admin Tool (see following figure).



Name	Color	Availability	Visibility	Customer readable	Icon
Deleted	example text	Comment, Incoming mail…	hidden	☐	✕
Internal_Important_Note	example text	Comment	all levels full	☐	✎
Unread_Email	example text	Incoming mail	all levels full	☐	📥
Problem_description	example text	Comment, Incoming mail…	1st level short, 2nd level…	☑	?
default_class	example text	Comment	all levels full	☐	💬
Read_Email	example text	Incoming mail	1st level short, 2nd level…	☐	✉
Unimportant	example text	Comment, Incoming mail…	2nd level short, 3rd leve…	☐	💡
Feedback	example text	Comment, Incoming mail	1st level short, 2nd level…	☑	⇆
Solution	example text	Comment, Incoming mail…	1st level short, 2nd level…	☑	✓

Fig. 2: ConSol*CM Admin Tool - Configuration: Classes of Text

You define a new class of text by clicking on ⊕ below the list. The following pop-up window appears:


Class of text modification
Class of text modification
Please edit the class of text
Class of text details

Name: Internal_Important_Note
Color:
Availability: [ ] Attachment [✓ Comment] [ ] Incoming mail [ ] Outgoing mail
Visibility: all levels full
Icon:
Customer readable: [ ]

Browse...
Localized values

Locale	Value
Deutsch	Interne wichtige Notiz
Englisch(default)	Internal Important Note
Polnisch	

Fig. 3: ConSol*CM Admin Tool - New Class of Text

Here, you have to define the class details:

- **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 by clicking on it. You can check the selected color in the *Preview* area. Click on *OK* to save your choice. Click on *Reset* if you want to return to the last saved color.

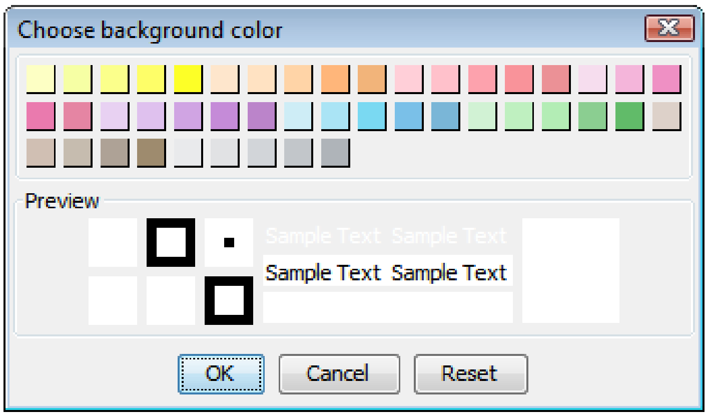


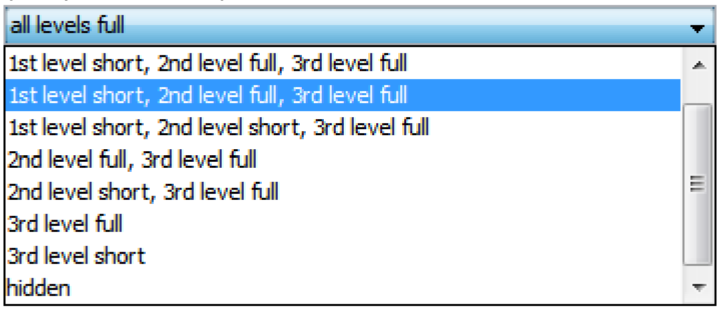
Fig. 4: ConSol*CM Admin Tool - Choose a Color for the Text Class

- **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**
  There are three ticket history levels History in the Web Client:
  - Basic (1st level)
  - Extended (2nd level)
  - Detail (3rd level)

  The terms *short* and *full* refer to the display mode the user has chosen:
  - short - *communication*
  - full - *display all entries*

  Select in the drop-down menu on which history levels the marked ticket information shall be visible (see picture below).



all levels full
1st level short, 2nd level full, 3rd level full
1st level short, 2nd level full, 3rd level full
1st level short, 2nd level short, 3rd level full
2nd level full, 3rd level full
2nd level short, 3rd level full
3rd level full
3rd level short
hidden

Fig. 5: ConSol*CM Admin Tool - Choose a Visibility Level

If you choose *hidden*, the marked ticket information will not be visible in the ticket history.

- **Icon**
  When you click into the box next to *Icon* you will get a selection of standard CM icons. Select one of these icons for the new class of text or load your own individual icon by clicking on the *Browse...* button.

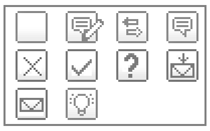


Fig. 6: ConSol*CM Admin Tool - Choose an Icon for the Text Class

- **Customer readable**
  Select this check box if ticket information marked with this class of text shall be visible for customers in CM.Track, the 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. In the Web Client the name will be displayed in the respective language of the locale of the web browser. If you do not make an entry here the object name, i.e. the content of the *Name* field, will be taken instead.

Click on OK to save the details of the new class of text and to close the window.

Assigning the Class of Text to a Queue

After assigning the class of text to a queue within the Queue Administration it will be available for tickets of this queue in the Web Client.

22.9.2 Edit a Class of Text

If you want to edit a class of text, select it in the list and click on . The same window as described above for creating a class will appear. You can modify all details and save your changes by clicking *OK*.

22.9.3 Delete 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 select it in the list and click on . If you confirm the following dialog with *Yes*, the class will be removed from the list and the system.

22.9.4 Setting the Default Class of Text

To define the default class of text, use the system property *cmweb-server-adapter*, *defaultContentEntryClassName* (see Appendix C). 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.

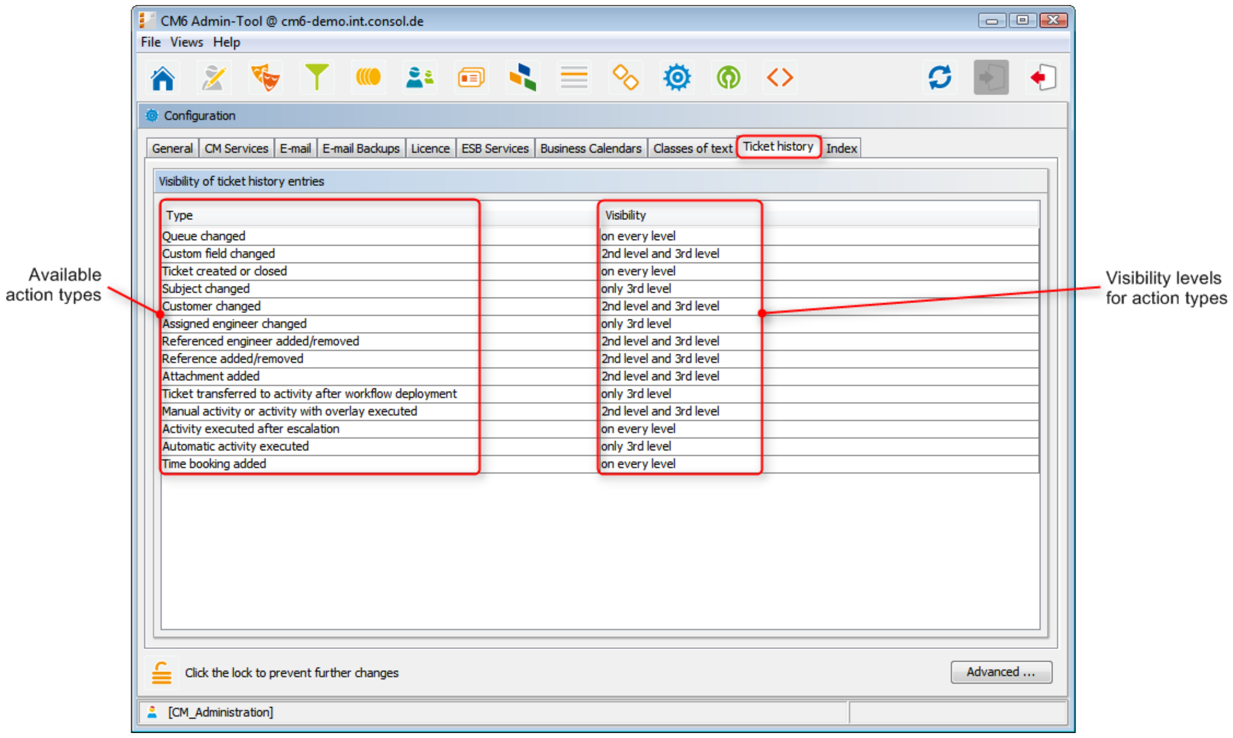
22.9.5 Working With Classes of Text in Scripts

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 when the class of text is not assigned to the queue! This can help automate processes.

22.10 Tab Ticket History

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



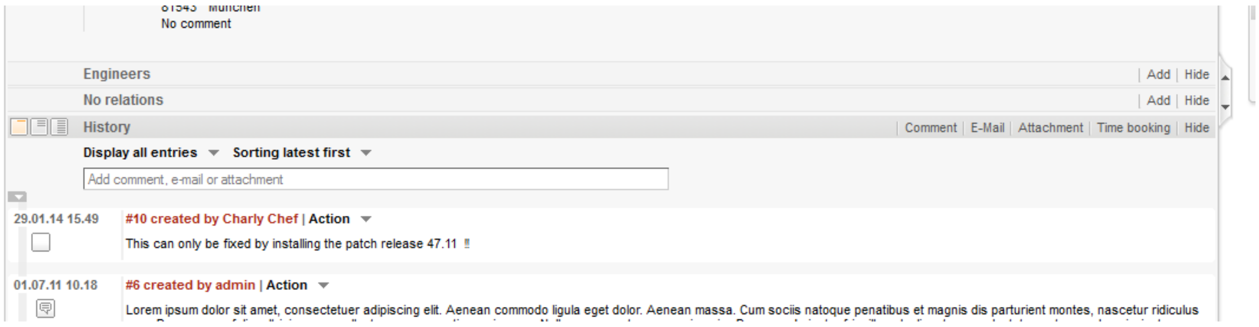
Type	Visibility
Queue changed	on every level
Custom field changed	2nd level and 3rd level
Ticket created or closed	on every level
Subject changed	only 3rd level
Customer changed	2nd level and 3rd level
Assigned engineer changed	only 3rd level
Referenced engineer added/removed	2nd level and 3rd level
Reference added/removed	2nd level and 3rd level
Attachment added	2nd level and 3rd level
Ticket transferred to activity after workflow deployment	only 3rd level
Manual activity or activity with overlay executed	2nd level and 3rd level
Activity executed after escalation	on every level
Automatic activity executed	only 3rd level
Time booking added	on every level

Fig. 1: ConSol*CM Admin Tool - Configuration: Ticket History

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

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

The following figures show the action type *time booking added* configured for *2nd level and 3rd level*.

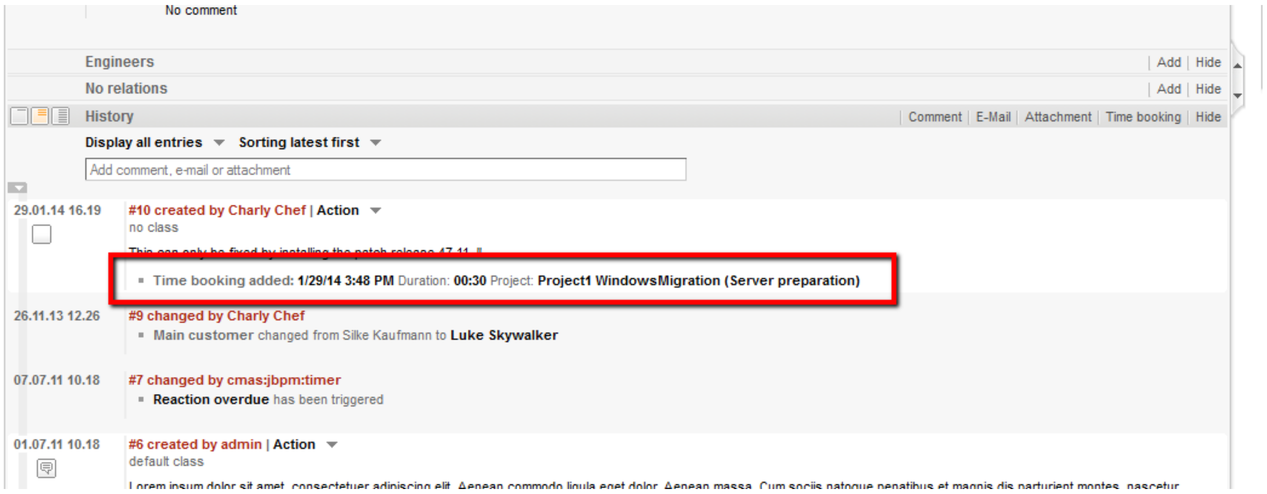

No comment

Engineers | Add | Hide
No relations | Add | Hide
History | Comment | E-Mail | Attachment | Time booking | Hide
Display all entries ▼ Sorting latest first ▼
Add comment, e-mail or attachment

29.01.14 15.49 #10 created by Charly Chef | Action ▼
This can only be fixed by installing the patch release 47.11

01.07.11 10.18 #6 created by admin | Action ▼
Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus

Fig. 2: ConSol*CM Web Client - Time Booking Entry Not Visible on 1st Level


No comment

Engineers | Add | Hide
No relations | Add | Hide
History | Comment | E-Mail | Attachment | Time booking | Hide
Display all entries ▼ Sorting latest first ▼
Add comment, e-mail or attachment

29.01.14 16.19 #10 created by Charly Chef | Action ▼
no class
▪ Time booking added: 1/29/14 3:48 PM Duration: 00:30 Project: Project1 WindowsMigration (Server preparation)

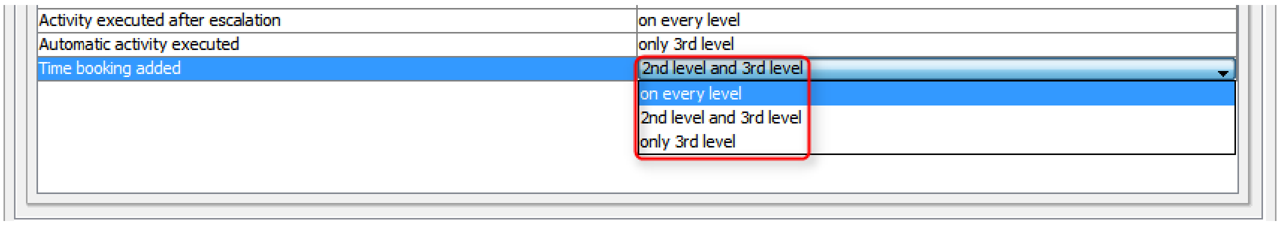
26.11.13 12.26 #9 changed by Charly Chef
▪ Main customer changed from Silke Kaufmann to Luke Skywalker

07.07.11 10.18 #7 changed by cmas:jbpm:timer
▪ Reaction overdue has been triggered

01.07.11 10.18 #6 created by admin | Action ▼
default class

Fig. 3: ConSol*CM Web Client - Time Booking Entry Visible on 2nd Level

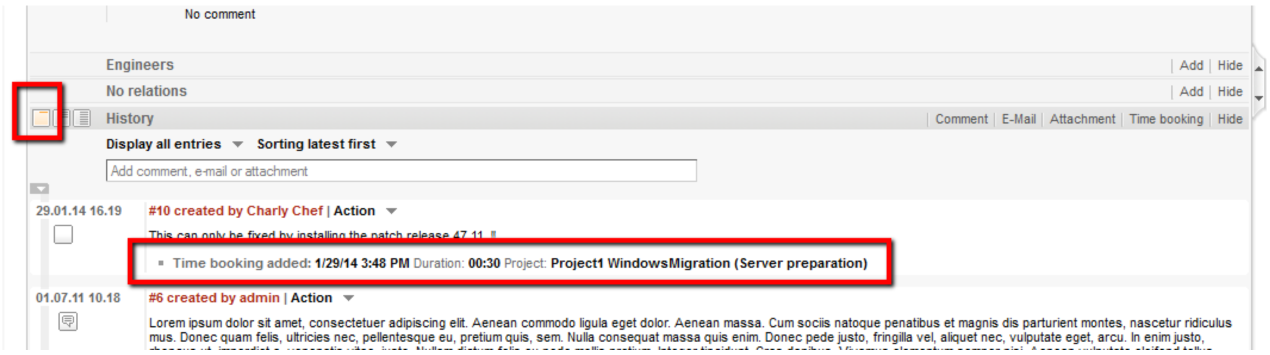
It is not possible to add new action types to the list. To edit the visibility for an existing entry, double-click on the visibility value you would like to modify and select the desired option from the drop-down menu.


Activity executed after escalation	on every level
Automatic activity executed	only 3rd level
Time booking added	2nd level and 3rd level

Drop-down: on every level / 2nd level and 3rd level / only 3rd level

Fig. 4: ConSol*CM Admin Tool - Selecting 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 in the tab *Ticket history* to be *on every level*:


No comment

Engineers | Add | Hide
No relations | Add | Hide
History | Comment | E-Mail | Attachment | Time booking | Hide
Display all entries ▼ Sorting latest first ▼
Add comment, e-mail or attachment

29.01.14 16.19 #10 created by Charly Chef | Action ▼
▪ Time booking added: 1/29/14 3:48 PM Duration: 00:30 Project: Project1 WindowsMigration (Server preparation)

01.07.11 10.18 #6 created by admin | Action ▼
Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo,

Fig. 5: ConSol*CM Web Client - Time Booking Entry Visible on 1st Level

22.11 Search Configuration and Indexer Management (Tab Index)

- Search Modes
    - Quick Search
    - Detailed Search
    - Search Using Intelligent (Autocomplete) Fields
- Fields which can be Searched
    - Ticket
        - Web Client Detailed Search
        - Web Client Quick Search (Search by Pattern)
    - Unit (Company or Customer)
        - Web Client Detailed Search
        - Web Client Quick Search (API Search by Pattern):
    - Engineer
        - Web Client Detailed Search
        - Web Client Autocomplete Search (Search by Pattern)
    - Fields which are Searchable by Default
- Administrator Tasks Concerning the Indexer
- Introduction to the ConSol*CM Indexer
- Indexer and Index Management Using the Admin Tool
    - Field Indexed Annotation
    - Indexer Management: Tab Index
- Indexer and Index-Relevant System Properties
- CM Indexer Services
- Systems with more than one Indexing Server

ConSol*CM provides a powerful search for all objects involved in the business processes, e.g. customers and tickets. Technically, the search is based on the *Indexer*, a module of ConSol*CM.

The following paragraphs will 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.

22.11.1 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 GUI. The display of the results (i.e. the fields and the order of the fields in the result list) can be formatted using templates, please see section Templates for Customer Data for details. Please keep in mind that you can adapt the size of the result list using the system property *cmweb-server-adapter*, *globalSearchResultSizeLimit*, see Appendix C (System Properties) for details.

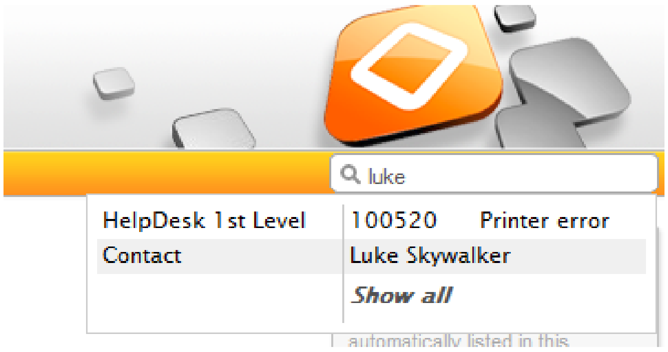
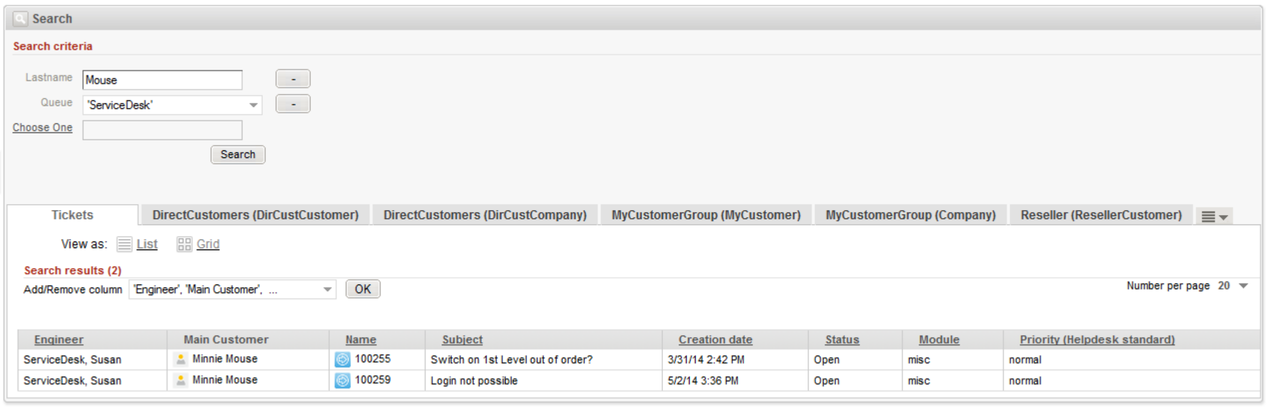


Fig. 1: ConSol*CM Web Client - Quick Search

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.


Search
Search criteria
Lastname
Mouse
Queue
'ServiceDesk'
Choose One
Search
Tickets
DirectCustomers (DirCustCustomer)
DirectCustomers (DirCustCompany)
MyCustomerGroup (MyCustomer)
MyCustomerGroup (Company)
Reseller (ResellerCustomer)
View as: List Grid
Search results (2)
Add/Remove column 'Engineer', 'Main Customer', ...
OK
Number per page 20

Engineer	Main Customer	Name	Subject	Creation date	Status	Module	Priority (Helpdesk standard)
ServiceDesk, Susan	Minnie Mouse	100255	Switch on 1st Level out of order?	3/31/14 2:42 PM	Open	misc	normal
ServiceDesk, Susan	Minnie Mouse	100259	Login not possible	5/2/14 3:36 PM	Open	misc	normal

Fig. 2: ConSol*CM Web Client - Detailed Search

Please keep in mind that the size 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*. See Appendix C (System Properties) for an explanation.

Information:

Please refer to the *ConSol*CM User Manual*, section *Searching for Tickets and Customers* to learn how to use the search functionality.

Search Using Intelligent (Autocomplete) Fields

This 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).

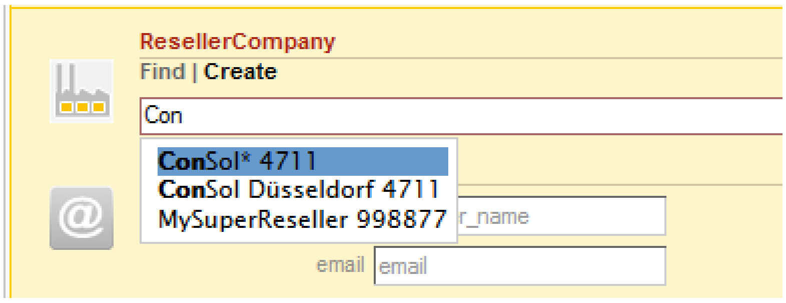

ResellerCompany
Find | Create
Con
ConSol* 4711
ConSol Düsseldorf 4711
MySuperReseller 998877
r_name
email email

Fig. 3: Search in an autocomplete field

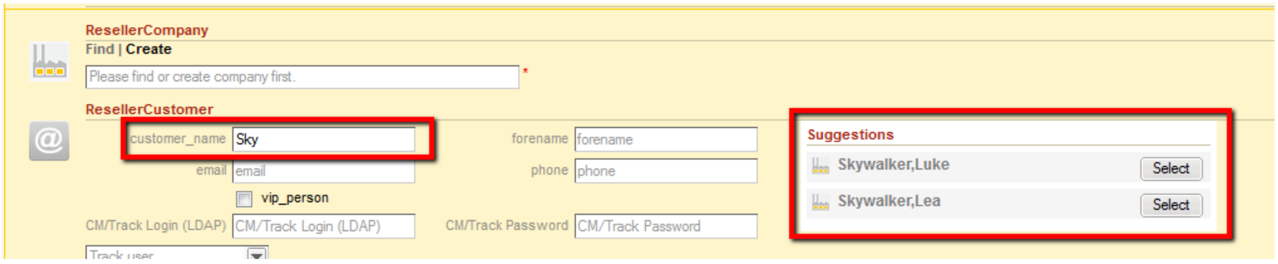

ResellerCompany
Find | Create
Please find or create company first. *
ResellerCustomer
customer_name Sky
forename forename
email email
phone phone
vip_person
CM/Track Login (LDAP) CM/Track Login (LDAP)
CM/Track Password CM/Track Password
Track user
Suggestions
Skywalker,Luke Select
Skywalker,Lea Select

Fig. 4: Suggestions for content of an autocomplete field

22.11.2 Fields which can be Searched

For custom fields and data object group fields which should be searched, the annotation *field indexed* has to be set. See section Field Indexed Annotation. This makes the field available for the Quick Search and for the Detailed Search.

Ticket

Web Client Detailed Search

- company
- creation date
- customer
- customer group
- custom field
- engineer
- pattern
- queue
- status
- subject
- scope - available when queue is set
- view

Web Client Quick Search (Search by Pattern)

- name
- subject
- localized scope name
- localized queue name
- content text
- attachment text (only if the *index.attachments* configuration property is set)
- engineer last name
- engineer first name
- referenced engineer last name
- referenced engineer first name
- referenced engineer localized ticket function name
- custom fields (annotated as *field indexed* = transitive, unit, local):
  - number fields
  - big decimal fields
  - localized enum fields
  - localized multi enum fields
  - string fields
  - short string fields
  - long string fields
  - reference fields
  - struct member fields (member fields must also have *field indexed* = transitive, unit, local annotation)
  - list member fields (member fields must also have *field indexed* = transitive, unit, local annotation)
- custom fields (annotated as indexed = transitive):
  - unit reference member fields (member fields must also have *field indexed =* transitive annotation)
  - contact reference member fields (member fields must also have *field indexed =* transitive annotation)

Unit (Company or Customer)

Web Client Detailed Search

- company
- creation date
- customer
- customer group
- custom field
- disabled units - available when unit result type is selected
- engineer
- pattern
- queue

- status
- subject
- scope - available when queue is set
- view

Web Client Quick Search (API Search by Pattern):

- custom fields (annotated as *field indexed* = transitive, unit, local):
    - number fields
    - big decimal fields
    - localized enum fields
    - localized multi enum fields
    - string fields
    - short string fields
    - long string fields
    - reference fields
    - struct member fields
    - list member fields
    - unit reference member fields (member fields must also have *field indexed* = transitive, unit annotation)

Engineer

Web Client Detailed Search

- name (Login)

Web Client Autocomplete Search (Search by Pattern)

- first name
- last name
- e-mail

For the use of search functionalities using the ConSol*CM API (in scripts), is is also required that the custom fields and data object group fields be indexed (e.g. for the use of *TicketCriteria* or *UnitCriteria*. For details, please refer to the *ConSol*CM Process Designer Manual*.

For the use of the REST API, indexing might also be required, please see the REST API documentation.

Fields which are Searchable by Default

- Engineer data
    - E-Mail

    - Firstname
    - Lastname
    - ID
- Ticket data
    - Subject
    - ID
    - History
    - Attachment content

22.11.3 Administrator Tasks Concerning the Indexer

For the administrator it is important to know how to configure ConSol*CM in a way that ...

- all required fields can be searched.
- no overhead is produced (i.e. not too many fields are configured for searching).
- the search results are displayed in the desired way.
    - in the result table in the Detailed Search
    - as suggestions during the use of intelligent fields

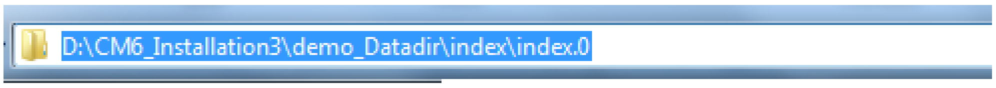
Those tasks will be described in the following sections.

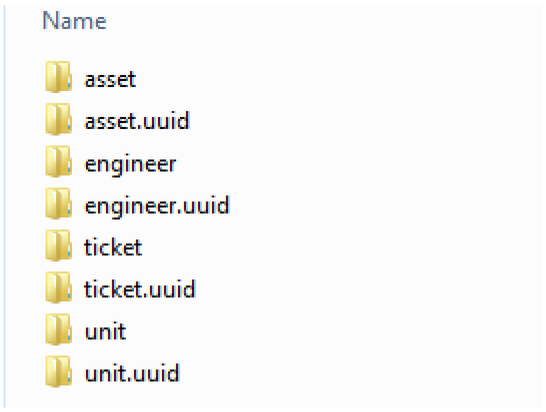
First, some background knowledge about the Indexer, the system which manages the search in ConSol*CM, is provided. This will help you as an administrator to look behind the scenes and understand the configuration in a better way.

22.11.4 Introduction to the ConSol*CM Indexer

The Indexer is a module of ConSol*CM which creates indexes. For each data field (custom field or data object group field) that should serve as search criterion (see next section), an index is created.

The indexes are stored on the hard disc in a sub-directory of the data directory that you have indicated during system set-up (the data directory is stored as a system property: *cmas-core-shared*, *data.directory*). The following picture shows an example for index files of a ConSol*CM installation (here used for a demo environment). The *demo_Datadir* is the data directory you have provided during set-up, all other directories are created automatically by ConSol*CM.


D:\CM6_Installation3\demo_Datadir\index\index.0


Name

- asset
- asset.uuid
- engineer
- engineer.uuid
- ticket
- ticket.uuid
- unit
- unit.uuid

Fig. 5: ConSol*CM Indexer - Directory demo_Datadir

Please make sure that ...

- the data directory is always available for the ConSol*CM server system if it has been created on another server and is linked to (or mounted on) the application server.
- that the ConSol*CM datadir is part of the daily backup (and can be restored if required).

If the index directory should be corrupt or the index should not be available, the index can be rebuilt or repaired. Please see the following sections for details.

22.11.5 Indexer and Index Management Using the Admin Tool

Field Indexed Annotation

By default, the entire ticket text and all attachments are indexed. For all custom fields and data object group fields, the field(s) which should be indexed has/have to have the annotation *field indexed*. Please refer to the sections Custom Field Administration, Data Object Group Field Management and GUI Design for Customer Data for details about setting annotations to custom fields (ticket data) and data object group fields (customer data). There are four possible values for this annotation:

- **local**
  Used for customer data. Only the unit (= data object) is given as a search result (e.g. when a field for a customer name is *indexed = local*, no company and no tickets are displayed when you search for the name of a contact, only the contact is listed).
- **unit**
  Used for customer data. Only the unit (= data object) and the parent unit (i.e. company) are given as a search result, no tickets are provided (e.g. when a field for a contact name is *indexed = unit*, the company is listed in the search result but no tickets of this contact are displayed when you search for the name of a user).

- **transitive**
  All data is displayed, this is the standard value for this annotation. If you are not sure what to choose, set *transitive*.
- **not indexed**
  The field is not indexed.

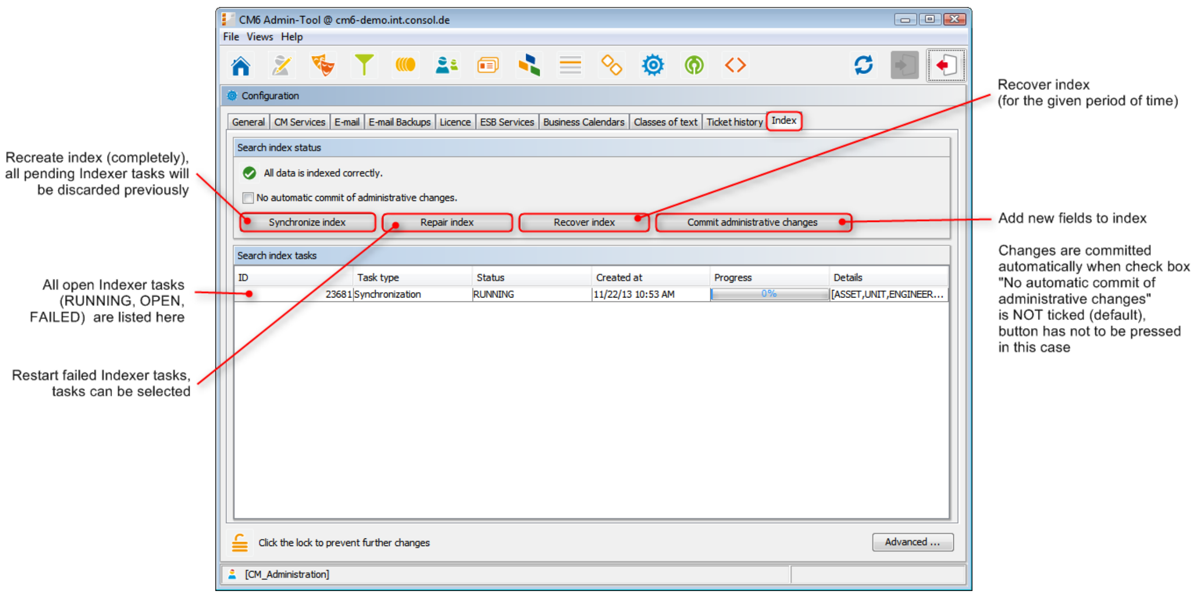
Nested fields have to have all the same index type, otherwise they cannot be searched. For example, when you work with a list of structs, the list, the struct and all custom fields which should be searched have to have the value *transitive* for the annotation *field indexed*.

Indexer Management: Tab Index

Usually, you do not have to do anything concerning the Indexer. ConSol*CM will handle everything regarding the indexing automatically. There are only two cases where you have to perform manual administrative operations:

1. You would like to change the configuration concerning the commitment of changes concerning the *field-indexed* annotation.
2. Errors have occurred in the indexing process.

In the Admin Tool, go to *Configuration* and use the tab *Index* to configure and manage the Indexer.


CM6 Admin-Tool @ cm6-demo.int.consol.de
File Views Help
Configuration
General | CM Services | E-mail | E-mail Backups | Licence | ESB Services | Business Calendars | Classes of text | Ticket history | Index
Search index status
Synchronize index
Repair index
Recover index
Commit administrative changes
Search index tasks
Click the lock to prevent further changes
Advanced ...
[CM_Administration]
Recreate index (completely), all pending Indexer tasks will be discarded previously
All open Indexer tasks (RUNNING, OPEN, FAILED) are listed here
Restart failed Indexer tasks, tasks can be selected
Recover index (for the given period of time)
Add new fields to index
Changes are committed automatically when check box "No automatic commit of administrative changes" is NOT ticked (default), button has not to be pressed in this case

All data is indexed correctly.

☐ No automatic commit of administrative changes.

ID	Task type	Status	Created at	Progress	Details
23681	Synchronization	RUNNING	11/22/13 10:53 AM	0%	[ASSET,UNIT,ENGINEER...

Fig. 6: ConSol*CM Admin Tool - Configuration: Tab Index

In the first line the current status of the Indexer is displayed (this is the value of the system property *cmas-core-index-common*, *index.status*):

- **GREEN** ✓
  All Indexer tasks have run correctly, no action required. At the beginning of the synchronization process, the index status is set to green. If it is completed successfully it remains green. If there is any problem it will change to yellow or red.

- **YELLOW**
  Fixable problems were identified, collected and persisted. The status is set when an administrative task (with auto-commit *off*) or a retry task is created.
- **RED**
  Errors have occurred. Please check. The index needs full synchronization.

The following operations can be performed:

- **Synchronize index**
  The index is rebuilt completely (from scratch), all open Indexer tasks are discarded.
- **Repair index**
  Indexer tasks which have not run successfully are restarted. The tasks can be selected in the Indexer task list.
- **Recover index**
  A time range can be selected. All changes which have been committed to ConSol*CM during this period of time will be (re-)indexed.
- **Commit administrative changes**
  Click this button to commit the changes when you have set a custom field or data object group field to *field-indexed* that was not indexed before. This has to be used if the check box *No automatic commit of administrative changes* has been selected. If the check box is inactive, the changes will be committed automatically when you have set the new annotation(s).

Attention:

There is a difference in CM versions concerning the meaning of *administrative changes* - this influences the manual index management!
In **versions prior to 6.9.3.0**, setting the annotation *field indexed* from *false* (or not set) to *true* is ⚠ **not** considered an administrative change. That means, when you have added the annotation *field indexed* = *true* for a custom field which was present before, you have to modify the index manually by using either *Synchronize index* or *Recover index*.
Starting with version 6.9.3.0, setting the annotation *field indexed* from *false* (or not set) to *true* **is** considered an administrative change. That means, when you have added the annotation *field indexed* = *true* for a custom field which was present before, you have to

- modify the index manually by clicking on *Commit administrative changes* in case *No automatic commit of administrative changes* **is** set.

or

- do nothing in case *No automatic commit of administrative changes* is **not** set.

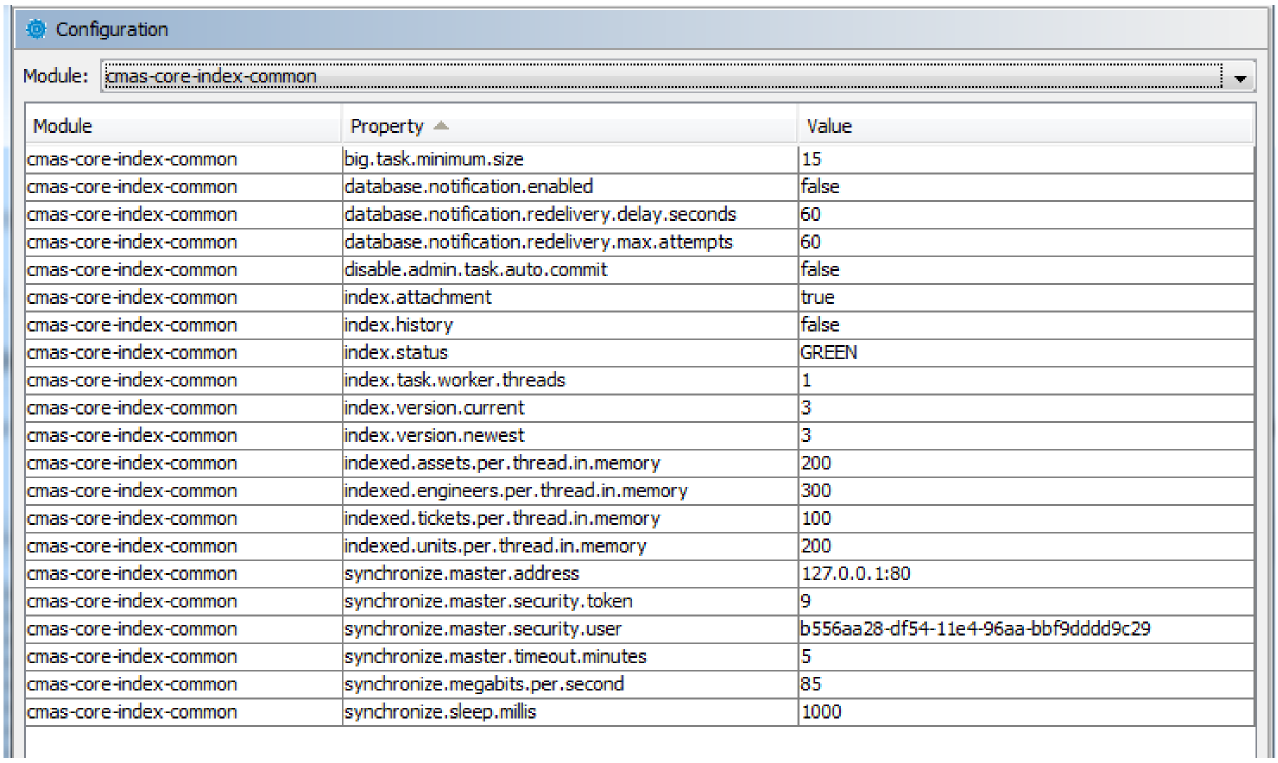
If there are open tasks in the Indexer task list, the following data is displayed for each task:

- **ID**
  Task ID
- **Task type**
  Three types are available:
  - **Synchronization**

        - Recreates whole index.
        - Triggered manually using the Admin Tool, *Synchronize index* command.
        - Before the start all other index tasks are removed.
    - **Administrative changes**
        - Created automatically when one of the following was updated:
          scope, queue, enum value, ticket function, ticket engineer, supported locale, role.
        - Processed automatically if *No automatic commit of administrative changes* option is unchecked.
        - Using *Commit administrative changes* command will start all administrative changes tasks.
    - **Retry**
        - Created automatically when error was encountered during index update process.
        - Holds information about entities which caused problems.
        - Using *Repair index* command will start all retry tasks.
- **Status**
  E.g. RUNNING
- **Created at**
  Time stamp when the task was created.
- **Progress**
  A progress bar that indicates the percentage of the task that has already been executed.
- **Details**
  A list of objects that are (re-)indexed in the task.

22.11.6 Indexer and Index-Relevant System Properties

The following system properties are also relevant for the Indexer, see following figure. Please refer to Appendix C for a detailed explanation of the Indexer system properties and Appendix D - Important System Properties - ordered by area of application, Indexer.



Module	Property	Value
cmas-core-index-common	big.task.minimum.size	15
cmas-core-index-common	database.notification.enabled	false
cmas-core-index-common	database.notification.redelivery.delay.seconds	60
cmas-core-index-common	database.notification.redelivery.max.attempts	60
cmas-core-index-common	disable.admin.task.auto.commit	false
cmas-core-index-common	index.attachment	true
cmas-core-index-common	index.history	false
cmas-core-index-common	index.status	GREEN
cmas-core-index-common	index.task.worker.threads	1
cmas-core-index-common	index.version.current	3
cmas-core-index-common	index.version.newest	3
cmas-core-index-common	indexed.assets.per.thread.in.memory	200
cmas-core-index-common	indexed.engineers.per.thread.in.memory	300
cmas-core-index-common	indexed.tickets.per.thread.in.memory	100
cmas-core-index-common	indexed.units.per.thread.in.memory	200
cmas-core-index-common	synchronize.master.address	127.0.0.1:80
cmas-core-index-common	synchronize.master.security.token	9
cmas-core-index-common	synchronize.master.security.user	b556aa28-df54-11e4-96aa-bbf9dddd9c29
cmas-core-index-common	synchronize.master.timeout.minutes	5
cmas-core-index-common	synchronize.megabits.per.second	85
cmas-core-index-common	synchronize.sleep.millis	1000

Fig. 7: Admin Tool: System Properties for Indexer

22.11.7 CM Indexer Services

For indexing, two ConSol*CM services are important:

- **Index changes notifier**
  Creates JMS (*Java Message Service*) messages with notifications that there has been one or more change(s) that concern(s) the index.
- **Index changes receiver**
  Reads JMS queue and starts update in Indexer.

Please see also section Tab CM Services.

Stopping *index changes receiver* is safe. After restart it will pick up all of the missing changes from the persistent store.

Stopping *index changes notifier is **NOT** safe.* If the indexer module discovers that the *notifier* is stopped and there is a message that has to be sent to the persistent store, the indexer will set the index status configuration property to *RED* i.e. signal that the index needs a full synchronization.

22.11.8 Systems with more than one Indexing Server

In case ConSol*CM is run in a cluster of application servers, each cluster node holds a local index in its *${cmas.data}/index/* directory. The index gets updated on the (single) master server, which is then polled by all slave servers to update their copies. As a result all indexing servers are eventually consistent, i.e. updates are not visible right away, usually with a small delay.

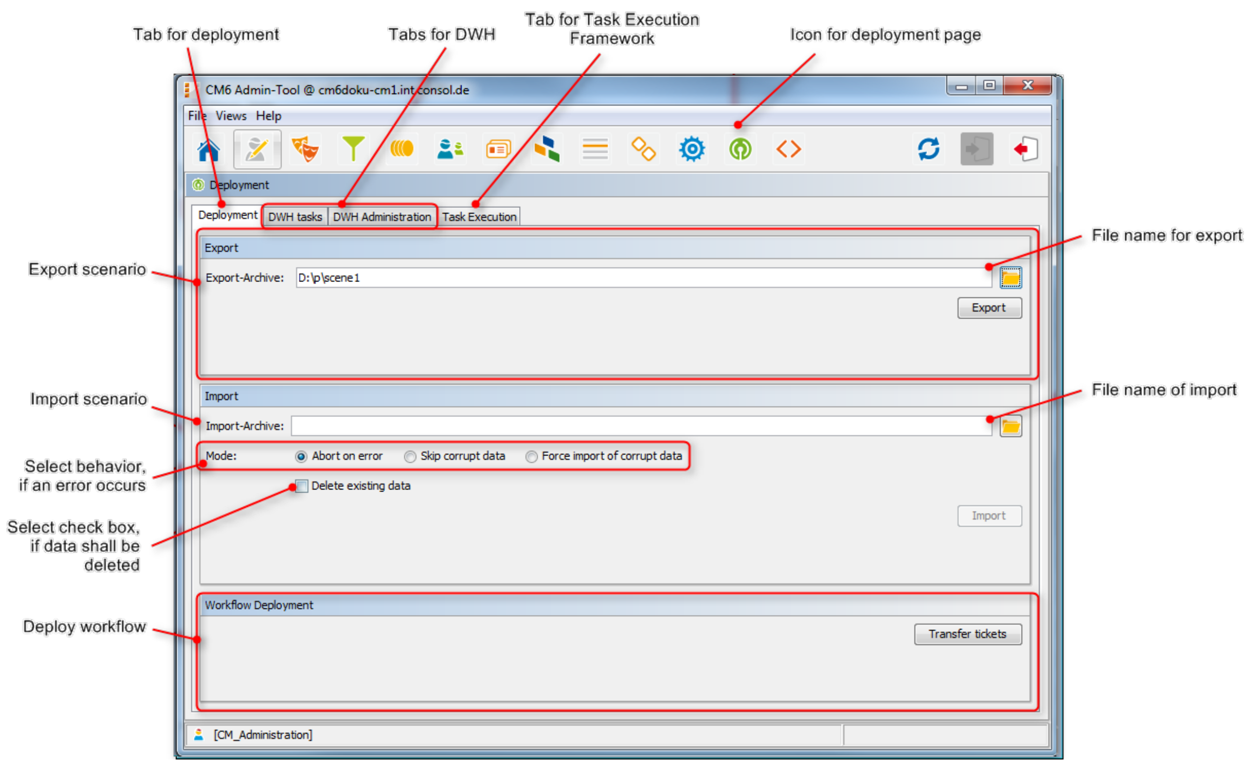
23 Deployment

- Introduction to Deployment in the Admin Tool
- Introduction to ConSol*CM Scenarios
- Tab Deployment
  - Export
  - Import
  - Workflow Deployment (for Deployment Error Recovery Only)
- Related Topics

23.1 Introduction to Deployment in the Admin Tool

On the tab *Deployment*, you can

- import and export scenarios (tab **Deployment**, will be explained in the subsequent sections in this chapter).
- manage the DWH (Data Warehouse, tabs **DWH tasks** and **DWH Administration**, please refer to chapter Data Warehouse (DWH) Management).
- manage tasks in the Task Execution Framework (TEF, tab **Task Execution**, please refer to section The Task Execution Framework (TEF).)



Export-Archive: D:\p\scene1

Import-Archive:

Mode: [✓ Abort on error] Skip corrupt data Force import of corrupt data

☐ Delete existing data

Fig. 1: ConSol*CM Admin Tool - Deployment Page

23.2 Introduction to ConSol*CM Scenarios

A scenario is a file in a proprietary ConSol*CM format (similar to *zip* and *tar*) that contains the data of a ConSol*CM installation. It can be exported from one 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 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-*
- all page customizations

A scenario **can** contain, depending on the selection of the administrator (see figure below):

- runtime data
- configuration data

A scenario will **never** contain:

- general (not customer-specific) system properties (e.g. mail server, LDAP directory etc.)

23.3 Tab Deployment

On the *Deployment* tab 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 CM installations. A typical example is transfer of the configuration from a test system to a production system.

Warning:

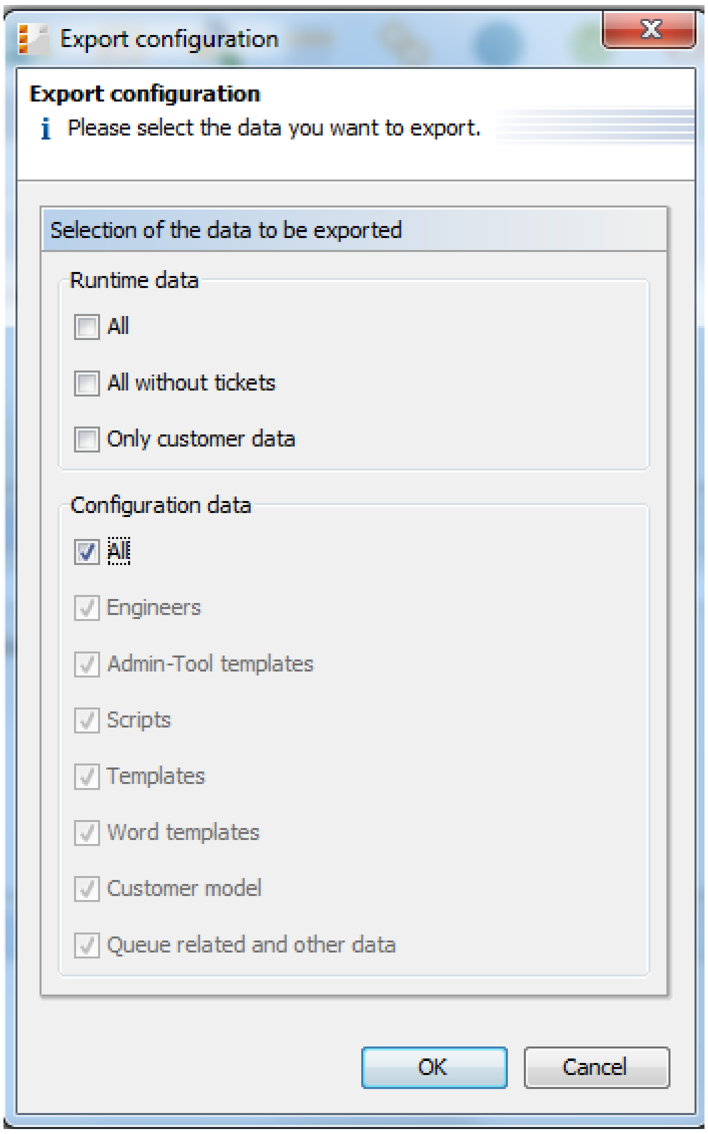
The import of external data can modify or delete existing data irrecoverably. Although the user is prompted for confirmation at critical points during deployment, this cannot prevent erroneous handling. Use this function only if you are very sure what you are doing. In case of doubt please ask the ConSol*CM support team or a ConSol*CM consultant for assistance.

23.3.1 Export

- **Export-Archive:**
  Enter path and name of the file you want to create. Alternatively, you can click on 📂 to open a selection window where you can search for a file.

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):


Export configuration

Export configuration

Please select the data you want to export.

Selection of the data to be exported

Runtime data

- [ ] All
- [ ] All without tickets
- [ ] Only customer data

Configuration data

- [x] All
- [x] Engineers
- [x] Admin-Tool templates
- [x] Scripts
- [x] Templates
- [x] Word templates
- [x] Customer model
- [x] Queue related and other data

Fig. 2: ConSol*CM Admin Tool - Deployment: Export Configuration

- **Runtime data**
  This refers to data that is stored as operating data, e.g. tickets and customer data.
  - **All**
    Ticket data and customer data is exported completely **and** the complete configuration is exported. When you select the check box *All*, all other check boxes are selected automatically.
  - **All without tickets**
    The complete installation beside the tickets is exported, i.e. the customer data **and** the complete configuration. When you select the check box *All without tickets*, all other check boxes except for *All* are selected automatically.
  - **Only customer data**
    Only customer data (i.e. the customer data model and the actual customer data) is exported. Nothing else. (The check box *Customer model* is checked automatically.)
- **Configuration data**
  This refers only to the configuration in the Admin Tool, no runtime data is exported.

- **All**
  The complete configuration is exported. When you select the check box *All*, all other check boxes under *Configuration data* are selected automatically.
- **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).
- **Templates**
  Only the templates that are stored in the Template Manager (see section The ConSol*CM Template Manager for details) are exported.
- **Word templates**
  Only the MS Word templates are exported, this is only relevant when CM.Doc is in operation (see section CM.Doc for details).
- **Customer model**
  Only the data object group fields that are used to define the customer model are exported. No runtime customer data is included.
- **Queue related and other data**
  Only queue configuration and general configuration settings are exported (workflows, queues, custom fields, enum values, MLAs, roles, views, properties, ...), in short: everything which is not included above.

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.

23.3.2 Import

The *general principle* of ConSol*CM scenario import is:

- If the check box *Delete existing data* has **not** been **selected**, the scenarios are **merged**, based on the following principles:
  - 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** the one of the original scenario.
    Example: For the field *priority*, there is the annotation *position = 0;2* in the imported scenario. The original scenario contains the value *position = 2;2* for the field *priority*, i.e. 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: In the imported scenario, there is the annotation *visibility = none* for the field *PersonID*. In the original scenario, the field *PersonID* is present, but does not have the annotation, i.e. 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/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 does no longer contain the annotation *visibility = none*, but the original scenario does contain the annotation, it will remain, i.e. in the resulting scenario the field *PersonID* is still invisible.
  - **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 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.
- If the check box *Delete existing data* has been **checked**, the entire system is deleted, i.e. **all** existing data are **deleted**. All data means:
  - Configuration data
  - Runtime data

  That means when *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.

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 on 📂 to open a selection window where you can search for the archive.
- **Mode:**
  Here you can choose what the import shall 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 when an error appeared. The problem can be probably handled afterwards in a short time. A new import might take longer to perform.
    Example: A referenced object is not found, e.g. 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 doing an error analysis.

Click on *Import* afterwards to start the data import.

23.3.3 Workflow Deployment (for Deployment Error Recovery Only)

Usually, all operations concerning workflow design and deployment are performed using the Process Designer. However, in case an error has occurred 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 it has to find the last consistent position in the process.
- **Restart process**
  The ticket goes back to the *START* node of the process/workflow.

Please read also the detailed explanation of the workflow deployment process in the *ConSol*CM Process Designer Manual*.

23.4 Related Topics

- Process Designer (see separate document *ConSol*CM Process Designer Manual*)

24 The Task Execution Framework (TEF)

- Introduction
- Admin Tool Scripts of Type Task
    - Introduction to Admin Tool Scripts of Type Task
    - Example Admin Tool Script of Type Task
    - Execution of a Task Using the Admin Tool
- Programming with Tasks
    - Introduction
    - Coding Examples
        - Creating a Task
        - Canceling (Killing) a Task
        - Repeating a Task
        - Defining the (First) Execution Date
        - Repeating a Task After an Error Occurred
    - Example for the Use of a Task Script

24.1 Introduction

Using the *Task Execution Framework* (TEF), ConSol*CM can perform various tasks which are not directly tied to or embedded in another object like a workflow, unit or resource activity or 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 CM script. The TEF tasks can be executed in an asynchronous manner. A new API has been added to ConSol*CM (in version 6.9.4.0) in order to provide the TEF extensions. TEF scripts can be started (i.e. the TEF API is available in):

- manually in the Admin Tool
- from workflow activity scripts
- from e-mail scripts
- from unit action scripts

A task is stored as Admin Tool Script of type *Task Definition.* This script type is explained in the respective section below.

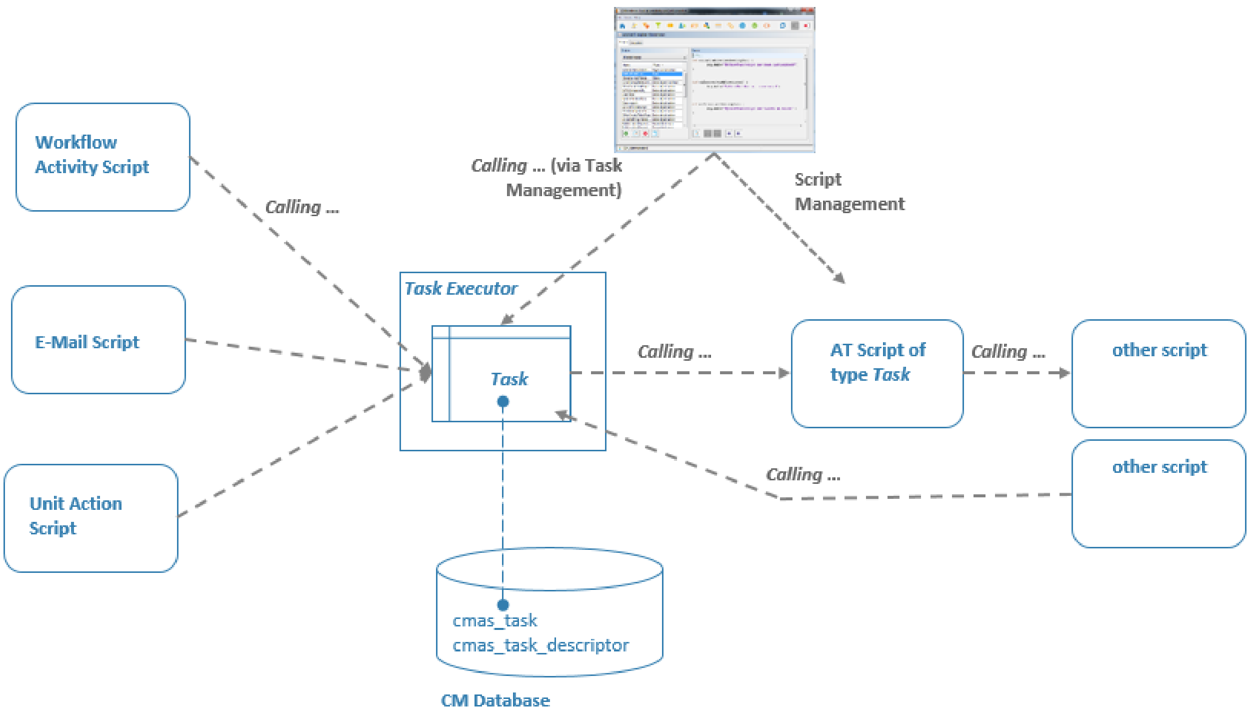


Fig. 1: ConSol*CM Task Execution Framework

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 of 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, tab *Task execution*, see section Task Execution using the Admin Tool. Here, a task can be started immediately, it can not 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
    - e-mail scripts
    - unit 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 or the start time of the task execution. Using the task descriptor, a newly defined task can be executed immediately or can be scheduled for a later execution time. This can be applied in scripts. Please see section Programming with Tasks for programming details.

24.2 Admin Tool Scripts of Type Task

24.2.1 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 provided when the script is created initially.

Methods in Admin-Tool Script of Type Task

```
def onInitialize(taskDescriptor) {}
def onExecute(taskDescriptor) {}
def onError(taskDescriptor) {}
def onCancel(taskDescriptor) {}
```

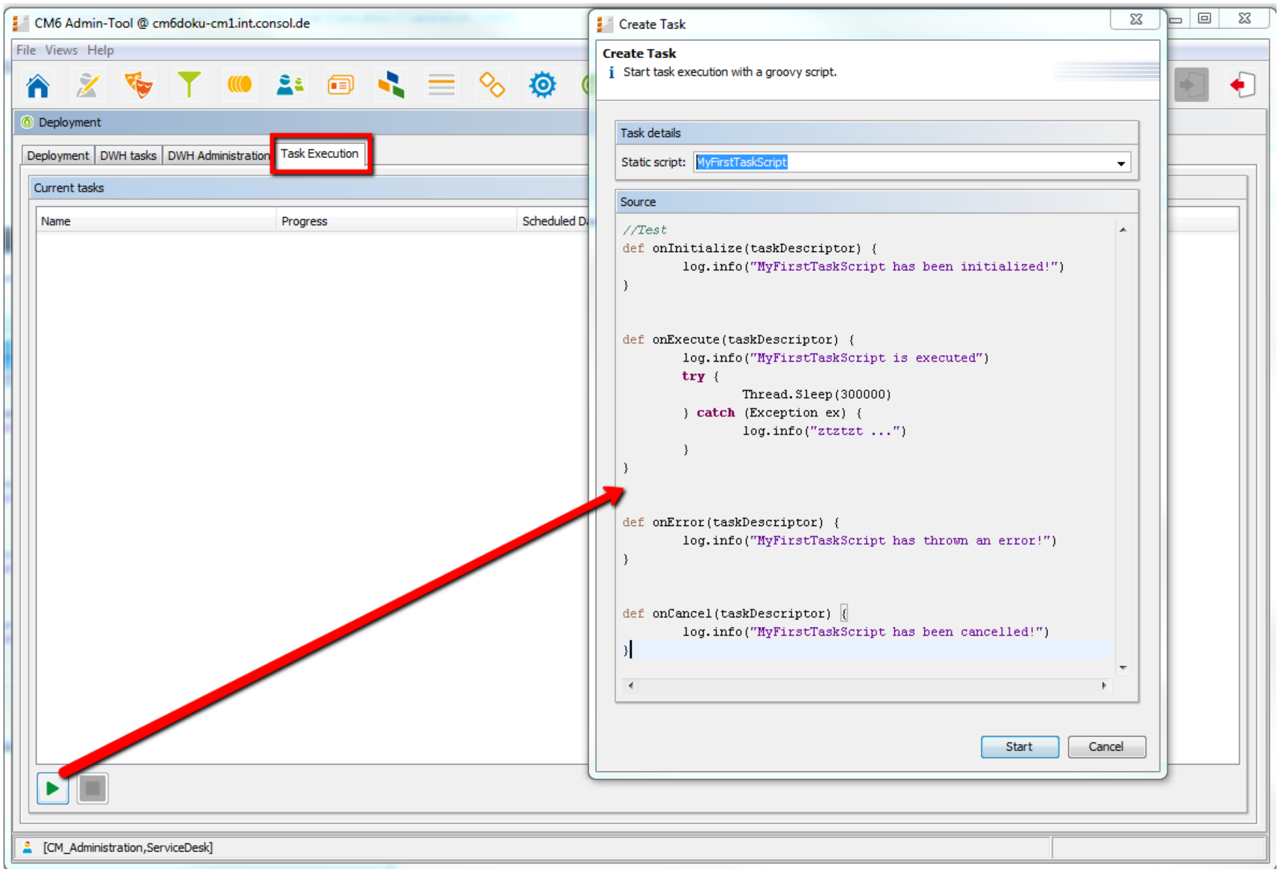
24.2.2 Example Admin Tool Script of Type Task

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!")
}
```

24.2.3 Execution of a Task Using the Admin Tool

In the Admin Tool, section *Deployment*, tab *Task Execution*, you can start tasks which have been defined as Admin Tool scripts before.



Create Task

Start task execution with a groovy script.

Task details

Static script: MyFirstTaskScript

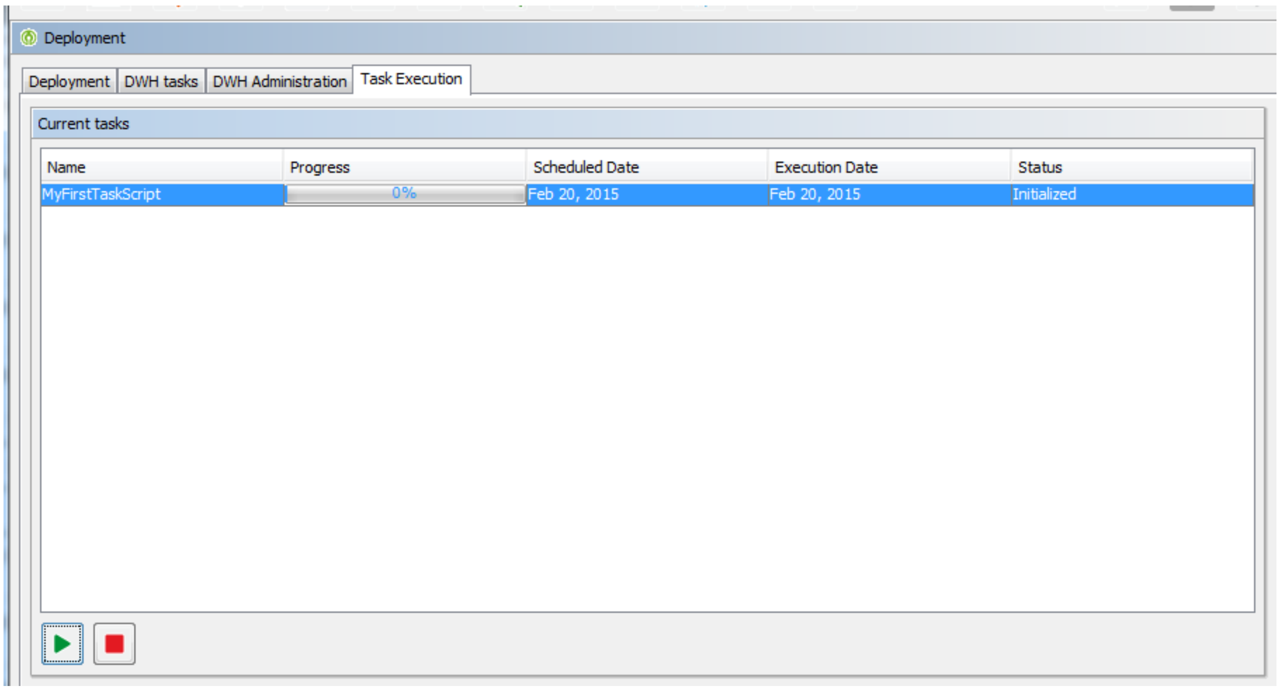
Source

```
//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!")
}
```

Fig. 2: Admin Tool 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* and the script will be executed 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 the explanation below.

When a task is running, a progress bar is shown. You can stop (cancel) the task using the *Stop* button.



Name	Progress	Scheduled Date	Execution Date	Status
MyFirstTaskScript	0%	Feb 20, 2015	Feb 20, 2015	Initialized

Fig. 3: Running task in the Admin Tool

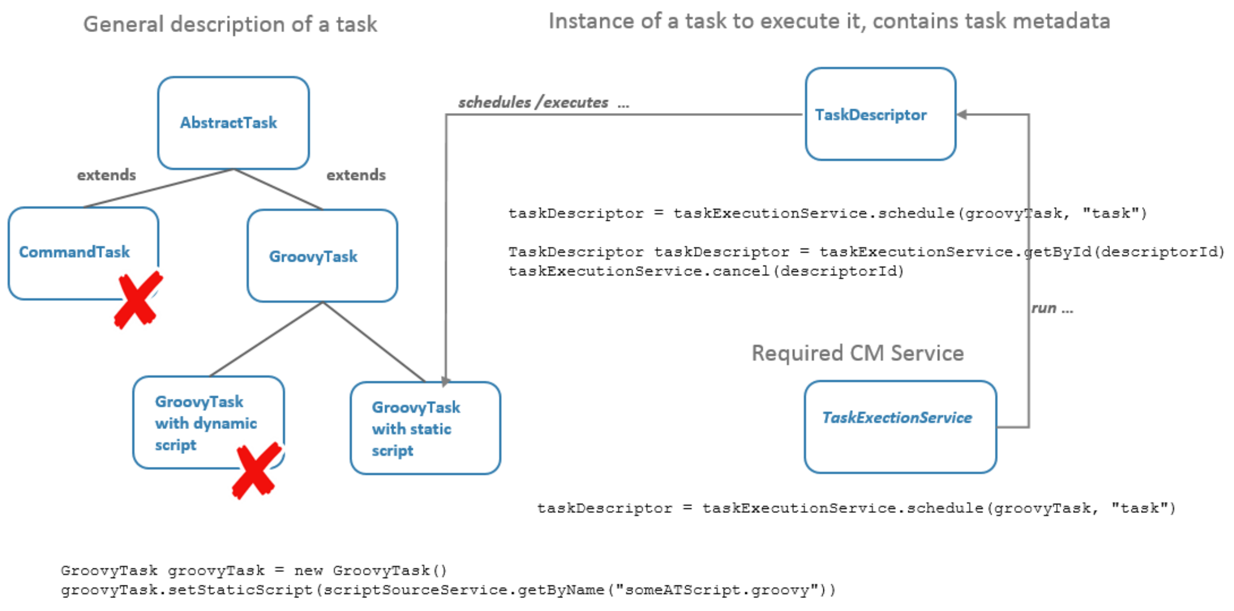
24.3 Programming with Tasks

24.3.1 Introduction

As of CM version 6.9.4, one type of tasks is available, the **Groovy Task with a static script**.This refers to the Admin Tool script which defines the task, see sections above.

The **Task Execution Service** (Groovy class TaskExecutionService, a singleton) runs in the background and scans the CM database for tasks (DB table cmas_task_descriptor) with the status INITIALIZED. Like all CM services it is implicitly available as object named *taskExecutionService* (see following examples). When the start date of the task has been reached, the task is started.

All parameters for the new task have to be set using the **task descriptor** (Groovy class TaskDescriptor), e. g. the start date of the task. The task descriptor will also provide information about the running task, like the task's progress.


General description of a task
Instance of a task to execute it, contains task metadata
AbstractTask
extends
extends
CommandTask
GroovyTask
GroovyTask with dynamic script
GroovyTask with static script
schedules /executes ...
TaskDescriptor
run ...
Required CM Service
TaskExectionService

```
taskDescriptor = taskExecutionService.schedule(groovyTask, "task")

TaskDescriptor taskDescriptor = taskExecutionService.getById(descriptorId)
taskExecutionService.cancel(descriptorId)
```

```
taskDescriptor = taskExecutionService.schedule(groovyTask, "task")
```

```
GroovyTask groovyTask = new GroovyTask()
groovyTask.setStaticScript(scriptSourceService.getByName("someATScript.groovy"))
```

Fig. 4: Some TEF Groovy classes

24.3.2 Coding Examples

Creating a Task

Example for creating a task descriptor

```
GroovyTask groovyTask = new GroovyTask();
groovyTask.setStaticScript(scriptSourceService.getByName("someATScript.groovy"));
taskDescriptor = taskExecutionService.schedule(groovyTask, "task");
```

Canceling (Killing) a Task

Part 1: Create the task descriptor and save the Id somewhere

Example for cancelling a task

```
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
```

Part 2, maybe also used at a later point of time:

Cancelling a task

```
taskExecutionService.cancel(myTaskDescriptorId)
```

Repeating a Task

If you set another execution date for a task after its job has been done, it will be rescheduled. So this code has to be used within the Admin Tool Task script.

Repeating a Task

```
def onInitialize(taskDescriptor) {}
def onExecute(taskDescriptor) {
    //some code to execute
    ...........................
    //here, we set the new future execution date for the task, we also need to return a special
steering object
    taskDescriptor.setExecutionDate(new Date(new Date().getTime() + 15000));
    return new ExecutionSpecification().setRetryRequested(true);
}
def onError(taskDescriptor) {}
def onCancel(taskDescriptor) {}
```

Defining the (First) Execution Date

For a script which should not be started immediately, you can define a start time in the *onInitialize()* method.

(Pre-) Scheduling a Task

```
def onInitialize(taskDescriptor) {
    taskDescriptor.setExecutionDate(yourDate)
}
```

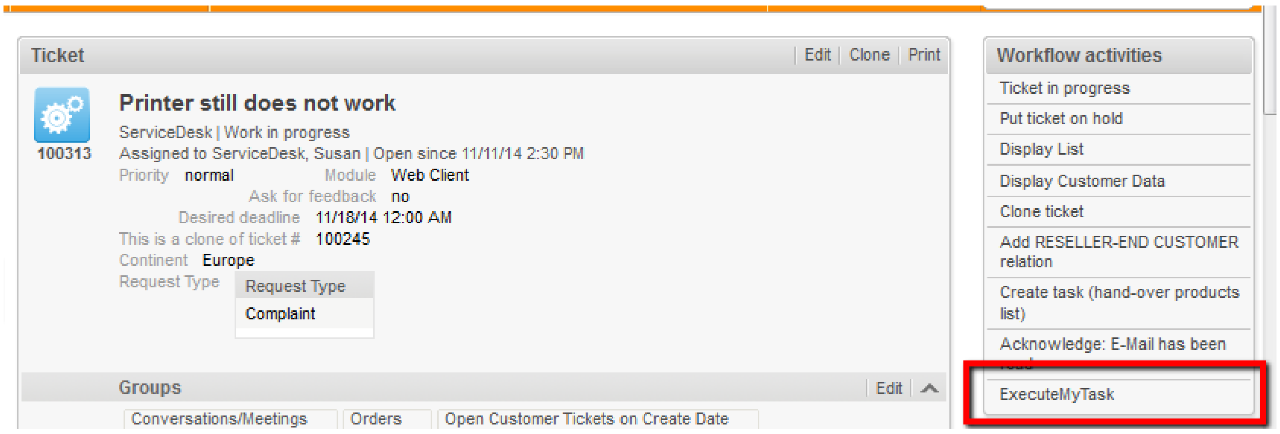
Repeating a Task After an Error Occurred

Repeating a Task after an Error Occured

```
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) {}
```

24.3.3 Example for the Use of a 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. The engineer does not have to do anything in between.


Ticket
Edit | Clone | Print
Printer still does not work
100313
ServiceDesk | Work in progress
Assigned to ServiceDesk, Susan | Open since 11/11/14 2:30 PM
Priority normal Module Web Client
Ask for feedback no
Desired deadline 11/18/14 12:00 AM
This is a clone of ticket # 100245
Continent Europe
Request Type Request Type Complaint
Groups Edit
Conversations/Meetings
Orders
Open Customer Tickets on Create Date
Workflow activities
Ticket in progress
Put ticket on hold
Display List
Display Customer Data
Clone ticket
Add RESELLER-END CUSTOMER relation
Create task (hand-over products list)
Acknowledge: E-Mail has been read
ExecuteMyTask
Fig. 5: Web Client view: Workflow activity for task execution

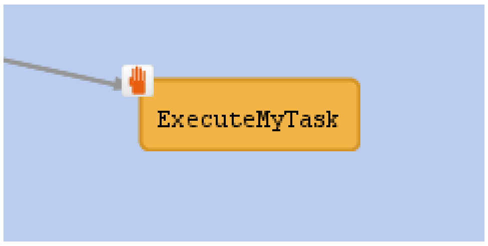


Fig. 6: Process Designer view: Workflow activity for task execution

Workflow activity script for task execution

```
def myNewTask = new GroovyTask()
```

```
myNewTask.setStaticScript(scriptSourceService.getByName("MyFirstTaskScript"))
def myTaskDescriptor = taskExecutionService.schedule(myNewTask, "myTaskGroup")
myTaskDescriptor.setExecutionDate(new Date())
```

Log output from 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}
```

25 CM6 Administrator Manual 6.9.4 - Script and Admin Tool Template Administration

25.1 Script and Admin Tool Template Administration

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 concerning this topic. However, various scripts are also stored in the Admin Tool, in the *Scripts* section. This will be explained in section Admin Tool Scripts.

Templates are also stored in several locations:

- Accessible via Web Client:
    - In the Template Manager (e-mail templates)
    - In Doc Template Manager (MS Word templates in CM.Doc)
- Accessible via Admin Tool:
    - In the *Scripts* section
        - Some special e-mail templates
        - Templates for customer data
        - <more templates depending on customizing>

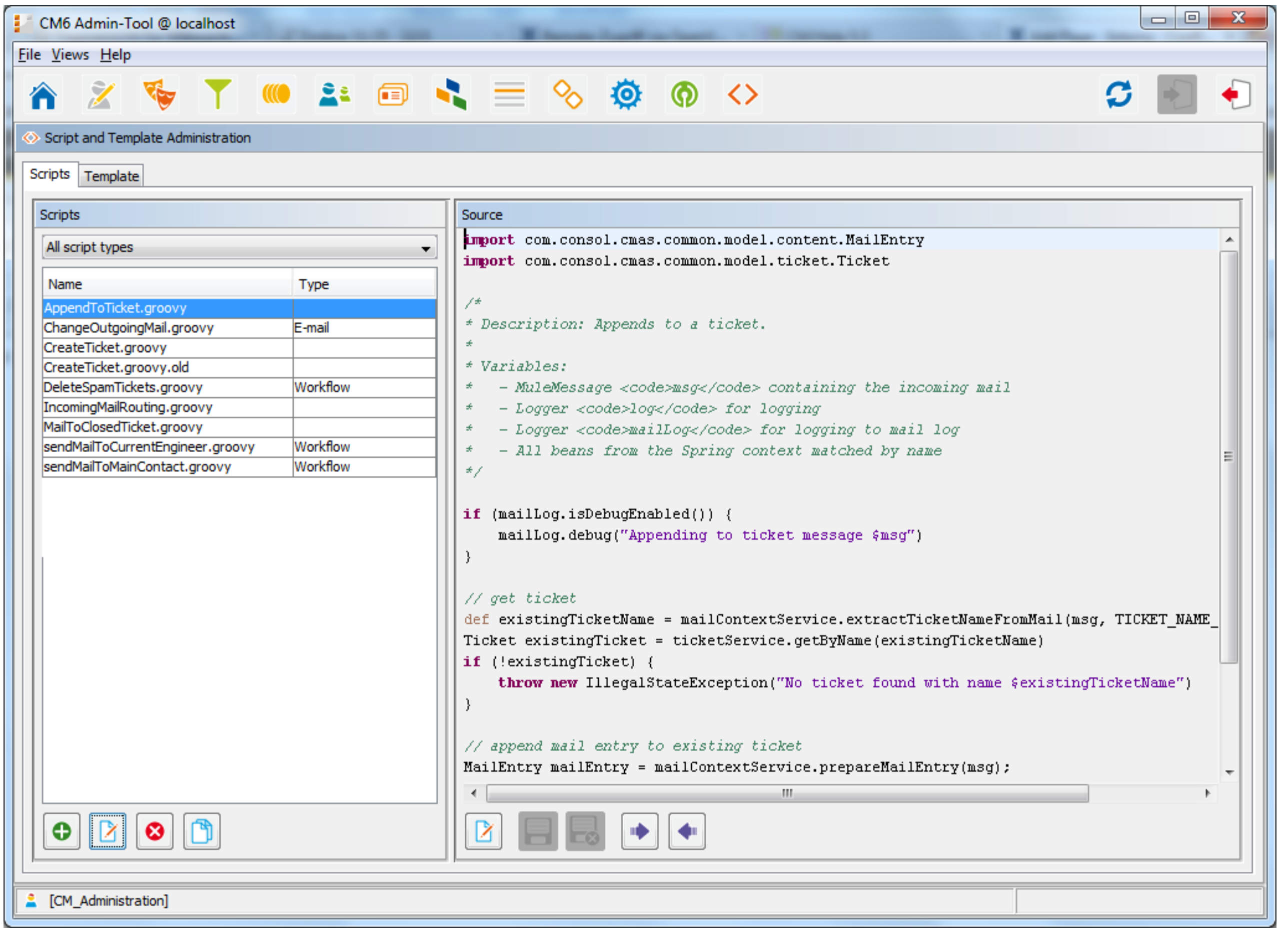
For explanations of the work with e-mail templates using the Template Manager and for configuring CM.Doc, please refer to section Working with Text Templates. For an explanation of templates in the Admin Tool, please read section Admin Tool Templates.

25.2 Introduction to Scripts in the Admin Tool

- The Source Code Editor
- Script Types
    - Scripts of Type Clone
    - Scripts of Type Data Object Action
    - Scripts of Type Data Object Condition
    - Scripts of Type Default Values
        - Overwrite Mode for Default Values Scripts
    - Scripts of Type Dependent Enum
    - Scripts of Type E-Mail
        - E-Mail Scripts for the Processing of Incoming E-Mails in ESB/Mule Mail Mode
        - E-Mail Scripts for the Processing of Incoming E-Mails in NIMH Mode
        - Differences between Scripts in ESB/Mule Mail and NIMH
        - E-Mail Scripts for Outgoing E-Mails
    - Scripts of Type Page Customization
    - Scripts of Type Task
    - Scripts of Type Workflow
    - Default Workflow Activity Script

Scripts are stored in the *Scripts* 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 *Script and Template Administration* in the Admin Tool. The tab *Scripts* is opened initially.



Name	Type
AppendToTicket.groovy	
ChangeOutgoingMail.groovy	E-mail
CreateTicket.groovy	
CreateTicket.groovy.old	
DeleteSpamTickets.groovy	Workflow
IncomingMailRouting.groovy	
MailToClosedTicket.groovy	
sendMailToCurrentEngineer.groovy	Workflow
sendMailToMainContact.groovy	Workflow

```
import com.consol.cmas.common.model.content.MailEntry
import com.consol.cmas.common.model.ticket.Ticket

/*
* Description: Appends to a ticket.
*
* Variables:
*   - MuleMessage <code>msg</code> containing the incoming mail
*   - Logger <code>log</code> for logging
*   - Logger <code>mailLog</code> for logging to mail log
*   - All beans from the Spring context matched by name
*/

if (mailLog.isDebugEnabled()) {
    mailLog.debug("Appending to ticket message $msg")
}

// get ticket
def existingTicketName = mailContextService.extractTicketNameFromMail(msg, TICKET_NAME_
Ticket existingTicket = ticketService.getByName(existingTicketName)
if (!existingTicket) {
    throw new IllegalStateException("No ticket found with name $existingTicketName")
}

// append mail entry to existing ticket
MailEntry mailEntry = mailContextService.prepareMailEntry(msg);
```

Fig. 1: ConSol*CM Admin Tool - Script and Template Administration

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. 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:
  - **Clone**
    Script which is executed when the *Clone* option is selected for a ticket. Has to be assigned to a queue. See section Queue Administration for details.
  - **Data object action**
    Script which is executed when a data object action has taken place, see section Action Framework for details.
  - **Data object condition**
    Script which is executed to evaluate if a data object action should be offered in the Web Client, see section Action Framework 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*, a structure that provides hierarchical lists. Please see section Scripts of Type Dependent Enum for details.
- **E-mail**
  Scripts of this type are used to manage incoming and outgoing e-mails. Please see section Scripts of Type E-mail for details.
- **Page customization**
  Scripts of this type are referenced by Page Customization settings. Please see section Page Customization for the Web Client Dashboard.
- **Task**
  Scripts of this type are used by the TEF (Task Execution Framework), please see section The Task Execution Framework (TEF).
- **Workflow**
  Scripts of this type are referenced from the workflow. Please see section Scripts of Type Workflow for details.

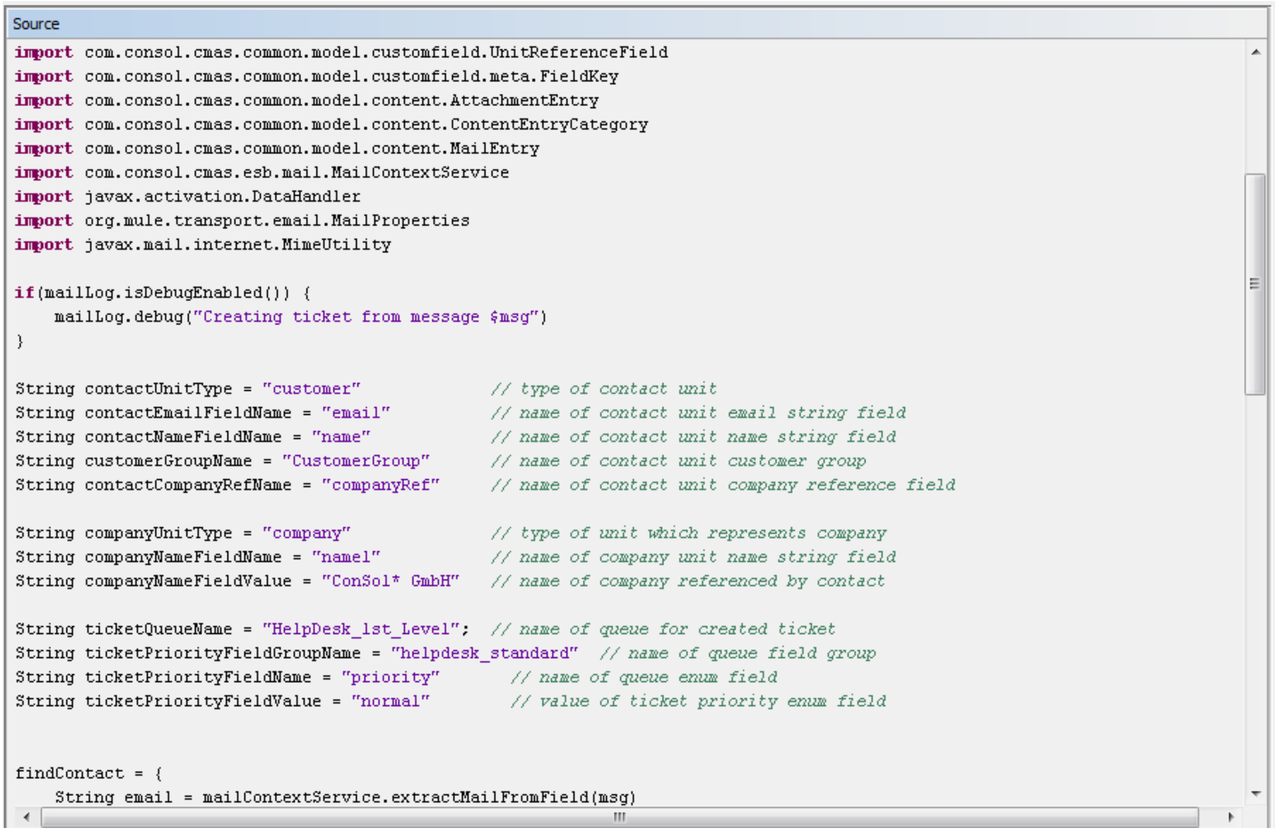
The buttons below the list offer the standard Admin Tool functionalities:

- Add a script
- Edit a script
- Delete a script
- Copy a script

On the right you see the *Source Code Editor*. The script that is selected in the list on the left is displayed. Here you can write the script source code when you have selected the *edit* mode.

25.2.1 The Source Code Editor

The Source Code Editor provides an editing panel with syntax highlighting. You have to check for correct code yourself.



```
import com.consol.cmas.common.model.customfield.UnitReferenceField
import com.consol.cmas.common.model.customfield.meta.FieldKey
import com.consol.cmas.common.model.content.AttachmentEntry
import com.consol.cmas.common.model.content.ContentEntryCategory
import com.consol.cmas.common.model.content.MailEntry
import com.consol.cmas.esb.mail.MailContextService
import javax.activation.DataHandler
import org.mule.transport.email.MailProperties
import javax.mail.internet.MimeUtility

if(mailLog.isDebugEnabled()) {
    mailLog.debug("Creating ticket from message $msg")
}

String contactUnitType = "customer"            // type of contact unit
String contactEmailFieldName = "email"         // name of contact unit email string field
String contactNameFieldName = "name"           // name of contact unit name string field
String customerGroupName = "CustomerGroup"     // name of contact unit customer group
String contactCompanyRefName = "companyRef"    // name of contact unit company reference field

String companyUnitType = "company"             // type of unit which represents company
String companyNameFieldName = "name1"          // name of company unit name string field
String companyNameFieldValue = "ConSol* GmbH"  // name of company referenced by contact

String ticketQueueName = "HelpDesk_1st_Level";  // name of queue for created ticket
String ticketPriorityFieldGroupName = "helpdesk_standard"  // name of queue field group
String ticketPriorityFieldName = "priority"      // name of queue enum field
String ticketPriorityFieldValue = "normal"       // value of ticket priority enum field


findContact = {
    String email = mailContextService.extractMailFromField(msg)
```

Fig. 2: ConSol*CM Admin Tool - Source Code Editor

In the lower section of the Source Code Editor, there are the following buttons:

- **Edit**
  Press this button to switch to *edit* mode in the Source Code Editor. When you open the *Script and Template Administration* in the Admin Tool, all scripts are in *read-only* mode to prevent an administrator from changing something accidentally.
- **Quit and save**
  Save the script and quit *edit* mode, i.e. switch to *read-only* mode again.
- **Quit without saving**
  Switch to *read-only* mode again, without saving the changes you might have made to the source code.
- **Open script from file**
  Import a script from a file. This will import the file and display it in the Source Code Editor. No checks are performed.
- **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 you are working on.

25.2.2 Script Types

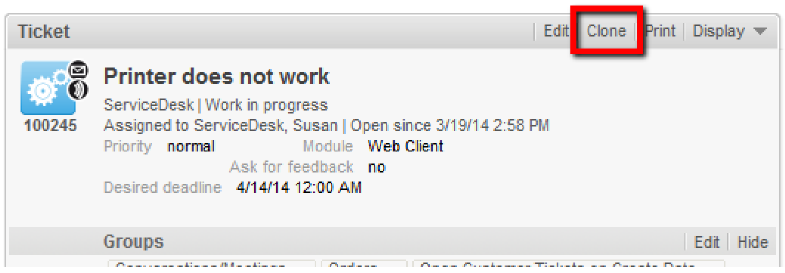
In the following section, the possible script types are explained. Some examples are provided to give you an impression of the potential of scripts in the system.

The following script types are available:

- Clone
- Data object action
- Data object condition
- Default values
- Dependent enums
- E-mail
- Page customization
- Task
- Workflow

Scripts of Type Clone

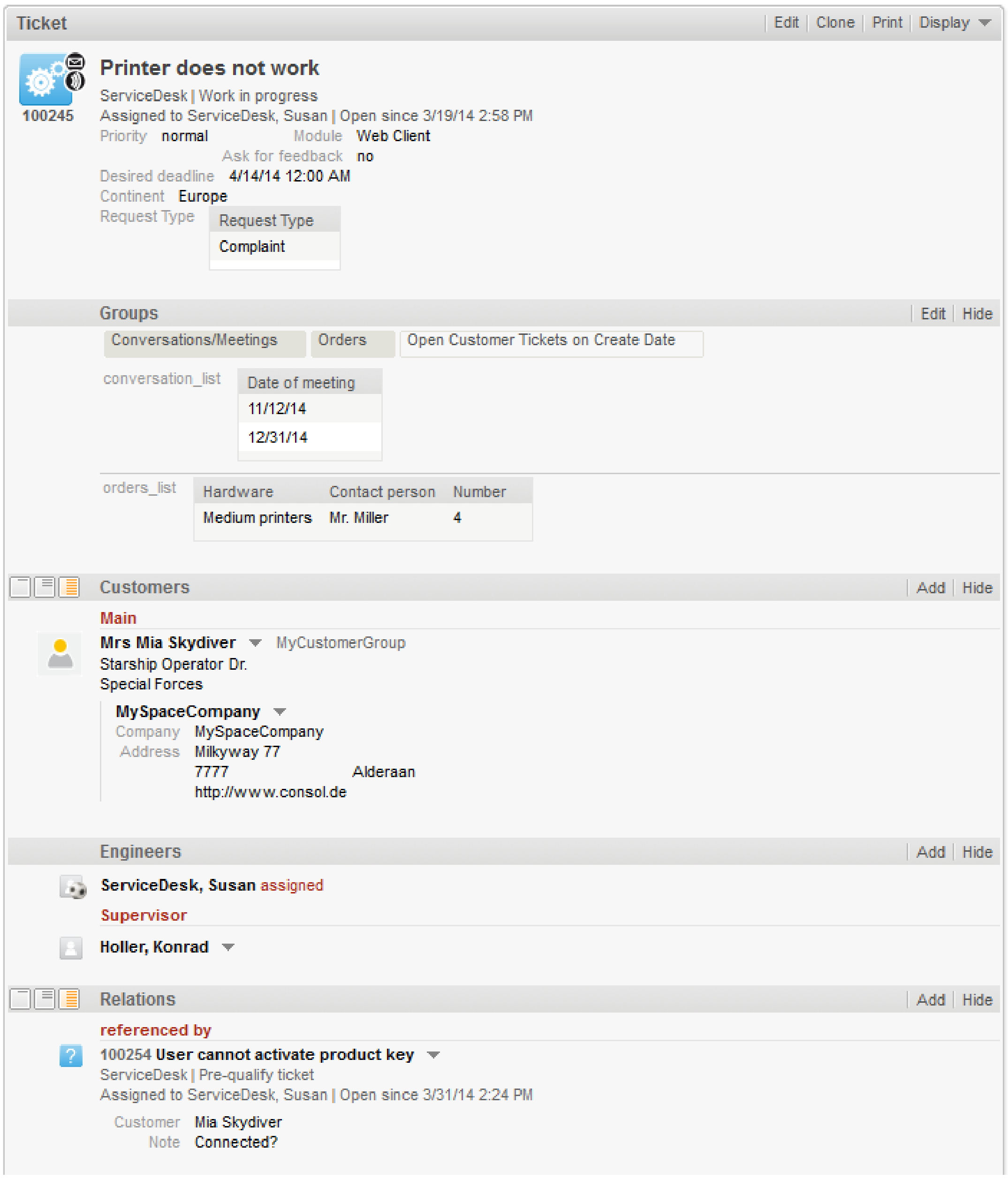
In the Web Client, a ticket can be cloned (duplicated) using the *Clone* option in the ticket menu.



Printer does not work
100245
ServiceDesk | Work in progress
Assigned to ServiceDesk, Susan | Open since 3/19/14 2:58 PM
Priority normal
Module Web Client
Ask for feedback no
Desired deadline 4/14/14 12:00 AM

Fig. 3: ConSol*CM Web Client - Clone Option in Ticket Menu

In case the queue where the ticket is located uses a *Clone* script (see section Queue Administration), this script will be executed when the engineer has clicked on *Clone*. The script can be used similar to a *Default values* script (see respective section below): you can preset values in the ticket which will be created. In the cloning process the values are pre-filled in the custom fields in the Web Client, i.e. before the ticket is generated. The engineer can change the values if required.


Ticket | Edit | Clone | Print | Display

Printer does not work
100245
ServiceDesk | Work in progress
Assigned to ServiceDesk, Susan | Open since 3/19/14 2:58 PM
Priority normal Module Web Client
Ask for feedback no
Desired deadline 4/14/14 12:00 AM
Continent Europe
Request Type

Request Type
Complaint

Groups | Edit | Hide
Conversations/Meetings | Orders | Open Customer Tickets on Create Date

conversation_list

Date of meeting
11/12/14
12/31/14

orders_list

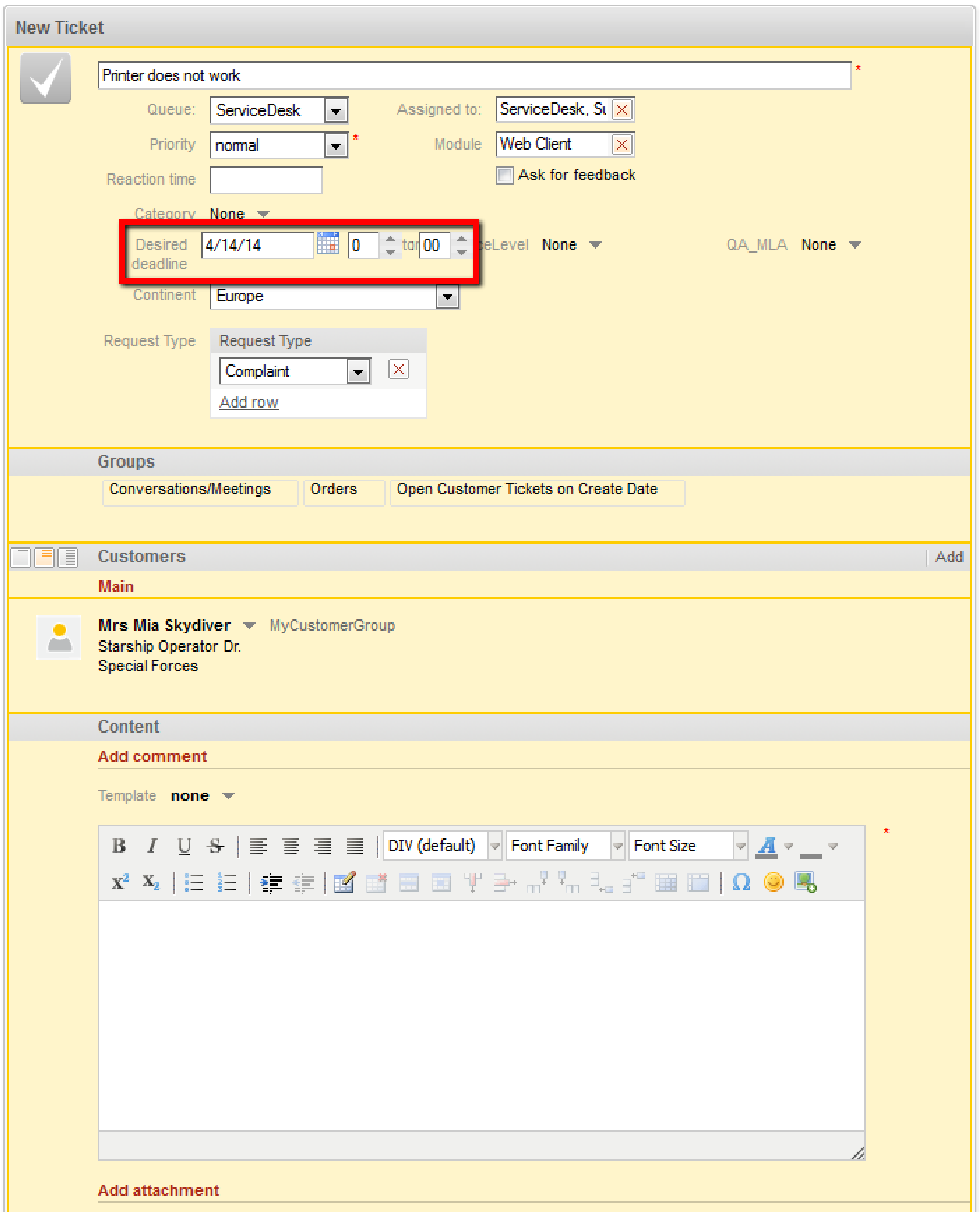
Hardware	Contact person	Number
Medium printers	Mr. Miller	4

Customers | Add | Hide
Main
Mrs Mia Skydiver MyCustomerGroup
Starship Operator Dr.
Special Forces
MySpaceCompany
Company MySpaceCompany
Address Milkyway 77
7777 Alderaan
http://www.consol.de

Engineers | Add | Hide
ServiceDesk, Susan assigned
Supervisor
Holler, Konrad

Relations | Add | Hide
referenced by
100254 User cannot activate product key
ServiceDesk | Pre-qualify ticket
Assigned to ServiceDesk, Susan | Open since 3/31/14 2:24 PM
Customer Mia Skydiver
Note Connected?

Fig. 4: ConSol*CM Web Client - Original Ticket


New Ticket

Printer does not work *

Queue: ServiceDesk
Assigned to: ServiceDesk, Su
Priority: normal *
Module: Web Client
Reaction time
Ask for feedback
Category: None
Desired deadline: 4/14/14 0 to 00
eLevel: None
QA_MLA: None
Continent: Europe
Request Type: Request Type
Complaint
Add row

Groups
Conversations/Meetings | Orders | Open Customer Tickets on Create Date

Customers
Add
Main
Mrs Mia Skydiver
MyCustomerGroup
Starship Operator Dr.
Special Forces

Content
Add comment
Template: none
DIV (default) | Font Family | Font Size
Add attachment

Fig. 5: ConSol*CM Web Client - Cloning the Ticket (without Clone Script)

Information:

When a ticket is cloned, the following data is transferred from the original ticket (compare the two images above):

- the ticket subject
- the queue
- the engineer (if it was set)
- the values of all custom fields (ticket data section and group section)
- the main customer
- additional customers

When a ticket is cloned, the following data is ⚠ **not** transferred from the original ticket (compare the two images above):

- all ticket text:
    - comments
    - e-mails
    - attachments
- the ticket history
- additional engineers
- ticket relations

Information:

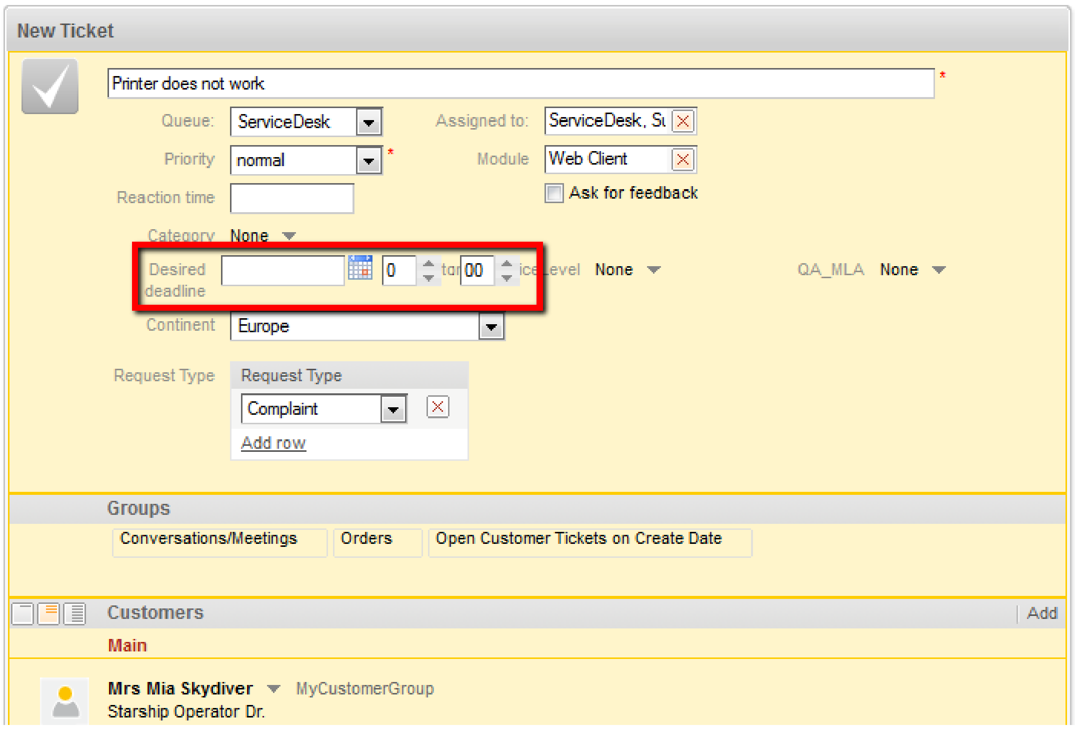
Please keep in mind that in a *Clone* script, you do not work in the workflow context! That means the *workflowApi* object (implementation of *WorkflowContextService*) is not available!

In the following example, the *Clone* script is used to reset the data field *Desired deadline* to avoid wrong service dates in the (cloned) *Service Desk* ticket.

Clone script to reset custom field for desired deadline

```
ticket.set("serviceDesk_fields.desiredDeadline", null)
```

When the script is assigned to the queue (*Service Desk* in the example), the field for the desired deadline is empty when the cloned ticket is offered to the engineer. See following image:



New Ticket

Printer does not work *

Queue:	ServiceDesk	Assigned to:	ServiceDesk, Sι
Priority	normal *	Module	Web Client
Reaction time			Ask for feedback
Category	None		
Desired deadline	0 to 00	ice Level None	QA_MLA None
Continent	Europe		
Request Type	Request Type: Complaint; Add row		

Groups

Conversations/Meetings | Orders | Open Customer Tickets on Create Date

Customers | Add

Main

Mrs Mia Skydiver MyCustomerGroup

Starship Operator Dr.

Fig. 6: ConSol*CM Web Client - Cloned Ticket (with Clone Script Assigned to the Queue)

Scripts of Type Data Object Action

See section Action Framework.

Scripts of Type Data Object Condition

See section Action Framework.

Scripts of Type Default Values

Sometimes it is required that a data field of a ticket has a certain value when the ticket is to be created, i.e. when the engineer presses *New ticket* and the respective form is opened in the GUI, one or more value(s) should be preset. This saves the engineer from setting the value every time, e.g. when the default priority is *normal*, this can be preset. In case *high* or *low* is required, the engineer can switch to another value.

This ConSol*CM behavior can be achieved by using one or more *Default values* scripts. The default values can be defined specifically for each queue. Please see the following example.

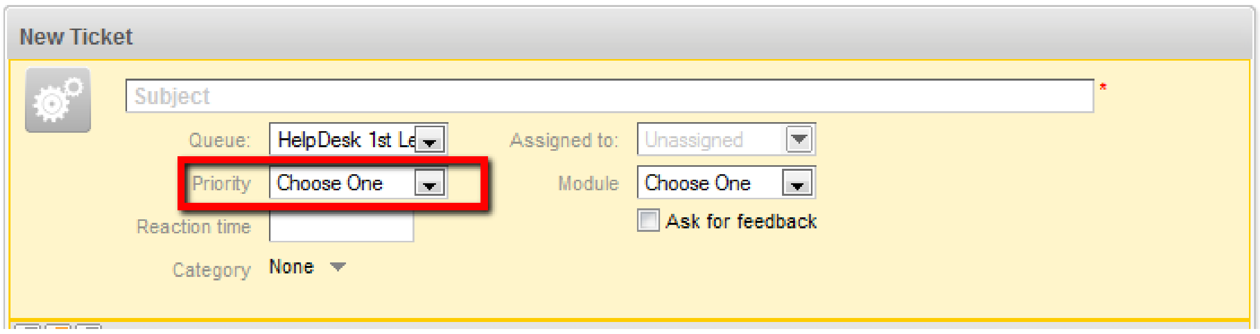


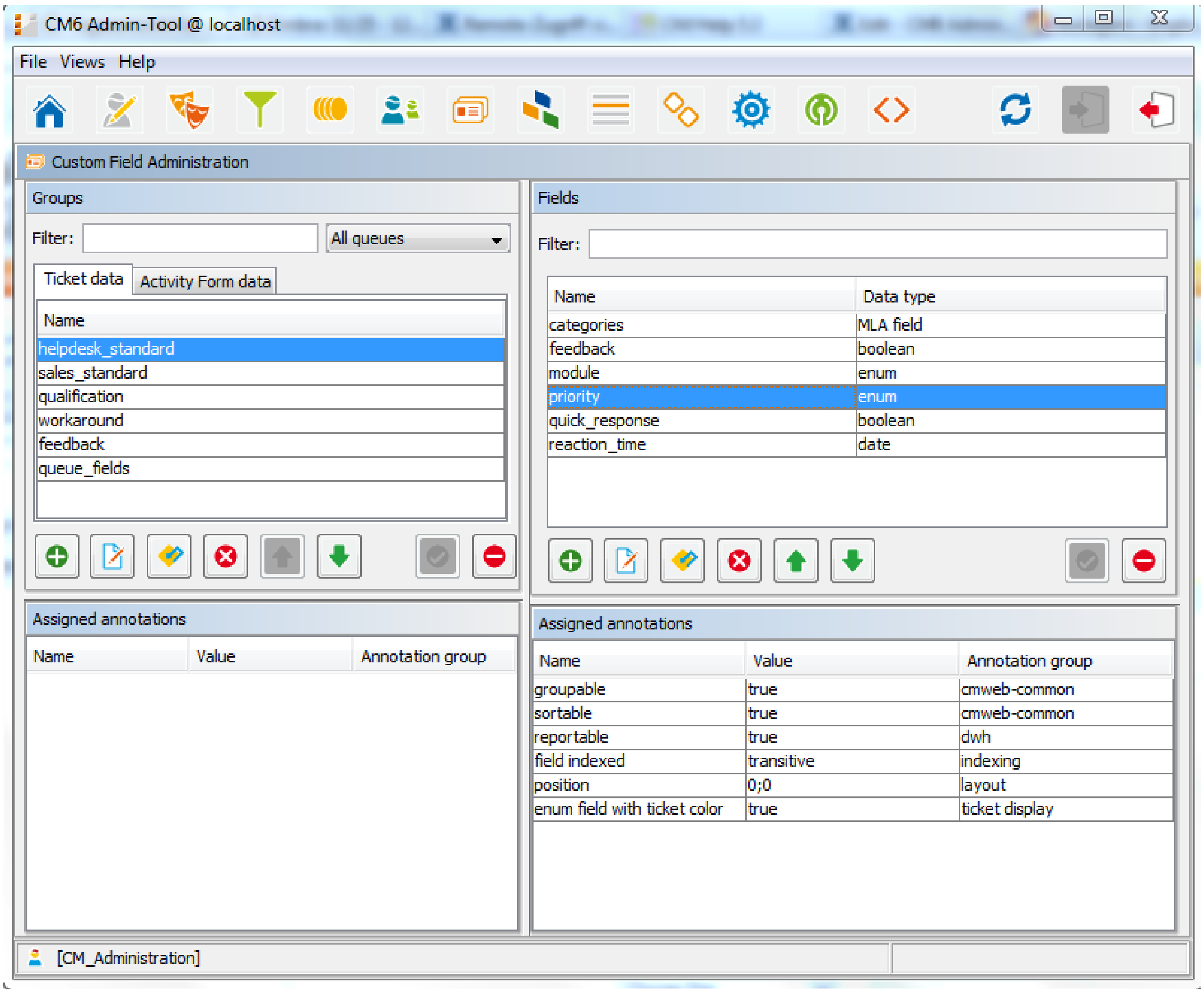
Fig. 7: ConSol*CM Web Client - New Ticket without Default Values

Without a default script, no value is set for the priority when an engineer creates a new ticket in the Web Client.

To define a default value, the script of type *Default values* has to be created. First, we have to look up where the respective custom field is to be found (in which custom field group and under which name) within the *Custom Field Administration*. See section Custom Field Administration for details.

Information:

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!



Name
helpdesk_standard
sales_standard
qualification
workaround
feedback
queue_fields

Name	Data type
categories	MLA field
feedback	boolean
module	enum
priority	enum
quick_response	boolean
reaction_time	date

Assigned annotations

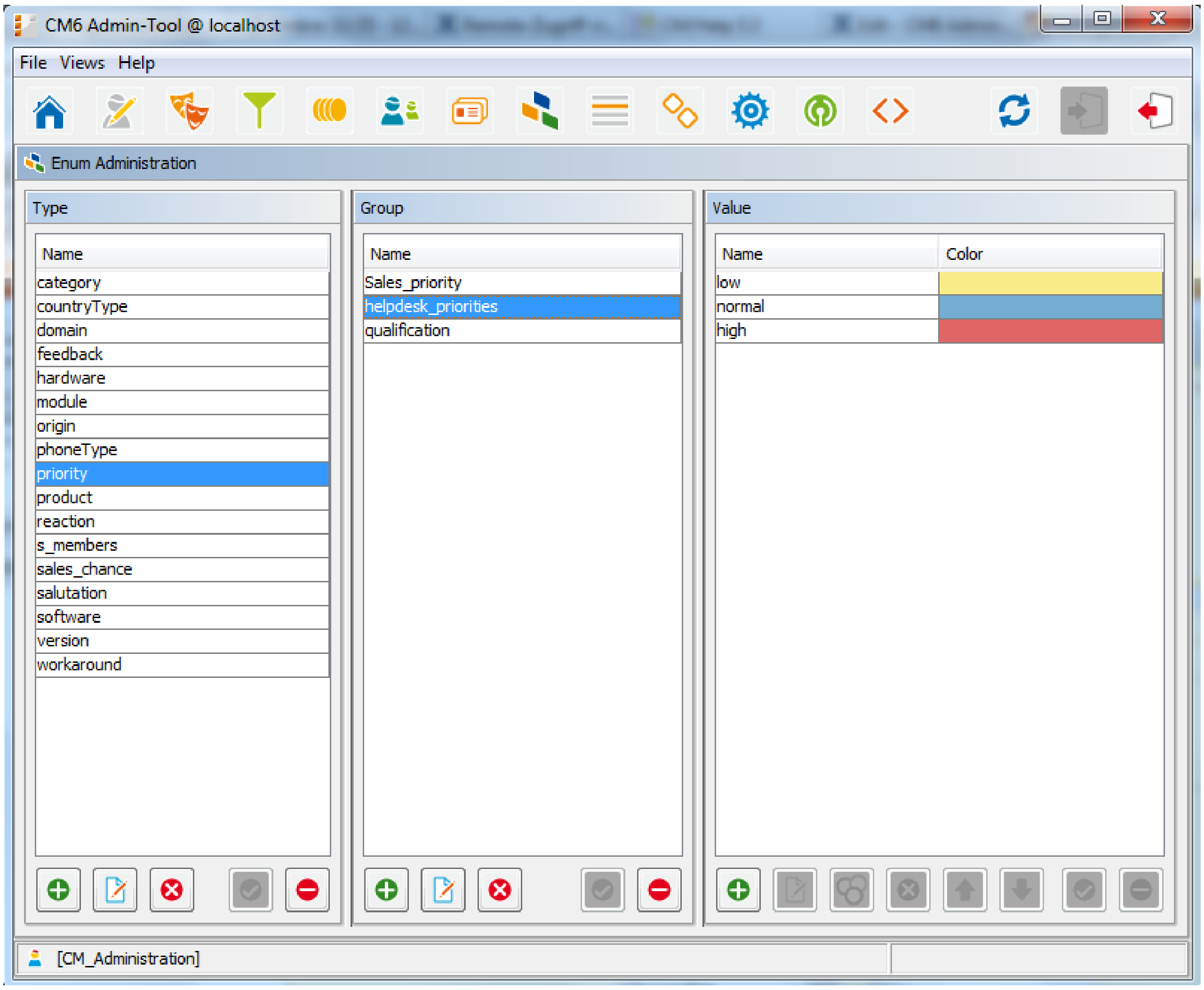
Name	Value	Annotation group

Assigned annotations

Name	Value	Annotation group
groupable	true	cmweb-common
sortable	true	cmweb-common
reportable	true	dwh
field indexed	transitive	indexing
position	0;0	layout
enum field with ticket color	true	ticket display

Fig. 8: ConSol*CM Admin Tool - Custom Field Administration

Since *priority* is an *enum* field (i.e. an ordered list), we have to check the possible values that can appear in the list and to look for the required default value within the *Enum Administration*.



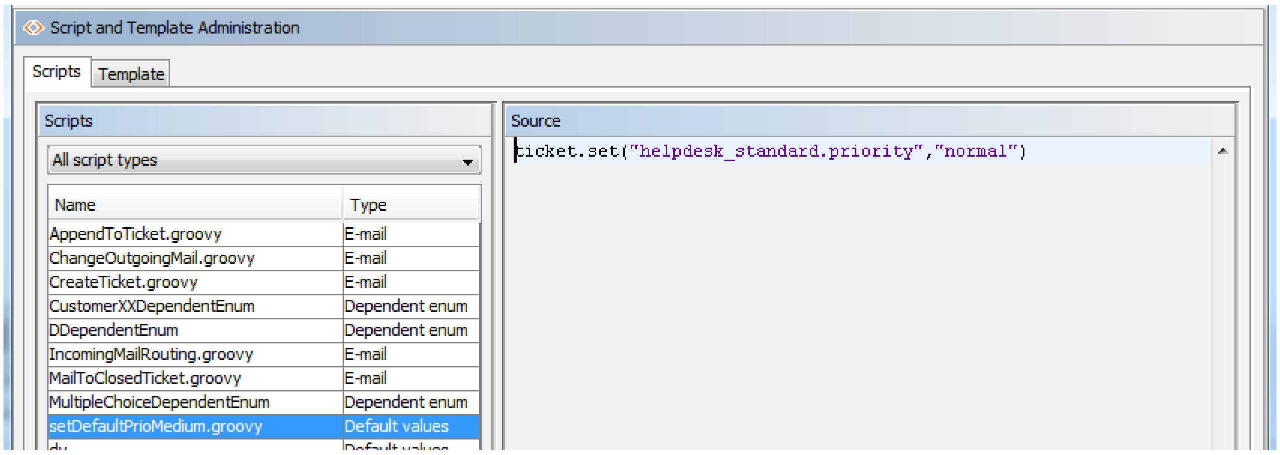
Type: Name
category
countryType
domain
feedback
hardware
module
origin
phoneType
priority
product
reaction
s_members
sales_chance
salutation
software
version
workaround

Group: Name
Sales_priority
helpdesk_priorities
qualification

Name	Color
low	🟨
normal	🟦
high	🟥

Fig. 9: ConSol*CM Admin Tool - Priority Values in Enum Administration

The group, the field, and the correct value can then be used in the respective Java method in the new script.

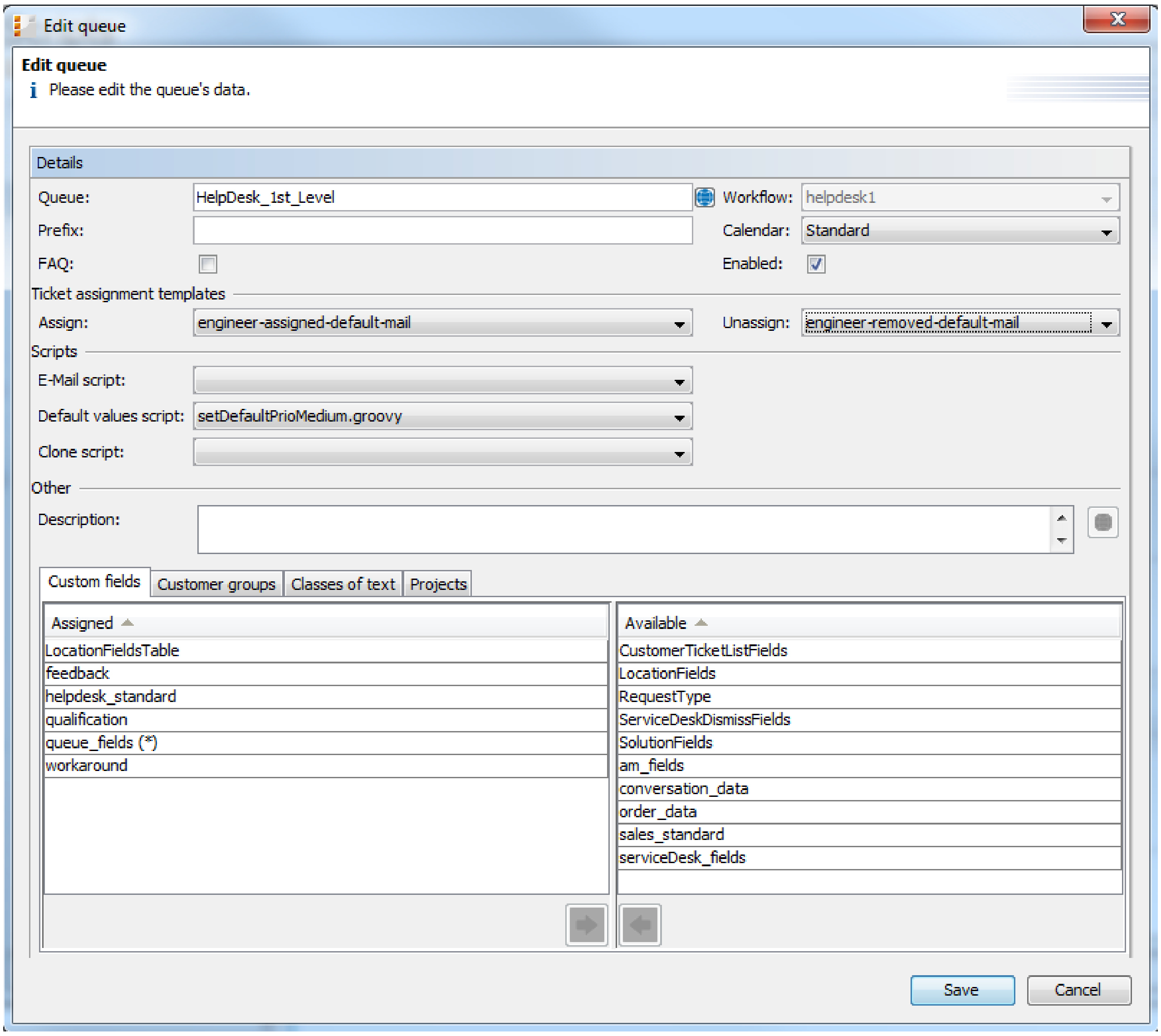


Name	Type
AppendToTicket.groovy	E-mail
ChangeOutgoingMail.groovy	E-mail
CreateTicket.groovy	E-mail
CustomerXXDependentEnum	Dependent enum
DDependentEnum	Dependent enum
IncomingMailRouting.groovy	E-mail
MailToClosedTicket.groovy	E-mail
MultipleChoiceDependentEnum	Dependent enum
setDefaultPrioMedium.groovy	Default values

```
ticket.set("helpdesk_standard.priority","normal")
```

Fig. 10: ConSol*CM Admin Tool - Include Group, Field, and Value in Script

In the *Queue Administration* we have to assign the script to the queue where the default value should be used.



Edit queue

Edit queue

Please edit the queue's data.

Details

Queue:	HelpDesk_1st_Level	Workflow:	helpdesk1
Prefix:		Calendar:	Standard
FAQ:	☐	Enabled:	✓

Ticket assignment templates

Assign:	engineer-assigned-default-mail	Unassign:	engineer-removed-default-mail

Scripts

E-Mail script:	
Default values script:	setDefaultPrioMedium.groovy
Clone script:	

Other

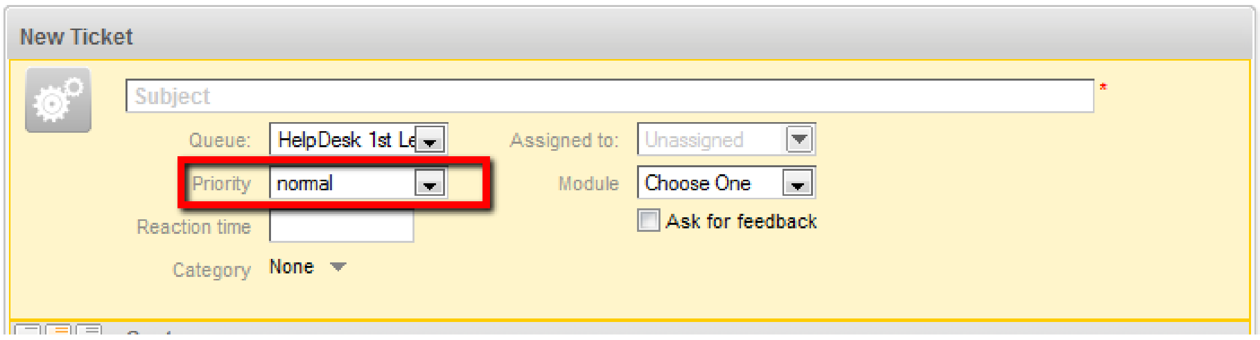
Description:

Custom fields | Customer groups | Classes of text | Projects

Assigned	Available
LocationFieldsTable	CustomerTicketListFields
feedback	LocationFields
helpdesk_standard	RequestType
qualification	ServiceDeskDismissFields
queue_fields (*)	SolutionFields
workaround	am_fields
	conversation_data
	order_data
	sales_standard
	serviceDesk_fields

Fig. 11: ConSol*CM Admin Tool - Assign Default Values Script to Queue in Queue Administration

When the engineer starts the *create ticket* operation now in the Web Client, the default value *normal* will be set in the *Priority* field.



New Ticket

	Subject		
Queue:	HelpDesk 1st Le	Assigned to:	Unassigned
Priority	normal	Module	Choose One
Reaction time			☐ Ask for feedback
Category	None		

Fig. 12: ConSol*CM Web Client - New Ticket with Default Value

Important information:

Please keep in mind that for every queue there can be only one *Default values* script. So if you have to define various default values, they have to be defined in one script. You might want to adapt the script name in this case.

In case the same default value has to be set in different queues and is set together with other default values, this has to be coded in one script for each queue also.

Overwrite Mode for Default Values Scripts

By default the fields of a ticket that are pre-filled by a *Default values* script are not overwritten, i.e. when the ticket is sent to another queue that has a different *Default values* script assigned, this second default script will try to set fields that were already filled by the first script. Since this is not possible the default value of the first script will be persistent.

If a *Default values* script should overwrite already existing values, the *overwrite* mode has to be activated. To activate this mode insert the following code at the beginning of your *Default values* script:

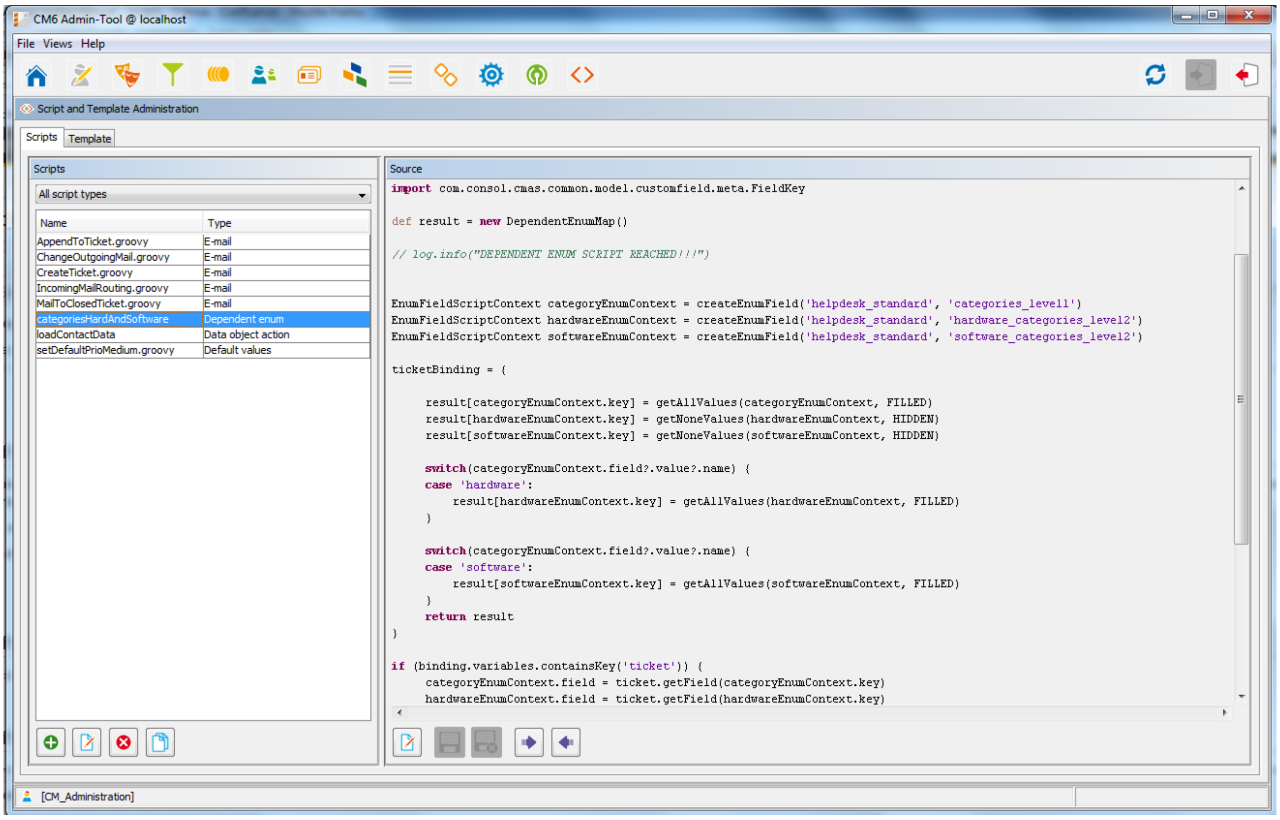
```
import com.consol.cmas.common.model.ticket.TicketPrototypeContext
TicketPrototypeContext.enableOverwriteMode()
```

Scripts of Type Dependent Enum

Dependent enums are hierarchical lists which provide a data structure similar to the one provided by *MLAs* (see section MLA Administration). In contrast to MLAs, with dependent enums only one level at a time is displayed. Depending on the value the user has selected in the list on this level, another list, the one on the sub-level, is opened. There do not have to be sub-lists for every list entry, so in graph notation, the list might be a combination of nodes and leaves. A dependent enum script is assigned to the custom field group where it is required.

Please see the following example:
In help desk tickets a category can be selected. When the user has selected *hardware* a hardware sub-list is displayed, when the user has selected *software*, the software sub-list is displayed. All custom fields have first to be defined as regular *enum* fields. In the script, the value of the first level list is checked and, depending on this value, another list is displayed in the second level. This can be used for as many levels as required, but please keep in mind that the editing of the script will become more and more complex.

The *Dependent enum* script is placed in the Admin Tool. Please ask our CM consultants for support when creating and/or editing the script since this is a rather complex task.

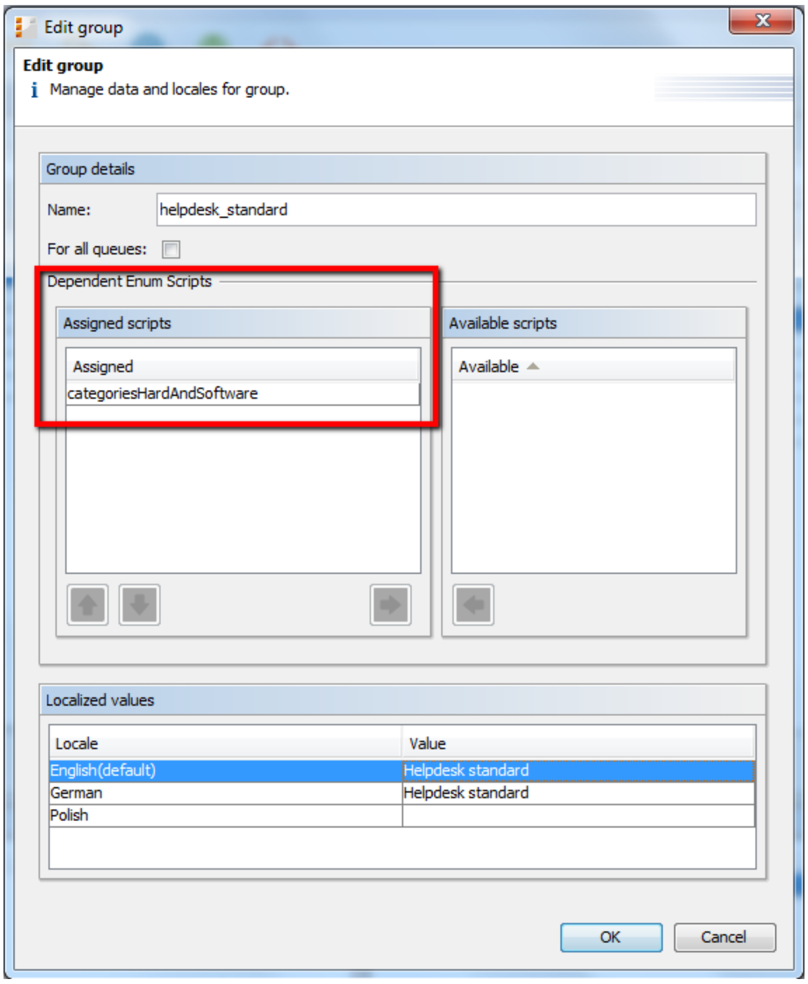


Name	Type
AppendToTicket.groovy	E-mail
ChangeOutgoingMail.groovy	E-mail
CreateTicket.groovy	E-mail
IncomingMailRouting.groovy	E-mail
MailToClosedTicket.groovy	E-mail
categoriesHardAndSoftware	Dependent enum
loadContactData	Data object action
setDefaultPrioMedium.groovy	Default values

```
import com.consol.cmas.common.model.customfield.meta.FieldKey

def result = new DependentEnumMap()

// log.info("DEPENDENT ENUM SCRIPT REACHED!!!")


EnumFieldScriptContext categoryEnumContext = createEnumField('helpdesk_standard', 'categories_level1')
EnumFieldScriptContext hardwareEnumContext = createEnumField('helpdesk_standard', 'hardware_categories_level2')
EnumFieldScriptContext softwareEnumContext = createEnumField('helpdesk_standard', 'software_categories_level2')

ticketBinding = {

     result[categoryEnumContext.key] = getAllValues(categoryEnumContext, FILLED)
     result[hardwareEnumContext.key] = getNoneValues(hardwareEnumContext, HIDDEN)
     result[softwareEnumContext.key] = getNoneValues(softwareEnumContext, HIDDEN)

     switch(categoryEnumContext.field?.value?.name) {
     case 'hardware':
         result[hardwareEnumContext.key] = getAllValues(hardwareEnumContext, FILLED)
     }

     switch(categoryEnumContext.field?.value?.name) {
     case 'software':
         result[softwareEnumContext.key] = getAllValues(softwareEnumContext, FILLED)
     }
     return result
}

if (binding.variables.containsKey('ticket')) {
     categoryEnumContext.field = ticket.getField(categoryEnumContext.key)
     hardwareEnumContext.field = ticket.getField(hardwareEnumContext.key)
```

Fig. 13: ConSol*CM Admin Tool - Dependent Enum Script

The *Dependent enum* script is assigned to the custom field group where it is required.



Edit group

Edit group

Manage data and locales for group.

Group details

Name: helpdesk_standard

For all queues: ☐

Dependent Enum Scripts

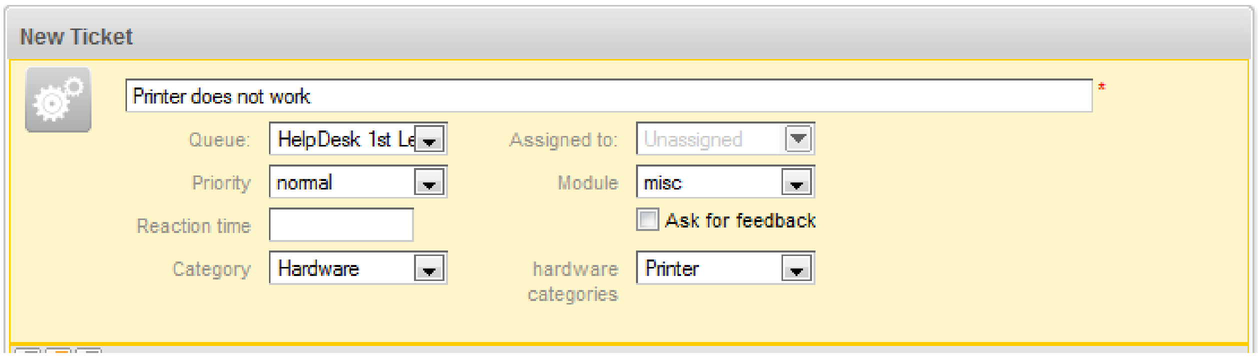
Assigned scripts	Available scripts
Assigned	Available
categoriesHardAndSoftware	

Localized values

Locale	Value
English(default)	Helpdesk standard
German	Helpdesk standard
Polish	

Fig. 14: ConSol*CM Admin Tool - Assign Dependent Enum Script to Custom Field Group

In the Web Client the engineer only sees the sub-list of the value selected in the first level list.

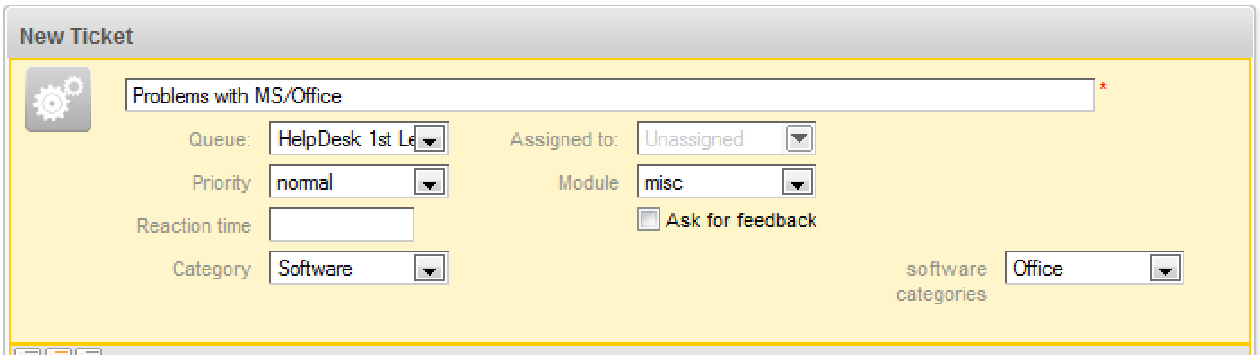


New Ticket

Printer does not work *

Queue:	HelpDesk 1st Le	Assigned to:	Unassigned
Priority	normal	Module	misc
Reaction time			☐ Ask for feedback
Category	Hardware	hardware categories	Printer

Fig. 15: ConSol*CM Web Client - Sub-List of Category Hardware


New Ticket

Problems with MS/Office *

Queue:	HelpDesk 1st Le	Assigned to:	Unassigned
Priority	normal	Module	misc
Reaction time			Ask for feedback
Category	Software	software categories	Office

Fig. 16: ConSol*CM Web Client - Sub-List of Category Software

Scripts of Type E-Mail

Scripts of this type are used for several functionalities. 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.

IMPORTANT INFORMATION

In ConSol*CM version 6.9.4, there are two modes to receive incoming e-mails:

- **Mule/ESB** - this has also been available in all previous CM versions
- **NIMH** (New Incoming Mail Handler) - new in version 6.9.4

For all configurations/settings which are valid for both modes, no further notes are added. For all settings which vary depending on the mode, this will be explained in separate (i.e. Mule/ESB or NIMH specific) sections.

E-Mail Scripts for the Processing of Incoming E-Mails in ESB/Mule Mail Mode

When an e-mail is received by ConSol*CM, it is processed by several scripts, see following figure.

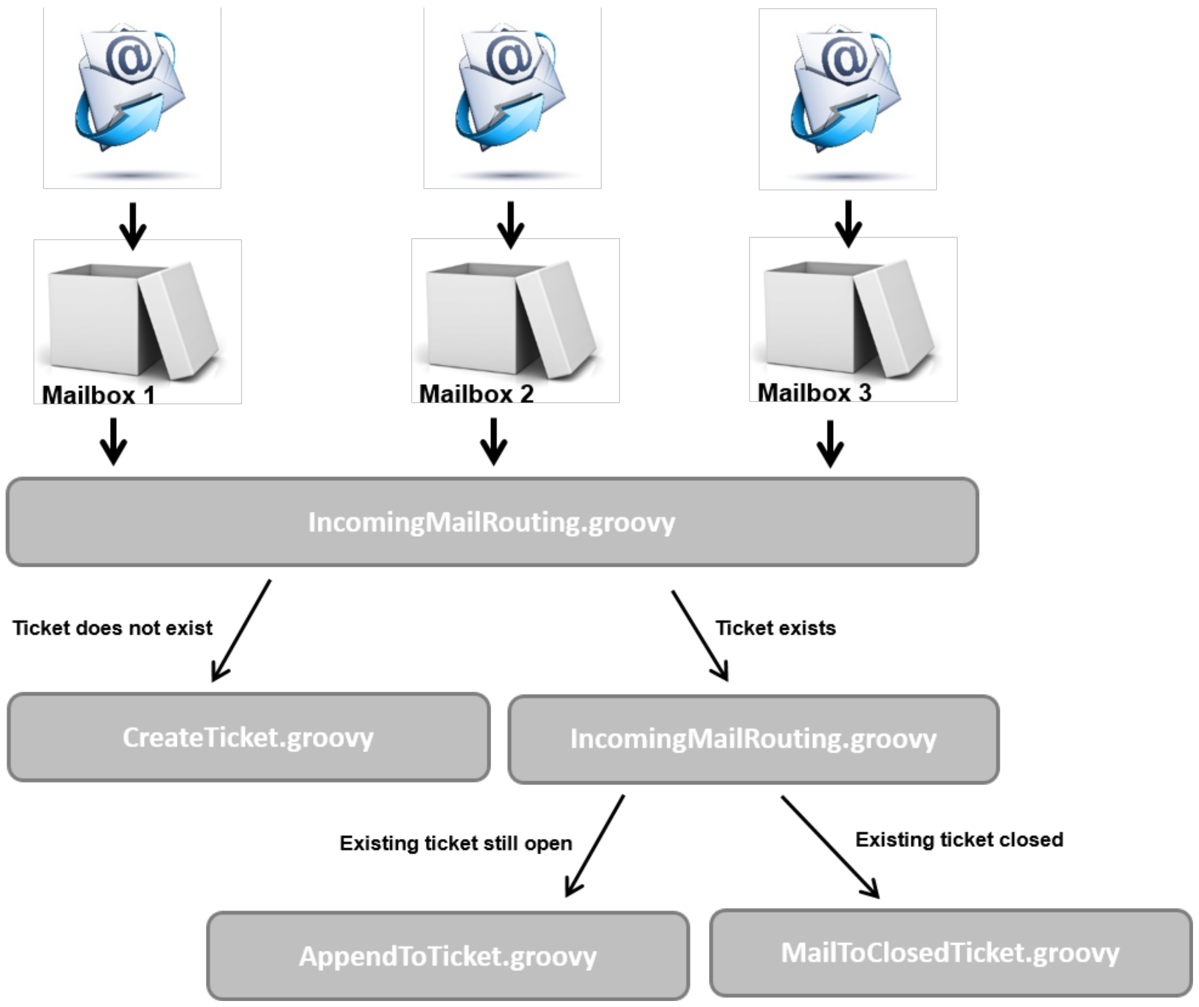


Fig. 17: ConSol*CM Admin Tool - E-Mail Scripts (ESB/Mule Mail)

- **IncomingMailRouting.groovy**
  Standard script. This is the first script that is executed when an e-mail comes in. Here, it is decided if a new ticket has to be created or if the e-mail concerns an existing open ticket (then *AppendToTicket.groovy* is executed) or if it concerns a closed ticket (then *MailToClosedTicket.groovy* is executed). This script does not require any changes to adapt it for a customer-specific environment.
- **CreateTicket.groovy**
  Standard script which is responsible for the creation of a ticket when an e-mail has been received in one of the ConSol*CM-configured mailboxes (see section Tab E-Mail for details). When the ticket subject does not match the regular expression for appending the e-mail to an existing ticket, this script is performed. All e-mails which are received by ConSol*CM (and have not been assigned to an existing ticket) are processed here, no matter from which mailbox they have been collected. In the script, the default queue for incoming e-mails has to be defined, more values of custom fields can be defined (like e.g. the default priority for e-mail tickets). Or decisions can be made in which queue the new ticket should be created, depending on the TO-address or other parameters. So usually, this script has to be adapted heavily. Please ask a CM consultant for support concerning this task.
- **AppendToTicket.groovy**
  Standard script which is responsible for appending an e-mail to a ticket that already exists. The

assignment of the e-mail to the ticket is performed using the comparison between the ticket subject and the required regular expression. Please see section Tab E-Mail for a detailed explanation of this context. Usually no changes are required for this script.

- **MailToClosedTicket.groovy**
  Standard script which is responsible for handling the e-mail when it concerns a closed ticket. The default system behavior is to create a new ticket for the customer (sender of the e-mail) and to create a reference to the old/closed ticket. So usually, no changes are required in this script.

E-Mail Scripts for the Processing of Incoming E-Mails in NIMH Mode

When an e-mail is received by ConSol*CM, it is processed by several scripts, see following figure.

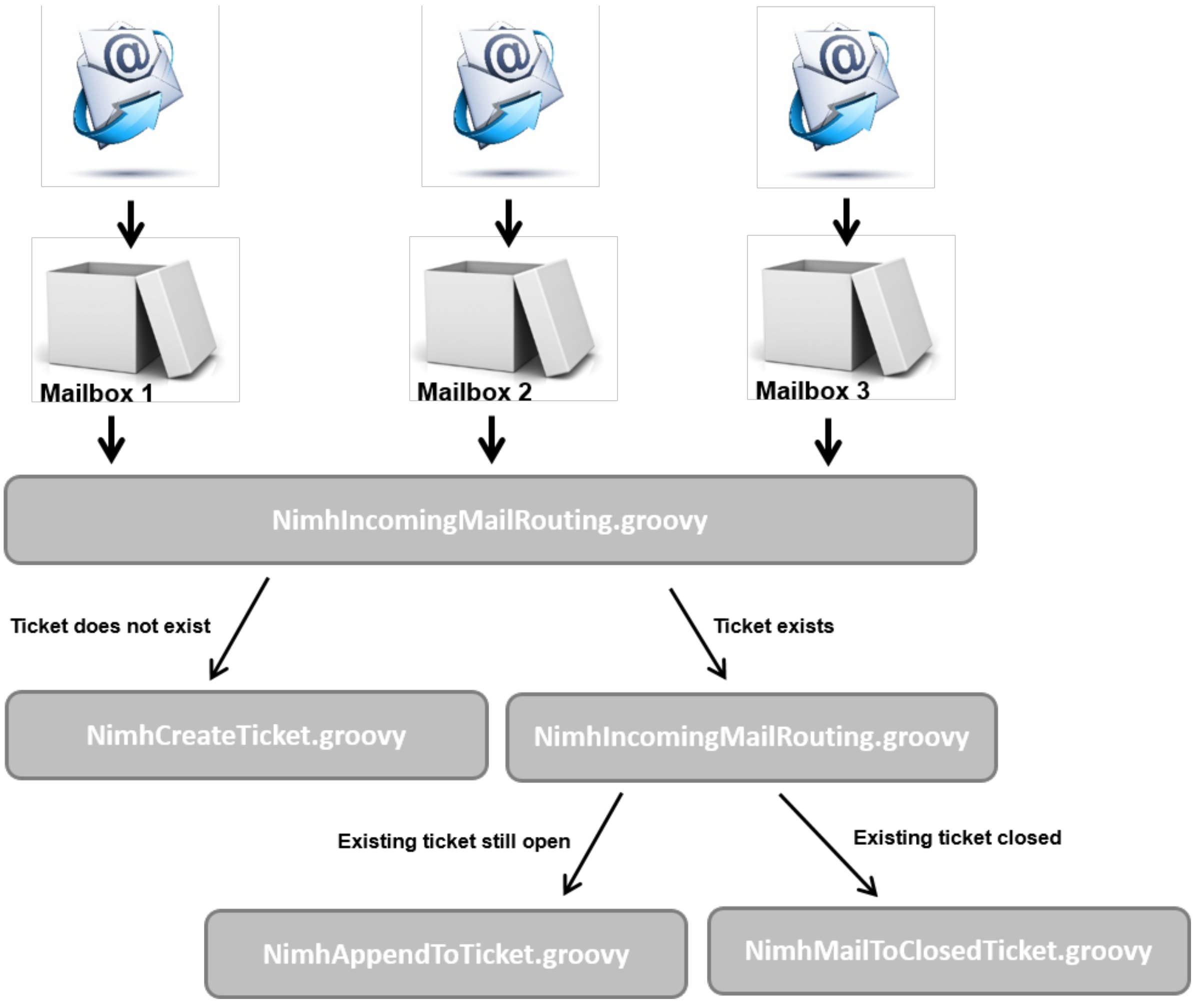

Mailbox 1
Mailbox 2
Mailbox 3
NimhIncomingMailRouting.groovy
Ticket does not exist
Ticket exists
NimhCreateTicket.groovy
NimhIncomingMailRouting.groovy
Existing ticket still open
Existing ticket closed
NimhAppendToTicket.groovy
NimhMailToClosedTicket.groovy

Fig. 18: ConSol*CM Admin Tool - E-Mail Scripts (NIMH)

- **NimhIncomingMailRouting.groovy**
  Standard script. This is the first script that is executed when an e-mail comes in. Here, it is decided if a new ticket has to be created or if the e-mail concerns an existing open ticket (then *AppendToTicket.groovy* is executed) or if it concerns a closed ticket (then *MailToClosedTicket.groovy* is executed). This script does not require any changes to adapt it for a customer-specific environment.
- **NimhCreateTicket.groovy**
  Standard script which is responsible for the creation of a ticket when an e-mail has been received in one of the ConSol*CM-configured mailboxes (see section Tab E-Mail for details). When the ticket subject does not match the regular expression for appending the e-mail to an existing ticket, this

script is performed. All e-mails which are received by ConSol*CM (and have not been assigned to an existing ticket) are processed here, no matter from which mailbox they have been collected. In the script, the default queue for incoming e-mails has to be defined, more values of custom fields can be defined (like e.g. the default priority for e-mail tickets). Or decisions can be made in which queue the new ticket should be created, depending on the *TO*-address or other parameters. So usually, this script has to be adapted heavily. Please ask a CM consultant for support concerning this task.

- **NimhAppendToTicket.groovy**
  Standard script which is responsible for appending an e-mail to a ticket that already exists. The assignment of the e-mail to the ticket is performed using the comparison between the ticket subject and the required regular expression. Please see section Tab E-Mail for a detailed explanation of this context. Usually no changes are required for this script.
- **NimhMailToClosedTicket.groovy**
  Standard script which is responsible for handling the e-mail when it concerns a closed ticket. The default system behavior is to create a new ticket for the customer (sender of the e-mail) and to create a reference to the old/closed ticket. So usually, no changes are required in this script.

Differences between Scripts in ESB/Mule Mail and NIMH

When NIMH is enabled, other Groovy classes have to be used in the e-mail scripts than in ESB/Mule Mail mode. Please see the mapping in the table below.

Mule	NIMH
mailContextService	mailSupportService
msg	mailHolder
mailLog	mailLog
spring cm services	spring cm services
all places where mailHolder ist used	pipeContext (used as parameter in mailSupportService invocations)

The steps you have to perform when you want to switch your ConSol*CM system from ESB/Mule Mail mode to NIMH mode are described in section E-Mail, Switching from ESB/Mule Mail to NIMH.

E-Mail Scripts for Outgoing E-Mails

For every queue, an *E-mail* script can be configured. Please see section Queue Administration for an explanation how to configure this. An e-mail which is written from a ticket in this queue (automatically by the workflow or manually by an engineer) passes through this script before it leaves the ConSol*CM system. So in this *E-mail* script you can change or set queue-specific settings for the outgoing e-mail. A common use case is the setting of a queue-specific REPLY-TO-address in order to use team-specific REPLY-TO-addresses.

An example of an outgoing *E-mail* script is the following script which is part of the CM default application:

- **ChangeOutgoingMail.groovy**
  Standard script that is not in operation but serves as a template for outgoing *E-mail* scripts. You might want to use them to configure queue-specific *E-mail* scripts.

Within the outgoing *E-mail* script, the Java object *mailEntry* is implicitly available as object *mail*. You have to set all required attributes for the outgoing e-mail using the *mail.setAttribute()* or *mail.setAttributes()* methods.

Example:

```
def queueReplyAddress = "serviceteam@mycompany.com"
// you might also use system properties for the queue-specific e-mail addresses and fetch an
// address using the configurationService!
mail.setAttribute('Reply-to', queueReplyAddress)
```

Common e-mail attributes are:

- BCC
- From
- Reply-to
- To
- CC
- Subject

In case you would like to read a very detailed description of the e-mail format, please refer to RFC 5322.

When you set the **REPLY-TO-address** in the queue-specific outgoing e-mail script, the *mail.reply.to* system property must not be set (because it would overwrite the configured value)! Leave the value for *mail.reply.to* empty.

That means when you use a queue-specific outgoing e-mail script for *one* queue you have to define outgoing e-mail scripts for *all* queues because the *mail.reply.to* property can no longer be used! In this case you will have to perform the following steps for each ⚠ queue where e-mails are relevant:

1. Create the queue-specific outgoing e-mail script for each specific queue, e.g. *ChangeOutgoingMail.groovy_Queue1,ChangeOutgoingMail.groovy_Queue2, ... ChangeOutgoingMail.groovy_Queue_n.*
2. Maybe create an outgoing e-mail script which should be used for all queues where no queue-specific script is required, e.g. *ChangeOutgoingMail.groovy_Queue_standard*
3. Assign this script to the queues in the Queue Administration.

In the **CM Web Client**, you will find the following behavior:

- When the REPLY-TO-address is set using the *mail.reply.to* system property, the *Reply to* field is displayed in the Ticket E-Mail Editor and pre-filled with the value of the system property, e.g. mymailaddress@consol.de. The engineer can use the field value or can also change it (which is not recommended!)
- When the *mail.reply.to* system property is empty, the *Reply to* field is not displayed in the Ticket E-Mail Editor in *edit* mode. In *display* mode, the REPLY-TO-address which was used is displayed. The REPLY-TO-address has to be set in the queue-specific outgoing e-mail script. If it is not set, either because an outgoing e-mail script is present where the REPLY-TO-address is not set or because there is no outgoing e-mail script defined for the queue, the e-mail is sent without a specific REPLY-TO-address (NULL).

Scripts of Type Page Customization

See section Configuration of Web Client Dashboard.

Scripts of Type Task

See section The Task Execution Framework (TEF)

Scripts of Type Workflow

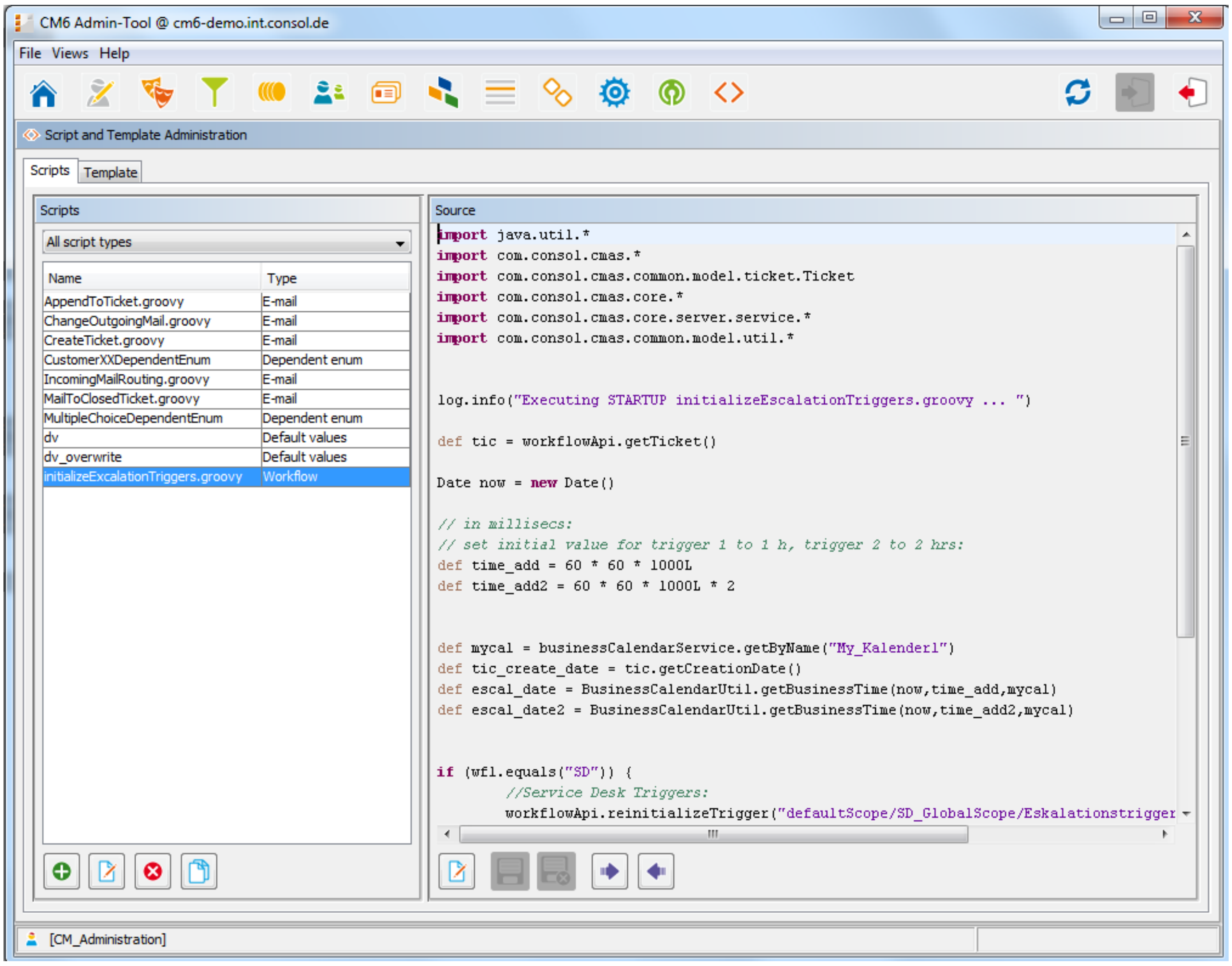
Scripts of that type are stored in the Admin Tool, because they are used in numerous workflow scripts, i.e. the code in the Admin Tool script is needed more than once in one or more workflow(s). It is easier, less error-prone, and less time-consuming to store the scripts at one central location (Admin Tool) and just reference them in the workflow(s) than to edit the same code at different locations in every workflow where it is used. Furthermore, during workflow development the Admin Tool script can be modified easily and the change is in operation at once whereas when editing a workflow it has to be deployed first.

Please see the *ConSol*CM Process Designer Manual* for a detailed introduction to workflow programming. A short example will be provided here.

This code in a workflow activity will only reference the script, e.g.:

```
scriptExecutionService.execute(scriptProviderService.createDatabaseProvider
("initializeEscalationTriggers.groovy"))
```

In the Admin Tool, the respective script is stored:



Name	Type
AppendToTicket.groovy	E-mail
ChangeOutgoingMail.groovy	E-mail
CreateTicket.groovy	E-mail
CustomerXXDependentEnum	Dependent enum
IncomingMailRouting.groovy	E-mail
MailToClosedTicket.groovy	E-mail
MultipleChoiceDependentEnum	Dependent enum
dv	Default values
dv_overwrite	Default values
initializeExcalationTriggers.groovy	Workflow

```
import java.util.*
import com.consol.cmas.*
import com.consol.cmas.common.model.ticket.Ticket
import com.consol.cmas.core.*
import com.consol.cmas.core.server.service.*
import com.consol.cmas.common.model.util.*


log.info("Executing STARTUP initializeEscalationTriggers.groovy ... ")

def tic = workflowApi.getTicket()

Date now = new Date()

// in millisecs:
// set initial value for trigger 1 to 1 h, trigger 2 to 2 hrs:
def time_add = 60 * 60 * 1000L
def time_add2 = 60 * 60 * 1000L * 2


def mycal = businessCalendarService.getByName("My_Kalender1")
def tic_create_date = tic.getCreationDate()
def escal_date = BusinessCalendarUtil.getBusinessTime(now,time_add,mycal)
def escal_date2 = BusinessCalendarUtil.getBusinessTime(now,time_add2,mycal)


if (wfl.equals("SD")) {
        //Service Desk Triggers:
        workflowApi.reinitializeTrigger("defaultScope/SD_GlobalScope/Eskalationstrigger
```

Fig. 19: ConSol*CM Admin Tool - Workflow Script

It is also possible to pass parameters (key-value pairs) to the Admin Tool script. This is explained in detail in the *ConSol*CM Process Designer Manual*.

Default Workflow Activity Script

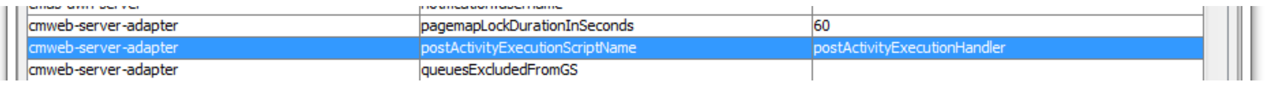
For certain use cases it might be required to execute a script when a ticket has run through a workflow activity. You might want to use this to display another ticket in the GUI after the workflow activity has been executed. From a user's (engineer's) point of view, the GUI *jumps* to the next ticket. The latter can be a child ticket or another ticket in a list, depending on the use case.

The system behavior is defined in an Admin Tool script. The name of the script has to be set in the system property *cmweb-server-adapter*, *postActivityExecutionScriptName*, see Appendix C (System Properties).

This script is executed after every ⚠ workflow activity. That means you have to insert all control mechanisms and *intelligence* into the script:

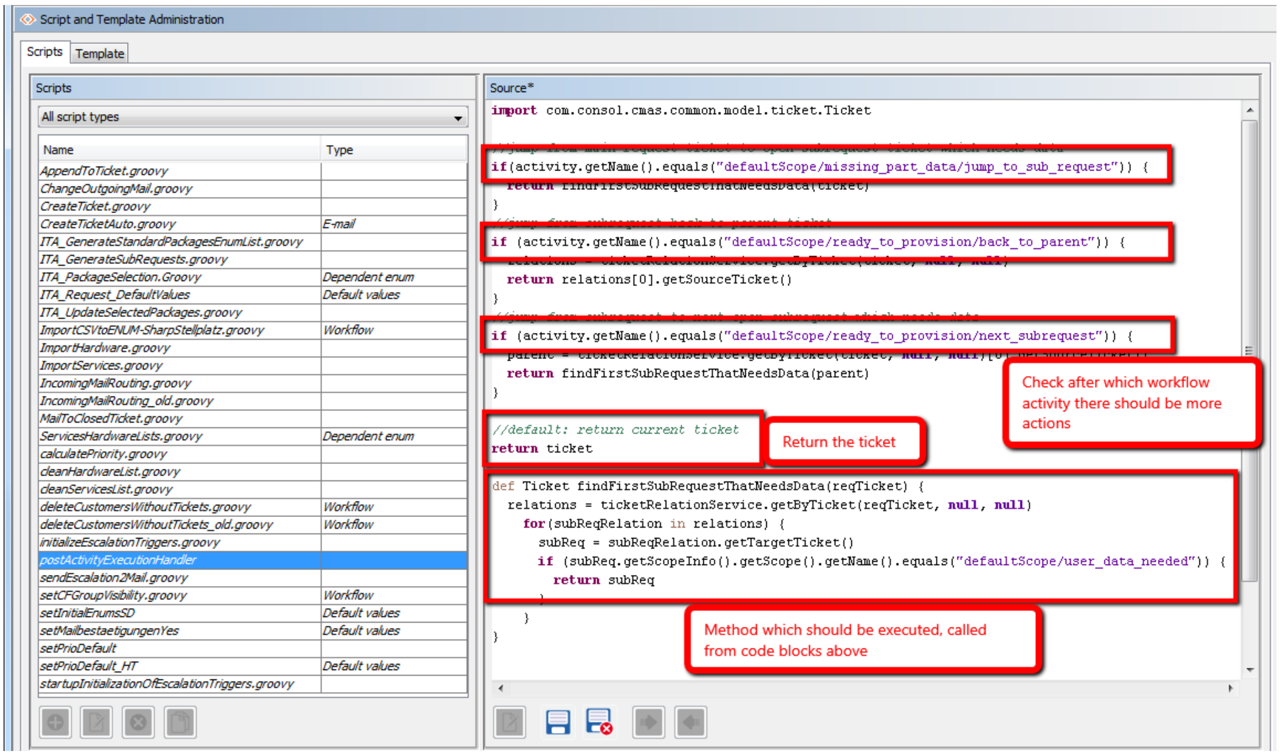
- After which activity the script should do something?
  (for all other activities, nothing will happen)
- What should happen?

Example: Jump to the next ticket in a list.



cmweb-server-adapter	pagemapLockDurationInSeconds	60
cmweb-server-adapter	postActivityExecutionScriptName	postActivityExecutionHandler
cmweb-server-adapter	queuesExcludedFromGS	

Fig. 20: Property for Definition of postActivityExecutionScriptName


Script and Template Administration
Scripts
Template
Scripts
All script types
Source*
Check after which workflow activity there should be more actions
Return the ticket
Method which should be executed, called from code blocks above

Name	Type
AppendToTicket.groovy	
ChangeOutgoingMail.groovy	
CreateTicket.groovy	
CreateTicketAuto.groovy	E-mail
ITA_GenerateStandardPackagesEnumList.groovy	
ITA_GenerateSubRequests.groovy	
ITA_PackageSelection.Groovy	Dependent enum
ITA_Request_DefaultValues	Default values
ITA_UpdateSelectedPackages.groovy	
ImportCSVtoENUM-SharpStellplatz.groovy	Workflow
ImportHardware.groovy	
ImportServices.groovy	
IncomingMailRouting.groovy	
IncomingMailRouting_old.groovy	
MailToClosedTicket.groovy	
ServicesHardwareLists.groovy	Dependent enum
calculatePriority.groovy	
cleanHardwareList.groovy	
cleanServicesList.groovy	
deleteCustomersWithoutTickets.groovy	Workflow
deleteCustomersWithoutTickets_old.groovy	Workflow
initializeEscalationTriggers.groovy	
postActivityExecutionHandler	
sendEscalation2Mail.groovy	
setCFGroupVisibility.groovy	Workflow
setInitialEnumsSD	Default values
setMailbestaetigungenYes	Default values
setPrioDefault	
setPrioDefault_HT	Default values
startupInitializationOfEscalationTriggers.groovy	

```
import com.consol.cmas.common.model.ticket.Ticket

//jump from main request ticket to open subrequest ticket which needs data
if(activity.getName().equals("defaultScope/missing_part_data/jump_to_sub_request")) {
  return findFirstSubRequestThatNeedsData(ticket)
}
//jump from subrequest back to parent ticket
if (activity.getName().equals("defaultScope/ready_to_provision/back_to_parent")) {
  relations = ticketRelationService.getByTicket(ticket, null, null)
  return relations[0].getSourceTicket()
}
//jump from subrequest to next open subrequest which needs data
if (activity.getName().equals("defaultScope/ready_to_provision/next_subrequest")) {
  parent = ticketRelationService.getByTicket(ticket, null, null)[0].getSourceTicket()
  return findFirstSubRequestThatNeedsData(parent)
}

//default: return current ticket
return ticket

def Ticket findFirstSubRequestThatNeedsData(reqTicket) {
  relations = ticketRelationService.getByTicket(reqTicket, null, null)
    for(subReqRelation in relations) {
      subReq = subReqRelation.getTargetTicket()
      if (subReq.getScopeInfo().getScope().getName().equals("defaultScope/user_data_needed")) {
        return subReq
      }
    }
}
```

Fig. 21: ConSol*CM Admin Tool - postActivityExecutionScript

25.3 Introduction to Templates in the Admin Tool

- The Admin Tool Template Editor
- Working with Admin Tool Templates
    - System Templates
    - Templates for Definition of Contact Format in GUI
    - Ticket Assignment Templates
    - Custom Defined Templates

In ConSol*CM, several types of templates are used:

- **E-mail and text templates**
  Stored either in the *Template Manager* or in the *Script and Template Administration* of the Admin Tool.
- **Non-e-mail templates**
  (e.g. templates for representation of contact data)
  Stored in the *Script and Template Administration* of the Admin Tool only.

In this chapter, the templates in the *Script and Template Administration* of the Admin Tool will be explained. Please see section The ConSol*CM Template Manager for a detailed introduction to the Template Manager (where e-mail and text templates are created and stored).

Admin Tool templates are written according to the *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.

25.3.1 The Admin Tool Template Editor

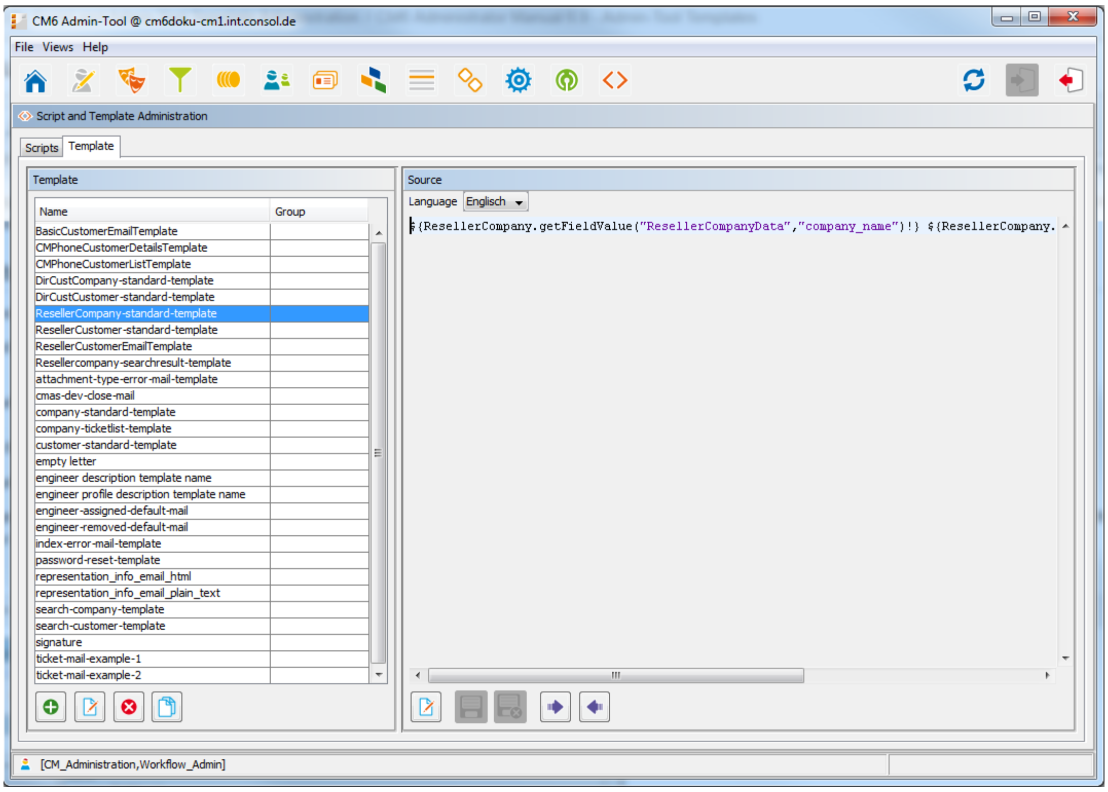
To work with templates, open the *Script and Template Administration* in the Admin Tool and switch to the *Templates* tab.

In the templates list, all templates are listed with:

- **Name**
  A template is referenced by its name when it is referenced by other objects.
- **Group**
  Groups help you sort the templates in the templates list. They do not have a technical implication.

To open a template in the editor panel, mark it in the list and open it by clicking on 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. Each template always has to be defined for the default language.


CM6 Admin-Tool @ cm6doku-cm1.int.consol.de
File Views Help
Script and Template Administration
Scripts
Template
Template
Source
Language
Englisch
[CM_Administration,Workflow_Admin]

Name	Group
BasicCustomerEmailTemplate	
CMPhoneCustomerDetailsTemplate	
CMPhoneCustomerListTemplate	
DirCustCompany-standard-template	
DirCustCustomer-standard-template	
ResellerCompany-standard-template	
ResellerCustomer-standard-template	
ResellerCustomerEmailTemplate	
Resellercompany-searchresult-template	
attachment-type-error-mail-template	
cmas-dev-close-mail	
company-standard-template	
company-ticketlist-template	
customer-standard-template	
empty letter	
engineer description template name	
engineer profile description template name	
engineer-assigned-default-mail	
engineer-removed-default-mail	
index-error-mail-template	
password-reset-template	
representation_info_email_html	
representation_info_email_plain_text	
search-company-template	
search-customer-template	
signature	
ticket-mail-example-1	
ticket-mail-example-2	

```
${ResellerCompany.getFieldValue("ResellerCompanyData","company_name")!} ${ResellerCompany.
```

Fig. 1: ConSol*CM Admin Tool - Template Editor

25.3.2 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 object group fields and custom fields are referenced by group name and field name, e.g. the company name within the data object group *ResellerCompany* will be referenced as shown in the following example.

```
${ResellerCompany.getFieldValue("ResellerCompanyData","company_name")!}
```

For a detailed explanation of the work with custom fields, please see section Custom Field Administration. Data object group fields are explained in section Setting Up the Customer Data Model.

Attention:

Do not use line breaks in template statements!

System Templates

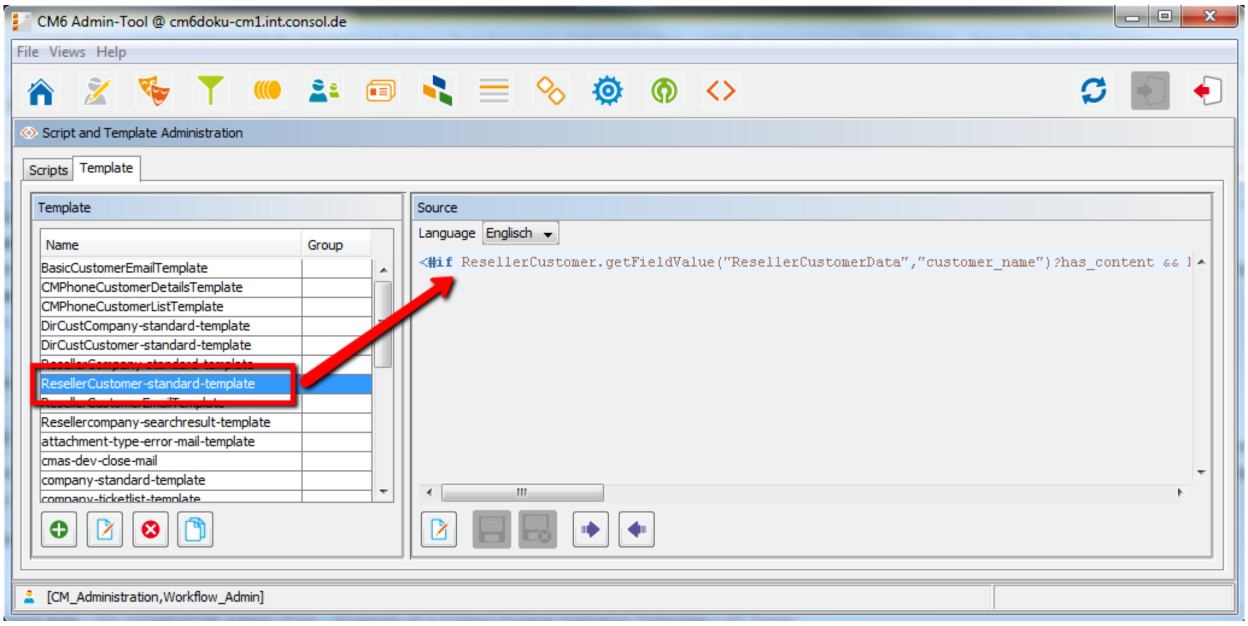
A default ConSol*CM installation comes with several system templates. They are used in standard situations like error messages to an administrator. Please see the following list for an overview of the system templates:

- **attachment-type-error-mail-template**
  An e-mail with this template is sent to the e-mail administrator (e-mail address given in system property *mail.process.error*) when the attachment type of an incoming or outgoing e-mail is not supported and thus the e-mail cannot be processed.
- **cmas-dev-close-mail**
  Not used and will be removed in one of the next ConSol*CM versions.
- **engineer description template name**
  Template used to render the engineer label, e.g. ticket owner**.**
- **engineer profile description template name**
  Template used to render the label on a header of the page, next to logout button**.**
- **index-error-mail-template**
  Not used and will be removed in one of the next ConSol*CM versions.
- **password-reset-template**
  Template for the body of the e-mail which is sent when a user requests a password reset (on login page)**.**
- **representation_info_email_html**
  All e-mails sent by CM6 to the represented engineer are also sent to the representing engineer (see *Global Permissions: Representation Permissions* in section Role Administration). The template is used to configure the text which is added to the forwarded e-mail**.**
- **representation_info_email_plain_text**
  Same as above, as plain text.

Templates for Definition of Contact Format in GUI

The appearance of contact data (e.g. name, phone number, and room number or name and forename only) in different sections of the Web Client can be formatted using templates. The definition has to be made for each data object so that specific templates can be used within each customer group. The configuration of templates for a data object is explained in section Parameters for Data Objects. Section Templates for Customer Data provides a detailed explanation of the templates used for customer data.

In the following example, the contact data within the *ResellerCustomer* data object should be represented in the standard template with first name and name.


CM6 Admin-Tool @ cm6doku-cm1.int.consol.de
File Views Help
Script and Template Administration
Scripts
Template
Template
Name
Group
Source
Language Englisch
[CM_Administration,Workflow_Admin]

BasicCustomerEmailTemplate
CMPhoneCustomerDetailsTemplate
CMPhoneCustomerListTemplate
DirCustCompany-standard-template
DirCustCustomer-standard-template
ResellerCustomer-standard-template
Resellercompany-searchresult-template
attachment-type-error-mail-template
cmas-dev-close-mail
company-standard-template

```
<#if ResellerCustomer.getFieldValue("ResellerCustomerData","customer_name")?has_content && 
```

Fig. 2: ConSol*CM Admin Tool - Example of a Contact Format Definition Template

Example of a contact format definition template

```
<#if ResellerCustomer.getFieldValue("ResellerCustomerData","customer_name")?has_content &&
ResellerCustomer.getFieldValue("ResellerCustomerData","forename")?
has_content>${ResellerCustomer.getFieldValue("ResellerCustomerData","customer_name")!},${Reselle
rCustomer.getFieldValue("ResellerCustomerData","forename")!}<#else> ${ResellerCustomer.
getFieldValue("ResellerCustomerData","customer_name")!}</#if>
```

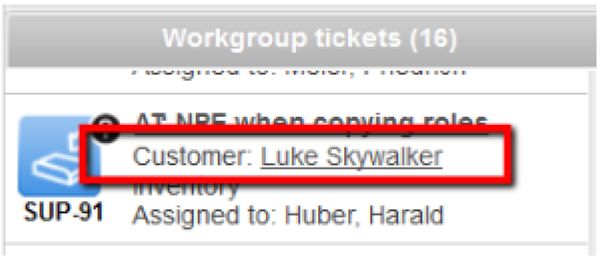
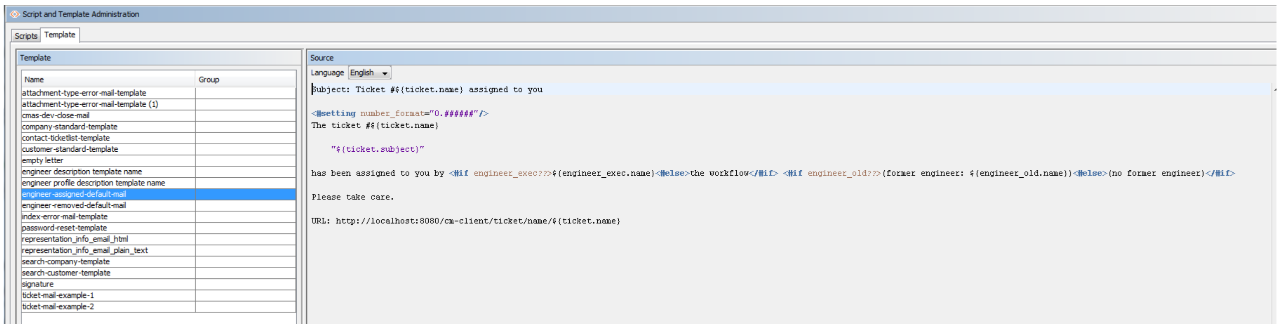


Fig. 3: ConSol*CM Web Client - Example of a Contact Format Definition Template

Ticket Assignment Templates

In the queue administration (see section Queue Administration), ticket-engineer-assignment templates can be selected. There are templates for the two use cases *assign* and *remove*. The *assign* template (*Assign*) is used as text template for an automatic e-mail, 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 e-mail which is sent by the system to the (old) engineer when a ticket has been removed from the engineer. You have to write and save the templates here in the *Template* section first. Then they will be available in the drop-down menu in the *Ticket assignment templates* section of the queue administration (see section Queue Administration).



Script and Template Administration

Scripts | Template

Template

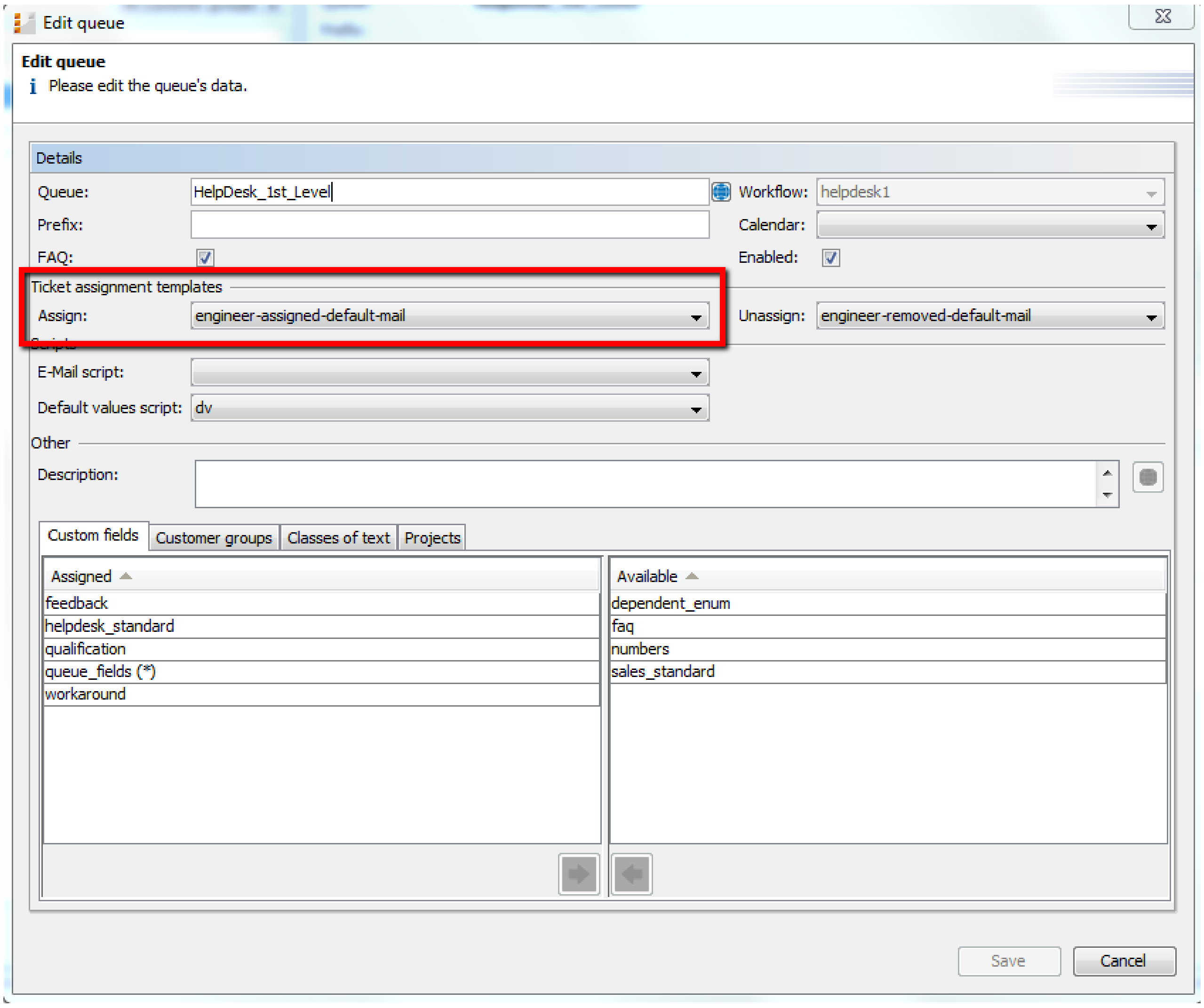
Name	Group
attachment-type-error-mail-template	
attachment-type-error-mail-template (1)	
cmas-dev-close-mail	
company-standard-template	
contact-ticketlist-template	
customer-standard-template	
empty letter	
engineer description template name	
engineer profile description template name	
engineer-assigned-default-mail	
engineer-removed-default-mail	
index-error-mail-template	
password-reset-template	
representation_info_email_html	
representation_info_email_plain_text	
search-company-template	
search-customer-template	
signature	
ticket-mail-example-1	
ticket-mail-example-2	

Source

Language: English

```
Subject: Ticket #${ticket.name} assigned to you

<#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}
```

Fig. 4: ConSol*CM Admin Tool - Example of an E-Mail Template for Ticket Assignment to an Engineer



Edit queue

Edit queue

Please edit the queue's data.

Details

Queue:	HelpDesk_1st_Level	Workflow:	helpdesk1
Prefix:		Calendar:	
FAQ:	☑	Enabled:	☑

Ticket assignment templates

Assign:	engineer-assigned-default-mail	Unassign:	engineer-removed-default-mail

Scripts

E-Mail script:	
Default values script:	dv

Other

Description:

Custom fields | Customer groups | Classes of text | Projects

Assigned	Available
feedback	dependent_enum
helpdesk_standard	faq
qualification	numbers
queue_fields (*)	sales_standard
workaround	

Fig. 5: ConSol*CM Admin Tool - Configuration of a Ticket Assignment Template for a Queue

Custom Defined Templates

A ConSol*CM administrator or workflow developer can define any template that is required and store it in the *Script and Template Administration*. When you use it in automatic e-mails which are sent by a workflow activity, you can always use the workflow API *renderTemplate()* method to reference a template. However, most e-mail templates should be managed using the Template Manager (see section The ConSol*CM Template Manager). There are only very few use cases which might require that e-mail templates or parts of e-mail templates have to be stored in the *Script and Template Administration* of the Admin Tool.

26 CM6 Administrator Manual 6.9.4 - Working with Text Templates

26.1 Working with Text 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 e-mails or ticket-comments where text, headers, and footers can be specified. Another example are documents which have to be edited using MS Word.

In both cases, the templates do not offer only texts 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 *Template Manager* for editing and managing e-mail and ticket-comment templates (see section The ConSol*CM Template Manager)

and

- *CM.Doc* for editing and managing MS Word templates (see section CM.Doc).

26.2 The ConSol*CM Template Manager

- Introduction to the Work with E-Mail and Ticket Text Templates
  - E-Mail Templates
    - Why E-Mail Templates?
    - E-Mails in ConSol*CM
  - Ticket Text Templates
    - Why Ticket Text Templates?
    - Ticket Text Templates in ConSol*CM
  - E-Mail and Ticket Text Templates in ConSol*CM
    - E-Mail and Ticket Text Template Components
    - Storage and Management of E-Mail and Ticket Text Templates
      - E-Mail Templates
      - Ticket Text Templates
- Introduction to the Template Manager
- Work with the Template Manager
  - Basic Template Manager GUI: The Template Library
    - Filter
    - Context
    - List
  - Create a New Template
    - Create a New Letter
    - The Library of Markers
    - Create a New Include or Workflow Include
    - Create New Text Blocks
    - Create and Use a Script
    - Binding Templates to Queues or to Specific Parameters
      - Hard and Soft Binding
- Migrating Templates from CM Version 6.8 and Less to CM Version 6.9 and Up
- Page Customization for E-Mail Template Functionalities

26.2.1 Introduction to the Work with E-Mail and Ticket Text Templates

Using the *Template Manager*, two types of templates can be defined:

- **E-mail templates** for e-mails written from the ConSol*CM system
  - manual e-mails (written by an engineer using the Ticket E-Mail Editor)
  - automatic e-mails initialized by the system (e.g. sent by a workflow script when a certain workflow activity is executed)
- **Text templates** for ticket texts (comments)
  - during ticket creation
  - during ticket editing

E-Mail Templates

Why E-Mail Templates?

When a system works with e-mails, several criteria have to be considered. If all those requirements are met, e-mail templates are a very helpful tool in every-day working life.

- The e-mails have to have a strictly defined layout, usually according to a company's CD (corporate design).
- The texts have to follow the company's letter/text guidelines.
- Texts that are used very frequently have to be provided by templates in order to save time and to avoid typos and other errors while typing the text.
- Customer-, system-, and engineer-specific data have to be integrated into the text.
- The template management should be performed by an administrator and/or power user. No system configuration by the software company should be required.

ConSol*CM provides the function set to take all those criteria into consideration.

E-Mails in ConSol*CM

E-mails are used for core functionalities in ConSol*CM. Those functionalities have been described in detail in chapter Tab E-mail, so here, only a short review is given.

ConSol*CM can receive and send e-mails. Sending e-mails can serve various purposes:

- **An engineer writes an e-mail directly from the ticket, using the Ticket E-Mail Editor.**
  This can be an e-mail to the customer, to a co-worker, or to any other person with a valid e-mail address. Often, there are standard texts which are used every day for several recipients. To avoid typing the same text over and over again, ConSol*CM offers e-mail templates. These are text templates where parameters like customer name, ticket number or engineer name and phone number can be integrated. When the template is used, the system fills in the parameters automatically with the valid data from the current ticket. The engineer can add more text or modify the text as required, so e-mail templates are not static, but dynamic.
  E-mails which are sent manually either do not use a template or are based on a template from the *Template Manager*. Templates from the *Script and Template Administration* of the Admin Tool are not available here.
- **The system sends an e-mail automatically.**
  This can be an internal e-mail like a reminder for an engineer when the ticket has entered the escalation status or an internal e-mail to a supervisor when a ticket needs approval to be continued. Or it can be an external e-mail to the customer like a confirmation of receipt or a notice that a ticket has been solved. The e-mail is generated automatically based on the respective e-mail template.This can be an e-mail template from the *Template Manager* or from the *Script and Template Administration* of the Admin Tool.

Ticket Text Templates

Why Ticket Text Templates?

Using ticket text templates, i.e. predefined text segments you can access when you create or edit a ticket, serves several purposes:

- You as ticket engineer save a lot of time by not typing the same text over and over again.
- You do not risk forgetting important points (e.g. in questions for a pre-qualification when you talk to your customer on the phone).
- You do not have to worry about typos.
- You do not have to look up ticket and/or customer data, because all data is integrated into the text automatically.

Ticket Text Templates in ConSol*CM

Ticket text templates are defined very similar to e-mail templates. Only the *Used within* parameter is set in a different way when the template is created using the Template Manager.

Attention:

Technically, there is no difference between e-mail and ticket text templates! So for each template you as a template manager working with the Template Manager can decide if the template should be used as ticket text template, as e-mail template, or both.

E-Mail and Ticket Text Templates in ConSol*CM

E-Mail and Ticket Text Template Components

In e-mail and ticket text templates in ConSol*CM you can use free text and all data that is available for a customer, an engineer, and/or the ticket. In section The Library of Markers all available components will be explained.

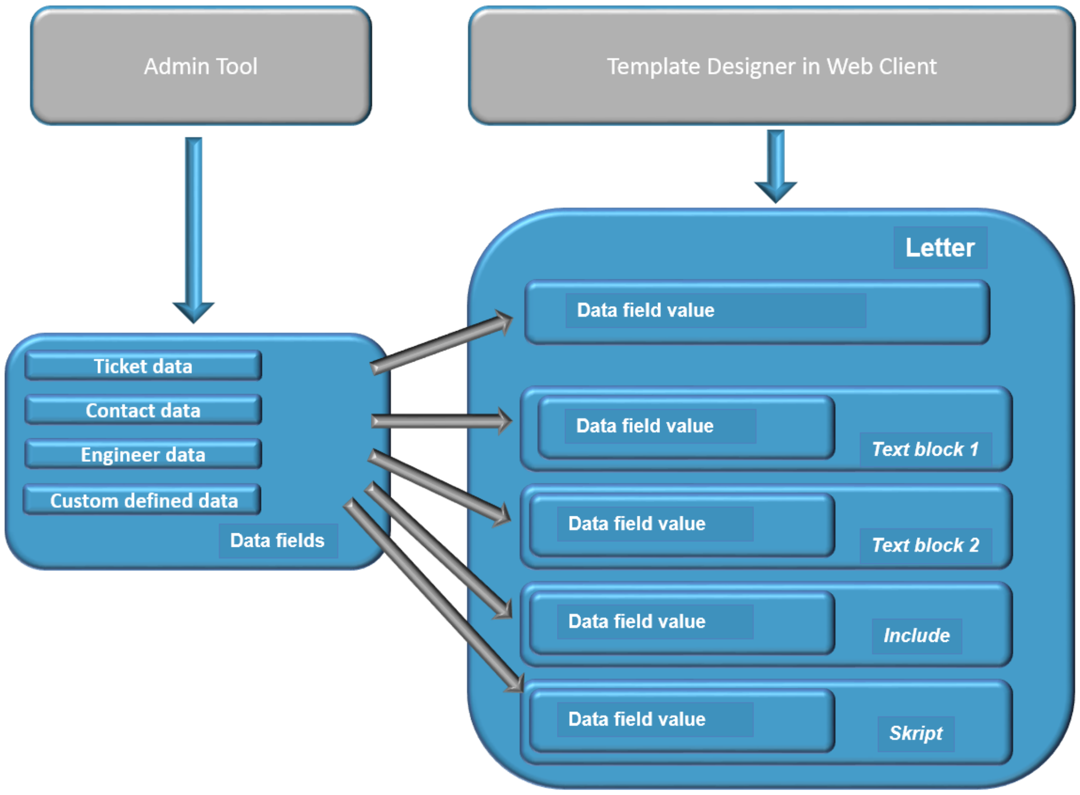


Fig. 1: ConSol*CM - Availabe Components / Data for E-Mail Templates

Please refer to the *ConSol*CM User Manual* section *Creating a New Ticket*, *Editing a Ticket*, and *Writing E-Mails from a Ticket* for a detailed description how to use the ticket editing functionalities and the Ticket E-Mail Editor.

Storage and Management of E-Mail and Ticket Text Templates

E-Mail Templates

E-Mail templates are stored and managed at two different locations in ConSol*CM:

1. In the *Template Manager*
2. In the *Script and Template Administration* of the Admin Tool (this will not be treated here, but in the respective section of this manual; see Admin Tool Templates)

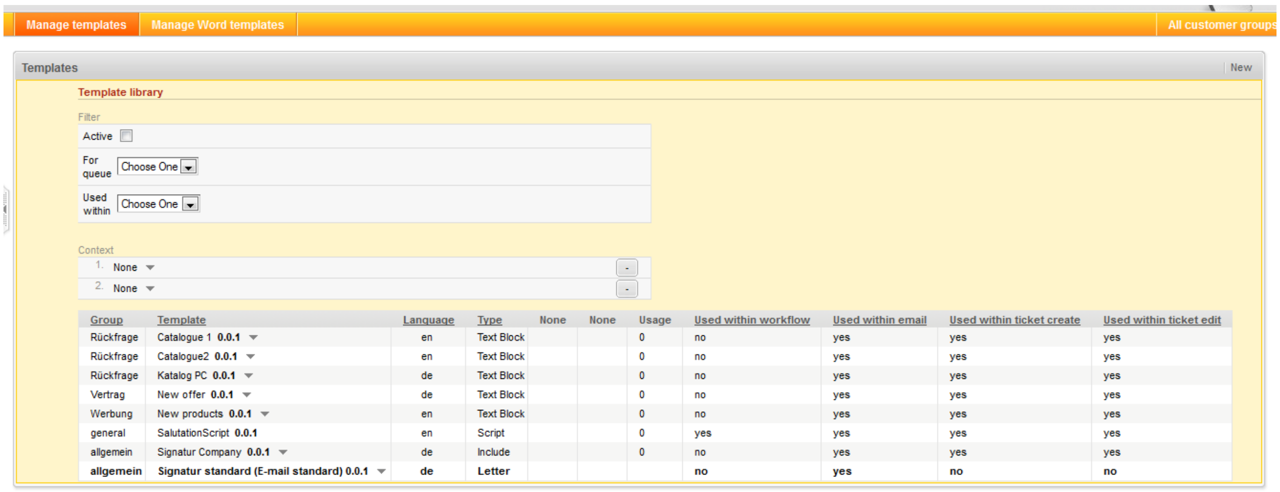
Ticket Text Templates

Ticket text templates are stored and managed at two locations in ConSol*CM:

1. In the *Template Manager* (here, ticket text templates are managed)
2. In *CM.Doc* (here, MS-Word documents can be stored for use with CM.Doc, see section CM.Doc)

26.2.2 Introduction to the Template Manager

The ConSol*CM Template Manager is a Web Client-based tool for the creation and management of e-mail and ticket text templates. See section Work with the Template Manager.



Filter

Active ☐

For queue: Choose One

Used within: Choose One

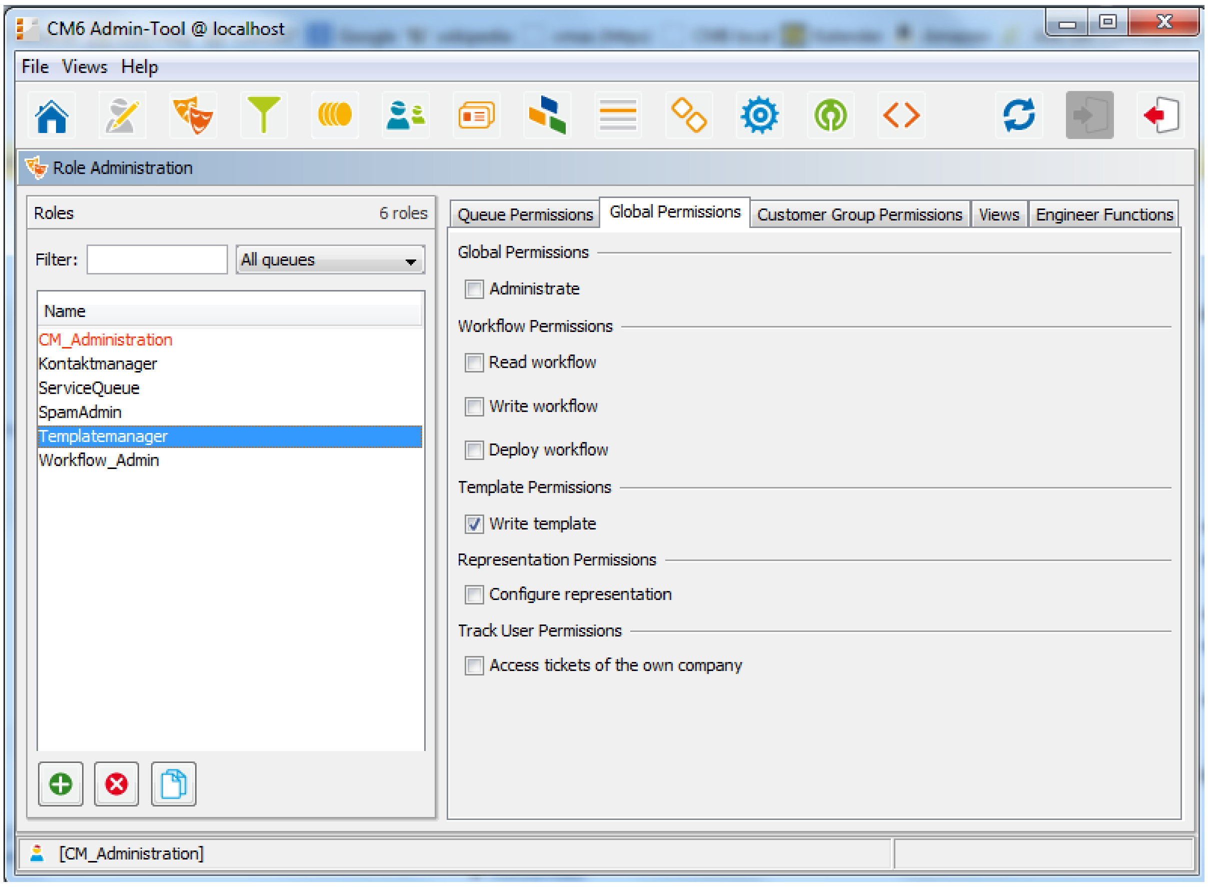
Context

1. None
2. None

Group	Template	Language	Type	None	None	Usage	Used within workflow	Used within email	Used within ticket create	Used within ticket edit
Rückfrage	Catalogue 1 **0.0.1**	en	Text Block			0	no	yes	yes	yes
Rückfrage	Catalogue2 **0.0.1**	en	Text Block			0	no	yes	yes	yes
Rückfrage	Katalog PC **0.0.1**	de	Text Block			0	no	yes	yes	yes
Vertrag	New offer **0.0.1**	de	Text Block			0	no	yes	yes	yes
Werbung	New products **0.0.1**	en	Text Block			0	no	yes	yes	yes
general	SalutationScript **0.0.1**	en	Script			0	yes	yes	yes	yes
allgemein	Signatur Company **0.0.1**	de	Include			0	no	yes	yes	yes
allgemein	**Signatur standard (E-mail standard) 0.0.1**	**de**	**Letter**				**no**	**yes**	**no**	**no**

Fig. 2: ConSol*CM Web Client - Template Manager

Every user who has been assigned a role with the permission *Write template* can access the item *Manage templates* in the main menu (which opens the Template Manager).



Roles 6 roles

Filter: All queues

Name
- CM_Administration
- Kontaktmanager
- ServiceQueue
- SpamAdmin
- Templatemanager
- Workflow_Admin

Global Permissions
- ☐ Administrate

Workflow Permissions
- ☐ Read workflow
- ☐ Write workflow
- ☐ Deploy workflow

Template Permissions
- ☑ Write template

Representation Permissions
- ☐ Configure representation

Track User Permissions
- ☐ Access tickets of the own company

[CM_Administration]

Fig. 3: ConSol*CM Admin Tool - Permissions for Role Templatemanager

Consulting Best Practice:

We recommend to create a role (*e.g. Templatemanager*) that has only the permission *Write template*, no queue permissions or other permissions are granted. Every user who should have access to the Template Manager can be given this role. That way, there is no merge between regular user permissions and Template Manager permissions and you can grant and retrieve the template manager permission in a very flexible way.

When the permission has been granted, the user has access to the main menu item *Manage templates*.

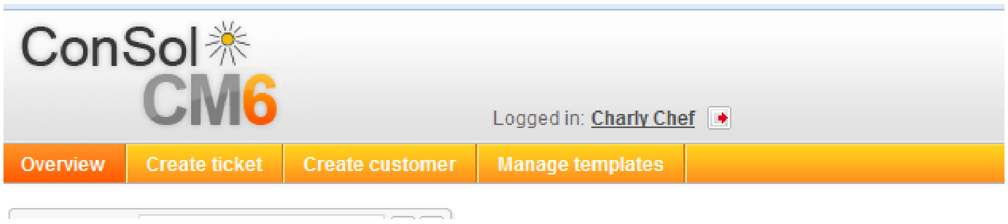
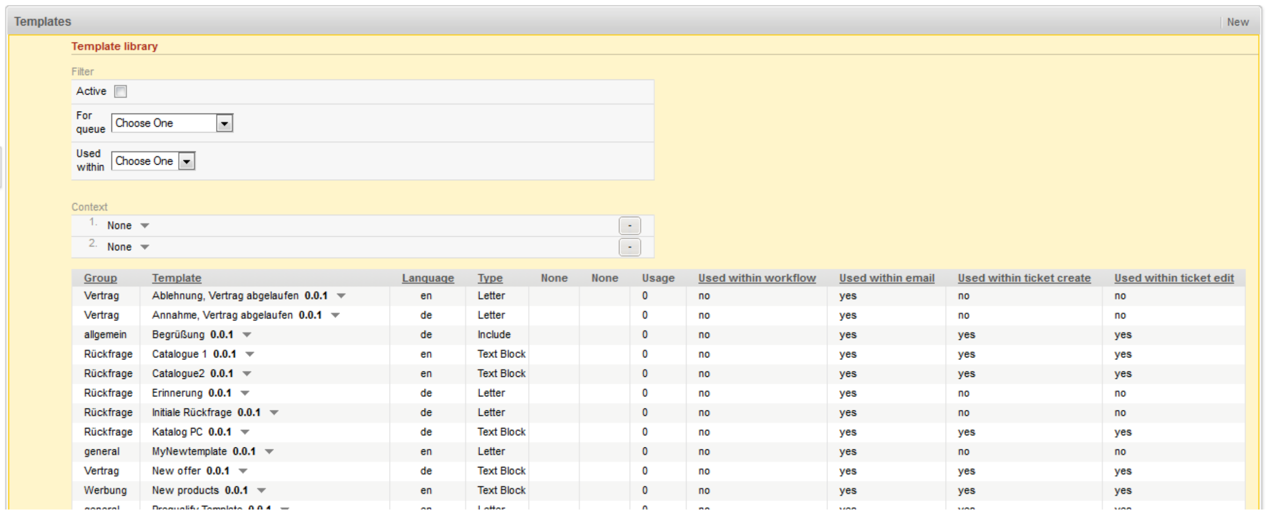


Fig. 4: ConSol*CM Web Client - Main Menu with Template Manager Access

26.2.3 Work with the Template Manager

Basic Template Manager GUI: The Template Library

When you open the Template Manager, the *Template library* is displayed:



Filter

Active	☐
For queue	Choose One
Used within	Choose One

Context

1. None
2. None

Group	Template	Language	Type	None	None	Usage	Used within workflow	Used within email	Used within ticket create	Used within ticket edit
Vertrag	Ablehnung, Vertrag abgelaufen **0.0.1**	en	Letter			0	no	yes	no	no
Vertrag	Annahme, Vertrag abgelaufen **0.0.1**	de	Letter			0	no	yes	no	no
allgemein	Begrüßung **0.0.1**	de	Include			0	no	yes	yes	yes
Rückfrage	Catalogue 1 **0.0.1**	en	Text Block			0	no	yes	yes	yes
Rückfrage	Catalogue2 **0.0.1**	en	Text Block			0	no	yes	yes	yes
Rückfrage	Erinnerung **0.0.1**	de	Letter			0	no	yes	no	no
Rückfrage	Initiale Rückfrage **0.0.1**	de	Letter			0	no	yes	no	no
Rückfrage	Katalog PC **0.0.1**	de	Text Block			0	no	yes	yes	yes
general	MyNewtemplate **0.0.1**	en	Letter			0	no	yes	no	no
Vertrag	New offer **0.0.1**	de	Text Block			0	no	yes	yes	yes
Werbung	New products **0.0.1**	en	Text Block			0	no	yes	yes	yes

Fig. 5: ConSol*CM Web Client - Template Manager

A list of all existing templates is shown.

Filter

You can filter the displayed list entries by using the filters in the upper part of the page:

- **Active**
  Only active templates are displayed (i.e. deactivated templates will not be displayed).
- **For queue**
  Only templates which have been assigned to the selected queue will be displayed. Only one queue can be selected. The display of templates which have not been assigned to a certain queue will not be affected here. They will be displayed.
- **Used within**
  Only the templates of the selected type will be displayed, i.e. e-mail, workflow, ticket texts (creation or editing). Only one type can be selected.

Context

This provides another kind of filter. Here you can select ticket data. For each selected criterion, one of the columns named *none* in the list will be named according to the selected parameter. For the templates where the selected custom fields are used, this will be indicated in the list.

List

The list contains the following columns. It can be sorted according to a column by clicking on the column header. Another click will reverse the display order.

- **Group**
  The group of a template does not have any technical or functional implications, it is only used to order the list in a certain way, i.e. to group templates with a common context.
- **Template**
  The template name. This is also used in workflows to indicate the required template and it is displayed in the Ticket E-Mail Editor or Ticket Comment Editor in the template selection.
- **Language**
  The language that has been selected during creation of the template (can be modified). The web browser of an engineer will display the template according to the browser locale. So when you need a template in different languages, make sure to set this value correctly.
- **Type**
  There are five different types of templates which will be explained in detail in the subsequent sections:
  - **Letter**
    This is the basic form of a template. *Letter* templates are offered in the Ticket E-Mail Editor or Ticket Comment Editor and can be used as workflow e-mail templates. All other template types are only sub-components of a *letter*.
  - **Include**
    This is a sub-component of a *letter* which can be used in *letters*. In this way, you can use the same text in several templates. A typical example is the signature of a company which is used in every other template. The signature should be defined as *include* and then be integrated in

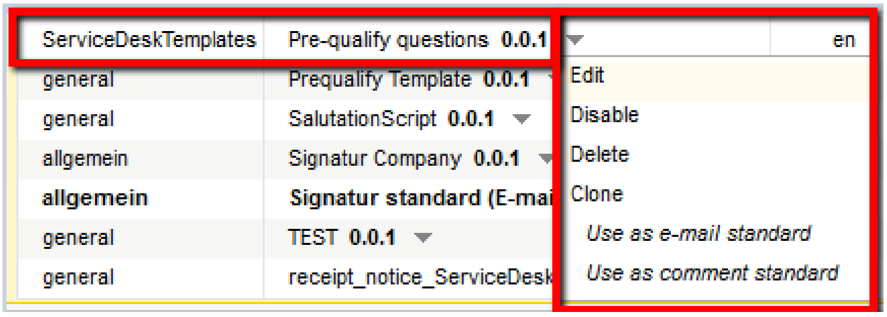
all other (*letter*) templates where the signature is required. Thus, the template administrator has to maintain the signature at exactly one location and can be sure that it is used in every other template correctly.

- **Workflow Include**
  This is the same as an *include* but used only for workflows.
- **Text Block**
  This is also a sub-component of a *letter*. It can be checked or unchecked during the writing of e-mails, i.e. the text will be displayed or not. A good example is the first analysis in a help desk team where the same questions are sent to every customer. One text block can contain hardware questions, one software questions. Depending on the purpose of the e-mail, the engineer uses either one.
- **Script**
  This template type is only available for administrators (i.e. a user who logs into the Web Client using an administrator account). Here, *intelligent* templates can be constructed like a template that sets *Dear Sir* for a male and *Dear Madam* for a female customer, depending on the value of the field *salutation*.

- **Context (here *None*)**
  Can be used to define dependencies or conditions (e.g. field values). Only when the condition is met the template is offered in the Web Client GUI, e.g. the template is only offered for tickets with high priority.

- **Usage**
  Indicates how often the template is used.
- **Used within workflow**
  Boolean. A template can be marked as *workflow template*. Then it is not available in the Ticket E-Mail Editor or Ticket Comment Editor but can only be used by the workflow for automatic e-mails.
- **Used within email**
  Boolean. All templates which have been marked as *Available in Email* will be marked *yes*.
- **Used within ticket create**
  Boolean. All templates which have been marked as *Available in Ticket create* will be marked *yes*.
- **Used within ticket edit**
  Boolean. All templates which have been marked as *Available in Ticket edit* will be marked *yes*.

For every template you can select an operation by using the context menu:



ServiceDeskTemplates	Pre-qualify questions 0.0.1	en
general	Prequalify Template 0.0.1	
general	SalutationScript 0.0.1	
allgemein	Signatur Company 0.0.1	
allgemein	Signatur standard (E-mai	
general	TEST 0.0.1	
general	receipt_notice_ServiceDesk	

Fig. 6: Template Manager - Context Menu of a Template

- **Edit**
  Edit the template. The same functionalities as described for creating a new template are available.

- **Disable**
  (or **Enable** for disabled templates)
  Only enabled (= active) templates are active and available in the system.
- **Delete**
  Delete the template. This is not possible when the template is used by a workflow or when an *include* or *text block* is used in other templates (*letter*s).
- **Clone**
  Create a copy of the template. A new name is required in this case.
- **Use as e-mail standard**
  (or **Unset standard** for the current standard template)
  Only one template can be marked as standard e-mail template. This will be automatically inserted into any e-mail that is opened in the Ticket E-Mail Editor. It can then be removed by the engineer or used in the e-mail. Usually a signature or footer is defined as standard template.
- **Use as comment standard**
  (or **Unset standard** for the current standard template)
  Only one template can be marked as comment standard template. This will be automatically inserted into a comment that is opened in the Ticket Comment Editor. It can then be removed by the engineer or used for the comment.

Information:

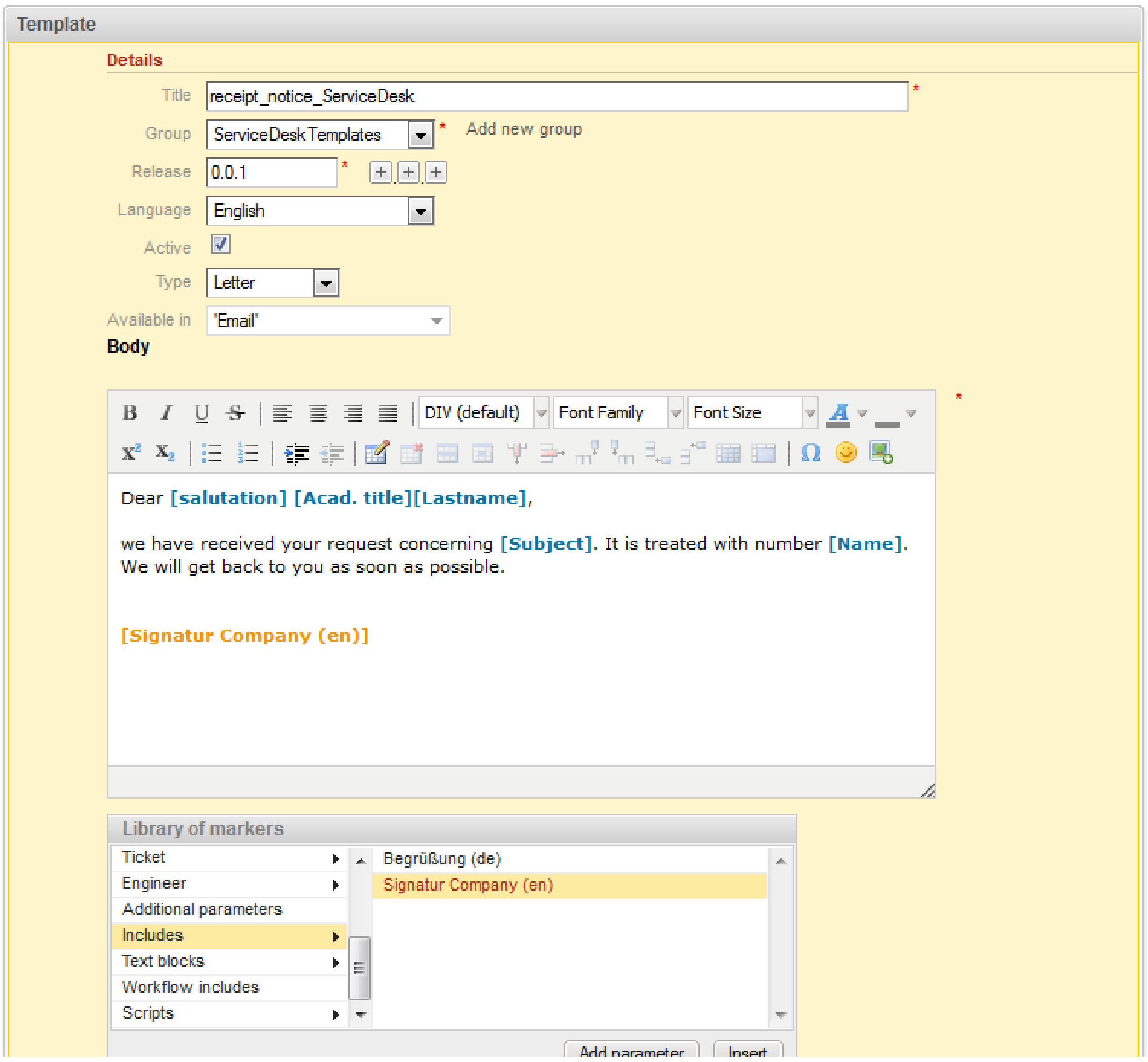
A standard template must not contain text blocks or variables.

Create a New Template

Here, an example for an e-mail template is shown. The same principle can be applied to ticket text templates.

Create a New Letter

To create a new template, click on the *New* link in the Template Manager. On the *New Template* page, you can enter all parameters for the new template. In our first example, a *letter* is created which serves as confirmation of receipt for the customer. It can be automatically sent from the workflow or be used in the Web Client.


Template

Details

Title	receipt_notice_ServiceDesk
Group	ServiceDeskTemplates
Release	0.0.1
Language	English
Active	✓
Type	Letter
Available in	'Email'

Body

Dear [salutation] [Acad. title][Lastname],

we have received your request concerning [Subject]. It is treated with number [Name].
We will get back to you as soon as possible.

[Signatur Company (en)]

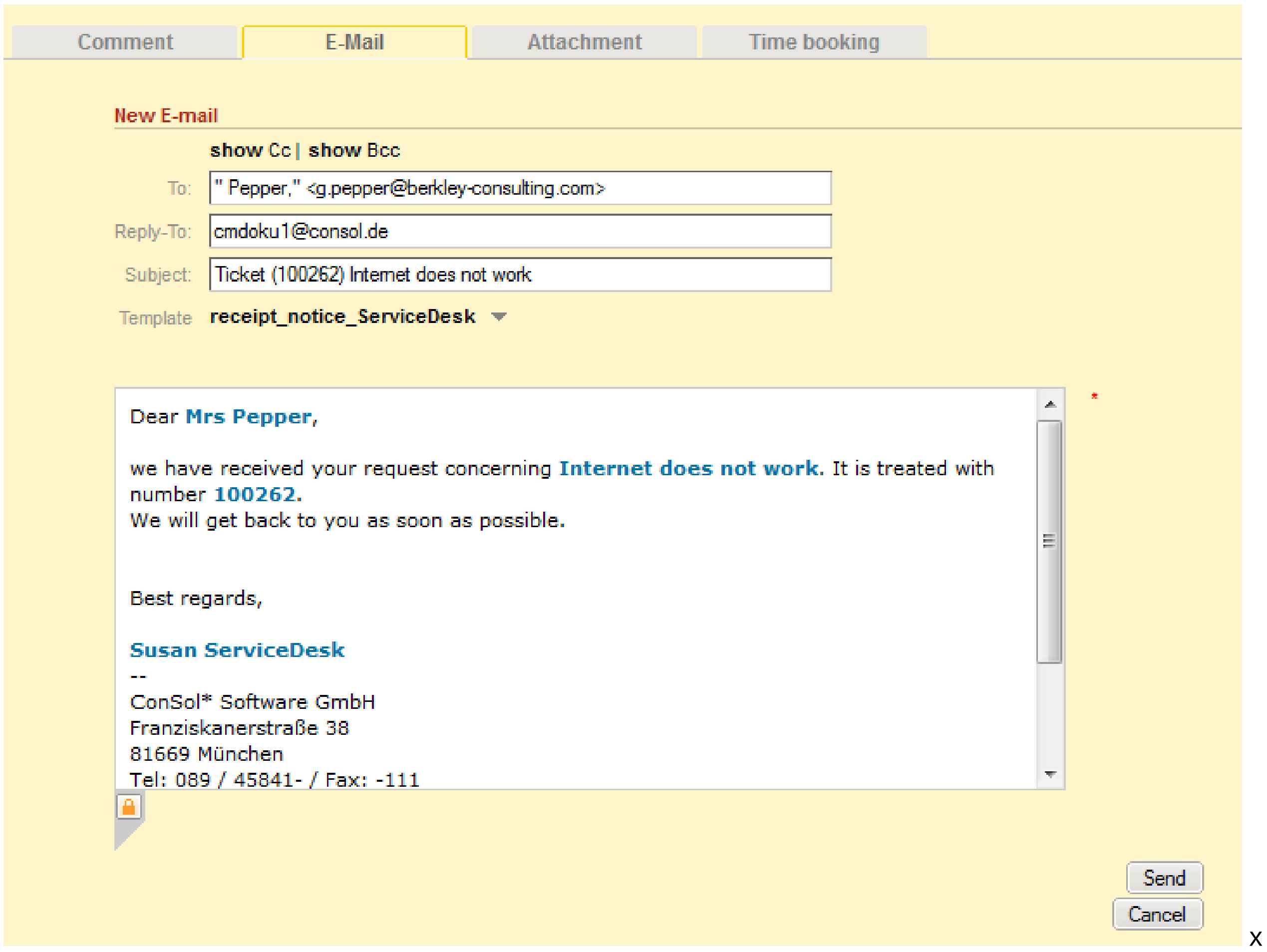
Fig. 7: ConSol*CM Web Client - Create a New Template

- **Title**
  The name of the template.
- **Group**
  The group (see previous section). You can either use an existing group or create a new one.
- **Release**
  If you want to set up a versioning system for the e-mail templates, you can set the release, i.e. version, here.
- **Language**
  Choose the language of the template. This can be important, if you work in an international team. ConSol*CM can be used in as many languages as required, this can be configured using the Admin Tool and in the Process Designer. To make sure the e-mails are sent in the correct language, the corresponding locale has to be set here.
- **Active**
  Select if the template should be active (= enabled) or inactive (disabled). This can be changed later, so you can design a template and work on it and set it *active* when you are finished.

- **Type**
  Select the type (letter, include, text block, script) of the template. See previous section for explanation.
- **Available in**
  - **Workflow**
    Select if the template should be available in workflows (i.e. not available in the Ticket E-Mail Editor or in the Ticket Comment Editor).
  - **Email**
    Select if the template should be available in e-mails.
  - **Ticket create**
    Select if the template should be available during ticket creation.
  - **Ticket edit**
    Select if the template should be available when the ticket is edited.
- **Body**
  Here you define the content of the template/letter. You can combine any free text and components of the *Library of Markers* (below the body, see section The Library of Markers for details). Write the text and select the desired element from the library by clicking on it and by pressing *Insert*.
- **Binding**
  Here you can define one or more
  - queue(s)
  - context(s)

  where the template should be available, see section Binding Templates to Queues or to Specific Parameters for details.

In the Web Client, i.e. in the Ticket E-Mail Editor, the template *receipt_notice_ServiceDesk* would have the following layout:



To: " Pepper," <g.pepper@berkley-consulting.com>
Reply-To: cmdoku1@consol.de
Subject: Ticket (100262) Internet does not work
Template receipt_notice_ServiceDesk

Dear Mrs Pepper,

we have received your request concerning Internet does not work. It is treated with number 100262.
We will get back to you as soon as possible.

Best regards,

Susan ServiceDesk
--
ConSol* Software GmbH
Franziskanerstraße 38
81669 München
Tel: 089 / 45841- / Fax: -111

Fig. 8: ConSol*CM Web Client - E-Mail Template in Ticket E-Mail Editor

The Library of Markers

The Library of Markers provides a collection of all data fields that are available in the system. These are:

- **Default fields**
  Like *queue* or *engineer* with all corresponding data like *queue name* or *engineer forename* or *engineer lastname*.
- **Ticket custom fields and/or data object group fields**
  That have been designed specifically for the system like e.g. *customer service number*.
- **Components of the Template Manager**
  That are used in other components, e.g. *includes* or *workflow includes*.
- **Scripts**
  That have been defined by an administrator and can help provide content in a dynamic way.

The following table provides examples for fields that could be found in a system. The names displayed in the Library of Markers are the localized names of the custom fields resp. of the data object group fields. If no localization is provided, the (technical) field name is displayed. If you would like to read the information about custom fields, please refer to chapter Custom Field Administration. For data object group fields (i.e. customer data), see section Setting Up the Customer Data Model.

Field Group or Main Component	(Example) Custom Field Resp. Data Object Group Field	Explanation
Customer data models		<entry point for all customer-specific fields>
Customer data models	Customer groups	<entry point for all customer-specific fields of the selected customer group>
<Customer or company>	Salutation	
	Academic title	
	Forename	
	Lastname	
	Phone	
	Email	
	<more fields depending on FlexCDM definition>	
Customer Group	Name of the customer group	
Queue	Name	The name of the queue where the ticket is being processed at the moment
Custom fields for queue	All custom fields of custom field groups that have been assigned to the queue	
Ticket	ID	The internal ticket ID, not displayed in the Web Client GUI
	Name	The ticket name, the *ID* in the Web Client GUI
	Subject	
	Engineer	The current engineer who owns the ticket. Can be *NULL* (empty) i f no engineer is set.
	Creation Date	Opening date of the ticket

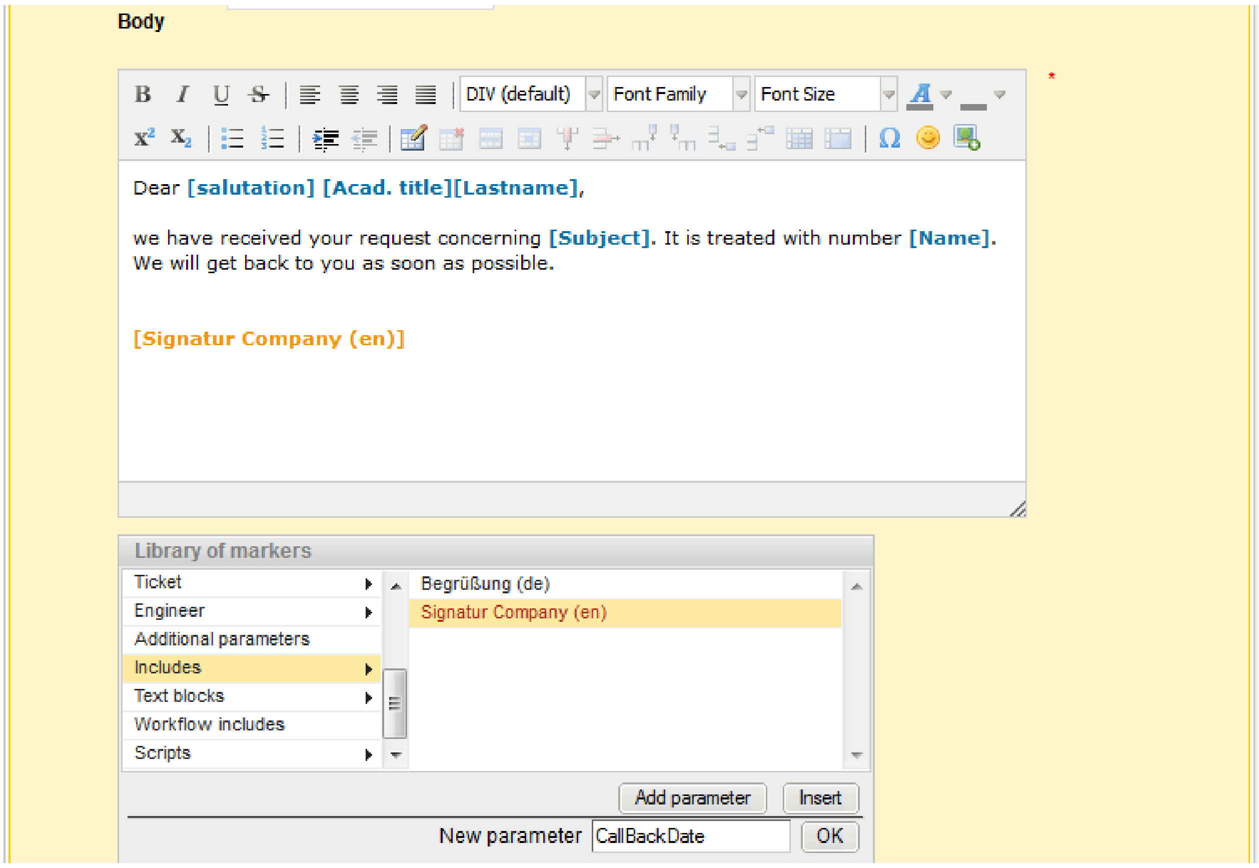
Field Group or Main Component	(Example) Custom Field Resp. Data Object Group Field	Explanation
Engineer	Login	The login of the engineer who is currently logged into the system
	Firstname	First name, last name, e-mail of the engineer, be sure that the respective field has been filled in the engineer data, see chapter Engineer Administration for details about engineer management.
	Lastname	
	Email	
Includes	<all available includes>	
Text Blocks	<all available text blocks>	
Workflow Includes	<all available workflow includes>	
Scripts	<all available scripts>	

Attention:

In case customer-specific fields are used in a template, this template will only be offered when the ticket is assigned to a main customer from the respective customer group!

Since data object group fields differ between customer data models, it might be necessary to define a similar template for several customer groups.

By clicking on *Add parameter* you can define a field where the engineer has to fill in data at run-time. When you have defined the field, it will be available under *Additional parameters*.


Body

Dear [salutation] [Acad. title][Lastname],

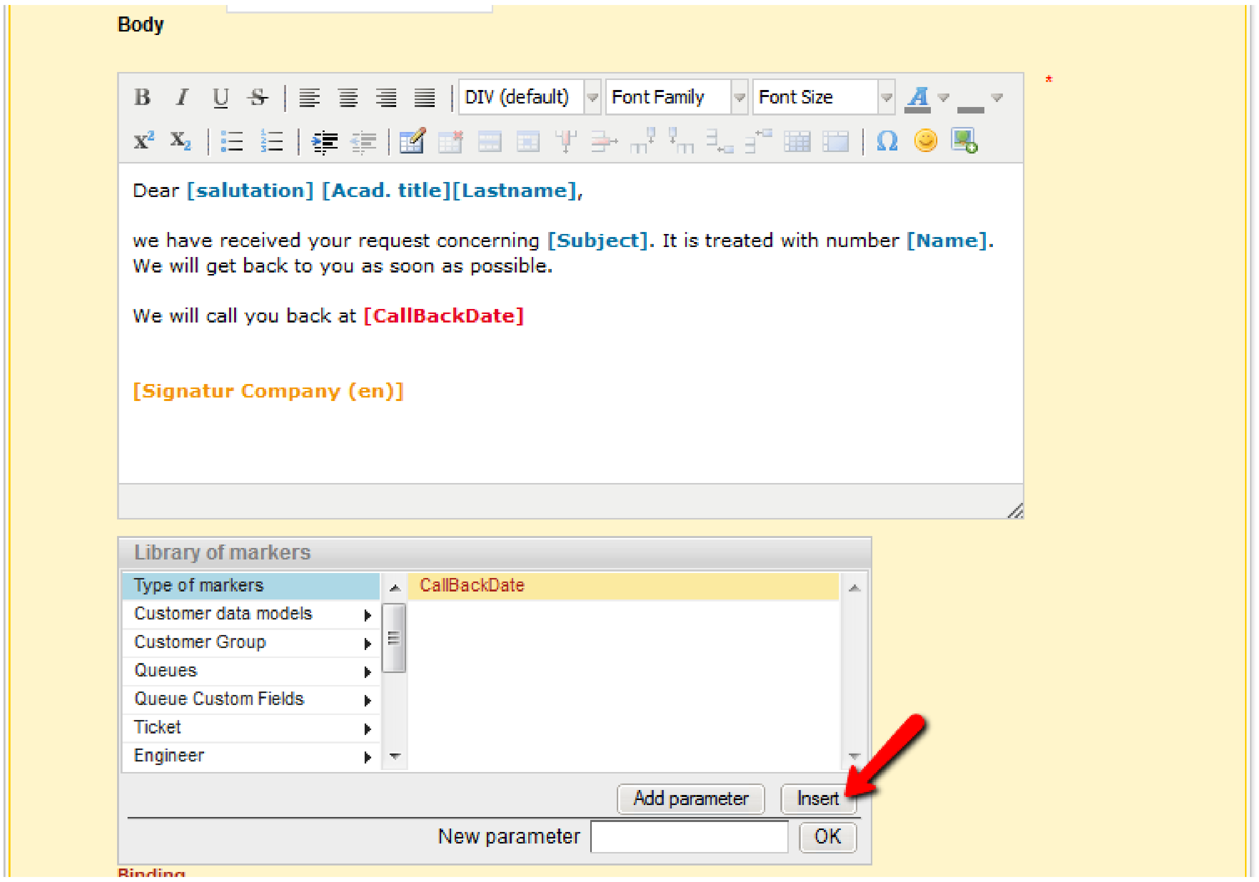
we have received your request concerning [Subject]. It is treated with number [Name].
We will get back to you as soon as possible.

[Signatur Company (en)]

Library of markers

Ticket	Begrüßung (de)
Engineer	Signatur Company (en)
Additional parameters	
Includes	
Text blocks	
Workflow includes	
Scripts	

Fig. 9: ConSol*CM Web Client - Template Manager: Add Parameter in Library of Markers


Body

Dear [salutation] [Acad. title][Lastname],

we have received your request concerning [Subject]. It is treated with number [Name].
We will get back to you as soon as possible.

We will call you back at [CallBackDate]

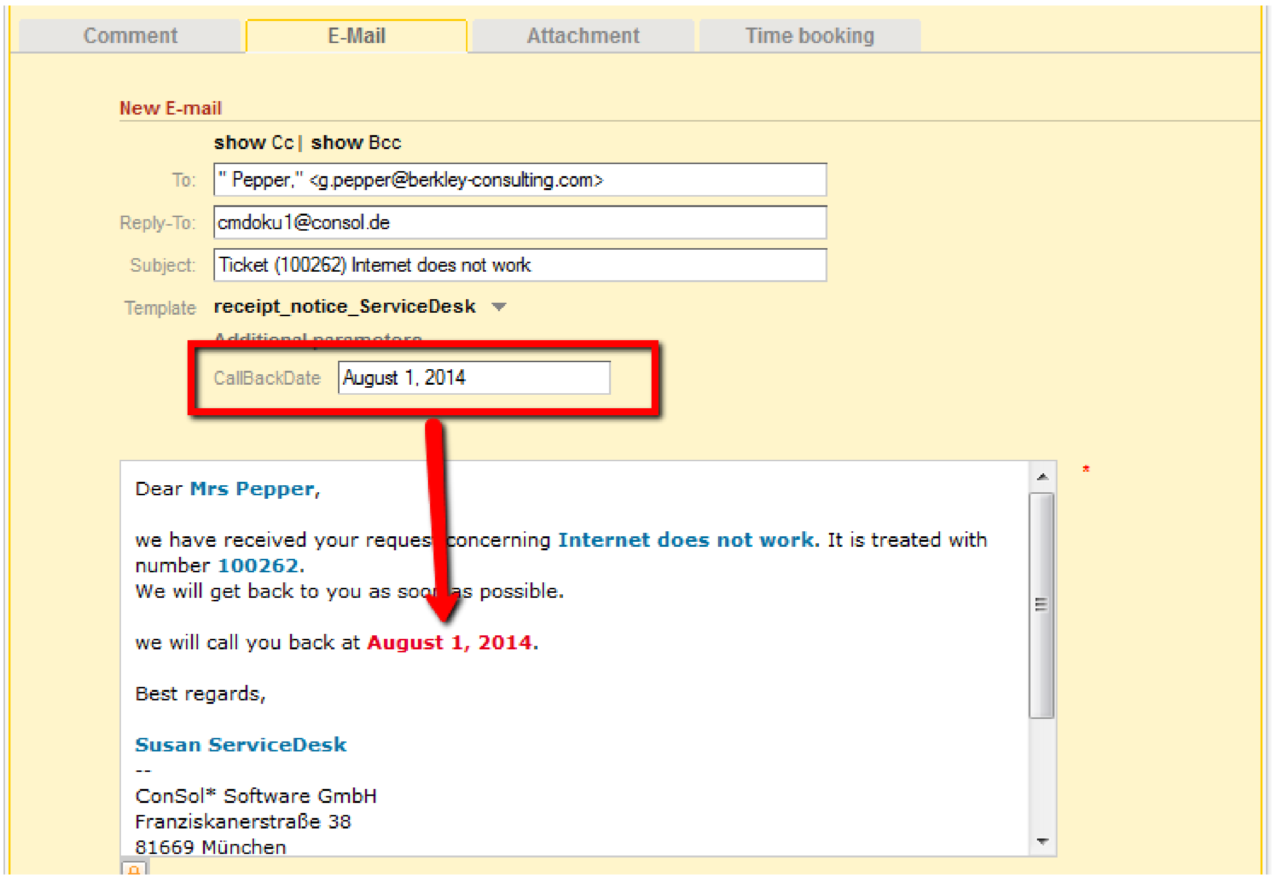
[Signatur Company (en)]

Library of markers

Type of markers	CallBackDate
Customer data models	
Customer Group	
Queues	
Queue Custom Fields	
Ticket	
Engineer	

Fig. 10: ConSol*CM Web Client - Template Manager: Library of Markers After Adding Parameter

When the engineer opens the Ticket E-Mail Editor in the ticket and enters data in the field (here *CallBackDate*), the (new) data is automatically written into the field in the template.



To: " Pepper," <g.pepper@berkley-consulting.com>
Reply-To: cmdoku1@consol.de
Subject: Ticket (100262) Internet does not work
Template receipt_notice_ServiceDesk
CallBackDate August 1, 2014

Dear **Mrs Pepper**,

we have received your request concerning **Internet does not work.** It is treated with number **100262**.
We will get back to you as soon as possible.

we will call you back at **August 1, 2014**.

Best regards,

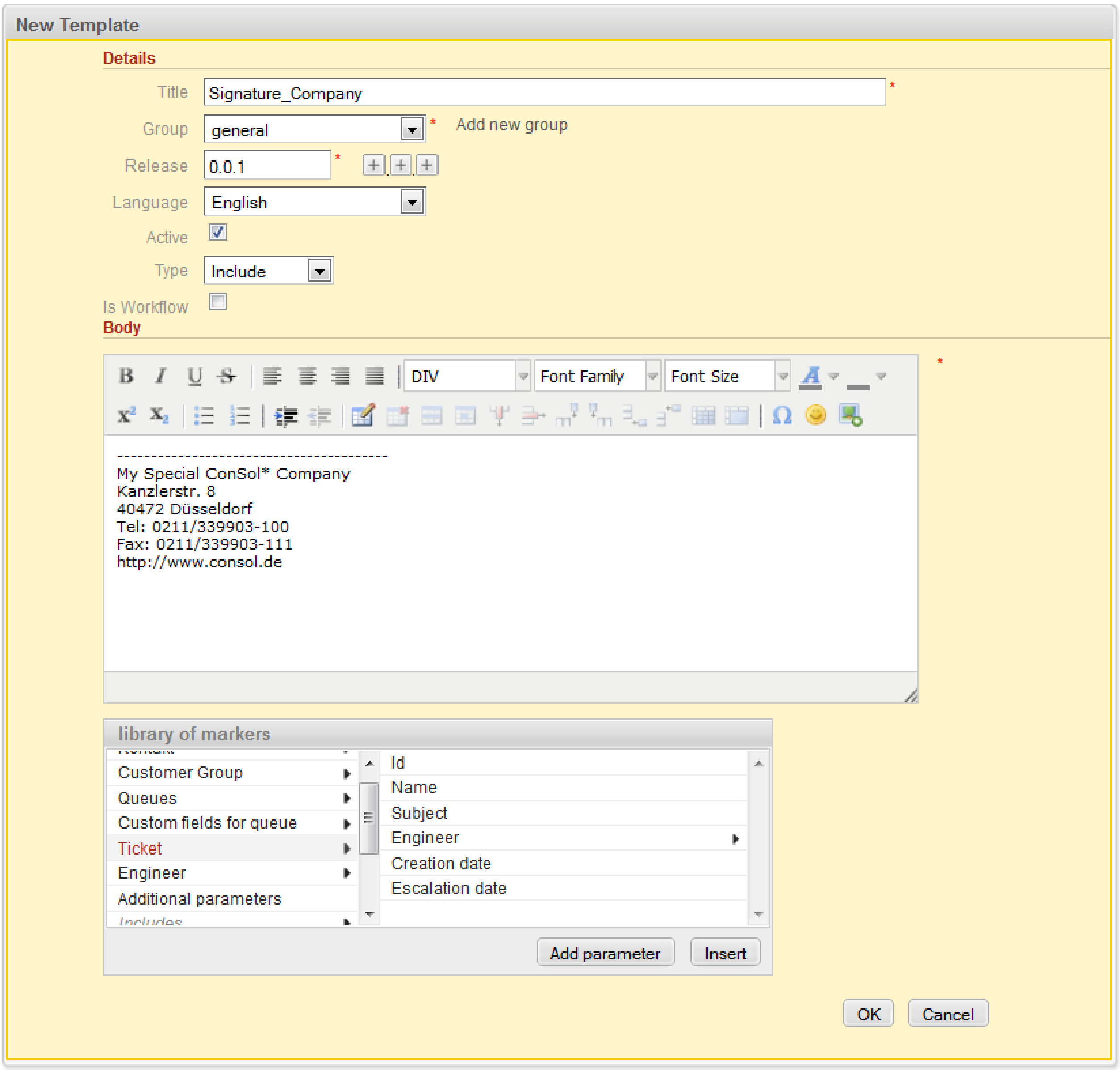
Susan ServiceDesk
--
ConSol* Software GmbH
Franziskanerstraße 38
81669 München

Fig. 11: ConSol*CM Web Client - Ticket E-Mail Editor: Enter Data for New Parameter

Create a New Include or Workflow Include

An *include* is a template that cannot be selected by the engineer directly (in the Ticket E-Mail or Comment Editor) but a component which is integrated in other e-mail or ticket text templates, mostly in *letters*.

A standard use case for an *include* is the signature, so we will show you the corresponding example. In order to define the standard company signature, define a signature as *include* and integrate it into the standard company signature which is a *letter*.



New Template

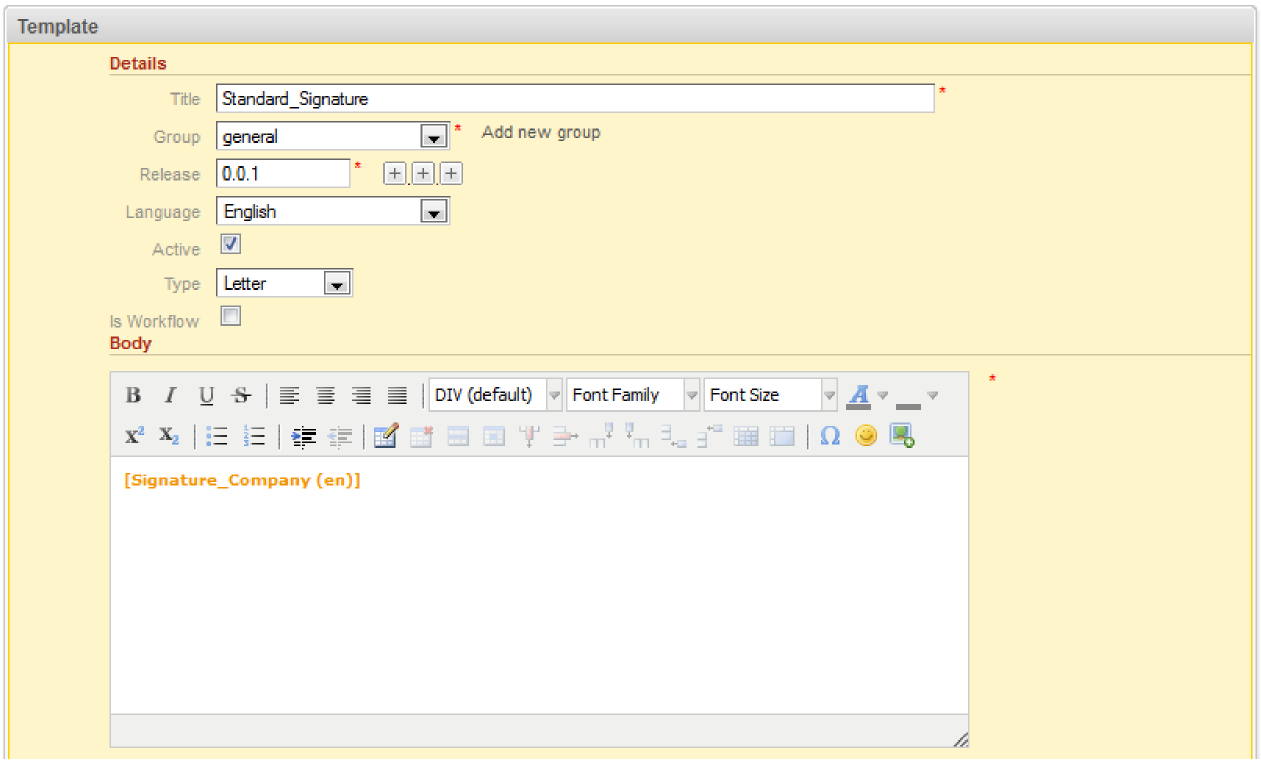
Details

Title	Signature_Company
Group	general
Release	0.0.1
Language	English
Active	☑
Type	Include
Is Workflow	☐

Body

```
----------------------------------------
My Special ConSol* Company
Kanzlerstr. 8
40472 Düsseldorf
Tel: 0211/339903-100
Fax: 0211/339903-111
http://www.consol.de
```

Fig. 12: ConSol*CM Web Client - Template Manager: Signature Include



Title	Standard_Signature
Group	general
Release	0.0.1
Language	English
Active	✓
Type	Letter
Is Workflow	☐

[Signature_Company (en)]

Fig. 13: ConSol*CM Web Client - Template Manager: Standard Signature Letter

If you follow the example, two purposes can be met:

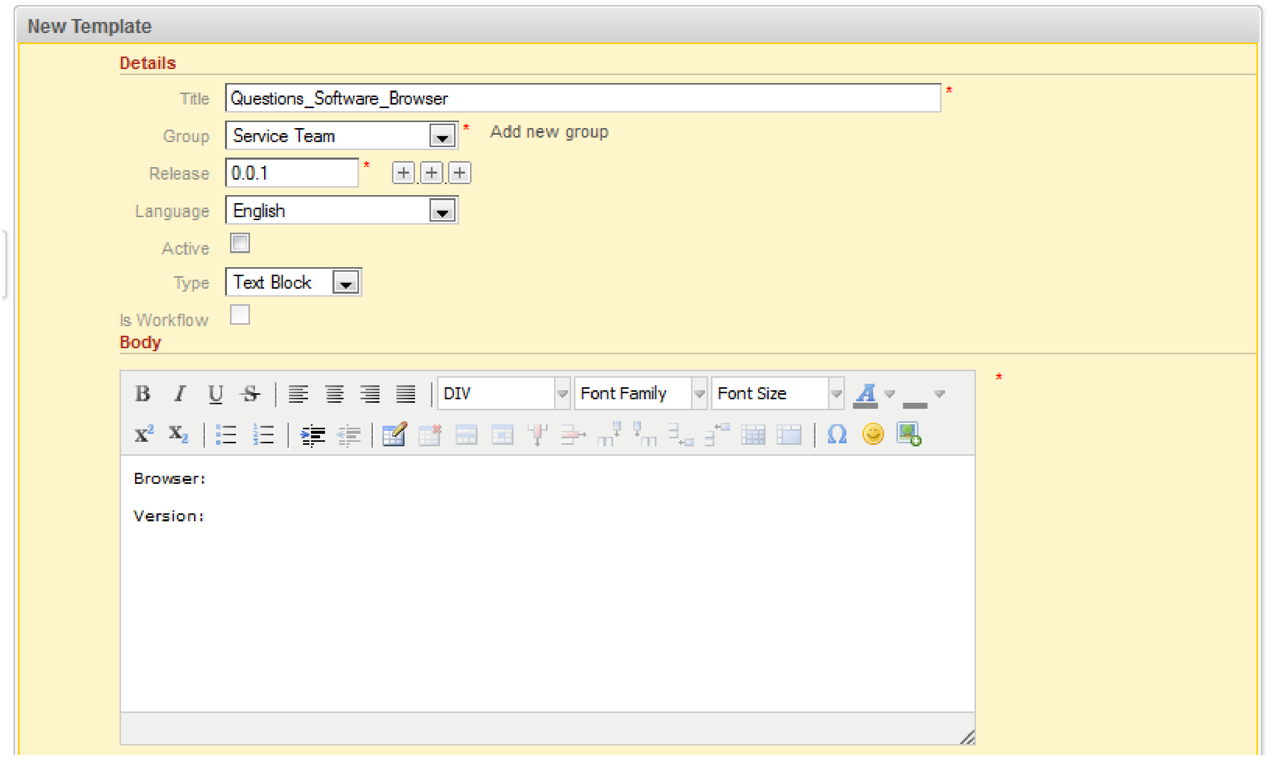
- The *Standard_Signature* can be defined as e-mail standard. In this way it will automatically be displayed for every new e-mail. Of course, the engineer can change the template.
- The *Signature_Company* can be used in any other template if required (compare image of the new template).

Create New Text Blocks

A *text block* is a template that cannot be selected by the engineer directly (in the Ticket E-Mail or Comment Editor) but a component which is integrated in other e-mail or ticket text templates, mostly in *letters*. Usually, several text blocks are offered in one *letter* so that the engineer can select which one(s) to use.

The following example shows how to use three *text blocks* to ask some initial analysis questions to the customer.

First, the *text blocks* are created:



New Template

Details

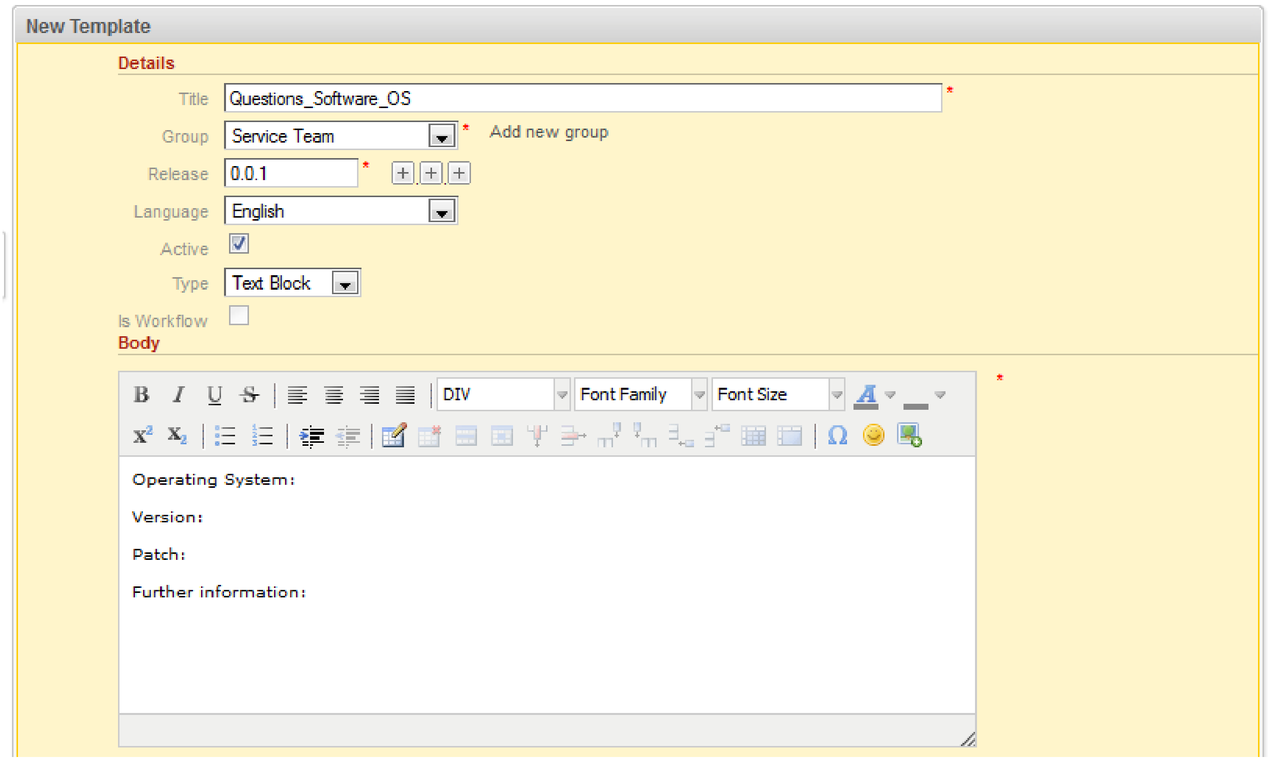
Title	Questions_Software_Browser *
Group	Service Team * Add new group
Release	0.0.1 *
Language	English
Active	☐
Type	Text Block
Is Workflow	☐

Body

Browser:

Version:

Fig. 14: ConSol*CM Web Client - Template Manager: Create First Text Block



New Template

Details

Title	Questions_Software_OS *
Group	Service Team * Add new group
Release	0.0.1 *
Language	English
Active	☑
Type	Text Block
Is Workflow	☐

Body

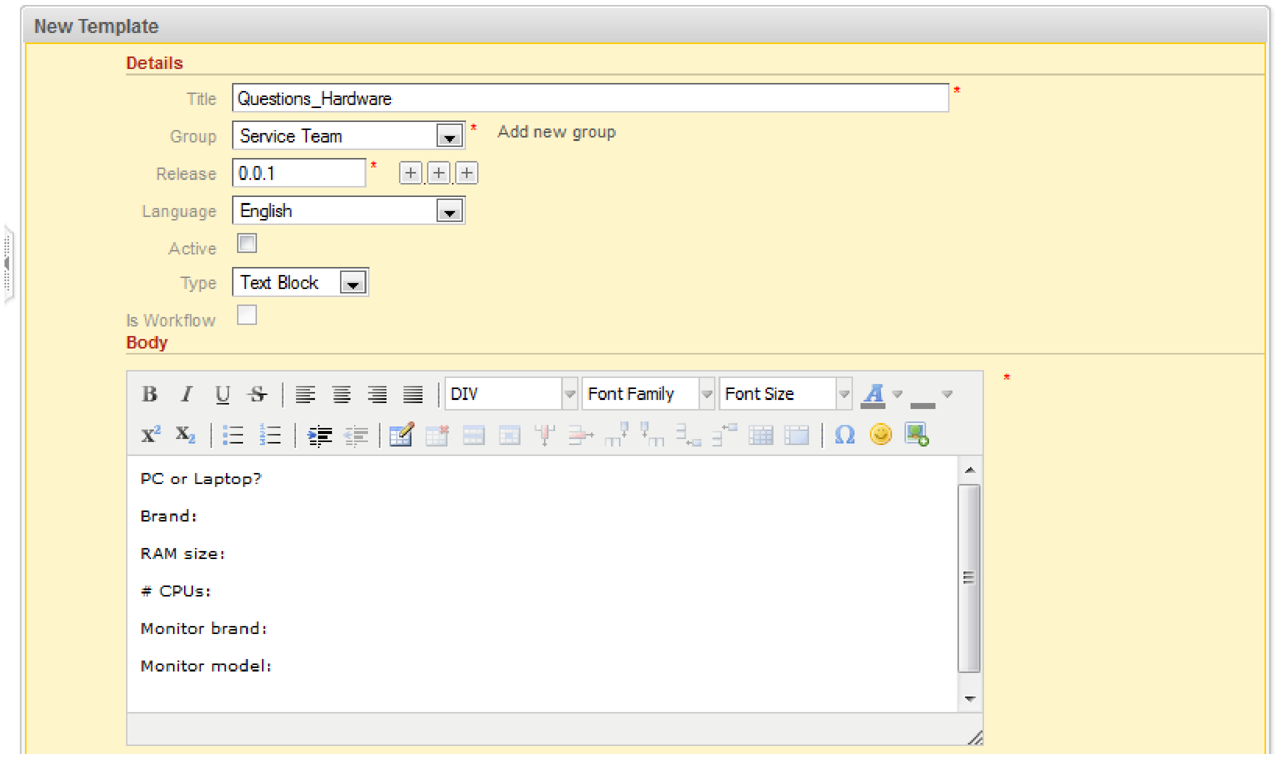
Operating System:

Version:

Patch:

Further information:

Fig. 15: ConSol*CM Web Client - Template Manager: Create Second Text Block


New Template

Details

Title	Questions_Hardware *
Group	Service Team * Add new group
Release	0.0.1 *
Language	English
Active	☐
Type	Text Block
Is Workflow	☐

Body *

PC or Laptop?

Brand:

RAM size:

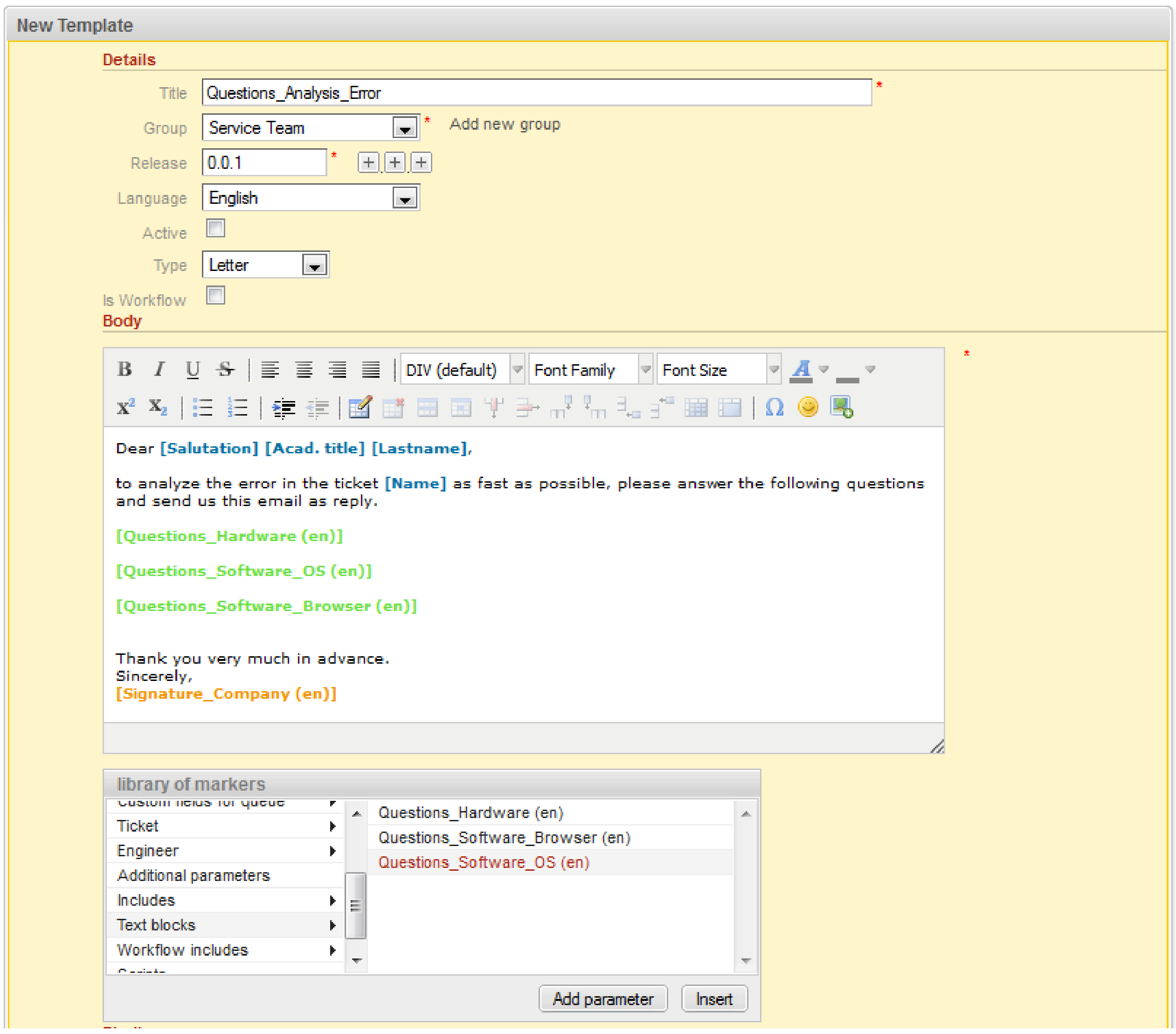
CPUs:

Monitor brand:

Monitor model:

Fig. 16: ConSol*CM Web Client - Template Manager: Create Third Text Block

Then the *letter* is created where the *text blocks* should be used:


New Template

Details

Title	Questions_Analysis_Error *
Group	Service Team * Add new group
Release	0.0.1 *
Language	English
Active	☐
Type	Letter
Is Workflow	☐

Body

Dear [Salutation] [Acad. title] [Lastname],

to analyze the error in the ticket [Name] as fast as possible, please answer the following questions and send us this email as reply.

[Questions_Hardware (en)]

[Questions_Software_OS (en)]

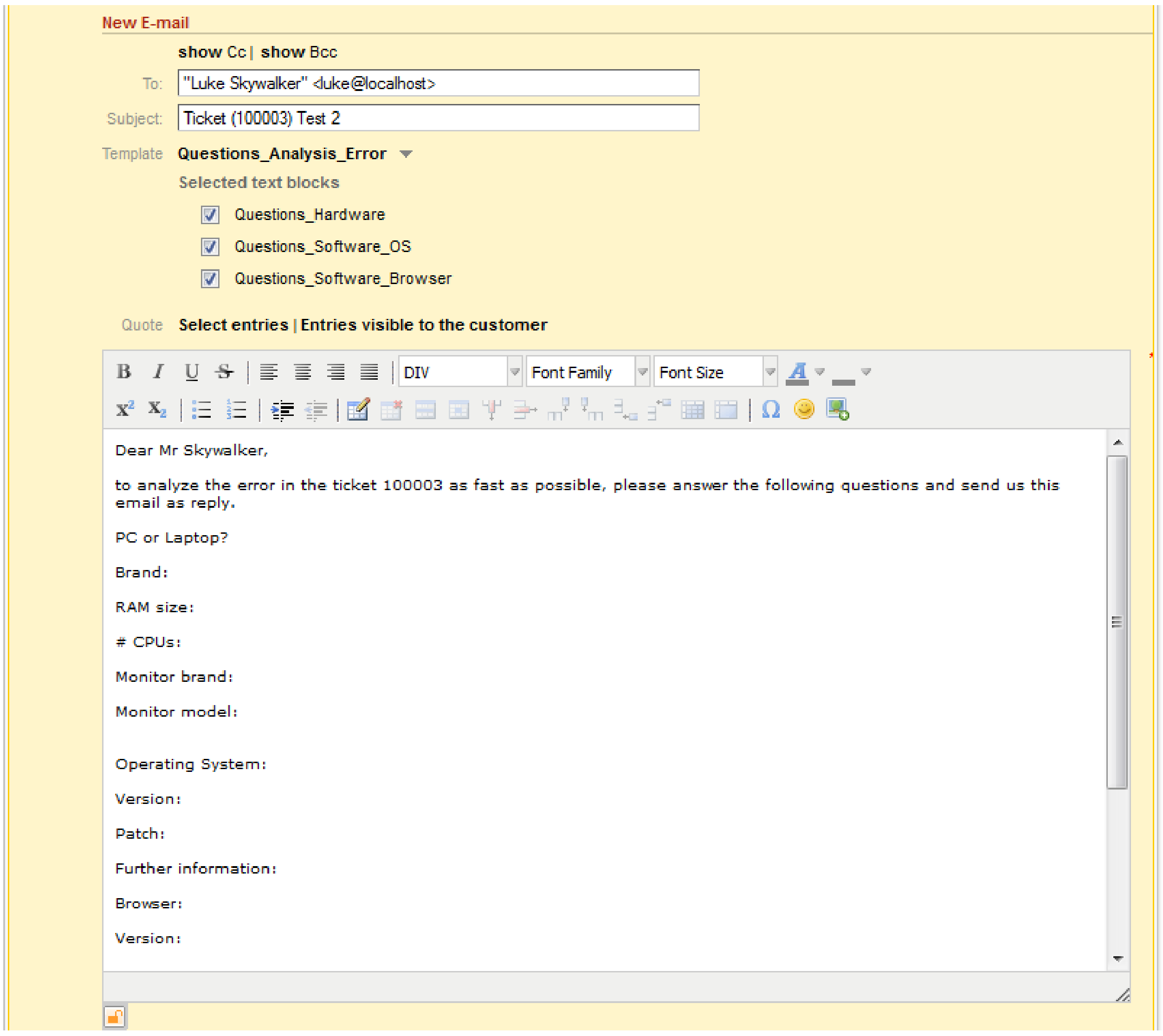
[Questions_Software_Browser (en)]

Thank you very much in advance.
Sincerely,
[Signature_Company (en)]

library of markers

Fig. 17: ConSol*CM Web Client - Template Manager: Create Letter for Text Blocks

In the Web Client, the engineer can then decide which *text blocks* to use and which ones to deactivate:



To: "Luke Skywalker" <luke@localhost>
Subject: Ticket (100003) Test 2
Template Questions_Analysis_Error
Selected text blocks
- [✓] Questions_Hardware
- [✓] Questions_Software_OS
- [✓] Questions_Software_Browser

Dear Mr Skywalker,

to analyze the error in the ticket 100003 as fast as possible, please answer the following questions and send us this email as reply.

PC or Laptop?

Brand:

RAM size:

CPUs:

Monitor brand:

Monitor model:

Operating System:

Version:

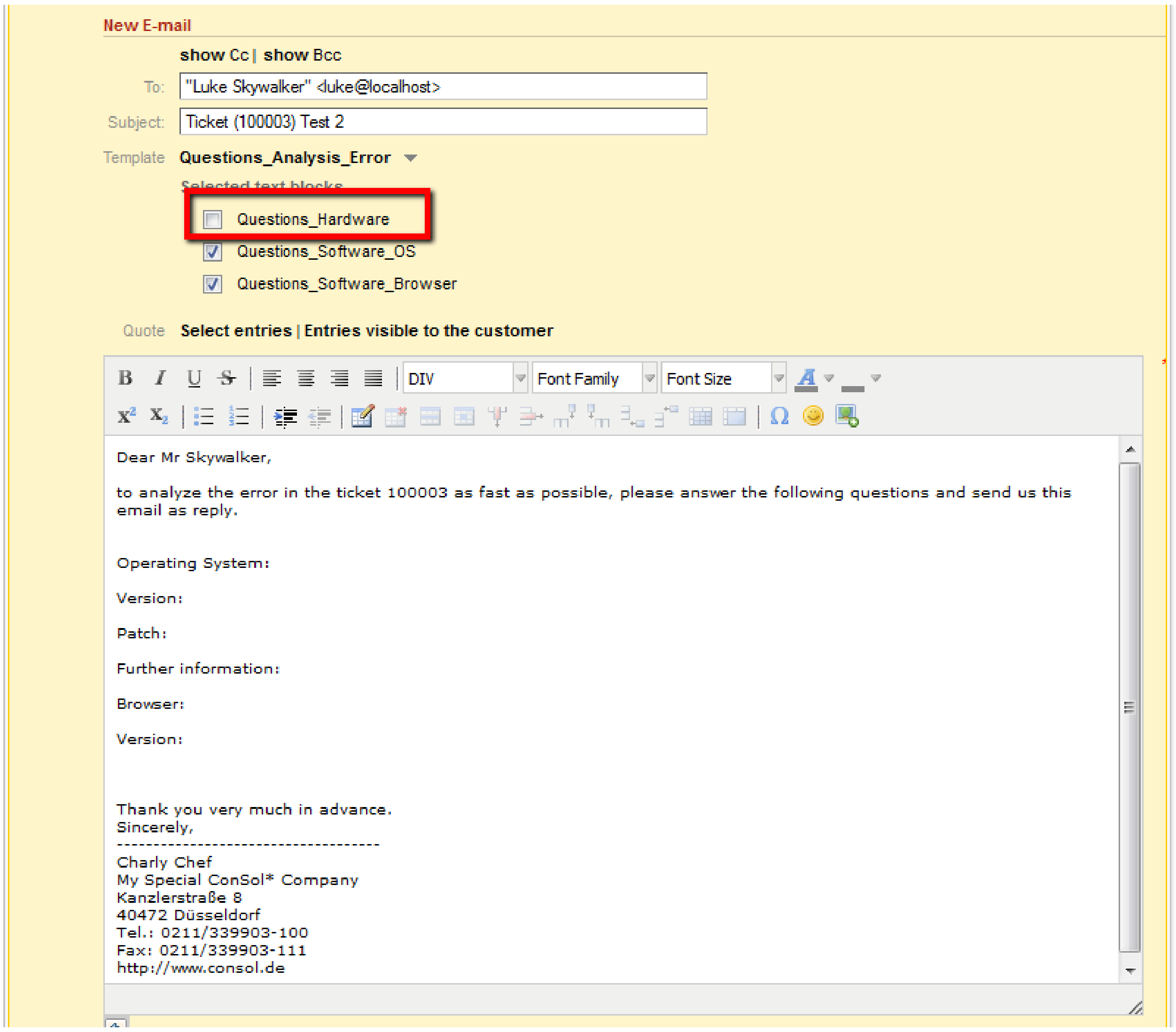
Patch:

Further information:

Browser:

Version:

Fig. 18: ConSol*CM Web Client - Ticket E-Mail Editor: All Text Blocks Selected



"Luke Skywalker" <luke@localhost>

Ticket (100003) Test 2

Questions_Analysis_Error

- [ ] Questions_Hardware
- [✓] Questions_Software_OS
- [✓] Questions_Software_Browser

```
Dear Mr Skywalker,

to analyze the error in the ticket 100003 as fast as possible, please answer the following questions and send us this
email as reply.


Operating System:

Version:

Patch:

Further information:

Browser:

Version:



Thank you very much in advance.
Sincerely,
------------------------------------
Charly Chef
My Special ConSol* Company
Kanzlerstraße 8
40472 Düsseldorf
Tel.: 0211/339903-100
Fax: 0211/339903-111
http://www.consol.de
```

Fig. 19: ConSol*CM Web Client - Ticket E-Mail Editor: One Text Block Deactivated

Create and Use a Script

Sometimes a system has to have a certain *intelligence* regarding the words and phrases used in e-mail templates, because they are not static but have to be adapted in a dynamic way. A standard example is the use of *Dear Sir* for male customers (salutation = "Mr") and *Dear Madam* for female customers (salutation = "Mrs").

Use cases like this can be covered using *scripts* in the Template Manager.

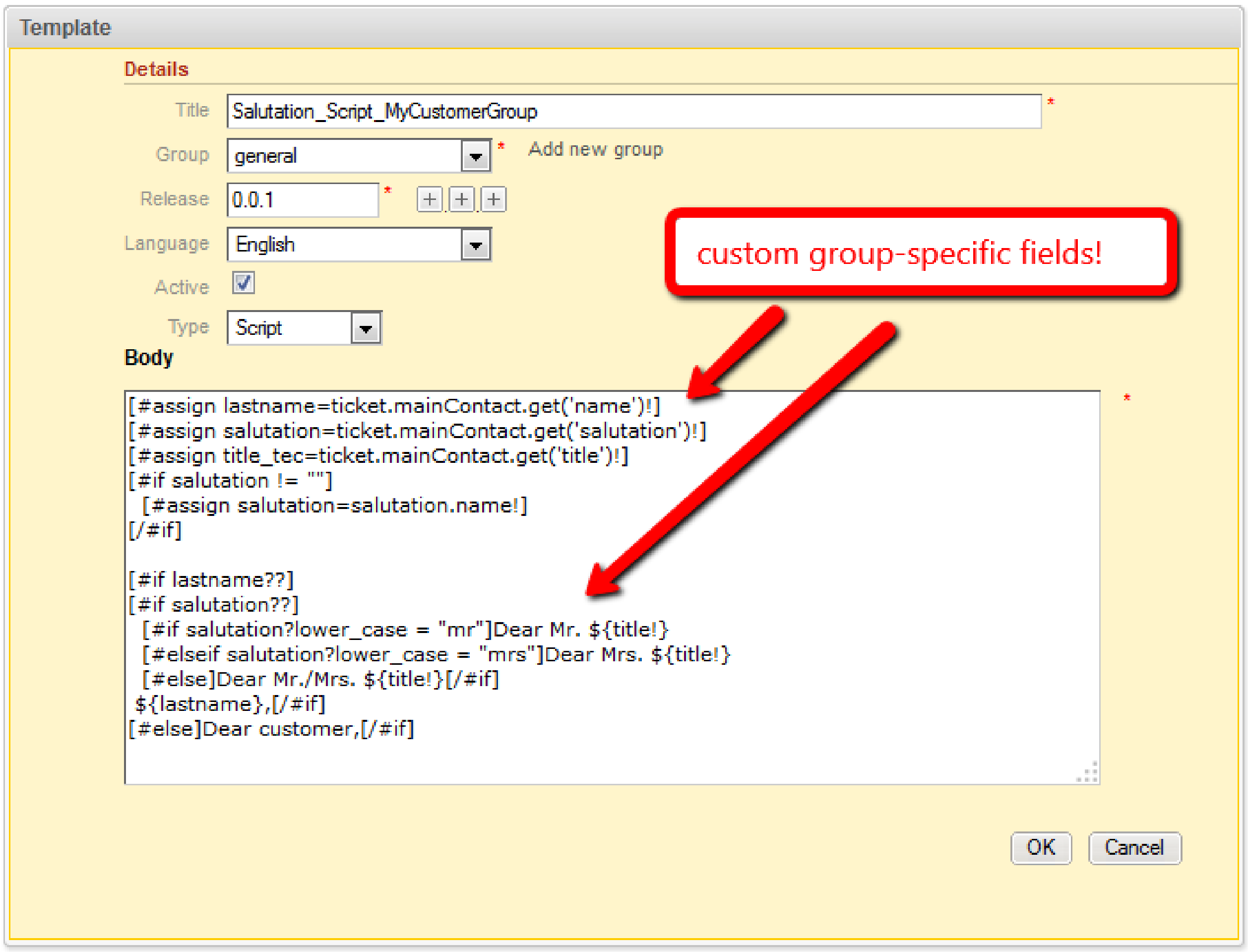
This can only be realized by a ConSol*CM administrator. When you log into the Web Client as a regular user with template managing permissions, you can define all template types but no *scripts*. In order to be able to select *Script* as template type, you have to log in with an administrator account.

For information about the syntax that is used, please see the following web links:

- FreeMarker
- FreeMarker directives

Please note that in the standard Freemarker notation, angle brackets (<>) are used for statements. Here, in an HTML environment, this cannot work, and therefore, instead of angle brackets, square brackets ([ ]) have to be used.

An example *script* is the following:



Title	Salutation_Script_MyCustomerGroup
Group	general
Release	0.0.1
Language	English
Active	✓
Type	Script

Body

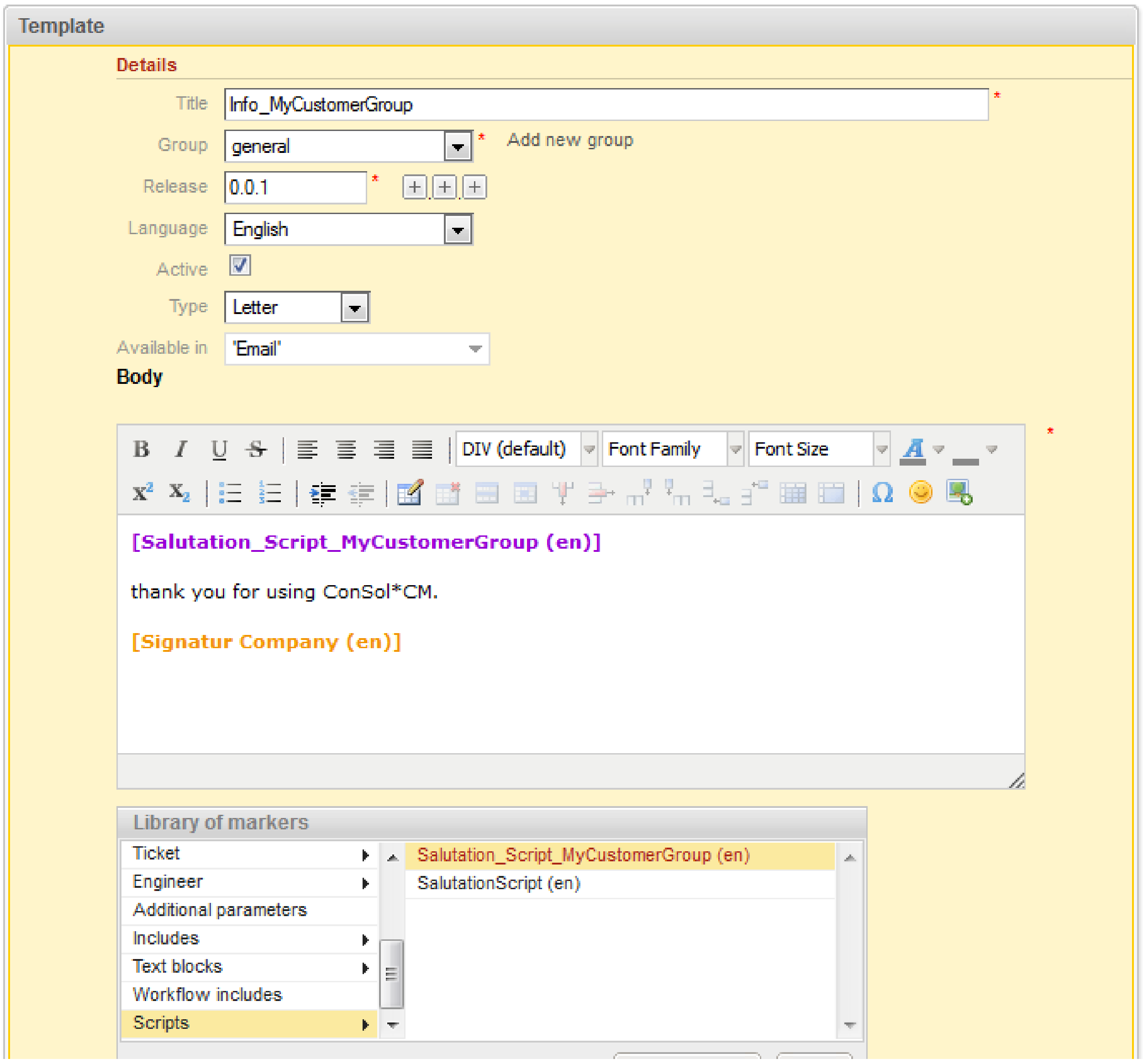
```
[#assign lastname=ticket.mainContact.get('name')!]
[#assign salutation=ticket.mainContact.get('salutation')!]
[#assign title_tec=ticket.mainContact.get('title')!]
[#if salutation != ""]
  [#assign salutation=salutation.name!]
[/#if]

[#if lastname??]
[#if salutation??]
  [#if salutation?lower_case = "mr"]Dear Mr. ${title!}
  [#elseif salutation?lower_case = "mrs"]Dear Mrs. ${title!}
  [#else]Dear Mr./Mrs. ${title!}[/#if]
 ${lastname},[/#if]
[#else]Dear customer,[/#if]
```

Fig. 20: ConSol*CM Web Client - Template Manager: Example Script for Salutation

Please keep in mind that ...

- the values of the fields are the technical values (in the example, the technical value of the field *salutation* is *mr* / *mrs*, the localized value for EN would be *Mr* / *Mrs*). Always use the technical values!
- the fields are the custom fields and data object group fields managed in the Admin Tool. Please refer to sections Custom Field Administration and Setting Up the Customer Data Model for details.

The *script* is then integrated into a *letter* template:



Title	Info_MyCustomerGroup
Group	general
Release	0.0.1
Language	English
Active	✓
Type	Letter
Available in	'Email'

[Salutation_Script_MyCustomerGroup (en)]

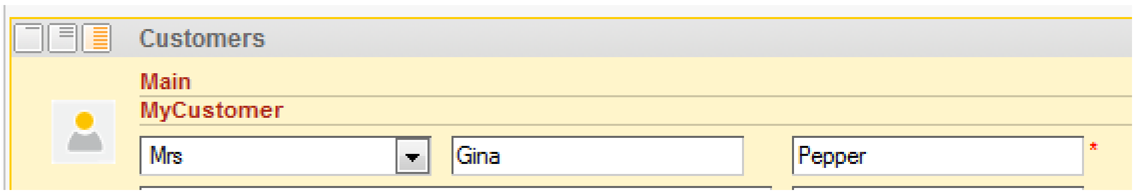
thank you for using ConSol*CM.

[Signatur Company (en)]

Fig. 21: ConSol*CM Web Client - Template Manager: Insert Script into Letter

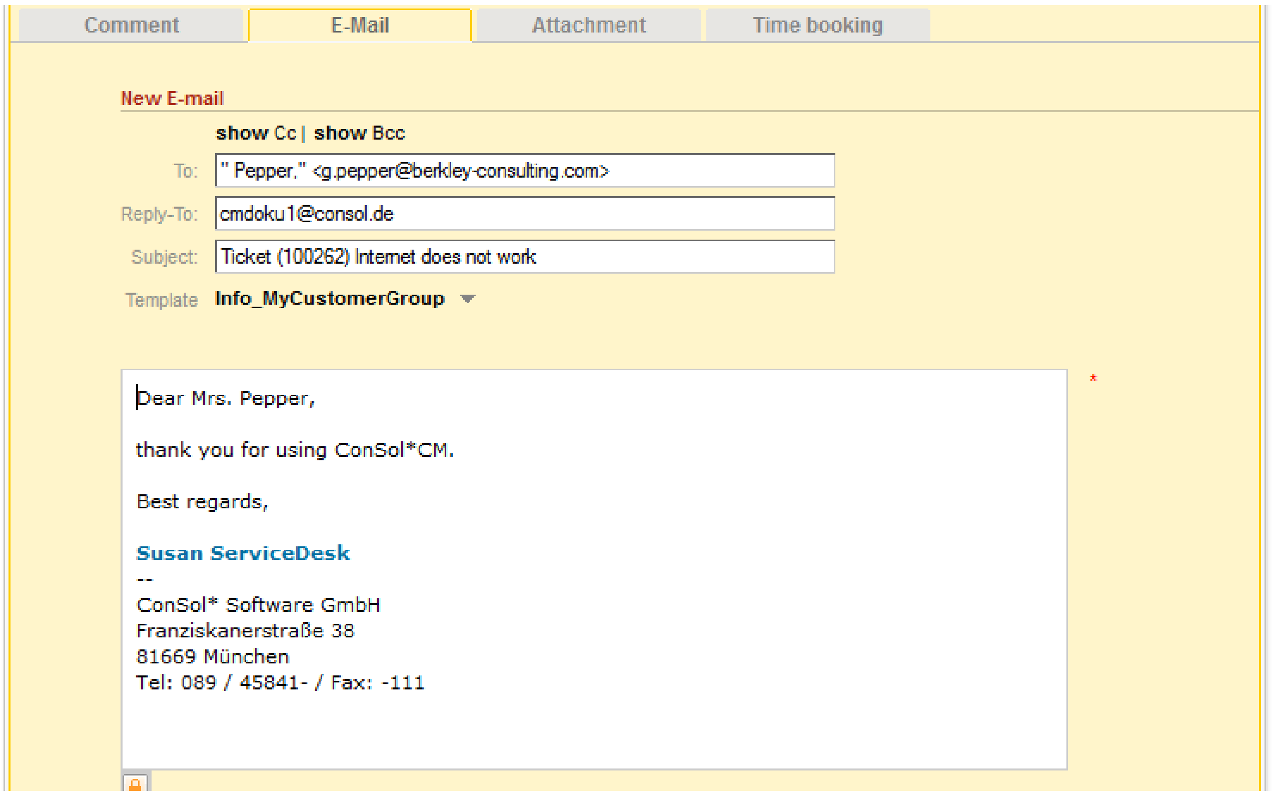
In the Web Client, the e-mails are formatted as requested.

Example 1 (for **Mrs**):



Mrs	Gina	Pepper

Fig. 22: ConSol*CM Web Client - E-Mails Formatted by Script (Customer Data, Example 1)



To: " Pepper," <g.pepper@berkley-consulting.com>
Reply-To: cmdoku1@consol.de
Subject: Ticket (100262) Internet does not work
Template Info_MyCustomerGroup

Dear Mrs. Pepper,

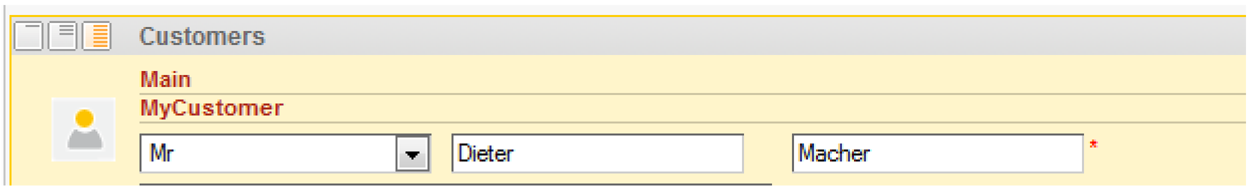
thank you for using ConSol*CM.

Best regards,

Susan ServiceDesk
--
ConSol* Software GmbH
Franziskanerstraße 38
81669 München
Tel: 089 / 45841- / Fax: -111

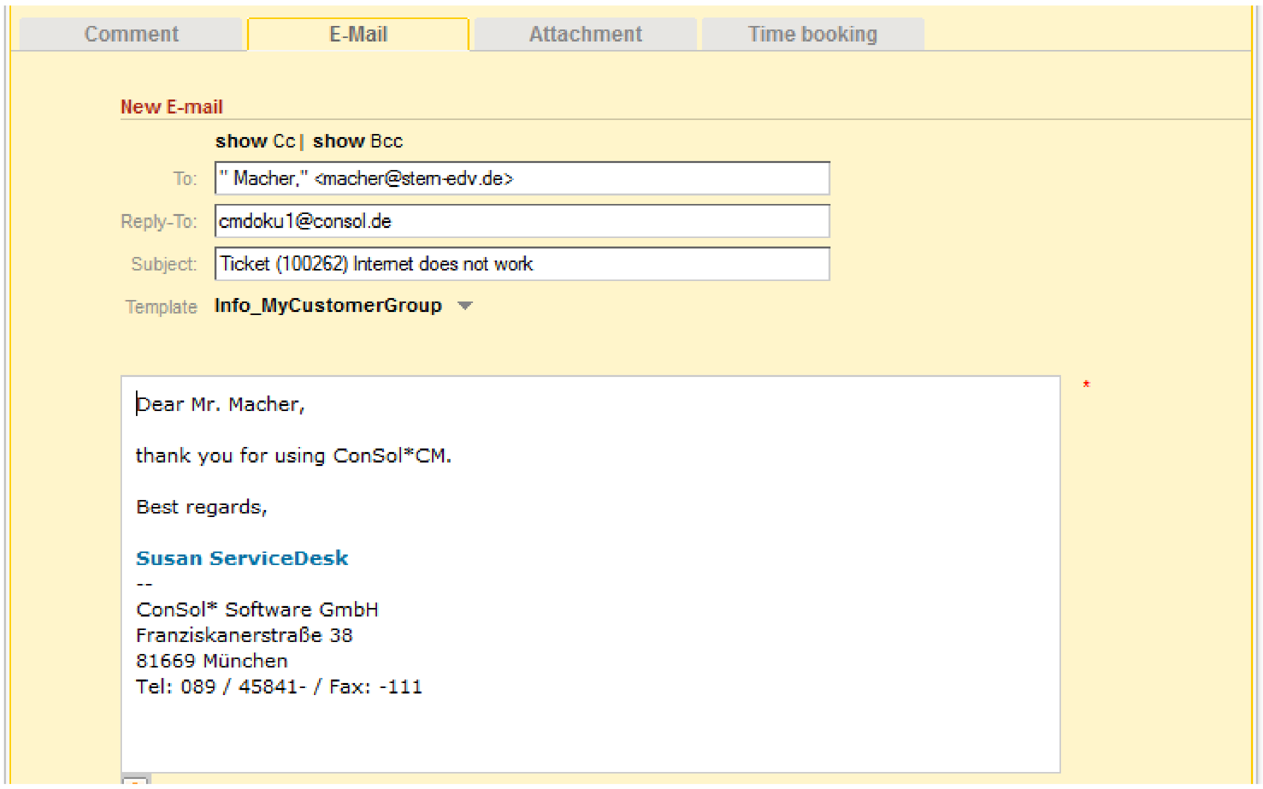
Fig. 23: ConSol*CM Web Client - E-Mails Formatted by Script (E-Mail, Example 1)

Example 2 (for **Mr**):



Mr | Dieter | Macher

Fig. 24: ConSol*CM Web Client - E-Mails Formatted by Script (Customer Data, Example 2)



To: " Macher," <macher@stern-edv.de>

Reply-To: cmdoku1@consol.de

Subject: Ticket (100262) Internet does not work

Template Info_MyCustomerGroup

Dear Mr. Macher,

thank you for using ConSol*CM.

Best regards,

Susan ServiceDesk
--
ConSol* Software GmbH
Franziskanerstraße 38
81669 München
Tel: 089 / 45841- / Fax: -111

Fig. 25: ConSol*CM Web Client - E-Mails Formatted by Script (E-Mail, Example 2)

Binding Templates to Queues or to Specific Parameters

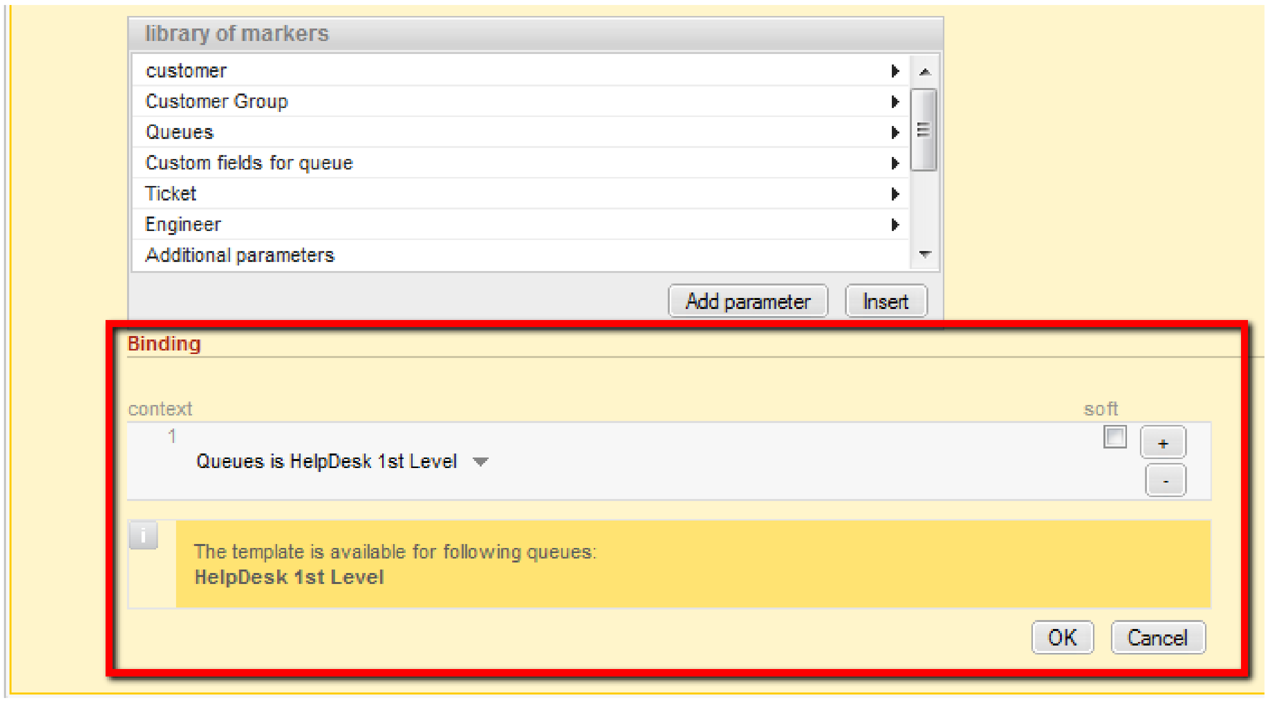
The section *Binding* is the last section of the *New Template* page of the Template Manager.

For every template you can decide if it should be displayed everywhere without any restrictions (i.e. in every queue and without regarding any parameters) or if it should be bound (= restricted) to specific criteria. This can be ...

- queues
- queue-specific parameters (e.g. display the template only when the priority *high* has been set for the ticket)

You can select queues and/or parameters by selecting a *context*, see the following pictures for examples. Use the "+" button to add more binding parameters or the "-" button to remove existing parameters.

The template should only be displayed in the *Helpdesk 1st Level* queue:

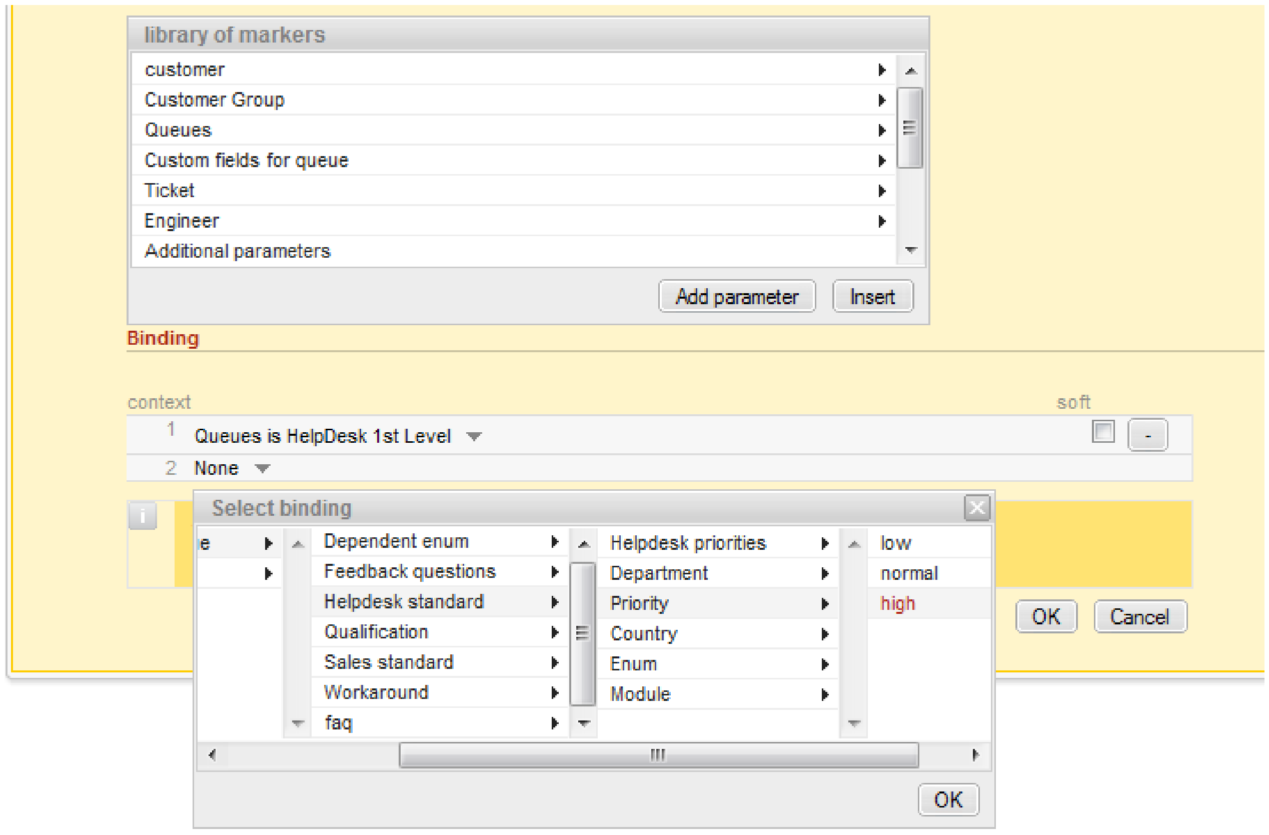


Queues is HelpDesk 1st Level

The template is available for following queues:
HelpDesk 1st Level

Fig. 26: ConSol*CM Web Client - Show Template for Specific Queue

The template should only be displayed in the *Helpdesk 1st Level* queue and only if the priority is *high*:



1 Queues is HelpDesk 1st Level
2 None

Fig. 27: ConSol*CM Web Client - Display Template for Specific Queue and Priority

Hard and Soft Binding

When the template is only displayed in one (or more) selected queue(s) as shown in the example above, the template is *bound* to those queues or to any other selected (restricting) parameter. There are two types of binding:

- **Hard binding**
  The check box *soft* is **not** checked:
  The template is only displayed (offered in the Ticket E-Mail Editor or Ticket Comment Editor) in the selected queues or for the selected parameters. The engineer who works with the template cannot change this configuration.
- **Soft binding**
  The check box *soft* is checked:
  As default setting, the template is only displayed (offered in the Ticket E-Mail Editor or Ticket Comment Editor) in the selected queues or for the selected parameters. However, the engineer can change the display by clicking on the button *More templates*. Then all softly bound templates are displayed as well.

26.2.4 Migrating Templates from CM Version 6.8 and Less to CM Version 6.9 and Up

When a ConSol*CM system is updated from a version 6.8 and less to a version 6.9 and up, the scripts in the Template Manager have to be checked and modified manually.

All parameters which refer to tickets and queues do not have to be changed. However, due to the new customer model, *FlexCDM*, the syntax for references to data object group fields has to be adapted.

26.2.5 Page Customization for E-Mail Template Functionalities

Please refer to section Page Customization to learn some details about how to adapt e-mail template parameters.

26.3 CM.Doc

- CM.Doc
  - Introduction to CM.Doc
  - Requirements for Using CM.Doc
  - Availability of CM.Doc
  - Configuring the ConSol*CM System for CM.Doc
  - Creating an Engineer Role with Permissions for the Doc Template Manager
  - Creating MS Word Templates and Making Them Available
    - Creating MS Word Templates
    - Creating a New MS Word Template, Step1: Creating the Template Object in the Template Library
      - Selection of the Data Object and Its Consequences
      - How to Control in which Tickets the MS Word Template will be Offered
    - Creating a New MS Word Template, Step2: Creating and Editing the Required MS Word Document
      - Details about ConSol*CM Keys for the Mail Merge Fields
  - Using MS Word Templates from within the Web Client
    - Creating a New MS Word Attachment Using a Word Template
    - Working with Existing MS Word Attachments

26.3.1 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 for other reasons. 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 MS Word documents.

Templates guarantee that ...

- all documents of one type are identical (text and layout).
- all documents match the company's CD (corporate design).
- no engineer has to 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/or of the respective company, additional contacts are NOT relevant in CM.Doc!)
- engineer data (name, phone number, e-mail address of the engineer who works on the case)

When CM.Doc is active in ConSol*CM, the engineer can select the required MS Word template in the ticket. The document is opened in MS Word 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 (MS Word) on their PCs.

With special adaptations implemented by the ConSol*CM consulting team, the CM system can be extended in a way that *.docx* documents can also be converted to *.pdf* files in order to make sure no further changes can be made to the document.

26.3.2 Requirements for Using CM.Doc

On the PC or laptop, the following requirements have to be met to use CM.Doc:

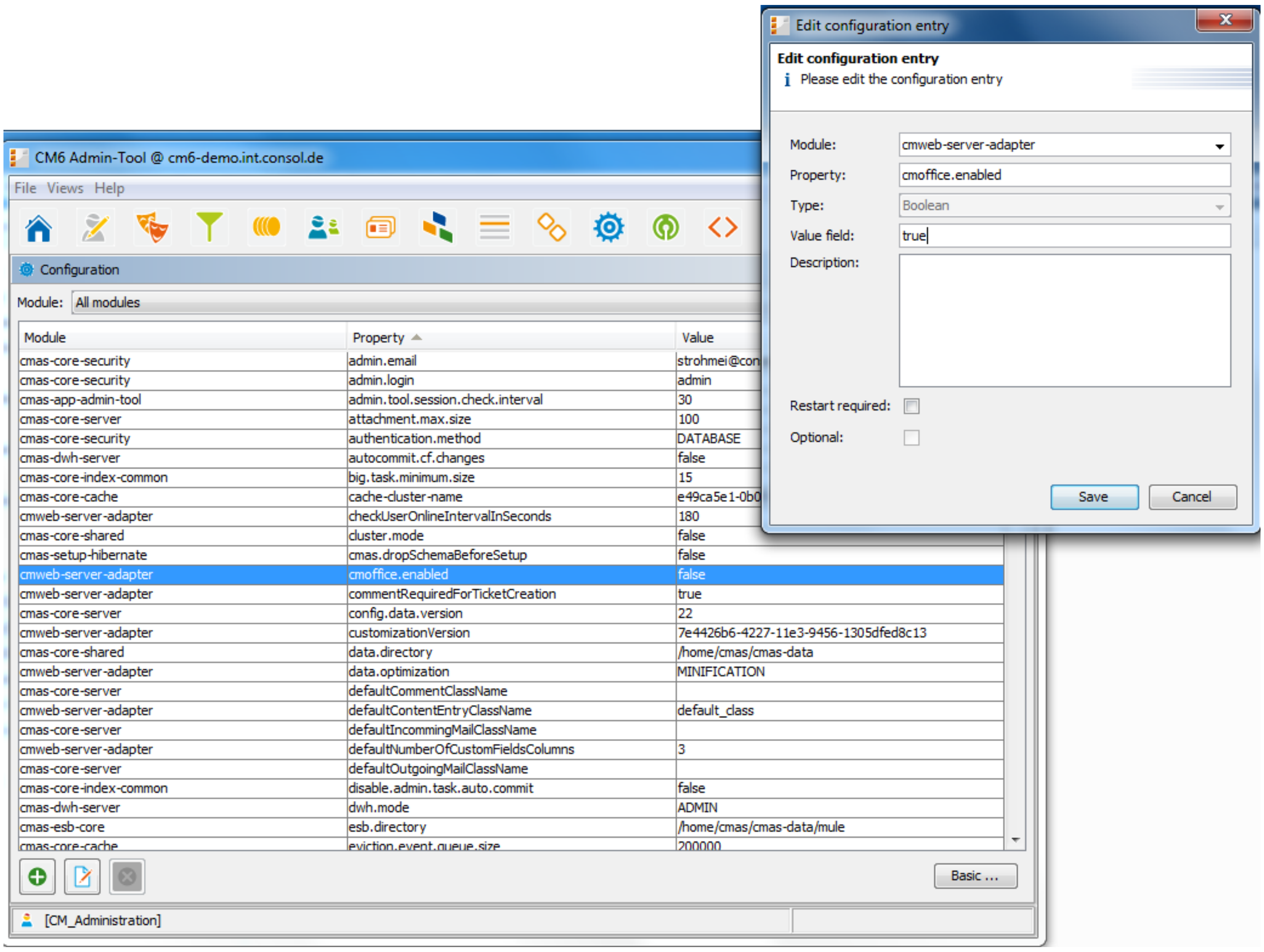
- A JRE (Java Runtime Environment) for the web browser has to be installed, because CM.Doc is based on Java applets. For the supported Java versions, please refer to the current *System Requirements*.
- MS Word has to be installed. For the supported MS Word versions, please refer to the current *System Requirements*.

26.3.3 Availability of CM.Doc

CM.Doc is available in ConSol*CM version 6.7 and higher and is part of the standard function set of the application.

26.3.4 Configuring the ConSol*CM System for CM.Doc

If you want to activate CM.Doc in your ConSol*CM system, the first step is to set the system property *cmweb-server-adapter*, *cmoffice.enabled* to *true*.



Module: All modules

Module	Property	Value
cmas-core-security	admin.email	strohmei@con
cmas-core-security	admin.login	admin
cmas-app-admin-tool	admin.tool.session.check.interval	30
cmas-core-server	attachment.max.size	100
cmas-core-security	authentication.method	DATABASE
cmas-dwh-server	autocommit.cf.changes	false
cmas-core-index-common	big.task.minimum.size	15
cmas-core-cache	cache-cluster-name	e49ca5e1-0b0
cmweb-server-adapter	checkUserOnlineIntervalInSeconds	180
cmas-core-shared	cluster.mode	false
cmas-setup-hibernate	cmas.dropSchemaBeforeSetup	false
cmweb-server-adapter	cmoffice.enabled	false
cmweb-server-adapter	commentRequiredForTicketCreation	true
cmas-core-server	config.data.version	22
cmweb-server-adapter	customizationVersion	7e4426b6-4227-11e3-9456-1305dfed8c13
cmas-core-shared	data.directory	/home/cmas/cmas-data
cmweb-server-adapter	data.optimization	MINIFICATION
cmas-core-server	defaultCommentClassName	
cmweb-server-adapter	defaultContentEntryClassName	default_class
cmas-core-server	defaultIncommingMailClassName	
cmweb-server-adapter	defaultNumberOfCustomFieldsColumns	3
cmas-core-server	defaultOutgoingMailClassName	
cmas-core-index-common	disable.admin.task.auto.commit	false
cmas-dwh-server	dwh.mode	ADMIN
cmas-esb-core	esb.directory	/home/cmas/cmas-data/mule
cmas-core-cache	eviction.event.queue.size	200000

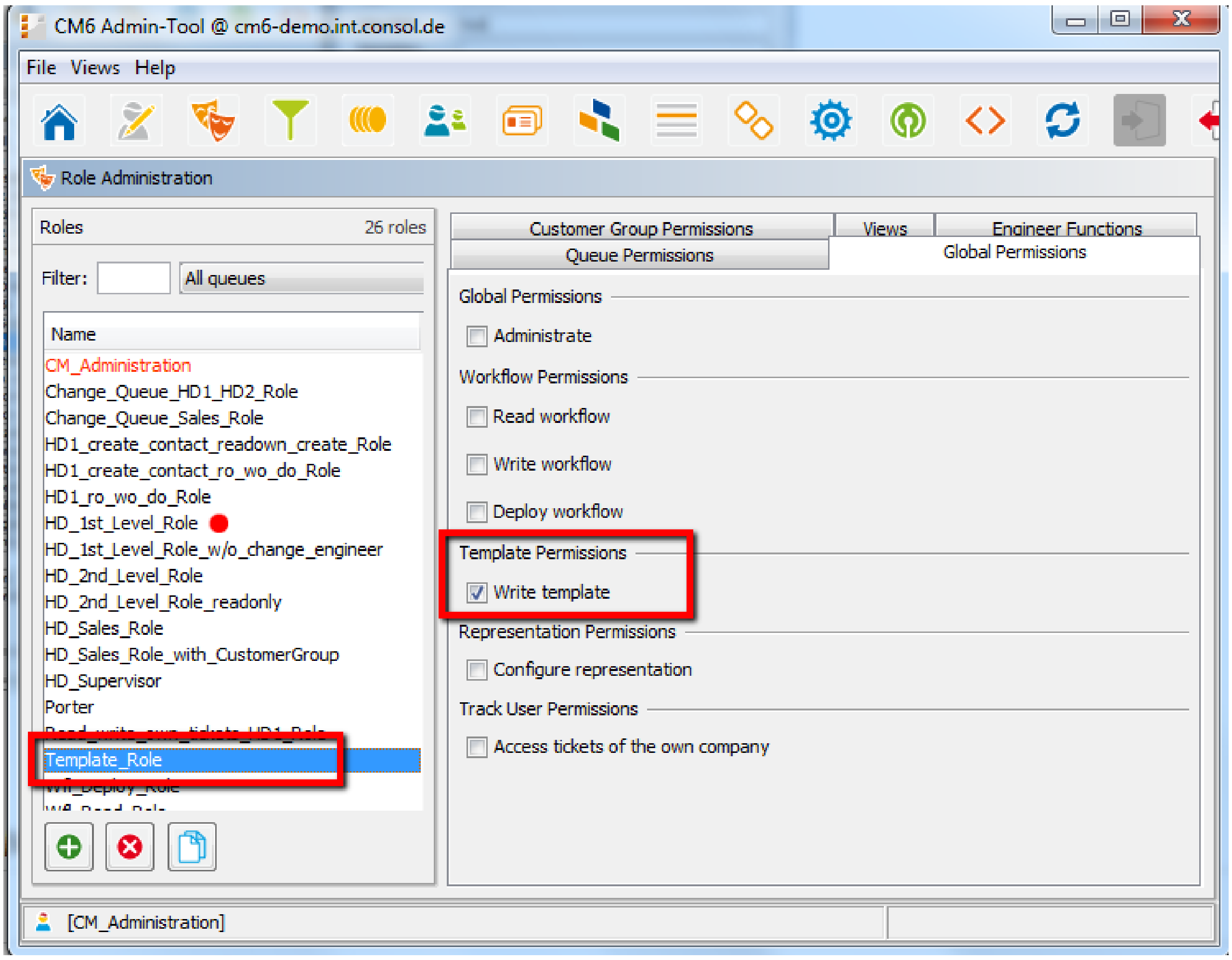
Edit configuration entry

Module:	cmweb-server-adapter
Property:	cmoffice.enabled
Type:	Boolean
Value field:	true
Description:	
Restart required:	☐
Optional:	☐

Fig. 1: ConSol*CM Admin Tool - Configuration of Property for CM.Doc

26.3.5 Creating an Engineer Role with Permissions for the Doc Template Manager

Only an engineer who has the permission *Write template* (see the following figure) can start the *Doc Template Manager* in the Web Client. So, as a second step, you have to create one or more role(s) with the respective permissions. For a detailed explanation about setting role permissions, please refer to section Role Administration.



Roles 26 roles

Filter: All queues

Name
CM_Administration
Change_Queue_HD1_HD2_Role
Change_Queue_Sales_Role
HD1_create_contact_readown_create_Role
HD1_create_contact_ro_wo_do_Role
HD1_ro_wo_do_Role
HD_1st_Level_Role ●
HD_1st_Level_Role_w/o_change_engineer
HD_2nd_Level_Role
HD_2nd_Level_Role_readonly
HD_Sales_Role
HD_Sales_Role_with_CustomerGroup
HD_Supervisor
Porter
Read_write_own_tickets_HD1_Role
Template_Role
Wf_Deploy_Role
Wf_Read_Role

Global Permissions

- [ ] Administrate

Workflow Permissions

- [ ] Read workflow
- [ ] Write workflow
- [ ] Deploy workflow

Template Permissions

- [✓] Write template

Representation Permissions

- [ ] Configure representation

Track User Permissions

- [ ] Access tickets of the own company

[CM_Administration]

Fig. 2: ConSol*CM Admin Tool - Necessary Template Permissions

Consulting Best Practice:

We recommend to create a role (*e.g. Template_Role*) that has only the permission *Write template*, no queue permissions or other permissions are granted. Every user who should have access to the Doc Template Manager can be given this role. That way, there is no merge between regular user permissions and Word template permissions and you can grant and retrieve the Word template permission in a very flexible way.

However, the permission *Write template* also grants access to the Template Manager (for text templates, see section The ConSol CM Template Manager).

26.3.6 Creating MS Word Templates and Making Them Available

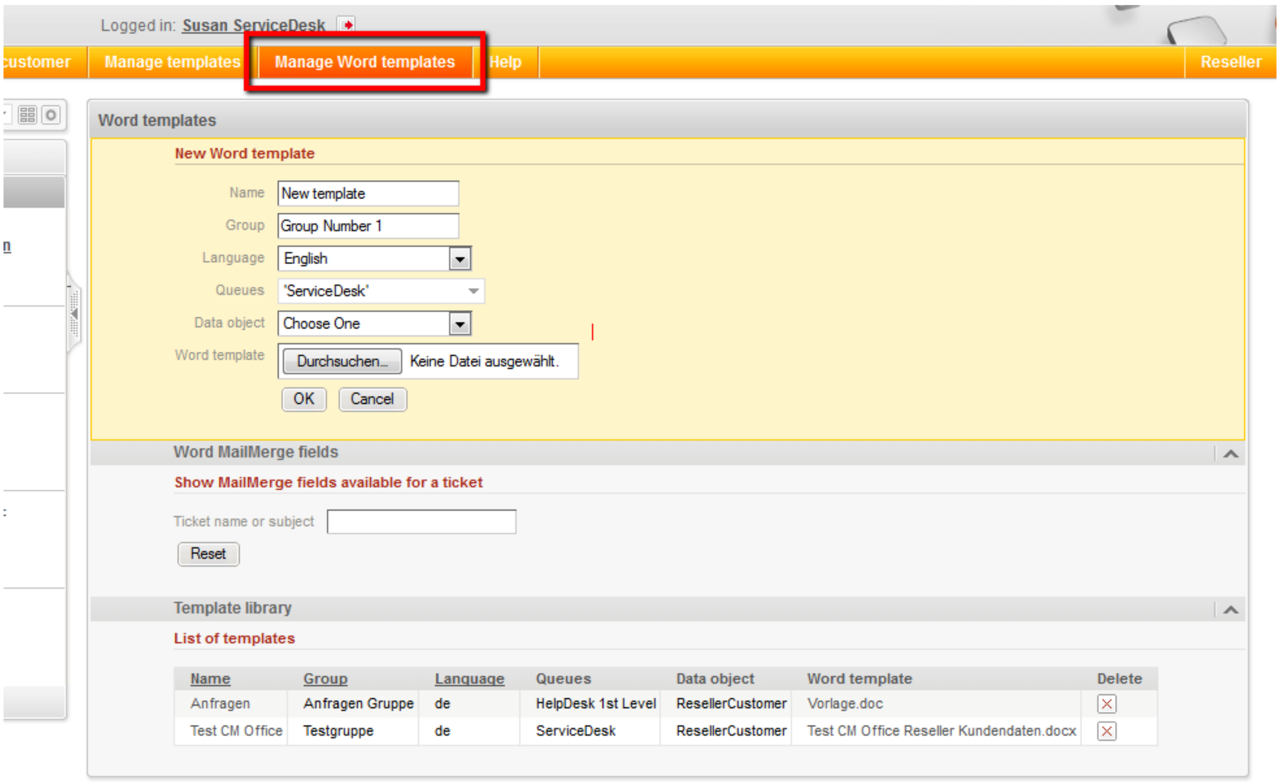
Creating MS Word Templates

As a third step, you have to create the MS Word templates. This is done using MS Word. Please create *.doc* or *.docx* files as templates, **not** *.dot* files!

Creating a New MS Word Template, Step1: Creating the Template Object in the Template Library

As a fourth step, you have to fill in the requested data fields as *merge* fields in the MS Word template, i.e. you create a CM.Doc template from your regular MS Word document. This is done using the ConSol*CM Web Client.

To perform steps three and four, log in to the Web Client and click on *Manage Word templates* in the main menu to open the *Doc Template Manager*.



New Word template

Name	New template
Group	Group Number 1
Language	English
Queues	'ServiceDesk'
Data object	Choose One
Word template	Keine Datei ausgewählt.

Show MailMerge fields available for a ticket

Ticket name or subject

List of templates

Name	Group	Language	Queues	Data object	Word template	Delete
Anfragen	Anfragen Gruppe	de	HelpDesk 1st Level	ResellerCustomer	Vorlage.doc	☒
Test CM Office	Testgruppe	de	ServiceDesk	ResellerCustomer	Test CM Office Reseller Kundendaten.docx	☒

Fig. 3: ConSol*CM Web Client - Doc Template Manager

The Doc 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 Doc Template Manager.
- **Language**
  Select the required language. The languages that have been activated in the Admin Tool are offered.
- **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 offered, please see section Selection of the Data Object and its consequences for a detailed explanation about which data object you have to select.
- **Word 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 an MS Word template, you have to select a data object, i.e. a customer object or a company object from your customer data model. If you are not familiar with the ConSol*CM customer data model concept, please read section The CM Customer Data Model - FlexCDM fist.

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)]

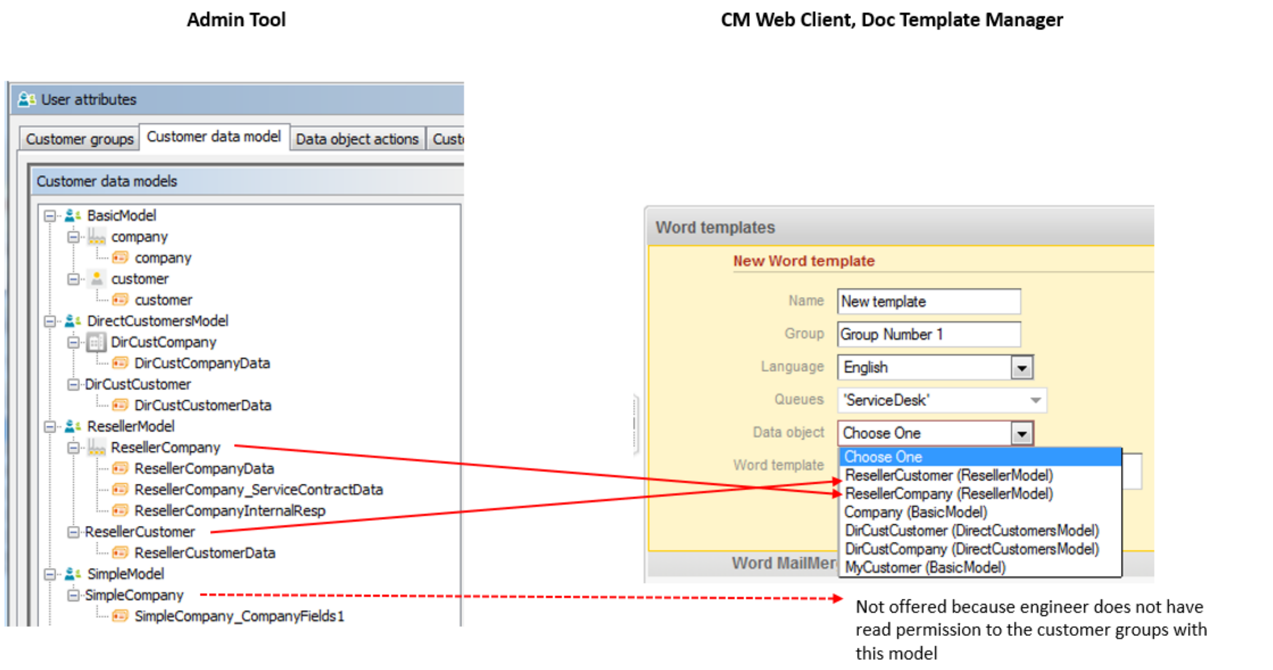
For a data object to be listed in the drop-down menu, some conditions have to be met:

1. The customer data model with the respective data object has to be created in the Admin Tool
2. The customer data model has to be assigned to at least one customer group
3. The engineer who works on the Word 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 MS Word template will be offered only in tickets which have main customers from a customer group with this customer data model. This means, if you want to use the same MS Word template text for two customer groups with two different customer data models, you have to create two ConSol*CM MS Word templates, one for each customer data model. Of course, you can use the same MS Word template for two or more customer groups which all have the same customer data model.

If you try to assign a data object to an MS Word 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 on company level, if for this customer data model the parameter *Company as customer* has been set.



Name: New template
Group: Group Number 1
Language: English
Queues: 'ServiceDesk'
Data object: Choose One

Choose One
ResellerCustomer (ResellerModel)
ResellerCompany (ResellerModel)
Company (BasicModel)
DirCustCustomer (DirectCustomersModel)
DirCustCompany (DirectCustomersModel)
MyCustomer (BasicModel)

Fig. 4: Selection of Data Object in Doc Template Manager

Please be aware that when you leave the *Data object* field empty, the MS Word template will theoretically be available for all customer groups, but there might be runtime errors in case 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 to always select a data object to keep the system in order.

How to Control in which Tickets the MS Word Template will be Offered

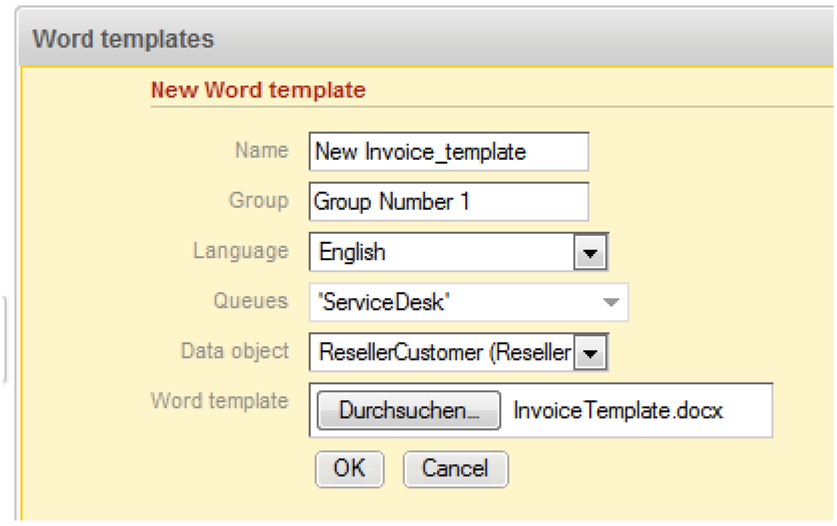
If the MS Word template is offered in a certain ticket (see section Using MS Word Templates from within the Web Client below) depends ob two parameters:

- the queue - the template is only offered for ticket in the selected queue(s)
- the customer data model - the template is only offered in tickets which have as main customer a customer object which is based on the selected data object

Example (we anticipate the use of MS Word templates in the Web Client, for details see section Using MS Word Templates from within the Web Client):

- First, the MS Word template is defined for the queue *Service Desk* and for the data object *ResellerCustomer* (first figure).
- In the Web Client the template can be used in a ticket which is in the queue *Service Desk* and which has as main customer an object of the *Reseller Customer* data object type (second figure).
- In the Web Client, the template cannot be used in a ticket which is in the queue *Service Desk* and which has as main customer of a data object which is NOT of the *Reseller Customer* data object type (third figure).

- In the Web Client, the template cannot be used in all tickets which are NOT in the queue Service Desk (not shown).



Word templates

New Word template

Name	New Invoice_template
Group	Group Number 1
Language	English
Queues	'ServiceDesk'
Data object	ResellerCustomer (Reseller
Word template	InvoiceTemplate.docx

Fig. 5: Definition of an MS Word Template for one queue and for a certain data object

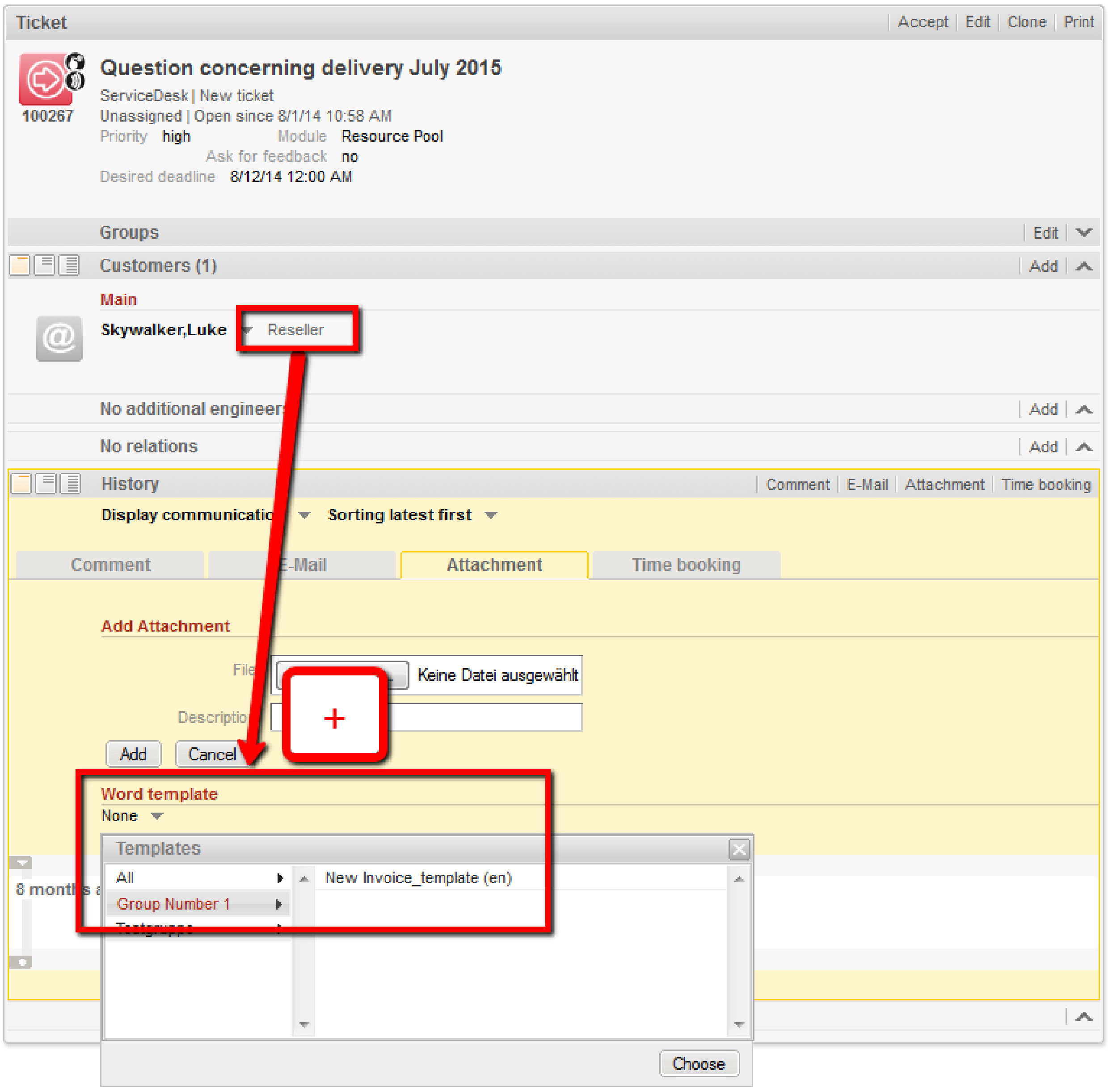
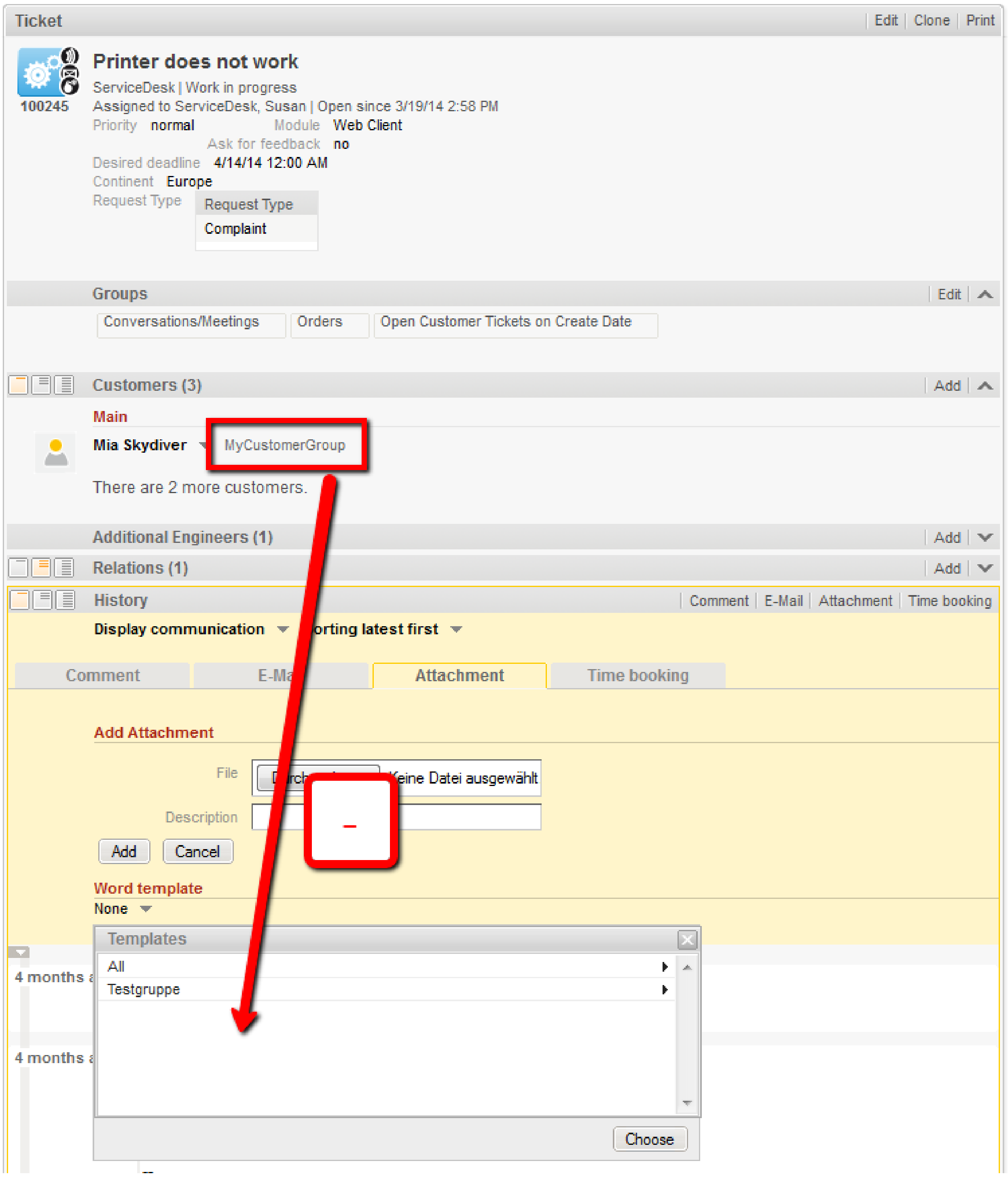


Fig. 6: MS Word Template offered because of queue and customer group


Ticket | Edit | Clone | Print

Printer does not work
100245
ServiceDesk | Work in progress
Assigned to ServiceDesk, Susan | Open since 3/19/14 2:58 PM
Priority normal | Module Web Client
Ask for feedback no
Desired deadline 4/14/14 12:00 AM
Continent Europe
Request Type: Request Type — Complaint

Groups | Edit
Conversations/Meetings | Orders | Open Customer Tickets on Create Date

Customers (3) | Add
Main
Mia Skydiver — MyCustomerGroup
There are 2 more customers.

Additional Engineers (1) | Add
Relations (1) | Add
History | Comment | E-Mail | Attachment | Time booking
Display communication — Sorting latest first

Add Attachment
File: Keine Datei ausgewählt
Description:
Add | Cancel
Word template: None

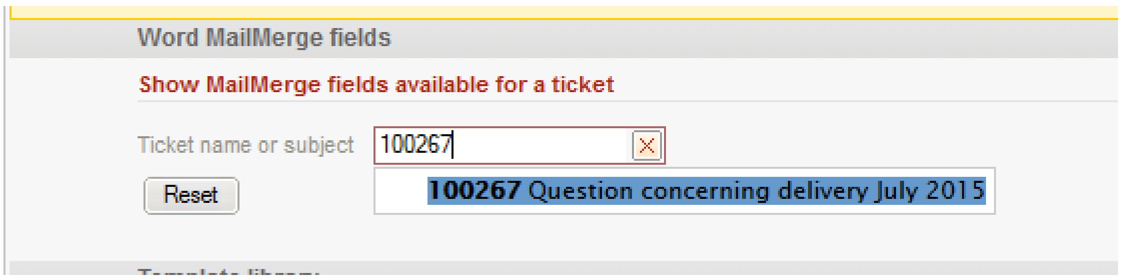
Templates
All
Testgruppe
Choose

Fig. 7: MS Word Template NOT offered because of queue and customer group

Creating a New MS Word Template, Step2: Creating and Editing the Required MS Word Document

When you have filled in all fields and clicked on *Ok*, the new template appears in the *Template library*, *List of templates*. You can perform the following step directly after entering the new template data or you mark the template name (in the *Word template* column) you would like to edit in the list.

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 requested fields by using the *Word 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.

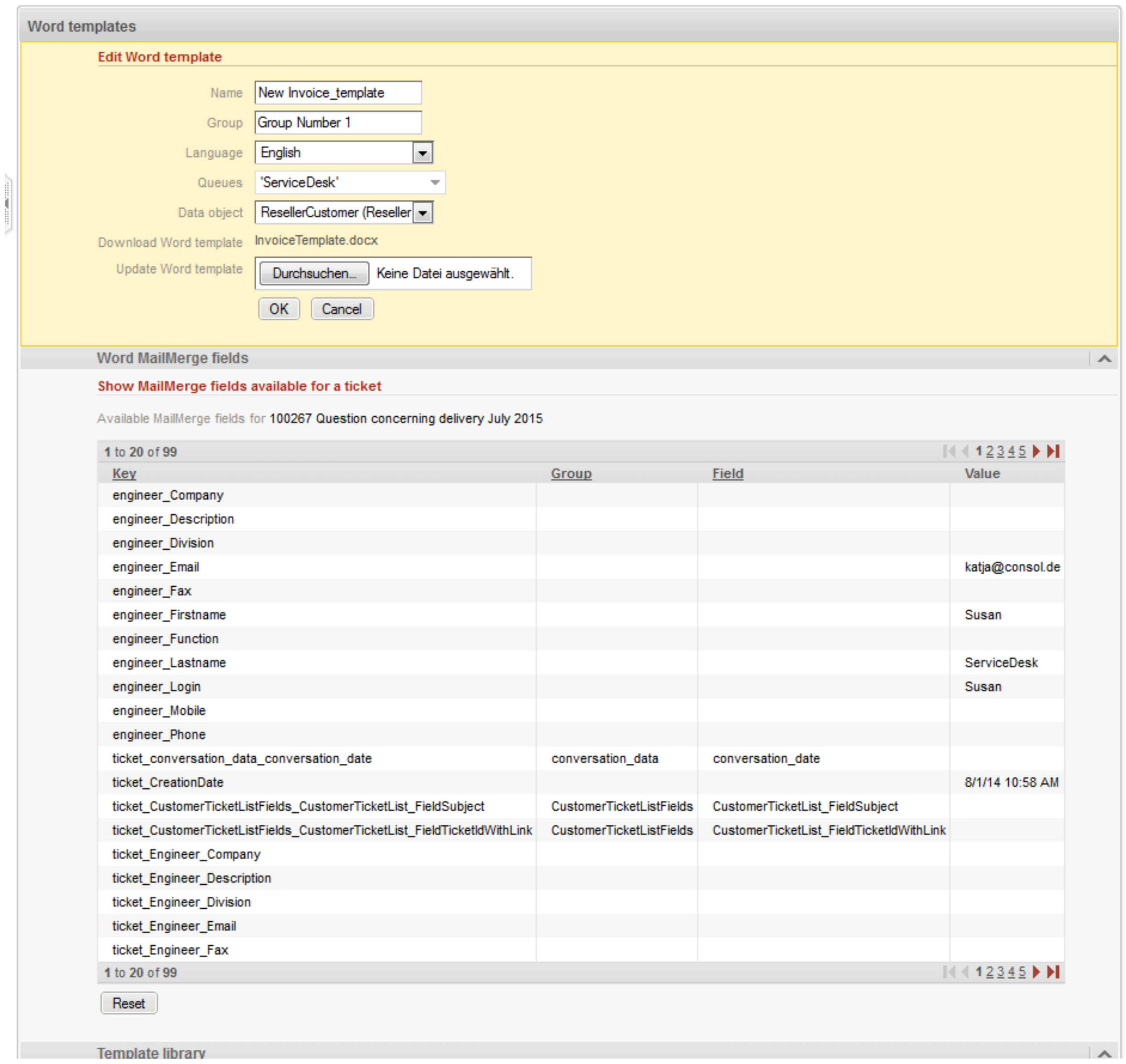


Ticket name or subject: 100267

100267 Question concerning delivery July 2015

Fig. 8: Selection of a Ticket to see all Required Data Fields

All available *MailMerge* fields are displayed:



Name	New Invoice_template
Group	Group Number 1
Language	English
Queues	'ServiceDesk'
Data object	ResellerCustomer (Reseller
Download Word template	InvoiceTemplate.docx
Update Word template	Durchsuchen... Keine Datei ausgewählt.

Available MailMerge fields for 100267 Question concerning delivery July 2015

1 to 20 of 99 — 1 2 3 4 5

Key	Group	Field	Value
engineer_Company			
engineer_Description			
engineer_Division			
engineer_Email			katja@consol.de
engineer_Fax			
engineer_Firstname			Susan
engineer_Function			
engineer_Lastname			ServiceDesk
engineer_Login			Susan
engineer_Mobile			
engineer_Phone			
ticket_conversation_data_conversation_date	conversation_data	conversation_date	
ticket_CreationDate			8/1/14 10:58 AM
ticket_CustomerTicketListFields_CustomerTicketList_FieldSubject	CustomerTicketListFields	CustomerTicketList_FieldSubject	
ticket_CustomerTicketListFields_CustomerTicketList_FieldTicketIdWithLink	CustomerTicketListFields	CustomerTicketList_FieldTicketIdWithLink	
ticket_Engineer_Company			
ticket_Engineer_Description			
ticket_Engineer_Division			
ticket_Engineer_Email			
ticket_Engineer_Fax			

1 to 20 of 99 — 1 2 3 4 5

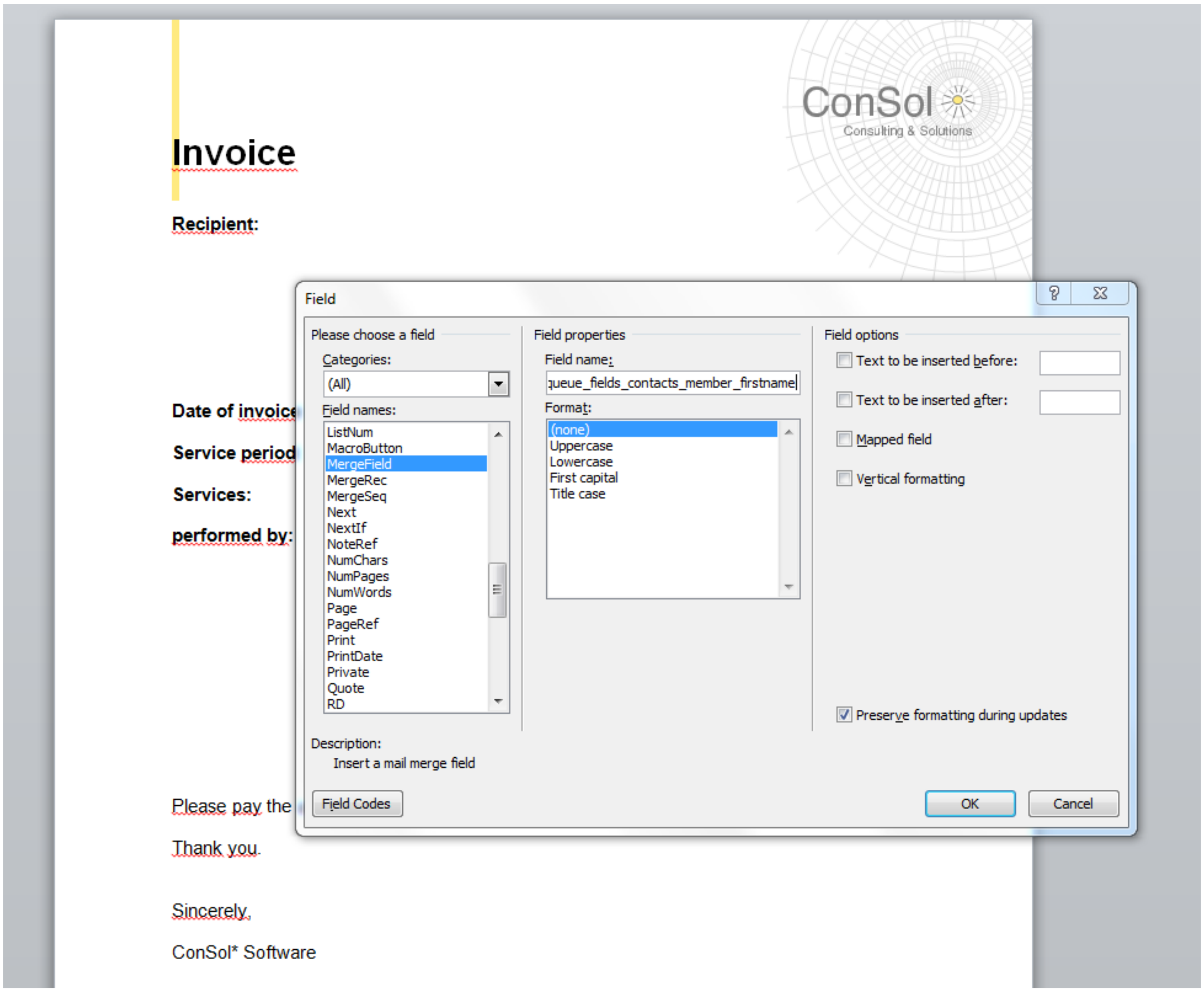
Fig. 9: ConSol*CM Web Client - Show Available MailMerge Fields

The list contains the following columns:

- **Key**
  The name of the field (this is the one you will have to use for the Merge Field later on)

- **Group**
  This is set for customer data and for ticket data. For customer data, the data object group is shown (here, only the data object groups from the selected data object are offered), for ticket data, the Custom Field Group is shown.
- **Field**
  This is set for customer data and for ticket data. For customer data, the data object group field name is shown (here, only the data object group fields from the selected data object are offered), for ticket data, the Custom Field name is shown.
- **Value**
  The value of this field in the selected ticket. You do not have to use this in the Word Template.

Open the MS Word template from the list in the Template Library or click on *Word template* , *Browse ...* to open/download the Word document. In this document, go to the position where you want to insert the first field (in our example this will be the customer name). Use *Insert* -> *Quick parts* -> *Field* to insert the *MergeField*. Copy and paste the key of the Merge Field you need into the respective field (*Field properties*, *Field name*) in MS Word:


Invoice

Recipient:

Date of invoice

Service period

Services:

performed by:

Please pay the

Thank you.

Sincerely,

ConSol* Software

Field

Please choose a field

Categories: (All)

Field names: ListNum, MacroButton, MergeField, MergeRec, MergeSeq, Next, NextIf, NoteRef, NumChars, NumPages, NumWords, Page, PageRef, Print, PrintDate, Private, Quote, RD

Field properties

Field name: queue_fields_contacts_member_firstname

Format: (none), Uppercase, Lowercase, First capital, Title case

Field options

Text to be inserted before:

Text to be inserted after:

Mapped field

Vertical formatting

Preserve formatting during updates [✓]

Description:

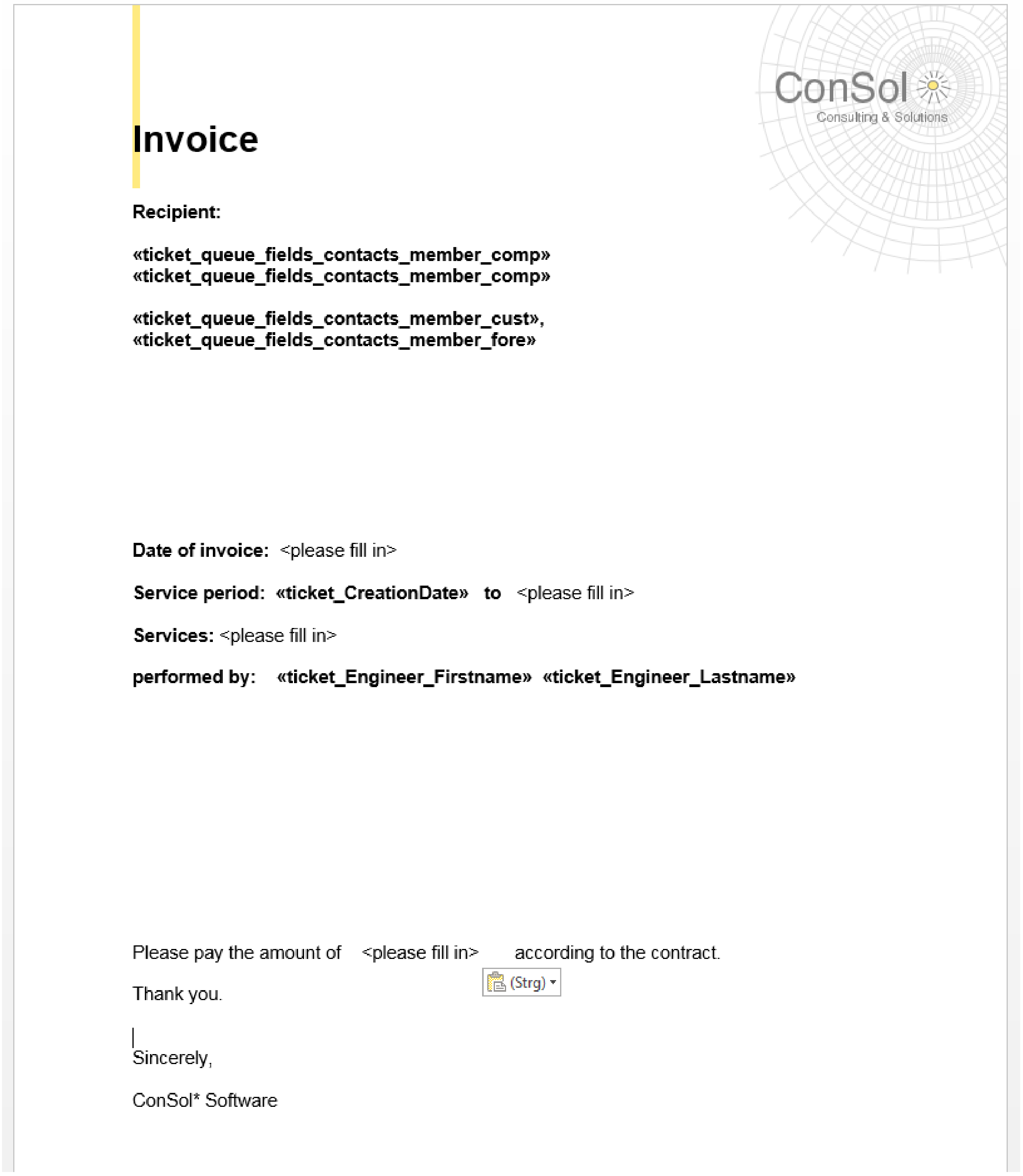
Insert a mail merge field

Fig. 10: MS Word - Insert MergeField into Document

Do this with all fields you would like to have pre-filled when the MS Word template is opened. In the end, your template might look like the example in the following figure.

You can use all fields which are offered in the ticket, i.e.

- Company fields (Data Object Group 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 data object group name as Group)
- Contact fields (Data Object Group 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 data object group name as Group)
- Ticket data fields (Custom Fields from the ticket, the respective keys start with *ticket_* and have a CF 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)
- Engineer data (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.)



Invoice

ConSol Consulting & Solutions

Recipient:

«ticket_queue_fields_contacts_member_comp»
«ticket_queue_fields_contacts_member_comp»

«ticket_queue_fields_contacts_member_cust»,
«ticket_queue_fields_contacts_member_fore»

Date of invoice: <please fill in>

Service period: «ticket_CreationDate» to <please fill in>

Services: <please fill in>

performed by: «ticket_Engineer_Firstname» «ticket_Engineer_Lastname»

Please pay the amount of <please fill in> according to the contract.

Thank you.

Sincerely,

ConSol* Software

Fig. 11: MS Word Example Template

Save the Word document in the local file system and then upload it using the field *Update Word template*. This will store the template in the Doc Template Manager.

Details about ConSol*CM Keys for the Mail Merge 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 data object group 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]_[data object group]_[data object group field], e.g., ticket_queue_fields_contacts_member_companyReferenceField_ResellerCompany_ResellerCompany **_company_name**

26.3.7 Using MS Word Templates from within the Web Client

Creating a New MS Word Attachment Using a Word Template

When MS Word templates are available for a queue, an engineer can use them by clicking on *Attachments* in the *History Section* of a ticket and by selecting the requested template. MS Word is started (if it is not already open) 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.

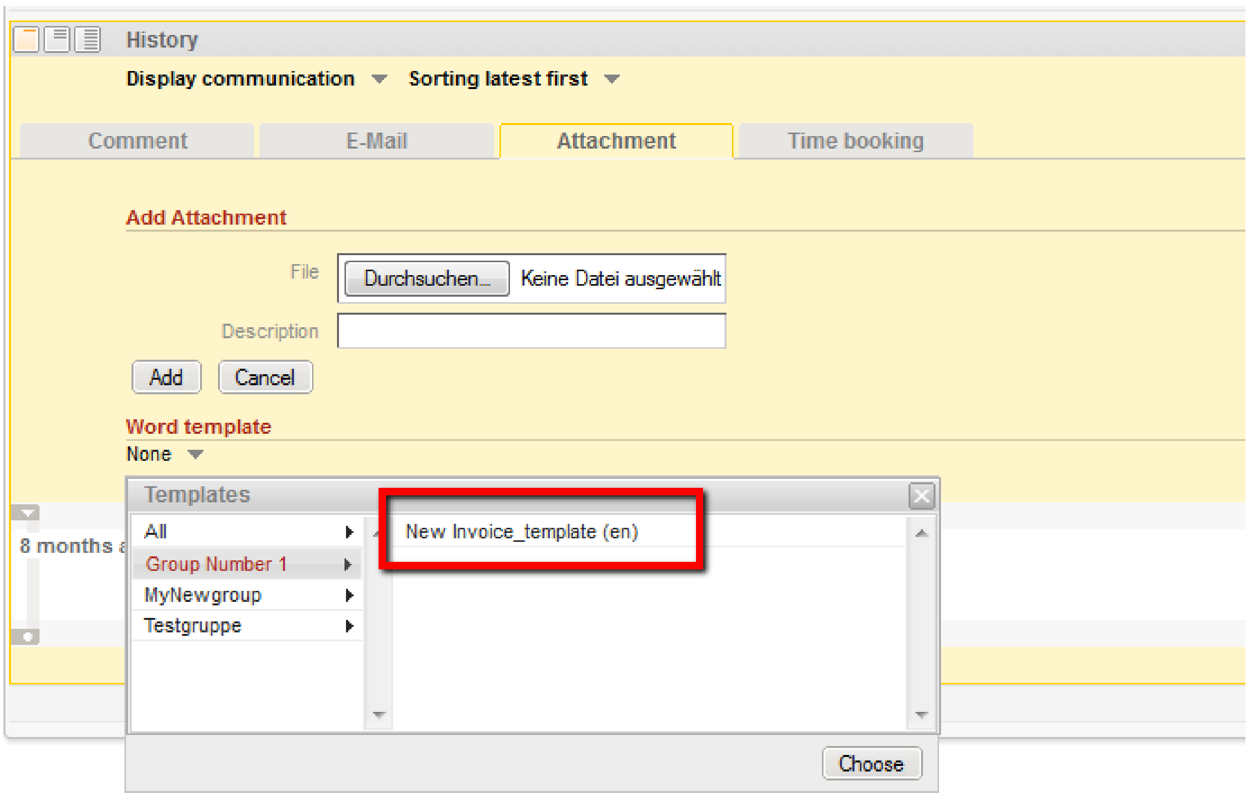
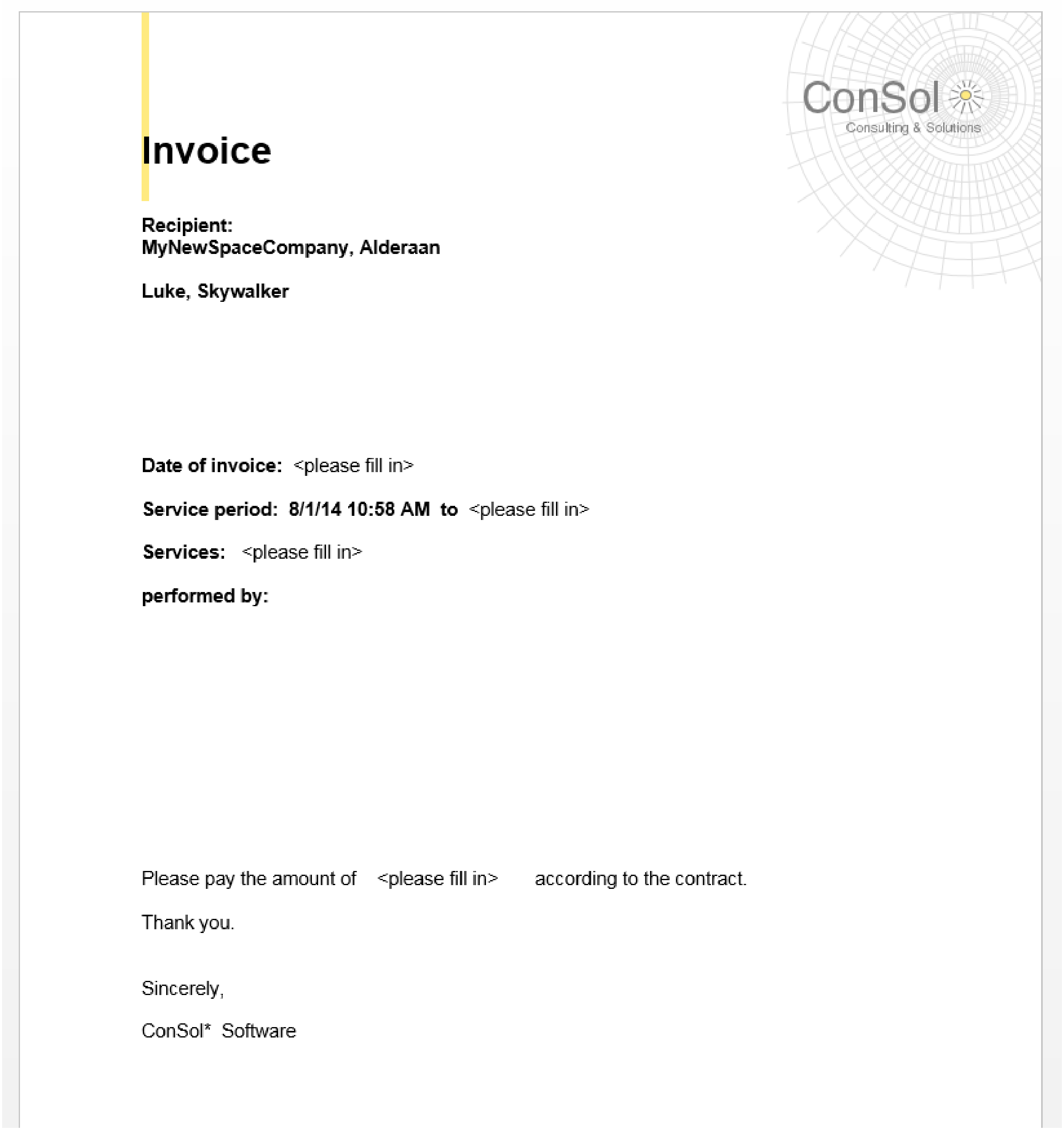


Fig. 12: ConSol*CM Web Client - MS Word Templates Available as Attachments



Invoice

Recipient:
MyNewSpaceCompany, Alderaan

Luke, Skywalker

Date of invoice: <please fill in>

Service period: 8/1/14 10:58 AM to <please fill in>

Services: <please fill in>

performed by:

Please pay the amount of <please fill in> according to the contract.

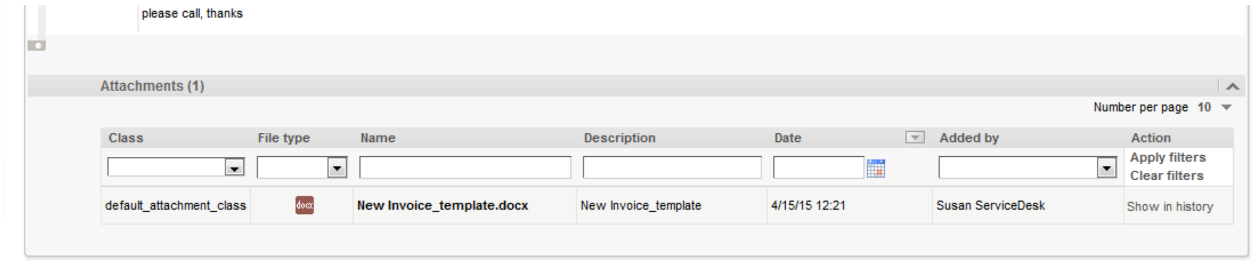
Thank you.

Sincerely,

ConSol* Software

Fig. 13: MS Word Example Document

The engineer can then edit the document if requested and save it. This will attach the new version of the document automatically to the ticket.



please call, thanks

Attachments (1)

Number per page 10

Class	File type	Name	Description	Date	Added by	Action
						Apply filters Clear filters
default_attachment_class	docx	New Invoice_template.docx	New Invoice_template	4/15/15 12:21	Susan ServiceDesk	Show in history

Fig. 14: Newly edited Word document as attachment at the ticket

From here on, the work with the Word document is exactly the same as the work with regular Word (.doc, .docx) attachments. See next section.

Working with Existing MS Word Attachments

An engineer can also open an MS Word document which has been attached to the ticket. As an engineer, click on the attachment name. This will open the file in MS Word. Edit the document and save it. A new version of the document will be attached to the ticket automatically.

27 Time Booking Using ConSol*CM

- General Introduction to Time Booking Using ConSol*CM
- Manual Time Bookings
    - Configuration of Manual Time Booking Using the Admin Tool
    - Manual Time Booking from a User's Point of View (Web Client)
    - Manual Time Bookings on the Engineer Profile Page
- Automatic Time Bookings (Available in CM Version 6.9.4.2 and Up)
    - Introduction to Automatic Time Booking
    - Configuration of Automatic Time Booking
    - Automatic Time Bookings on the Engineer Profile Page
- DWH Reports
- Page Customization for Time Booking

27.1 General Introduction to Time Booking Using ConSol*CM

In ConSol*CM, working hours can be booked on tickets. There are two booking modes:

1. Manual bookings, see section Manual Time Bookings
2. Automatic bookings (available in CM versions 6.9.4.2 and up), see section Automatic Time Bookings

27.2 Manual Time Bookings

In ConSol*CM, an engineer can book working hours on a ticket. Those working hours can then be reported.

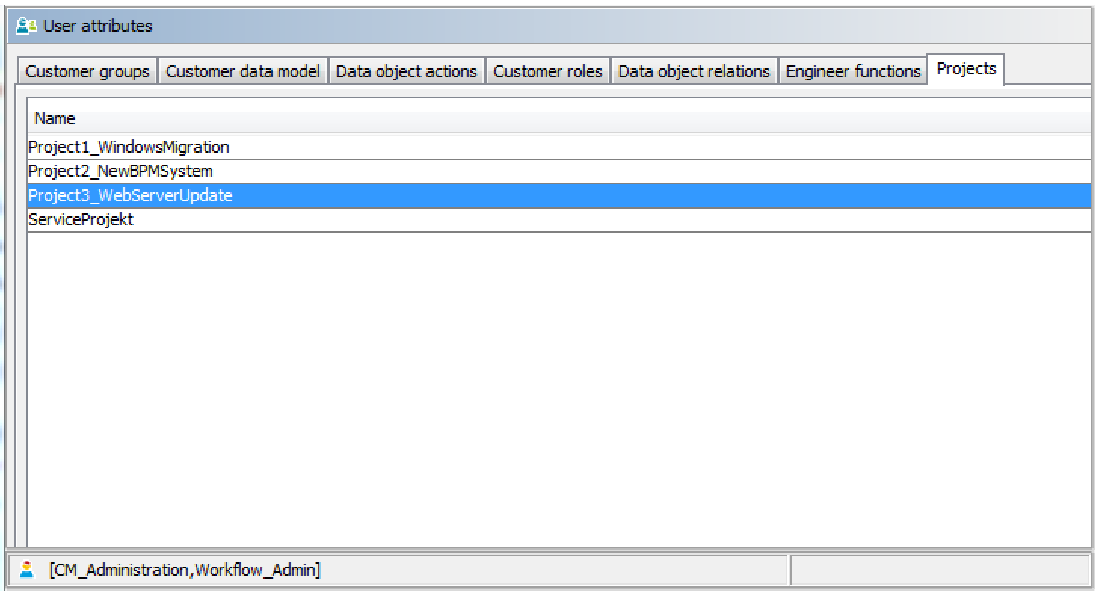
Working hours are either booked on projects which have to be assigned to one or more queues or they are booked directly on tickets. For example, if your company plans to perform a migration from Windows 7 to Windows 8 clients and all the working hours should be registered for this migration project, the ConSol*CM administrator has to create a migration project and assign it to all queues where tasks for this project might be completed. Then engineers can book times on the project and can see their own reports for the project. Additionally, a report over all time bookings, of all engineers, might be implemented using the DWH (data warehouse, see section Data Warehouse (DWH) Management).

27.2.1 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 tab *Projects*, see section Additional User Attributes.
2. Assign one or more projects to the desired queues within the Queue Administration.

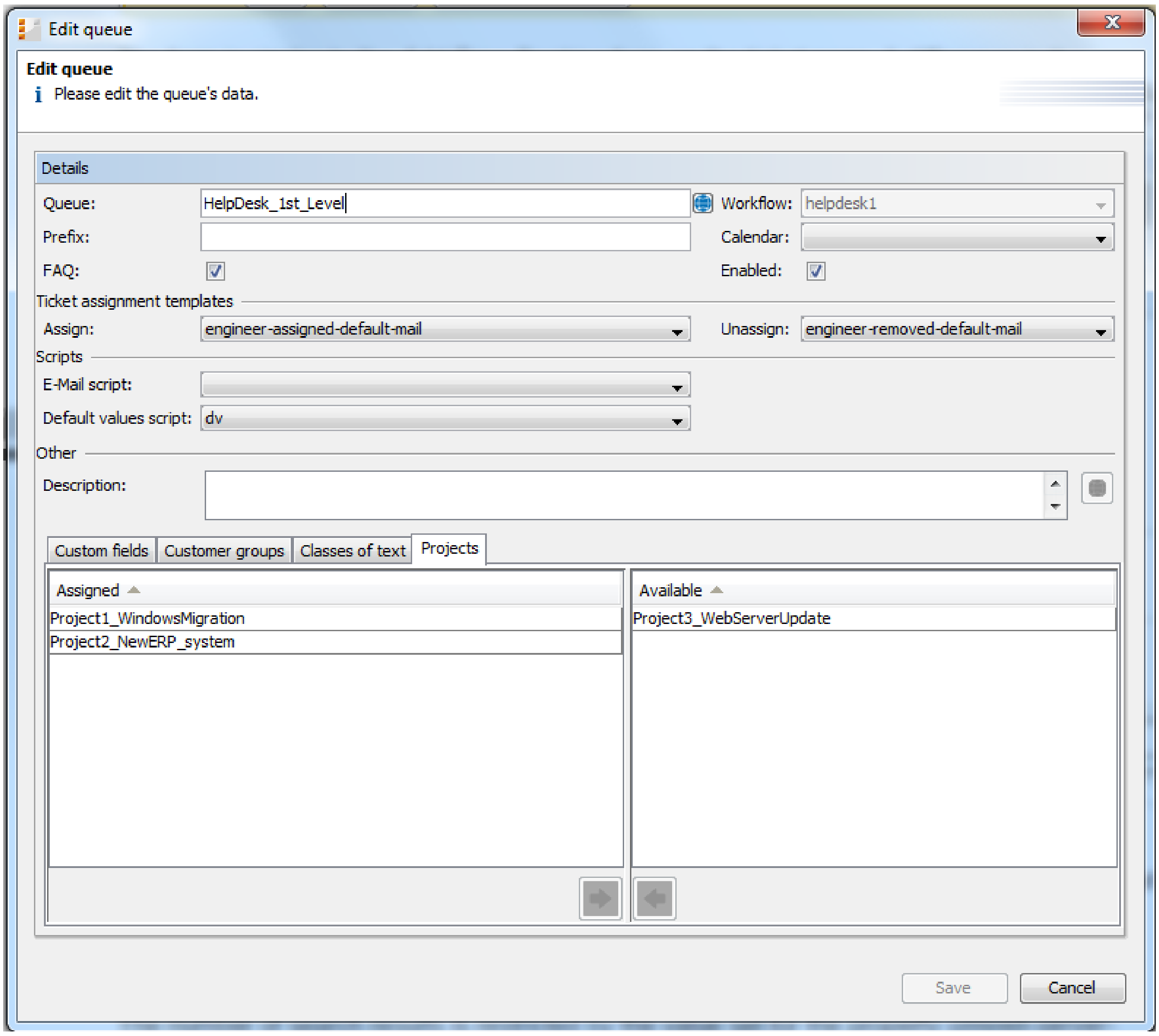
In the following example, three projects are created. Engineers in the *HelpDesk_1st_Level* queue should be able to book working hours on two of them. Thus, the two projects have to be assigned to the *HelpDesk_1st_Level* queue.



Name
Project1_WindowsMigration
Project2_NewBPMSystem
Project3_WebServerUpdate
ServiceProjekt

[CM_Administration,Workflow_Admin]

Fig. 1: ConSol*CM Admin Tool - User Attributes: Management of Projects


Edit queue

Edit queue

Please edit the queue's data.

Details

Queue:	HelpDesk_1st_Level	Workflow:	helpdesk1
Prefix:		Calendar:	
FAQ:	✓	Enabled:	✓

Ticket assignment templates

Assign:	engineer-assigned-default-mail	Unassign:	engineer-removed-default-mail

Scripts

E-Mail script:	
Default values script:	dv

Other

Description:

Custom fields | Customer groups | Classes of text | Projects

Assigned	Available
Project1_WindowsMigration	Project3_WebServerUpdate
Project2_NewERP_system	

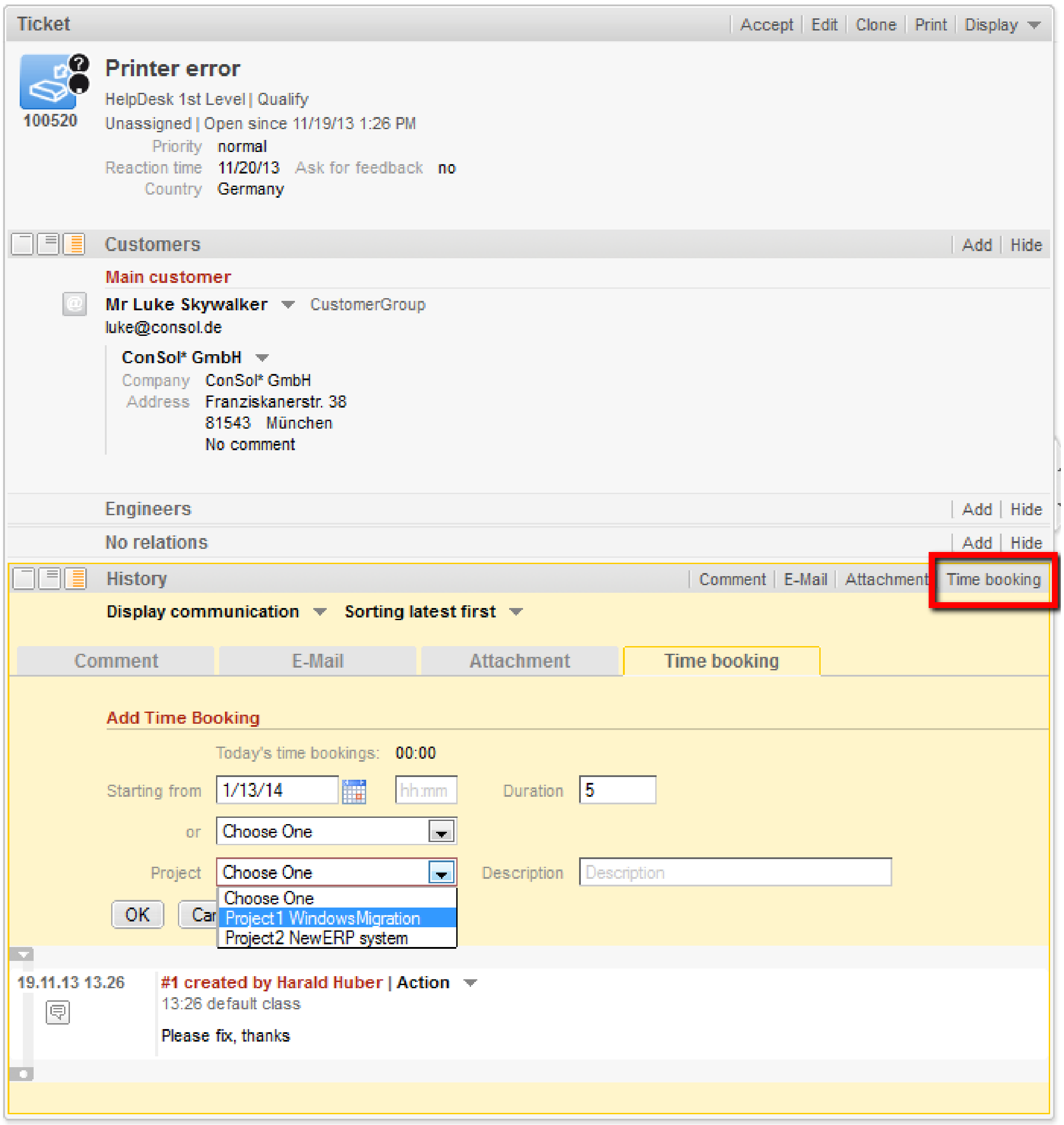
Fig. 2: ConSol*CM Admin Tool - Queue Administration: Assigning Projects to a Queue

27.2.2 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 short overview is provided.

The user (engineer) can book working hours on a ticket using two different modes:

1. Using the *Time booking* section in a **ticket** to book working hours directly to this ticket.


Ticket | Accept | Edit | Clone | Print | Display

100520

Printer error

HelpDesk 1st Level | Qualify

Unassigned | Open since 11/19/13 1:26 PM

Priority	normal		
Reaction time	11/20/13	Ask for feedback	no
Country	Germany		

Customers | Add | Hide

Main customer

Mr Luke Skywalker CustomerGroup

luke@consol.de

ConSol* GmbH

Company	ConSol* GmbH
Address	Franziskanerstr. 38
	81543 München
	No comment

Engineers | Add | Hide

No relations | Add | Hide

History | Comment | E-Mail | Attachment | Time booking

Display communication | Sorting latest first

Comment | E-Mail | Attachment | Time booking

Add Time Booking

Today's time bookings: 00:00

Starting from	1/13/14	hh:mm	Duration	5
or	Choose One			
Project	Choose One		Description	Description

Choose One
Project1 WindowsMigration
Project2 NewERP system

OK | Can

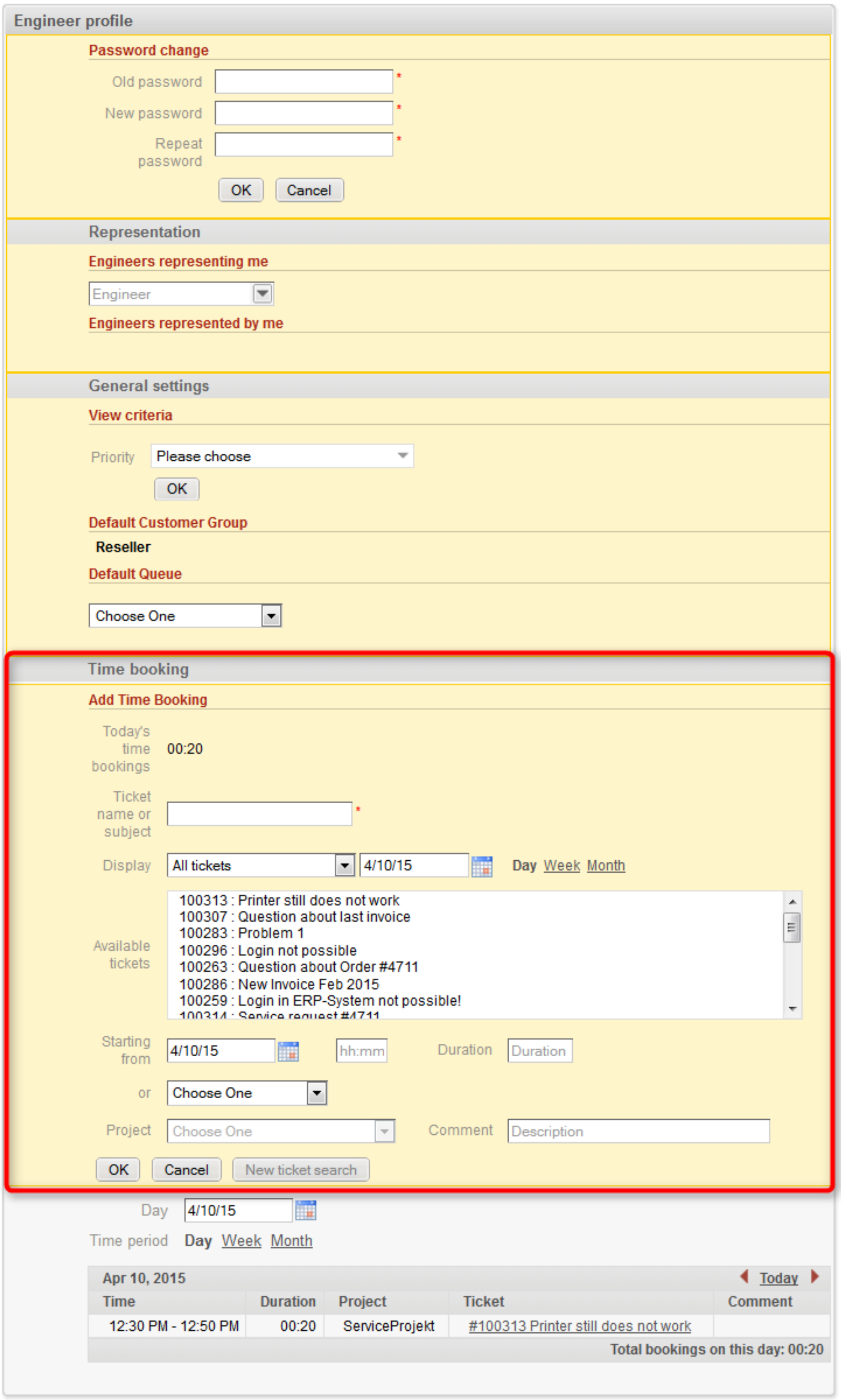
19.11.13 13.26 #1 created by Harald Huber | Action

13:26 default class

Please fix, thanks

Fig. 3: ConSol*CM Web Client - Time Booking in a Ticket

2. Using the *Time booking* section in the **engineer profile page** to book working hours on a selected ticket (only tickets where the engineer has performed certain activities and tickets owned by the engineer can be selected).

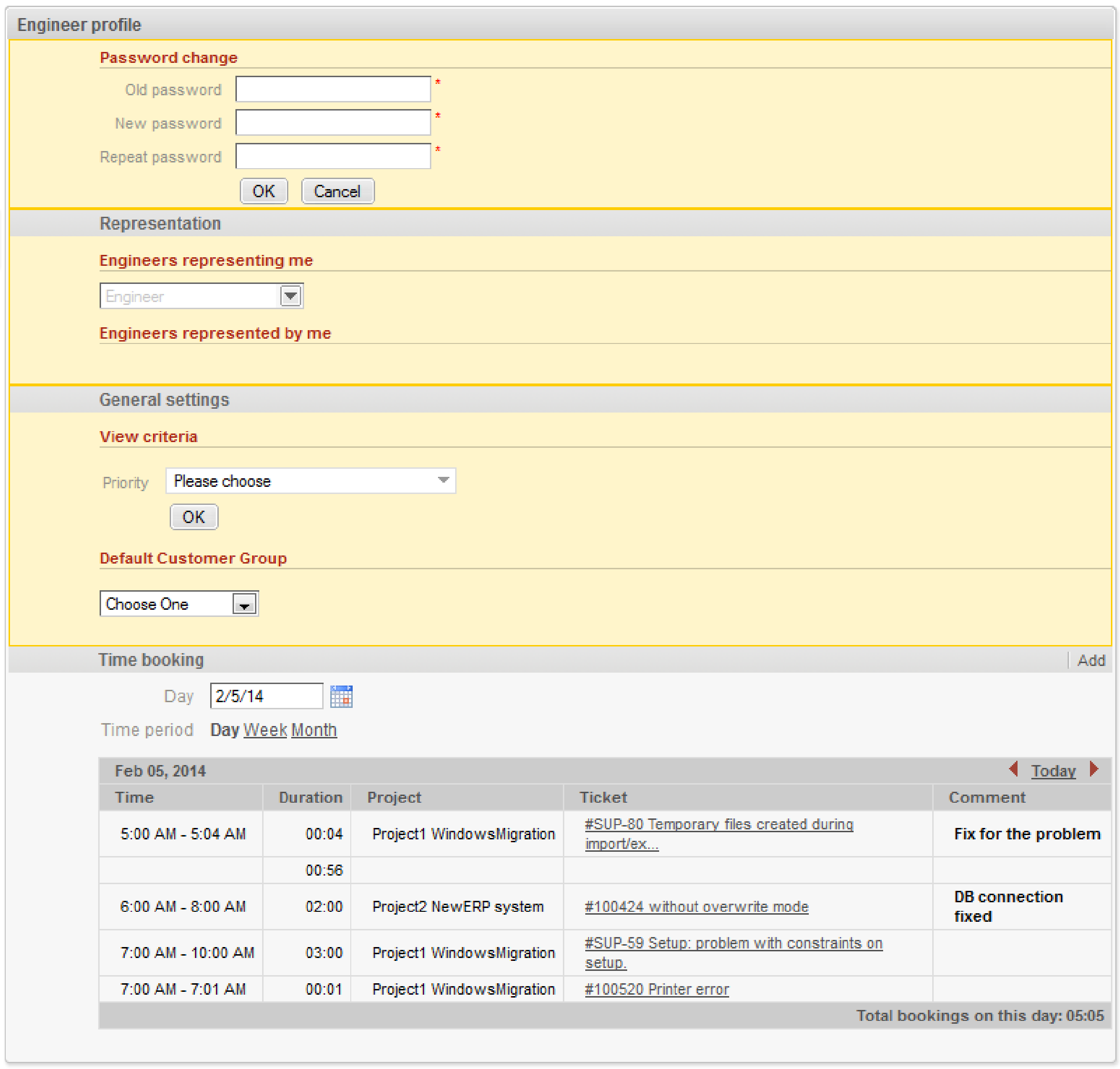


Time	Duration	Project	Ticket	Comment
12:30 PM - 12:50 PM	00:20	ServiceProjekt	#100313 Printer still does not work	
			Total bookings on this day: 00:20	

Fig. 4: ConSol*CM Web Client - Time Booking in the Engineer Profile Page

27.2.3 Manual Time Bookings on the Engineer Profile Page

Engineers can see a list of their time bookings on the engineer profile page. An example is shown in the following figure.



Engineer profile

Password change

Old password *
New password *
Repeat password *

OK Cancel

Representation

Engineers representing me

Engineer

Engineers represented by me

General settings

View criteria

Priority: Please choose

OK

Default Customer Group

Choose One

Time booking — Add

Day: 2/5/14

Time period: Day Week Month

Feb 05, 2014 — Today

Time	Duration	Project	Ticket	Comment
5:00 AM - 5:04 AM	00:04	Project1 WindowsMigration	#SUP-80 Temporary files created during import/ex...	Fix for the problem
	00:56			
6:00 AM - 8:00 AM	02:00	Project2 NewERP system	#100424 without overwrite mode	DB connection fixed
7:00 AM - 10:00 AM	03:00	Project1 WindowsMigration	#SUP-59 Setup: problem with constraints on setup.	
7:00 AM - 7:01 AM	00:01	Project1 WindowsMigration	#100520 Printer error	

Total bookings on this day: 05:05

Fig. 5: ConSol*CM Web Client - Time Booking Report in the Engineer Profile

As an engineer, you can select if 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.

27.3 Automatic Time Bookings (Available in CM Version 6.9.4.2 and Up)

27.3.1 Introduction to Automatic Time Booking

If required, 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 pressed.
- **Duration of creating a ticket**
  Started when *Create ticket* is selected and stopped when the *Create* button is pressed.

Time booking is suspended when a ticket is transferred to the workspace and continued when the ticket is brought back to the active work.

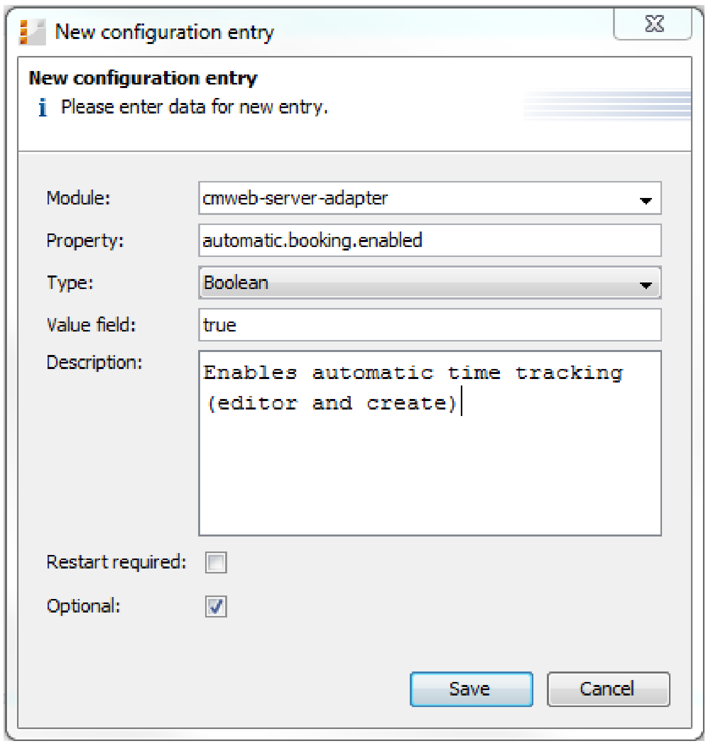
No times are booked on the ticket in case ...

- an operation is canceled
- the engineer logs out manually
- the engineer session is ended automatically

Times are always booked in minutes and always rounded up to the next minute.

27.3.2 Configuration of Automatic Time Booking

In order to enable this functionality in your CM system, set the system property *cmweb-server-adapter, automatic.booking.enabled* to *true*. In CM versions 6.9.4.1 and lower, the property does not exist and has to be added manually.


New configuration entry

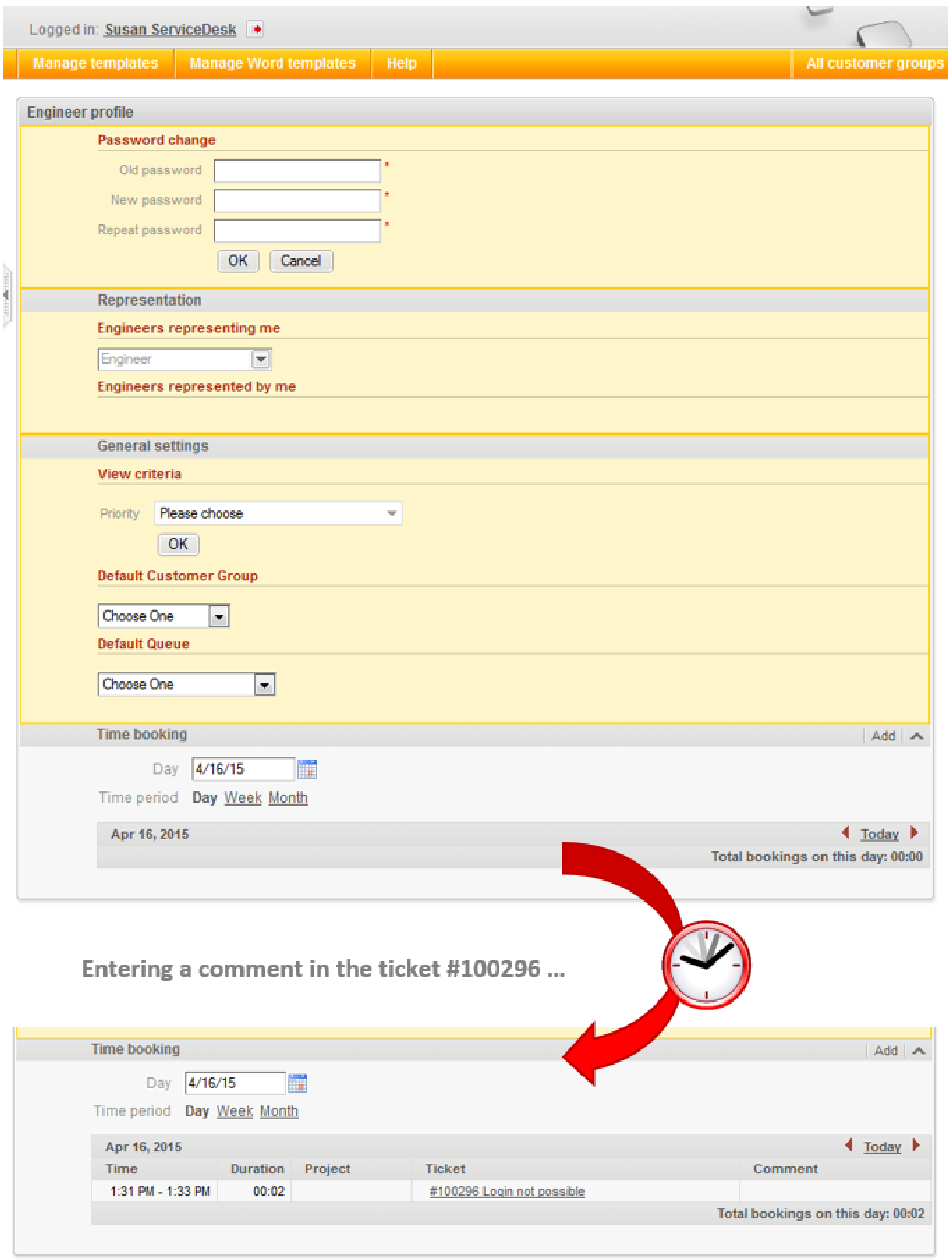
New configuration entry
Please enter data for new entry.

Module:	cmweb-server-adapter
Property:	automatic.booking.enabled
Type:	Boolean
Value field:	true
Description:	Enables automatic time tracking (editor and create)
Restart required:	☐
Optional:	☑

Fig. 6: Set System Property to Switch on Automatic Time Booking

27.3.3 Automatic Time Bookings on the Engineer Profile Page

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.



Time	Duration	Project	Ticket	Comment
1:31 PM - 1:33 PM	00:02		#100296 Login not possible	

Total bookings on this day: 00:02

Fig. 7: Automatically Booked Time in Time Booking Report on Engineer Profile Page

27.4 DWH Reports

If your company would like to have reports on 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.

27.5 Page Customization for Time Booking

In case the time booking feature is not required, you as an administrator 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

28 CM6 Administrator Manual 6.9.4 - Authentication Methods for Engineers in the CM Web Client

28.1 Authentication Methods for Engineers in the Web Client

For the authentication of engineers in the Web Client, you can configure one of the following three methods:

1. **Standard**
   All user (engineer) data is kept in the ConSol*CM database. User name and password are set on the Engineer Administration page, see section Engineer Administration.
2. **LDAP**
   The credentials are kept in an *LDAP* server. ConSol*CM sends a request to this server for the engineer authentication, see section LDAP Authentication.
3. **Kerberos SSO**
   The credentials are taken from a valid session using *Kerberos*, see section Single Sign-On with ConSol*CM Using Kerberos.

28.2 Introduction to ConSol*CM LDAP Authentication

- Configuring the System to Enable LDAP Authentication
  - Configuring the System During Initial Set-Up
  - Switching the Authentication Mode to LDAP in a Running System
- Managing Engineer Accounts for LDAP Authentication
- Using LDAPS (LDAP over SSL)
  - Introduction
  - Preparations
  - LDAPS Configuration in the ConSol*CM Admin Tool (System Properties)

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 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 CM server sends a request with the engineer's user name and password and checks with the LDAP servers if those credentials are correct.

When the credentials are correct, the approval is sent back to the CM server and the engineer is logged into the Web Client.

Important information:

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. Every user who should work with the system as an engineer has to be created as an engineer account in the engineer administration!

Please see also the following picture for an explanation of the process:

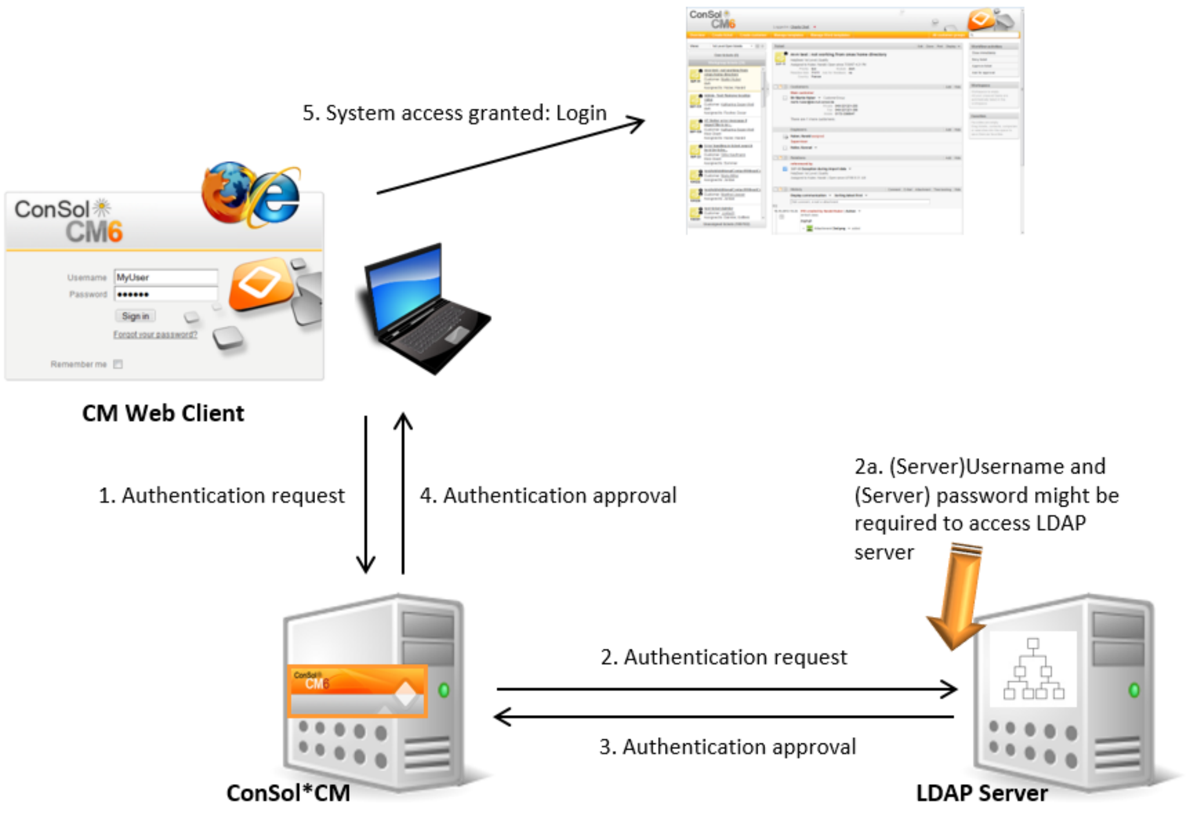

5. System access granted: Login
ConSol CM6
Username MyUser
Password ••••••
Sign in
Forgot your password?
Remember me
CM Web Client
1. Authentication request
4. Authentication approval
2a. (Server)Username and (Server) password might be required to access LDAP server
2. Authentication request
3. Authentication approval
ConSol*CM
LDAP Server

Fig. 1: ConSol*CM LDAP Authentication Process

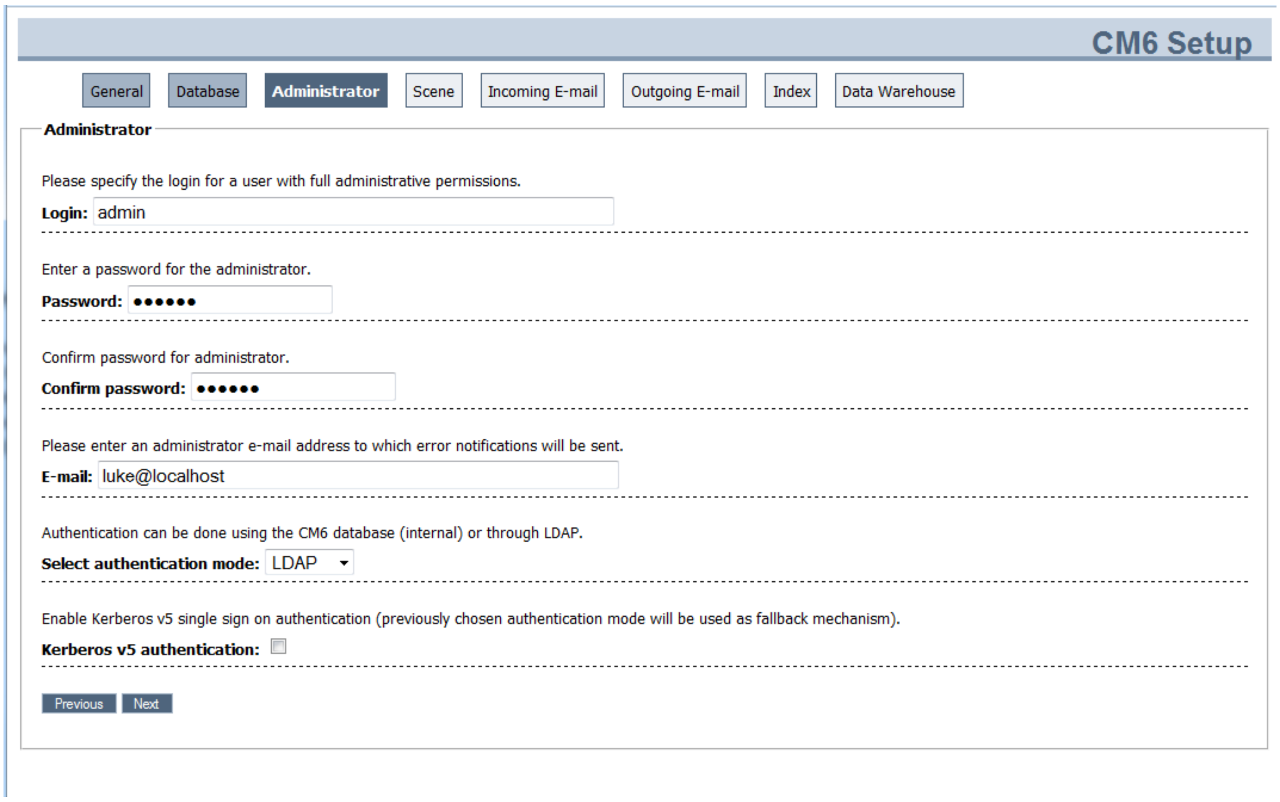
28.2.1 Configuring the System to Enable LDAP Authentication

There are two ways you can enable the ConSol*CM system to use LDAP authentication:

1. Select *LDAP authentication* during system set-up and enter the requested parameters (system properties) after the set-up
2. Set up the system with the regular authentication mechanism and switch to LDAP later on, i.e. enter the remaining requested parameters (system properties) later on

Configuring the System During Initial Set-Up

During system set-up you can select LDAP as authentication mode. 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 subsequent section for an explanation.



Administrator

Please specify the login for a user with full administrative permissions.

Login: admin

Enter a password for the administrator.

Password: ●●●●●●

Confirm password for administrator.

Confirm password: ●●●●●●

Please enter an administrator e-mail address to which error notifications will be sent.

E-mail: luke@localhost

Authentication can be done using the CM6 database (internal) or through LDAP.

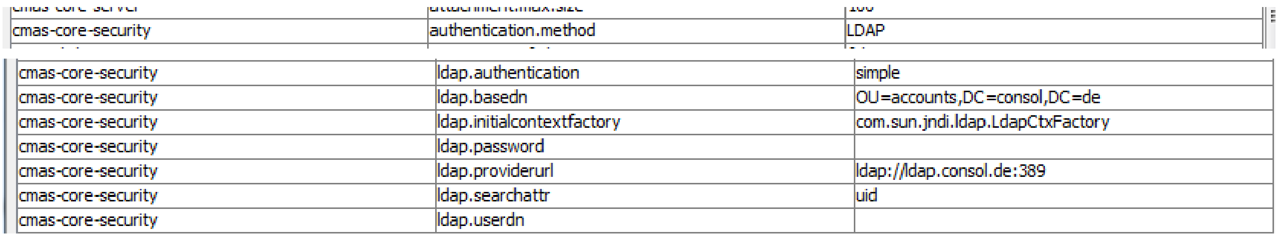
Select authentication mode: LDAP

Enable Kerberos v5 single sign on authentication (previously chosen authentication mode will be used as fallback mechanism).

Kerberos v5 authentication: ☐

Fig. 2: ConSol*CM System Set-Up - Authentication Mode LDAP

Switching the Authentication Mode to LDAP in a Running System



cmas-core-security	authentication.method	LDAP
cmas-core-security	ldap.authentication	simple
cmas-core-security	ldap.basedn	OU=accounts,DC=consol,DC=de
cmas-core-security	ldap.initialcontextfactory	com.sun.jndi.ldap.LdapCtxFactory
cmas-core-security	ldap.password	
cmas-core-security	ldap.providerurl	ldap://ldap.consol.de:389
cmas-core-security	ldap.searchattr	uid
cmas-core-security	ldap.userdn	

Fig. 3: ConSol*CM Admin Tool - Properties for LDAP Authentication

Required values for LDAP authentication (they are set via *system properties*, please see Appendix C for an explanation):

- **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. Usually should 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 LDAP server to look up users. Only needed if look-up cannot be done anonymously.

  Attention:

  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 for a first check.

- **ldap.providerurl**
  The complete URL for the LDAP server:

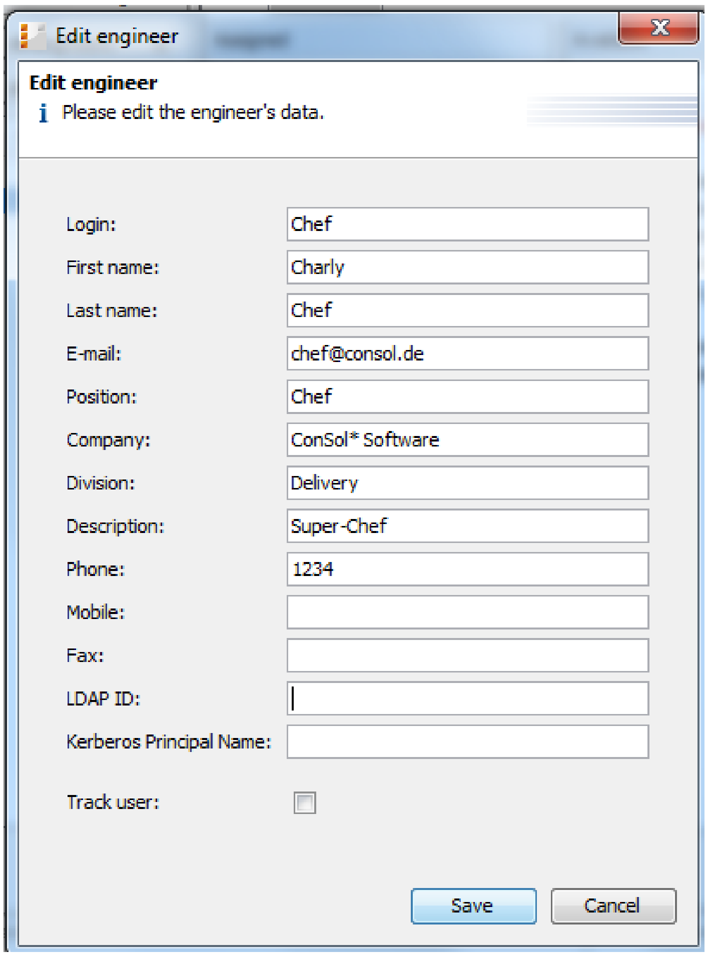
  ```
  ldap://<HOSTNAME>:<LDAP PORT>
  ```

- **ldap.searchattr**
  Search attribute for looking up LDAP entry connected to CM login, i.e. the attribute which is used as user name for the authentication.

28.2.2 Managing Engineer Accounts for LDAP Authentication

Use the *Engineer Administration* 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. Please refer to section Engineer Administration for details concerning the other (non LDAP-related) data fields.



Edit engineer

Please edit the engineer's data.

Login:	Chef
First name:	Charly
Last name:	Chef
E-mail:	chef@consol.de
Position:	Chef
Company:	ConSol* Software
Division:	Delivery
Description:	Super-Chef
Phone:	1234
Mobile:	
Fax:	
LDAP ID:	
Kerberos Principal Name:	
Track user:	☐

Fig. 4: ConSol*CM Admin Tool - Engineer Administration

- **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 you can fill in this field. During the LDAP authentication process, this LDAP ID is used as the user name which is looked up in the LDAP directory in the *ldap.searchattr* node.

28.2.3 Using LDAPS (LDAP over SSL)

Introduction

Per default , when an LDAP client accesses an LDAP server, the information is transferred by default 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 is can use certificates. One way to do this is described in the following section for a Linux machine.

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 <beliebig> -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 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 CM server as shown in the following example:

- cmas-core-security;ldap.authentication = simple
- cmas-core-security;ldap.basedn;DC = myDC,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

28.3 Single Sign-On with ConSol*CM Using Kerberos

- Single Sign-On with ConSol*CM Using Kerberos
  - Configuration of Kerberos Single Sign-On
    - Introduction
    - Requirements
  - Setting Up the System
    - Domain Controller
      - Registering the ConSol*CM Server Machine
      - Registering the ConSol*CM Server User
      - Generating the keytab File
    - ConSol*CM Server
      - Enabling Kerberos in ConSol*CM
      - Configuring Kerberos
      - File keytab
    - Client Machine
      - Internet Explorer
      - Firefox
  - Using the System
    - Single Sign-On from the User's Point of View
    - Multi Domains Single Sign-On
    - Mapping Kerberos User Name to Engineer Name
    - Starting and Stopping Kerberos Authentication

28.3.1 Configuration of Kerberos Single Sign-On

Introduction

The *single sign-on* feature relating to ConSol*CM allows users to authenticate against ConSol*CM automatically with their *Windows* credentials.

This authentication mechanism ...

- works completely transparent, no user interaction (i.e. filling in login screen) is required,
- does not interfere with existing authentication mechanisms. If Kerberos authentication fails, whatever authentication mechanism was configured (e.g. LDAP or database authentication) is used.

The single sign-on feature is based on the *Kerberos V5* protocol, which is integrated in the *Windows Active Directory*. All information is encrypted using *RC4* and *HMAC*.

The web server works as a *non-Windows Kerberos service* and can be installed on any operating system /application server.

Client and web server use GSSAPI and SPNEGO to exchange authentication information.

This guide shows you how to set up single sign-on in a Windows (Active Directory) environment as this is the most common scenario for our customers.

Requirements

For Kerberos-based single sign-on you need:

- Domain controller for the Windows domain
- ConSol*CM server
- Windows clients

28.3.2 Setting Up the System

Domain Controller

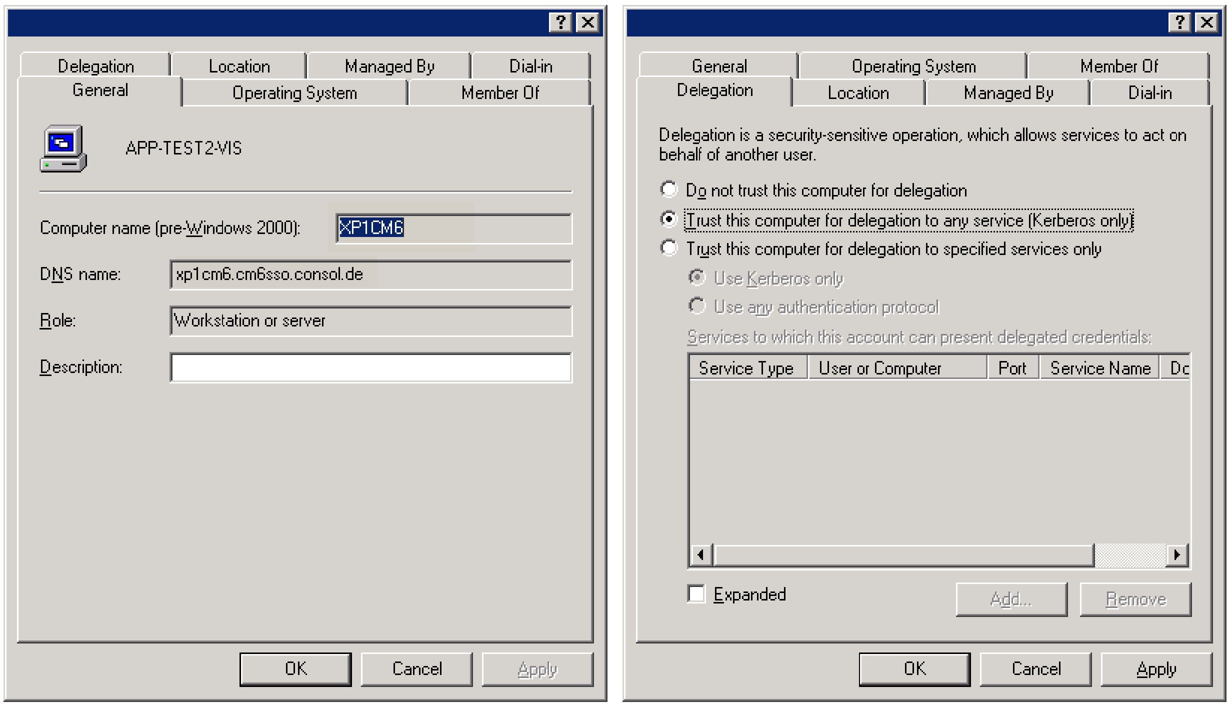
The first step is to configure the domain controller so that it knows the ConSol*CM server. In our example the domain controller is called *win2003srv*, the domain is *CM6SSO.CONSOL.DE*.

Registering the ConSol*CM Server Machine

First, the ConSol*CM server machine needs to be registered in the Active Directory of the domain controller. In our example it is the computer *xp1cm6*.

Attention:

The radio button *Trust this computer for delegation to any service (Kerberos only)* must be activated!



Delegation | Location | Managed By | Dial-in
General | Operating System | Member Of

APP-TEST2-VIS

Computer name (pre-Windows 2000): XP1CM6

DNS name: xp1cm6.cm6sso.consol.de

Role: Workstation or server

Description:

OK | Cancel | Apply

General | Operating System | Member Of
Delegation | Location | Managed By | Dial-in

Delegation is a security-sensitive operation, which allows services to act on behalf of another user.

- Do not trust this computer for delegation
- [✓ Trust this computer for delegation to any service (Kerberos only)]
- Trust this computer for delegation to specified services only
  - [✓ Use Kerberos only]
  - Use any authentication protocol

Services to which this account can present delegated credentials:

Service Type	User or Computer	Port	Service Name	Dc

[ ] Expanded

Add... | Remove

OK | Cancel | Apply

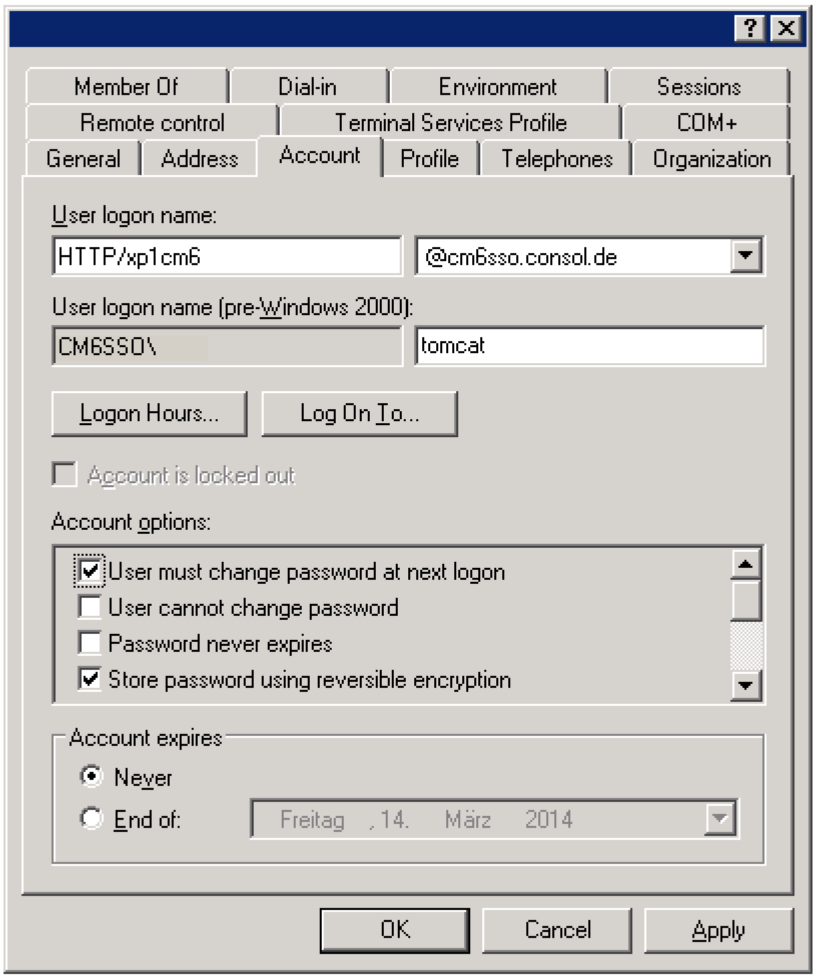
Fig. 1: Registration of ConSol*CM Server Machine

Registering the ConSol*CM Server User

Second, the user under which the ConSol*CM server process will run, is created and registered in the Active Directory, in our example the user *tomcat*.

The following account options must be enabled:

- account is trusted for delegation
- no Kerberos pre-authentication needed



User logon name: HTTP/xp1cm6 @cm6sso.consol.de

User logon name (pre-Windows 2000): CM6SSO\ tomcat

[ ] Account is locked out

Account options:

[✓] User must change password at next logon
[ ] User cannot change password
[ ] Password never expires
[✓] Store password using reversible encryption

Account expires
(●) Never
( ) End of: Freitag , 14. März 2014

Fig. 2: Registration of ConSol*CM Server User

Generating the keytab File

On the domain controller the ConSol*CM server is created as a new Kerberos service, additionally a Kerberos *keytab* file is generated. This file will be needed later on the ConSol*CM server machine. This *keytab* file contains the shared secret key of the service.

```
C:\Programme\Support Tools>ktpass /out tomcat.keytab /ptype KRB5_NT_PRINCIPAL /princ HTTP
/xp1cm6.cm6sso.consol.de@CM6SSO.CONSOL.DE /pass consol.123 /mapuser tomcat /crypto rc4-hmac-nt
```

Attention:

If *ktpass* is not available, the *Windows Server 2003 Support Tools* must be installed, available here.

ConSol*CM Server

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 set-up, 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 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 class path:

- Under JBoss this can be:
  `../jboss/server/{domain}/conf/cm6-kerberos.properties`
- Under WebLogic this can be:
  `../{domain}/cm6-kerberos.properties`

In case you have a cluster of more than one ConSol*CM servers in operation, each server has to have a separate properties file.

Properties in this file should contain:

- Reference to a Kerberos config file (e.g. *krb5.ini* or *krb5.conf*)
- One or more service principal(s), i.e. reference to *keytab* file

Example for cm6-kerberos.properties:

```
# path to kerberos configuration
kerberos.config.location=C:\\conf\\krb5.ini
# one or more service principals (principal = path to keytab file)
HTTP/xp1cm6.cm6sso.consol.de@CM6SSO.CONSOL.DE=C:\\conf\\tomcat.keytab
```

Example for krb5.ini:

```
[libdefaults]
   default_realm = CM6SSO.CONSOL.DE
   default_tkt_enctypes = rc4-hmac des-cbc-md5 des-cbc-crc des3-cbc-sha1
   default_tgs_enctypes = rc4-hmac des-cbc-md5 des-cbc-crc des3-cbc-sha1
[realms]
   CM6SSO.CONSOL.DE = {
      kdc = w2003srvcm6
      admin_server = w2003srvcm6:88
   }
[domain_realm]
   .w2003srvcm6 = CM6SSO.CONSOL.DE
   w2003srvcm6 = CM6SSO.CONSOL.DE
```

File keytab

Copy the *keytab* file you generated on the domain controller to the location you specified in the *cm6-kerberos.properties* config file.

Attention:

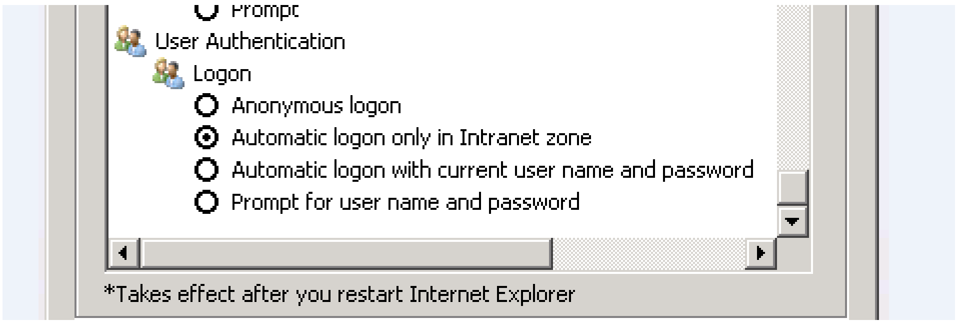
You have to restart the ConSol*CM server process!

Client Machine

Internet Explorer

The *Internet Explorer* needs to be configured so that automatic login is enabled. By default, this is allowed in the *medium-low* security setting, which by default is set for the *local intranet zone*.

In detail, the following settings for login behavior are available.



Prompt
User Authentication
Logon
Anonymous logon
[✓ Automatic logon only in Intranet zone]
Automatic logon with current user name and password
Prompt for user name and password

*Takes effect after you restart Internet Explorer

Fig. 3: Internet Explorer Login Configuration

Settings and resulting behavior:

- **Anonymous logon**
  No single sign-on is possible, user will get 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 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

In 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://xp1cm6* if that is the URI).

You can set this property also on the file system. Open the file

C:\Dokumente und Einstellungen\[USER]\Anwendungsdaten\Mozilla\Firefox\Profiles\XYZ.default\prefs.js

and add/replace the following line:

```
user_pref("network.automatic-ntlm-auth.trusted-uris", "http://xp1cm6");
```

Attention:

You have to restart Firefox after this change.

28.3.3 Using the System

Single Sign-On from the User's Point of View

An engineer using single sign-on to log into ConSol*CM will notice that ...

- 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 ConSol*CM Web Client main screen. Here, a message is displayed:

You have been automatically logged in and a new session has been created for you.

It is still possible to login as another ConSol*CM user by clicking on 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 you will enable single sign-on, 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/cm6.consol.de@CONSOL.DE=/etc/krb5_consolde.keytab
HTTP/cm6.consol.de@CONSOL.PL=/etc/krb5_consolpl.keytab
```

Mapping Kerberos User Name to Engineer Name

Using Kerberos-based single sign-on, the Kerberos principal (i.e. the user's OS login) has to be mapped to a ConSol*CM engineer name.

By default, this mapping is done using one of the two following 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 *cmas-core-security*, *kerberos.v5.username.regexp* property is taken and applied to the principal. The result of this will be taken and a ConSol*CM engineer with this login will be searched:
  - First matching regular expression group (in brackets) will be used as engineer login name, e.g. the default property value *(.*)@.** will convert *Huber@cm6sso.consol.de* to *Huber*.

If further customization is needed please refer to *UsernameAdapter interface javadoc*.

Starting and Stopping Kerberos Authentication

Kerberos authentication can be started/stopped in the Admin Tool -> page *Configuration* -> tab *CM Services* -> *Kerberos v5 authentication provider*, see section Tab CM Services.

29 CTI with ConSol*CM: CM.Phone

- Introduction to CM.Phone
  - Incoming Calls
  - Outgoing Calls
- CM.Phone Set-Up
  - System Requirements
  - Components Required for CM.Phone Set-Up
  - Installing CM.Phone on the Application Server
    - Basic CM.Phone Server Installation
    - Configuring CM.Phone on the Application Server
      - Configuring the Client Pop-Up Window
        - CallNotification.html
        - ContactData.html
        - ContactList.html
        - Links/Buttons within the Templates
      - Replacing the CM.Phone Pop-Up Window by a Custom Web Application
  - Installing CM.Phone on Each Windows Client
    - Engineer Authentication Modes
- Configuration of CM.Phone in the Admin Tool
  - Set the Annotations for the Data Object Group 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
  - Change the Prefix for Outgoing Calls
- Troubleshooting
  - Logging
  - Registration as phone: protocol handler

29.1 Introduction to CM.Phone

CM.Phone is a distinct ConSol*CM module which has to be licensed in addition to the core CM system.

CM.Phone is a Windows client application for 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.

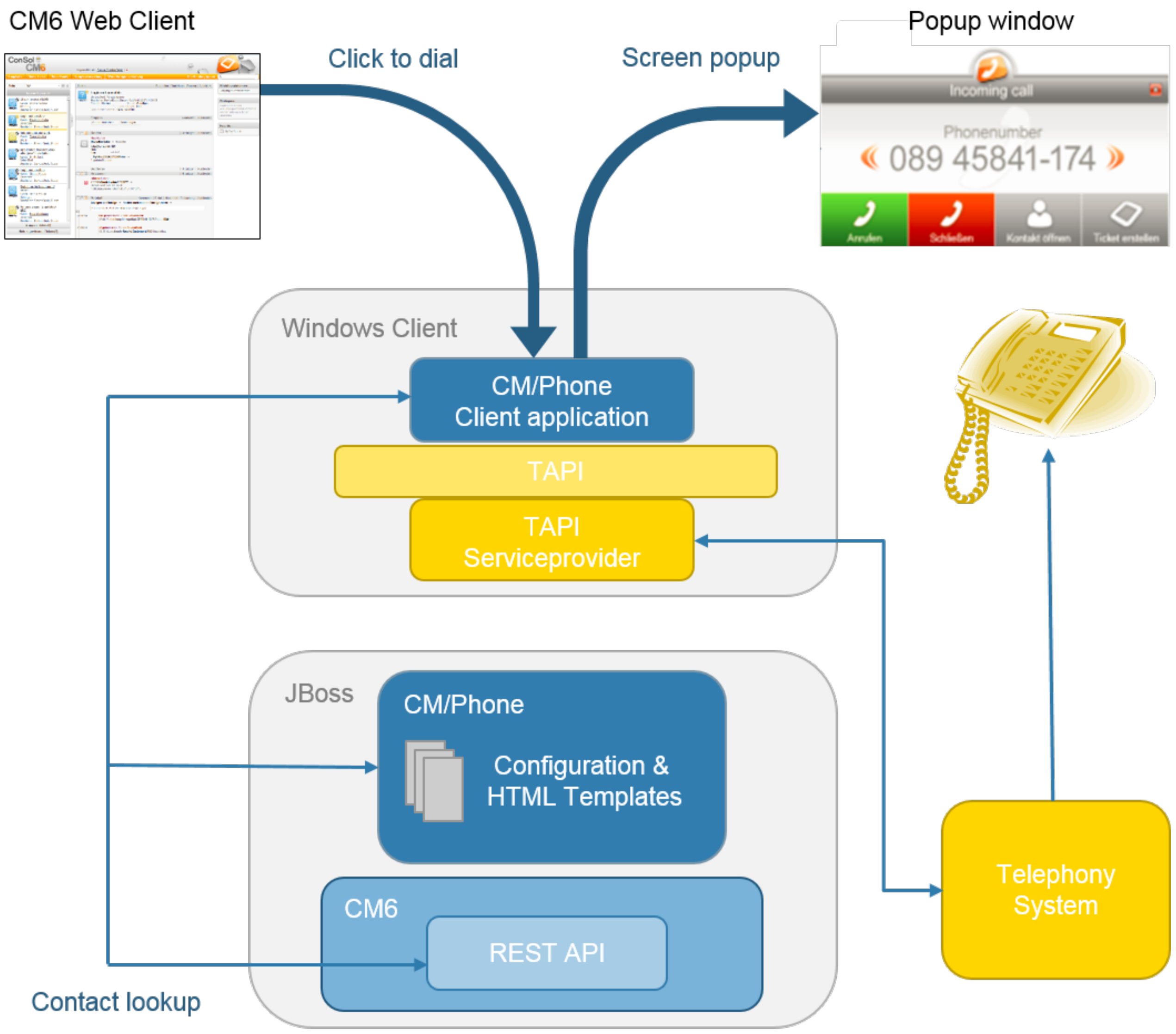


Fig. 1: CM.Phone - Basic Principle

29.1.1 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 created, displaying the phone number of the caller. The CM customer database is searched for matches for this contact. If one or more matches have been found, a contact list is offered for selection. Engineers can then decide if they want to create a ticket for the contact or if they want to have the contact page displayed. In case no corresponding contact data matches the phone number, just the calling number is displayed and the option *Create contact* is offered.

Attention:

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 CM server. These templates are loaded by the CM.Phone client application during startup. The information displayed in the pop-up window (data object group fields from the customer data model) can be customized by editing the template files (see section Configuring the Client Pop-Up Window).

The following options can be selected in the pop-up window, if exactly one contact matches in the CM database:

- **Open contact**
  Opens the contact page (contact/company) in the Web Client (alternatively *Create contact* will be offered, if the caller is unknown in CM).
- **Create ticket**
  Opens the *Create ticket* page for this found (or new) contact in the Web Client.
- **Call back**
  Will be available in case of a missed call.
- **Close**
  Closes the CM.Phone pop-up window.

In case, the contact is not yet present in the CM system, the caller's phone number will be used to fill-in the phone number field in the contact data (data object group fields) annotated as *dialable*. This will be done for new contacts and newly created tickets. Should multiple fields be annotated as *dialable* the first one will be pre-filled. In case the user has access to multiple customer groups the respective *dialable* phone number fields of each customer group will be pre-filled.

29.1.2 Outgoing Calls

The engineer can start an outgoing call directly by clicking on a phone number (e.g. in the customer data) in a data object group 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.

29.2 CM.Phone Set-Up

29.2.1 System Requirements

Please refer to the *System Requirements* of the ConSol*CM version which is installed in your environment for detailed information concerning server and client requirements for CM.Phone set-up.

29.2.2 Components Required for CM.Phone Set-Up

For CM.Phone set-up you, as an administrator, need:

- The license for CM.Phone, please ask your ConSol*CM consultant.
- The CM.Phone *.war* file for deployment in the application server.
- A *TAPI 3* driver in the telephone system.

29.2.3 Installing CM.Phone on the Application Server

Basic CM.Phone Server Installation

The CM.Phone module is delivered as a *.war* package. The *.war* package is provided as folder (not as packed file), because in this way you can easily access the configuration files located in this path.

For the server installation, perform the following steps:

1. Copy the *cm-phone.war* folder to *<CM6 path>/server/cmas/deploy*.
2. Adapt the configuration as needed (see sections below).
3. Restart the application server.

To check if the application was deployed correctly, visit the web URI:

http://<CM server>:<CM port>/cm-phone/

You should see the welcome page of the application with the link to the CM.Phone installer download.

Configuring CM.Phone on the Application Server

All application parameters are set in the file **cmphone-config.xml** which is located under *<CM6 path>/server/cmas/deploy/cm-phone.war*.

This file is loaded by the CM.Phone client application during start-up and each time the settings dialog is closed with *OK*.

However, this file should not be edited in order to configure the system. Please make sure that all configuration parameters are set using the Admin Tool.

Configuring the Client Pop-Up Window

The contents of the pop-up window are based on the HTML templates in the following path: *<CM6 path> /server/cmas/deploy/cm-phone.war/templates*. The main directory contains the templates for the default language (of the client system!):

- CallNotification.html
- ContactData.html
- ContactList.html

For each additional language which should be supported, a folder with the name of the locale has to be created (e.g. *de* for *German*) which contains localized copies of the template files.

The templates are used to render the contact details in the pop-up window. Since every customer may want to see different information in the pop-up, the content can be adapted by editing the HTML files. The templates contain tags which are replaced with actual values by the client application during a call. Those templates are Admin Tool templates which have to be defined for each customer group, please see section Configure the Admin Tool Templates for Customer Data for Each Customer Group for details.

If required, you can change the names of the templates. You can use any file name you want for the three HTML files, just make sure you have entered the correct values in the *config* file.

CallNotification.html

This is the first template which is displayed as soon as an incoming call is detected by the CM.Phone client. This window only displays the calling number since at this time there is no customer data available.

Available tags:

- **[phonenumber]**
  Phone number of the caller.
  Example: <h1>Phonenumber: [phonenumber]</h1>
- **[calltime]**
  Time of the call.
- **[content]**
  This will show additional information within the pop-up window:
  A *Loading* icon during the contact look-up or an error message if something went wrong during the look-up, e.g. wrong user name or password, etc.

Attention:

These tags are case sensitive and must be lower case.

ContactData.html

This will display the actual contact details if the look-up successfully found a matching contact for the phone number. In this template all data object group fields of the customer data model can be used as tags. This way, the displayed contact details can be adapted to any customer's need.

All tags from the *CallNotification* template are available (see above), plus the following tags:

- **[contact.id]**
  CM internal ID of the contact. This ID may be used to create additional links into the CM Web Client.
- **[contactContent]**
  Here, the contact data is filled in according to the template which is defined for each customer group in the Admin Tool under <CustomerGroupObject> -> Templates -> *CMPhone Customer Details*.

ContactList.html

In case the look-up finds more than one contact for a phone number, the *ContactList* template is displayed in the pop-up window. For each found contact, a row is added in the contact table within the template.

Here, the contact data is filled in according to the template which is defined for each customer group in the Admin Tool under <CustomerGroupObject> -> Templates -> *CMPhone Customer List*.

Links/Buttons within the Templates

Four buttons are configured in the standard templates:

- **Close**
  Closes the pop-up window.
- **Call**
  Starts an outgoing phone call to the calling number.
- **Open Contact**
  Opens the contact in the CM Web Client.
- **Create Ticket**
  Opens the Web Client in the *Create ticket* page.

These buttons can also be customized within the templates. Buttons may be removed or additional buttons or links may be added as required. Each button refers to an HTML link.

For CM.Phone there are two types of links available:

1. **External links**
   These links will open a browser window to display the page.
   For example, the link *http://heldesk/cm-client/contact/[contact.id]* will open the CM Web Client and display the selected contact.
   For these links, all data object group fields of the contact data may be used to create the URL. This way, additional functions may be added by creating a link to a customer-specific web application and by passing user data from CM as parameters, e.g.: *https://intranet.mycompany.de/index.php?id=234&id_person=[customer.personid]*.
2. **CM.Phone internal links**
   These links are only valid within the pop-up window.
   Format: http://cmphone/<command>/?<parameter>
   The following commands are available:
   a. **contactdata**
      Displays the *ContactData* template for the selected contact.
      Parameter: Contact Id
      This command is used in the *ContactList* template to allow the user to select and display a

specific contact:
http://cmphone/contactdata/?[contact.id]

b. **contactlist**
   Displays the *ContactList* template.
   This command is used to allow the user to go back to the list of contacts from the *Contact Data* page:
   http://cmphone/contactlist
c. **call**
   Starts an outgoing phone call to the phone number of the contact:
   http://cmphone/call
d. **runcmd**
   Starts a local application on the client PC.
   Parameter: Command line of the application
   This may be used for instance to start a database application and pass a user ID as a command line parameter, e.g. :
   http://cmphone/runcmd/?dbapp.exe +userid=[customer.userid]
e. **close**
   Closes the pop-up window:
   http://cmphone/close

Replacing the CM.Phone Pop-Up Window by a Custom Web Application

In case a customer does not want to use the pop-up window from CM.Phone but instead requires a custom web application to be opened for a phone call, this can be done by setting the *OnCallCmd* parameter in the *cmphone-config.xml* file. If it is set to an external URL, a browser window will be opened with this URL for each phone call. The pop-up window will not be displayed.

29.2.4 Installing CM.Phone on Each Windows Client

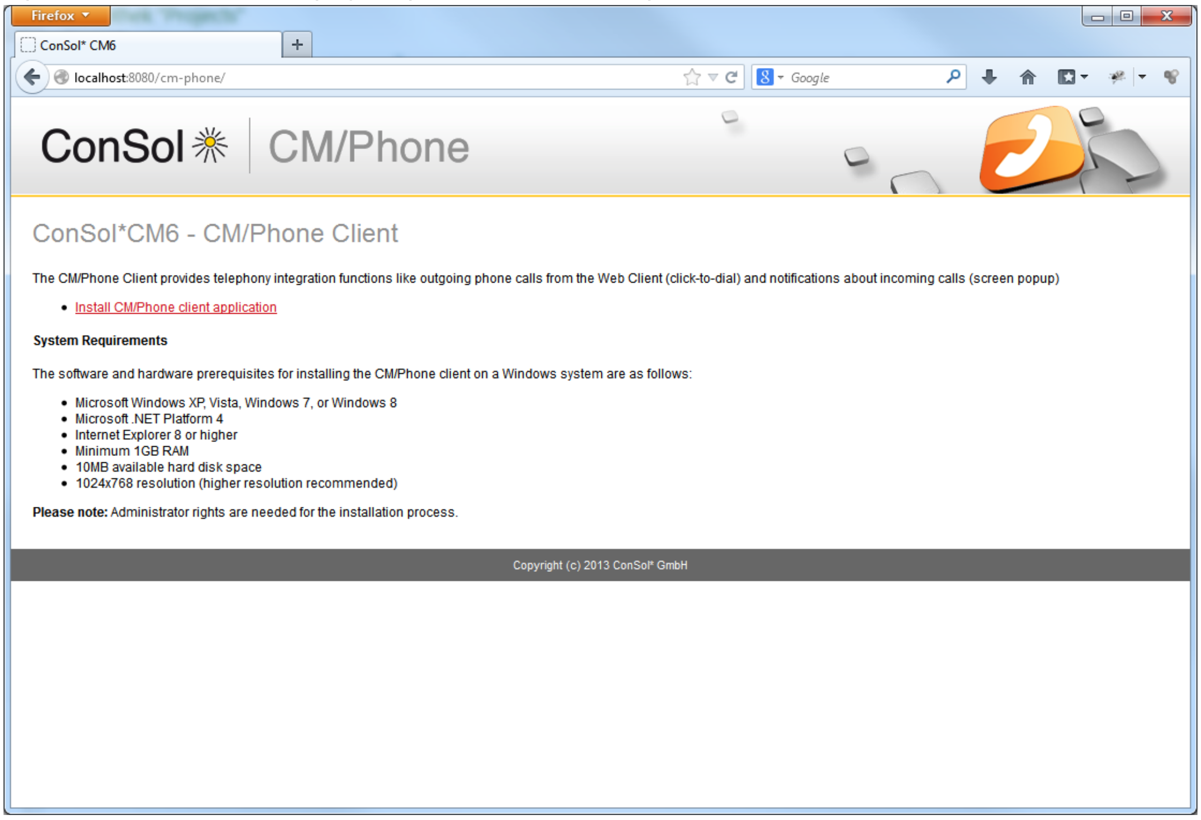
The CM.Phone client application has to be installed locally on all client PCs that need CTI functionality. The set-up package has to be downloaded from the CM.Phone start page on the CM server.

Attention:

Administrator rights are needed on the client PC to install the CM.Phone application. The reason for this is the registration as *phone: protocol handler* which requires a registry key to be written.

For the client installation, perform the following steps:

1. Open the CM.Phone start page *http://<CM6-URL>/cm-phone*:



ConSol*CM6 - CM/Phone Client

The CM/Phone Client provides telephony integration functions like outgoing phone calls from the Web Client (click-to-dial) and notifications about incoming calls (screen popup)

- Install CM/Phone client application

System Requirements

The software and hardware prerequisites for installing the CM/Phone client on a Windows system are as follows:

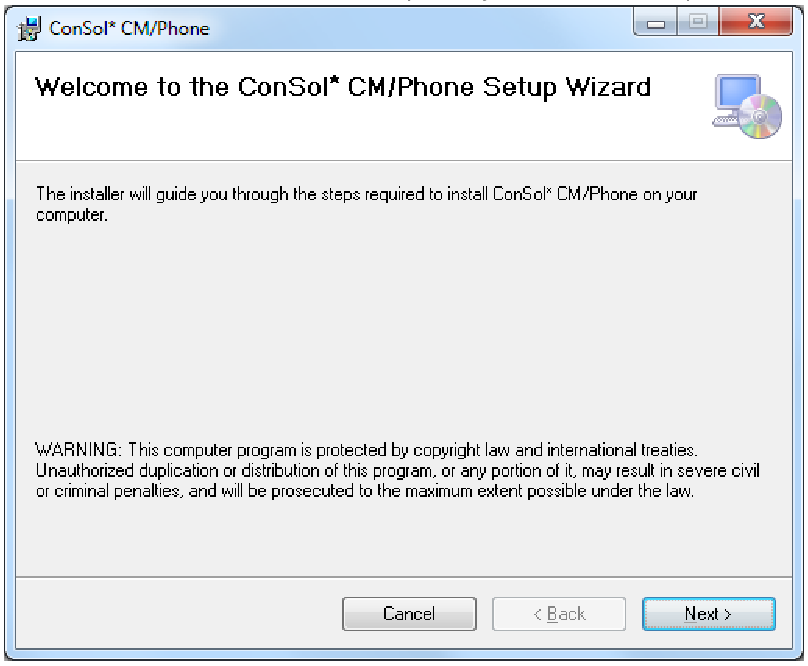
- Microsoft Windows XP, Vista, Windows 7, or Windows 8
- Microsoft .NET Platform 4
- Internet Explorer 8 or higher
- Minimum 1GB RAM
- 10MB available hard disk space
- 1024x768 resolution (higher resolution recommended)

Please note: Administrator rights are needed for the installation process.

Copyright (c) 2013 ConSol* GmbH

Fig. 2: CM.Phone Client Set-Up (1)

2. Download and run the installation package *CMPhoneSetup.msi*:



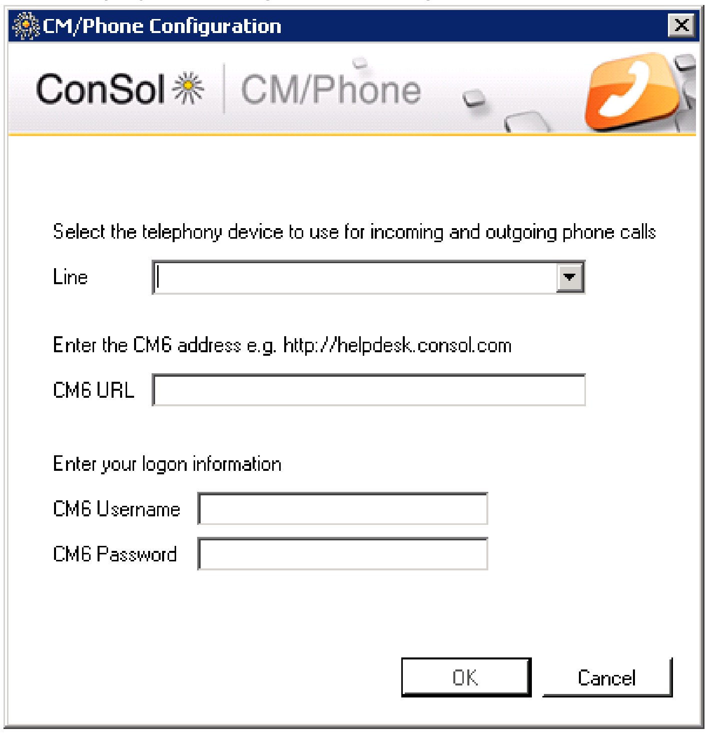
Welcome to the ConSol* CM/Phone Setup Wizard

The installer will guide you through the steps required to install ConSol* CM/Phone on your computer.

WARNING: This computer program is protected by copyright law and international treaties. Unauthorized duplication or distribution of this program, or any portion of it, may result in severe civil or criminal penalties, and will be prosecuted to the maximum extent possible under the law.

Fig. 3: CM.Phone Client Set-Up (2)

3. Start the CM.Phone application. Start *All programs* -> *ConSol CM6* -> *CM Phone.* After the first start it will display the configuration dialog:



Select the telephony device to use for incoming and outgoing phone calls

Line

Enter the CM6 address e.g. http://helpdesk.consol.com

CM6 URL

Enter your logon information

CM6 Username

CM6 Password

Fig. 4: CM.Phone Client Set-Up (3)

Fill in the following fields:

- **Line**
  Select the TAPI line which should be used.
- **CM6 URL**
  Enter the URL of the CM system. The basic URL is required, e.g. *http://myserver:8080*.
- **CM6 Username**
  Your CM user name (the user has to have enough rights to search for data objects through CM REST API !).
- **CM6 Password**
  Your CM password.

Information:

The configuration dialog can be opened anytime by opening the context menu of the CM. Phone notification icon in the task bar and selecting *Settings ...* .

Engineer Authentication Modes

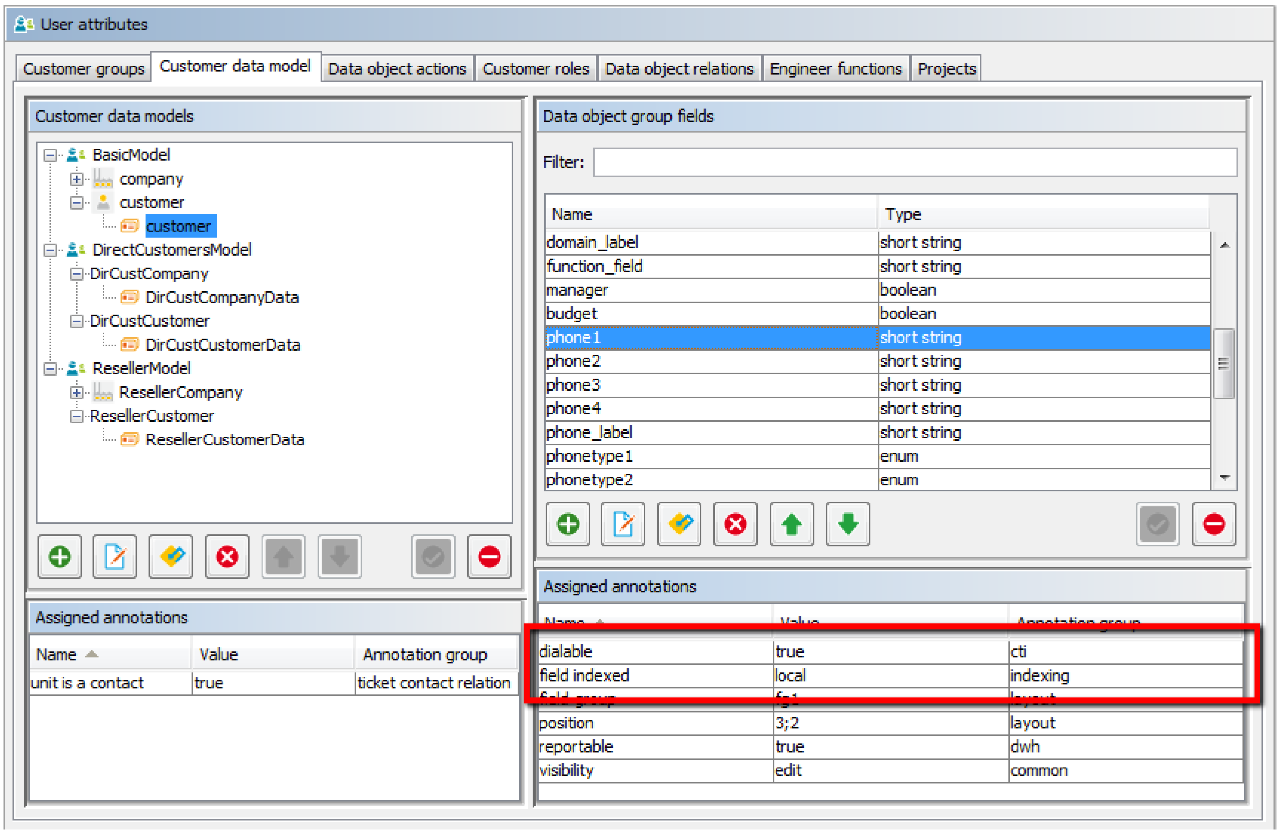
The ConSol*CM CTI client will not work together with SSO authentication mode (see section Single Sign-On with ConSol*CM Using Kerberos).

29.3 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 data object group 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 prefix for outgoing calls.

29.3.1 Set the Annotations for the Data Object Group Fields Which Contain Phone Numbers



Name	Type
domain_label	short string
function_field	short string
manager	boolean
budget	boolean
phone1	short string
phone2	short string
phone3	short string
phone4	short string
phone_label	short string
phonetype1	enum
phonetype2	enum

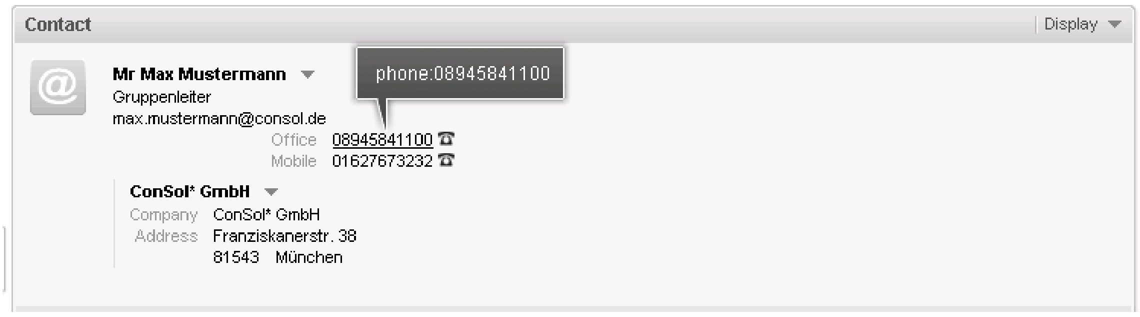
Assigned annotations

Name	Value	Annotation group
unit is a contact	true	ticket contact relation

Assigned annotations

Name	Value	Annotation group
dialable	true	cti
field indexed	local	indexing
field group	fg1	layout
position	3;2	layout
reportable	true	dwh
visibility	edit	common

Fig. 5: ConSol*CM Admin Tool - Annotations for Data Object Group Fields with Phone Numbers



Mr Max Mustermann
Gruppenleiter
max.mustermann@consol.de
Office 08945841100
Mobile 01627673232

ConSol* GmbH
Company ConSol* GmbH
Address Franziskanerstr. 38
81543 München

Fig. 6: ConSol*CM Web Client - Dialable Number When Using CM.Phone

Two annotations are required for the data object group fields which contain phone numbers:

- **dialable =** ***true***
  This configures the phone numbers as dialable links in the Web Client.
- **field-indexed =** ***local***
  This makes the field searchable which is important for the customer look-up.

29.3.2 Configure the Admin Tool Templates for Customer Data for Each Customer Group

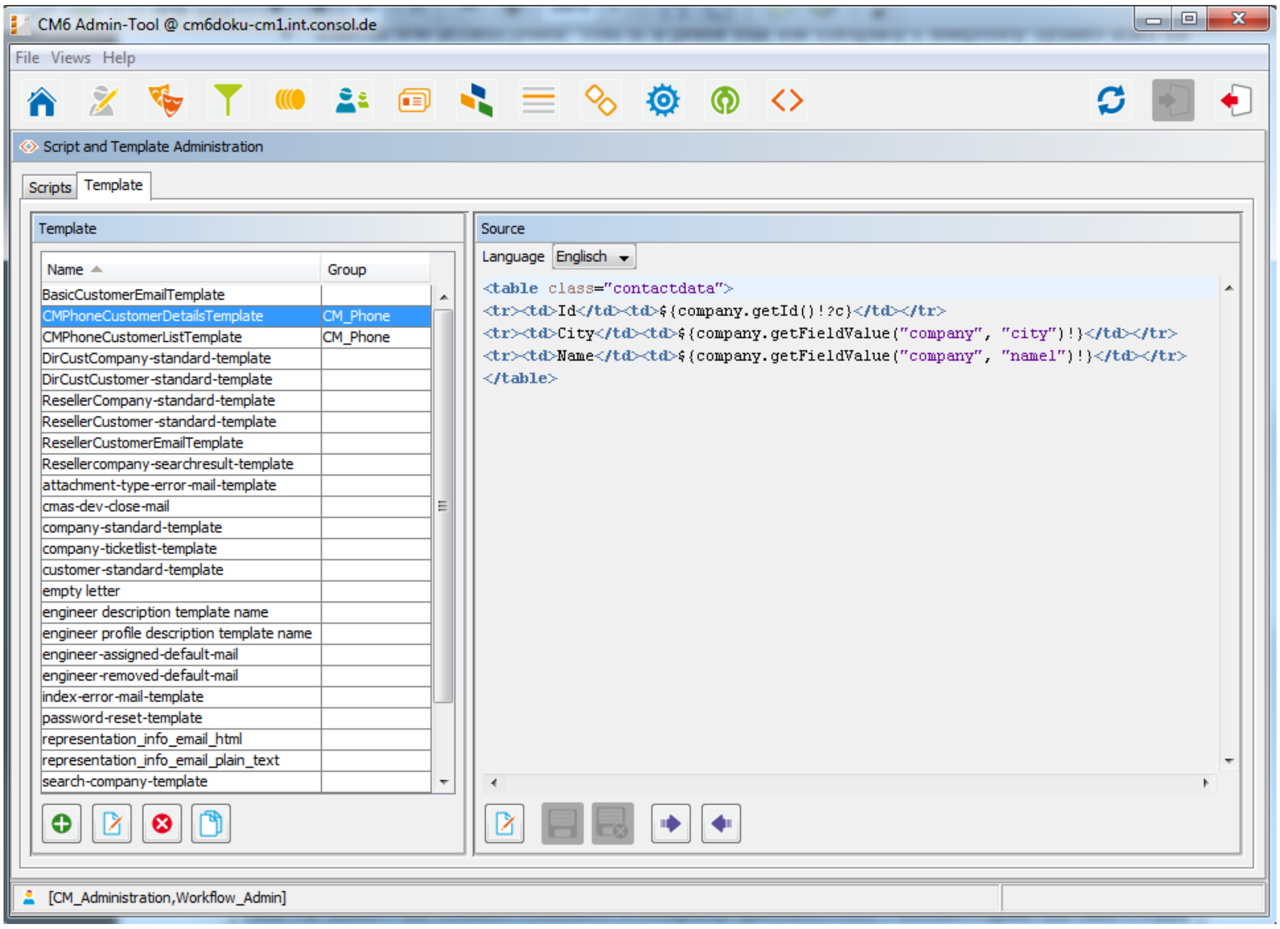
The customer data model configuration also allows for two more contexts/types of data object templates now:

- CMPhone customer details
- CMPhone customer list

They are used for defining how CM.Phone should render incoming call information. The first one is used for exactly one data object matching the phone number and the second one is used for multiple matches, so that the engineer has to select the correct customer.

You have to perform two steps:

1. Write the templates and store them in the *Script and Template Administration* of the Admin Tool.
2. Assign the templates to customer data models (*User attributes* - tab *Customer data model*).

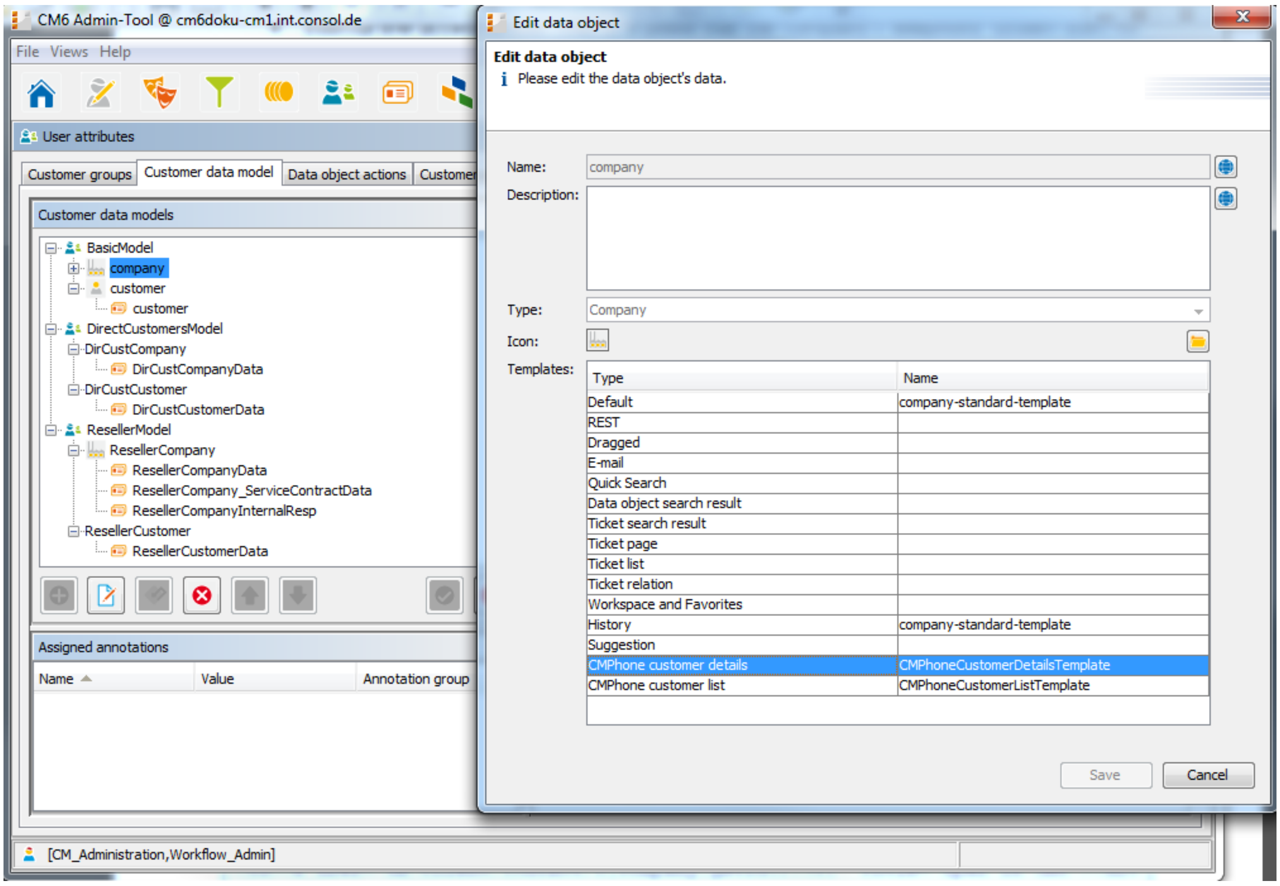

CM6 Admin-Tool @ cm6doku-cm1.int.consol.de
File Views Help
Script and Template Administration
Scripts
Template
Template
Source
Language Englisch

Name	Group
BasicCustomerEmailTemplate	
CMPhoneCustomerDetailsTemplate	CM_Phone
CMPhoneCustomerListTemplate	CM_Phone
DirCustCompany-standard-template	
DirCustCustomer-standard-template	
ResellerCompany-standard-template	
ResellerCustomer-standard-template	
ResellerCustomerEmailTemplate	
Resellercompany-searchresult-template	
attachment-type-error-mail-template	
cmas-dev-close-mail	
company-standard-template	
company-ticketlist-template	
customer-standard-template	
empty letter	
engineer description template name	
engineer profile description template name	
engineer-assigned-default-mail	
engineer-removed-default-mail	
index-error-mail-template	
password-reset-template	
representation_info_email_html	
representation_info_email_plain_text	
search-company-template	

```
<table class="contactdata">
<tr><td>Id</td><td>${company.getId()!?c}</td></tr>
<tr><td>City</td><td>${company.getFieldValue("company", "city")!}</td></tr>
<tr><td>Name</td><td>${company.getFieldValue("company", "name1")!}</td></tr>
</table>
```

[CM_Administration,Workflow_Admin]

Fig. 7: ConSol*CM Admin Tool - Example Template for Rendering Customer Data for Display in CM.Phone



- BasicModel
  - company
  - customer
    - customer
- DirectCustomersModel
  - DirCustCompany
    - DirCustCompanyData
  - DirCustCustomer
    - DirCustCustomerData
- ResellerModel
  - ResellerCompany
    - ResellerCompanyData
    - ResellerCompany_ServiceContractData
    - ResellerCompanyInternalResp
  - ResellerCustomer
    - ResellerCustomerData

Edit data object

Please edit the data object's data.

Name: company

Description:

Type: Company

Icon:

Templates:

Type	Name
Default	company-standard-template
REST	
Dragged	
E-mail	
Quick Search	
Data object search result	
Ticket search result	
Ticket page	
Ticket list	
Ticket relation	
Workspace and Favorites	
History	company-standard-template
Suggestion	
CMPhone customer details	CMPhoneCustomerDetailsTemplate
CMPhone customer list	CMPhoneCustomerListTemplate

Fig. 8: ConSol*CM Admin Tool - Assignment of CM.Phone Templates for Customer Data to Customer Groups

29.3.3 Configure the Phone Number Format for Each Customer Group

The format defined here is used to transform incoming numbers (from the respective customer group) to a common canonical form exchange format. 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 data object. 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.

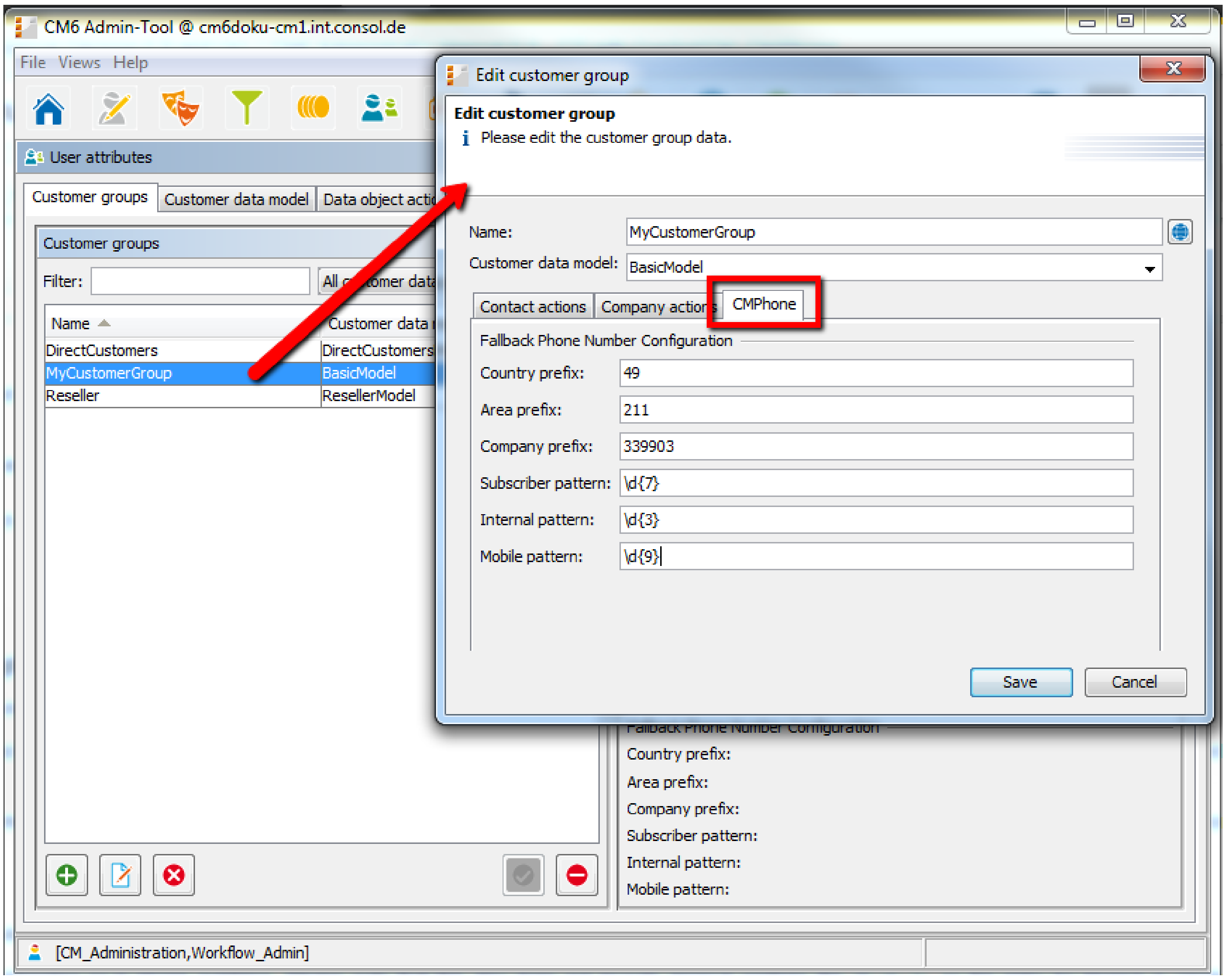
The topic *User attributes* has to be selected after logging in to the Admin Tool for this configuration. On the tab *Customer groups* the desired customer group has to be selected for editing. After clicking the *Edit* button below the list of customer groups the edit dialog opens which now contains a new tab titled *CMPhone*.

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 if an outgoing call is going to be performed within the same country or not. Several phone providers do not handle canonical (so theoretically correct) numbers for domestic calls. Therefore the country prefix has to be cut-off 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 if the number provided is a full subscriber number (eventually 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 also a country prefix will be added, so the result will be: 49600289906. If the engineer enters a full number starting with "+" or "0" then the configuration is skipped - CM assumes no number conversion is required.



Edit customer group

Please edit the customer group data.

Name	Customer data model
DirectCustomers	DirectCustomers
MyCustomerGroup	BasicModel
Reseller	ResellerModel

Name: MyCustomerGroup

Customer data model: BasicModel

Fallback Phone Number Configuration

Country prefix:	49
Area prefix:	211
Company prefix:	339903
Subscriber pattern:	\d{7}
Internal pattern:	\d{3}
Mobile pattern:	\d{9}

Fig. 9: ConSol*CM Admin Tool - 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.

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, if it is already fully qualified. Otherwise 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 for a valid number to be dialed. In all other cases the supplied original number cannot be used for making a connection.

29.3.4 Set the System Properties

There are three new properties with relevance for CM.Phone in CM to be set in the Admin Tool. 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**
  This is the local country code. The value is an international country code like *49* for Germany, for example. Default value is *49*.

- **internal.line.access.prefix**
  This is a prefix that the company’s telephony system asks 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.

29.3.5 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 CM consultant for advice.

29.4 Troubleshooting

29.4.1 Logging

For debugging purposes, a log file may be activated on the client. In order to do this, the log configuration file *log4net.xml* in the installation path of the client, usually *<Program Files>\ConSol\CMPhone*, has to be configured.

Since most users do not have *write* access to the *Program Files* directory, the log file path has to be set to a folder that is writable for the user, e.g.:

Log4net configuration for CM.Phone logging

```
<file value="c:\temp\cmphone.log" />
<appender name="RollingFileAppender" type="log4net.Appender.RollingFileAppender">
    <file value=" c:\temp\cmphone.log" />
     <appendToFile value="true" />
     <rollingStyle value="Size" />
     <maxSizeRollBackups value="10" />
     <maximumFileSize value="1MB" />
     <staticLogFileName value="true" />
    <layout type="log4net.Layout.PatternLayout">
         <conversionPattern value="%date %-5level %logger - %message%newline" />
    </layout>
</appender>
```

Notes:

- Special characters and/or whitespace do not have to be masked.
- Use the following to write the log file into the user's home directory:

```
<file value="${USERPROFILE}\phone.log" />
```

29.4.2 Registration as phone: protocol handler

If the client application cannot be installed by the users themselves because of insufficient access rights, the application may be distributed by an administrator employing a software distribution system. In that case, the application needs to be registered on the client as *phone: protocol handler* by creating the appropriate registry keys:

- [HKEY_CLASSES_ROOT\phone]
  Please make sure to set the space in *URL Protocol*, otherwise it will not work.
- [HKEY_CLASSES_ROOT\phone\DefaultIcon]
  @="C:\\Program Files (x86)\\ConSol\\CMPhone\\cmphone.ico"

- [HKEY_CLASSES_ROOT\phone\shell]
- [HKEY_CLASSES_ROOT\phone\shell\open]
- [HKEY_CLASSES_ROOT\phone\shell\open\command]
  @="C:\\Program Files (x86)\\ConSol\\CMPhone\\cmphone.exe" (example)

30 Data Warehouse (DWH) Management

- Introduction
  - Data Warehouse
  - ConSol*CM Data Warehouse and ConSol*CM Reporting Framework
- DWH Management Using the Admin Tool
  - DWH Administration Overview
  - Basic DWH Configuration
  - Initialization of the DWH
  - First DWH Synchronization
  - DWH Synchronization During System Operation
  - DWH Tasks
  - DWH Troubleshooting and Repair
- DWH-Related System Properties

Attention:

To set up a DWH, a ConSol*CM Reporting Framework (CMRF) that is up and running is required. If your system does not include a CMRF yet, please talk to your ConSol*CM manager or contact ConSol* Software.

30.1 Introduction

30.1.1 Data Warehouse

A data warehouse is a collection of data from one or more systems and/or databases that provides the basis for reporting and for data analysis. Often, the data has been combined or rearranged (integrated) in a way that a perfect basis for reporting and for data analysis is provided.

30.1.2 ConSol*CM Data Warehouse and ConSol*CM Reporting Framework

A ConSol*CM default installation comprises all modules that are required to build a data warehouse. The core component is the **ConSol*CM Reporting Framework (CMRF)**.

This is a Java EE application which synchronizes the data between the ConSol*CM database and the DWH database. Please see the following picture for system architecture examples with DWH and CMRF. We recommend that you use two servers, one for ConSol*CM and one for CMRF. Please refer to the current *System Requirements* for information about the supported application servers and RDBMS.

There are two different modes to synchronize the DWH with the CM database:

- **LIVE mode**
  In this mode, every change that has been submitted to the CM database is immediately synchronized with the DWH.
- **ADMIN mode**
  In this mode, the administrator has to trigger the synchronization manually.

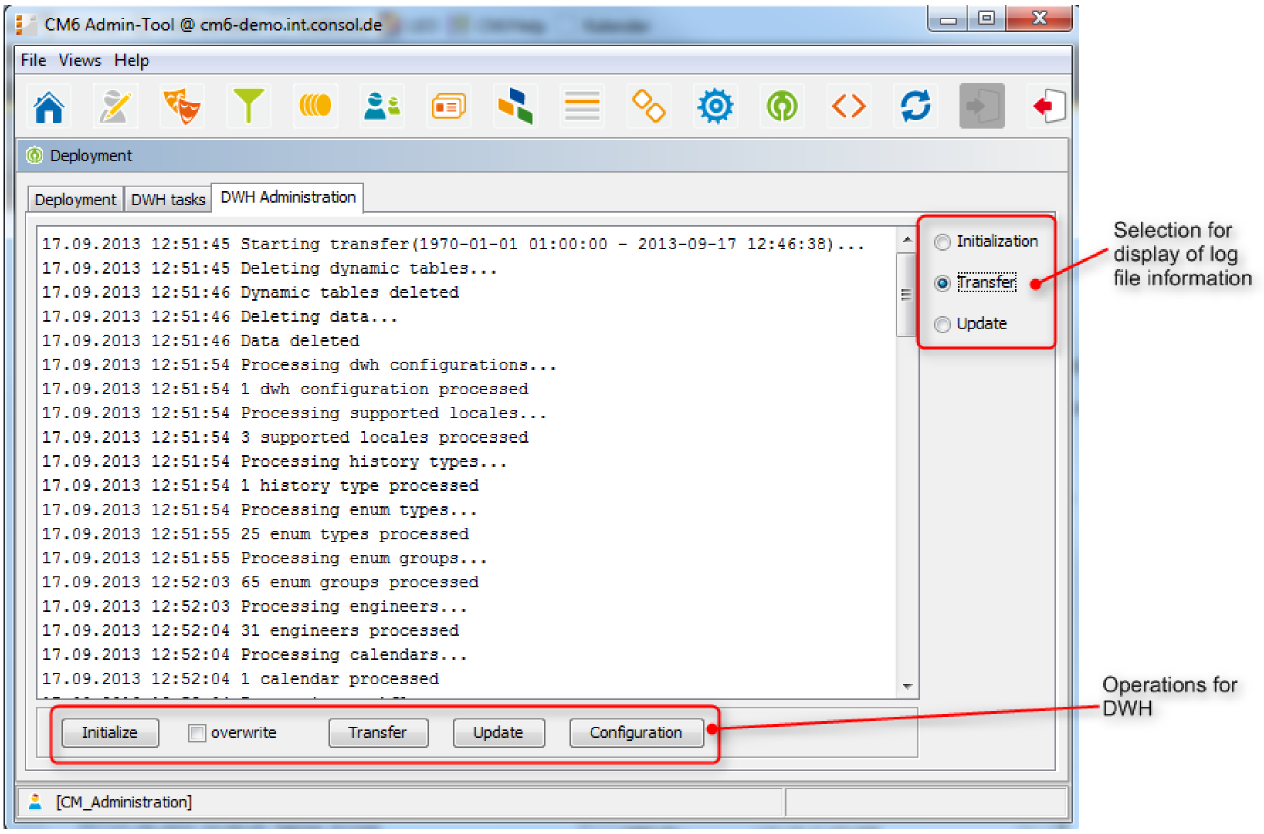
Attention:

Only data from custom fields and data object group fields with the annotation *reportable = true* will be synchronized with the DWH!

30.2 DWH Management Using the Admin Tool

30.2.1 DWH Administration Overview

To manage the DWH, use the *DWH Administration* tab under *Deployment*.



```
17.09.2013 12:51:45 Starting transfer(1970-01-01 01:00:00 - 2013-09-17 12:46:38)...
17.09.2013 12:51:45 Deleting dynamic tables...
17.09.2013 12:51:46 Dynamic tables deleted
17.09.2013 12:51:46 Deleting data...
17.09.2013 12:51:46 Data deleted
17.09.2013 12:51:54 Processing dwh configurations...
17.09.2013 12:51:54 1 dwh configuration processed
17.09.2013 12:51:54 Processing supported locales...
17.09.2013 12:51:54 3 supported locales processed
17.09.2013 12:51:54 Processing history types...
17.09.2013 12:51:54 1 history type processed
17.09.2013 12:51:54 Processing enum types...
17.09.2013 12:51:55 25 enum types processed
17.09.2013 12:51:55 Processing enum groups...
17.09.2013 12:52:03 65 enum groups processed
17.09.2013 12:52:03 Processing engineers...
17.09.2013 12:52:04 31 engineers processed
17.09.2013 12:52:04 Processing calendars...
17.09.2013 12:52:04 1 calendar processed
```

Fig. 1: ConSol*CM Admin Tool - DWH Administration

In the center area, the log file information of DWH operations is displayed. Use the radio buttons on the right-hand side to select which log file should be displayed. The DWH operations are available as buttons in the row below the center area.

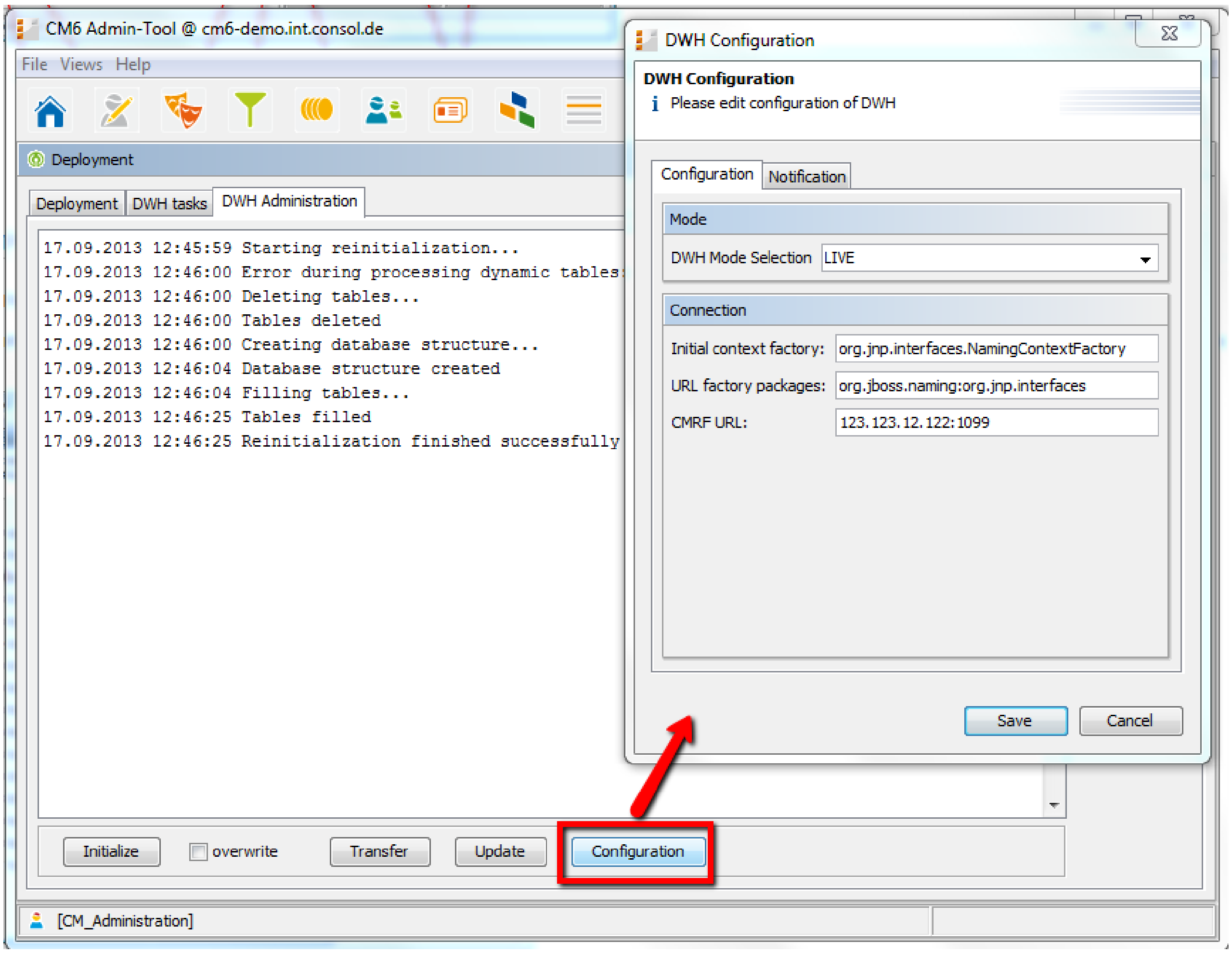
- **Initialize**
  Create tables during DWH set-up, see Initialization of the DWH.
- **overwrite**
  Used for re-initialization, see Initialization of the DWH.
- **Transfer**
  Start initial data transfer after set-up, see First DWH Synchronization.
- **Update**
  Transfer new/additional data to the DWH, see DWH Synchronization During System Operation.
- **Configuration**
  Open DWH configuration panel, see Basic DWH Configuration.

The radio buttons and the buttons for DWH operations are not coupled with one another, i.e. when you select an operation, the log file display is not changed. See the following paragraphs for detailed explanations about all operations.

30.2.2 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 database server has to be available for the CMRF server.

In order to prepare the system for the DWH synchronization, you have to configure the database and the DWH mode. Open the *Deployment* section of the Admin Tool and open the tab *DWH Administration*. Click on *Configuration*, open the tab *Configuration*, and insert all values of the CMRF server.



```
17.09.2013 12:45:59 Starting reinitialization...
17.09.2013 12:46:00 Error during processing dynamic tables:
17.09.2013 12:46:00 Deleting tables...
17.09.2013 12:46:00 Tables deleted
17.09.2013 12:46:00 Creating database structure...
17.09.2013 12:46:04 Database structure created
17.09.2013 12:46:04 Filling tables...
17.09.2013 12:46:25 Tables filled
17.09.2013 12:46:25 Reinitialization finished successfully
```

DWH Mode Selection	LIVE
Initial context factory:	org.jnp.interfaces.NamingContextFactory
URL factory packages:	org.jboss.naming:org.jnp.interfaces
CMRF URL:	123.123.12.122:1099

Fig. 2: ConSol*CM Admin Tool - DWH Configuration: File Card Configuration

At *DWH Mode Selection*, choose one of the available options:

- **LIVE**
  In this mode, every change that has been submitted to the CM database is immediately synchronized with the DWH.
- **ADMIN**
  In this mode, the administrator has to trigger the synchronization manually.
- **OFF**
  No transfer of data to the DWH.

You can also see the DWH mode that is in operation by taking a look at the corresponding DWH system property *cmas-dwh-server*, *dwh.mode* (see Appendix C).

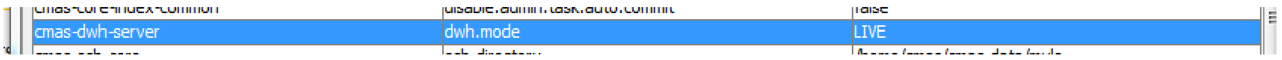

cmas-dwh-server dwh.mode LIVE

Fig. 3: ConSol*CM Admin Tool - System Property for DWH Mode

For the connection, the following parameters are required:

- For **JBoss**:
  - **Initial context factory**
    The Java class that is used for the connection. No changes are required here since ConSol*CM selects the correct value during system set-up.
  - **URL factory packages**
    The Java package that comprises the required connection classes. No changes are required here since ConSol*CM selects the correct value during system set-up.
  - **CMRF URL**
    The URL of the CMRF, i.e. the URL to which the CM system should connect in order to provide the information about new synchronization tasks. The general notation

    ```
    <CMRF_HOST_IP>:<JNDI_PORT>
    ```

    (i.e. *192.168.0.1:1099*) can be used. Please note that the default JNDI port is *1099*. In case you are using different JBoss port mappings, then the JNDI port will also differ. I.e. when using *ports-01* then the JNDI port is *1199*, for *ports-02* it is *1299*, and so on.
- For **Weblogic**:
  - **Initial context factory**
    The Java class that is used for the connection. Use:

    ```
    weblogic.jndi.WLInitialContextFactory
    ```

  - **URL factory packages**
    The Java package that comprises the required connection classes. Use:

    ```
    weblogic.jndi.factories:weblogic.corba.j2ee.naming.url:weblogic.corba.client.
    naming
    ```

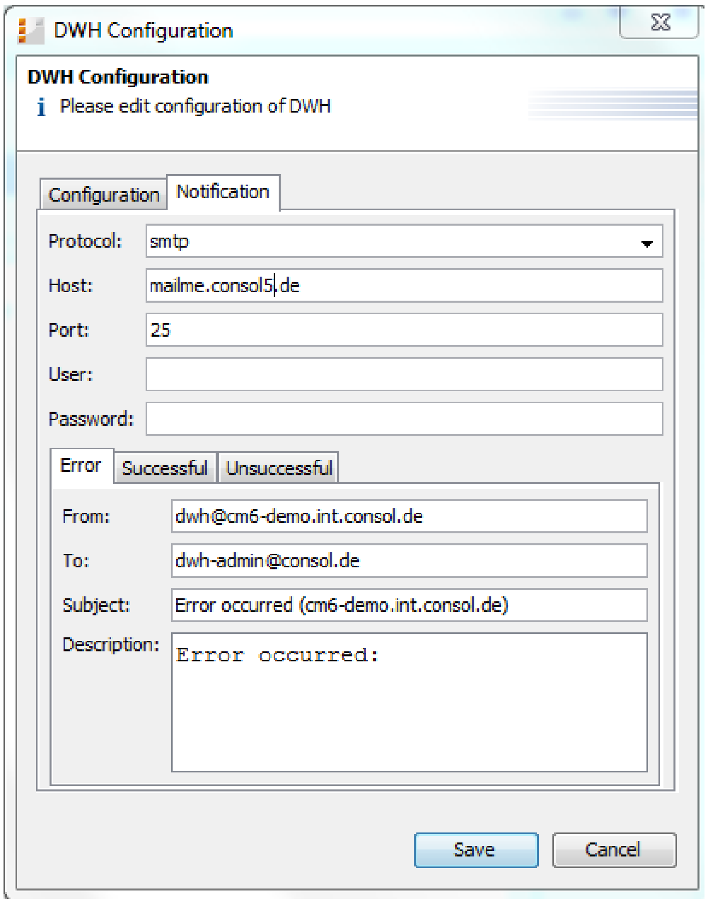
- **CMRF URL**
  The URL of the CMRF, i.e. the URL to which the CM system should connect in order to provide the information about new synchronization tasks. The *t3* protocol has to be used, i.e.

  ```
  t3://<CMRF_HOST_IP>:<JNDI_PORT>
  ```

  (i.e. *t3://localhost:7010*).

In the tab *Notification* you can configure the format of the messages (e-mails) which are sent by the system concerning DWH operations. These might be errors, success messages, or an information about an unsuccessful operation.

The values are saved in the DWH notification properties (see Appendix C for details).



Protocol: smtp
Host: mailme.consol5.de
Port: 25
User:
Password:
From: dwh@cm6-demo.int.consol.de
To: dwh-admin@consol.de
Subject: Error occurred (cm6-demo.int.consol.de)
Description: Error occurred:

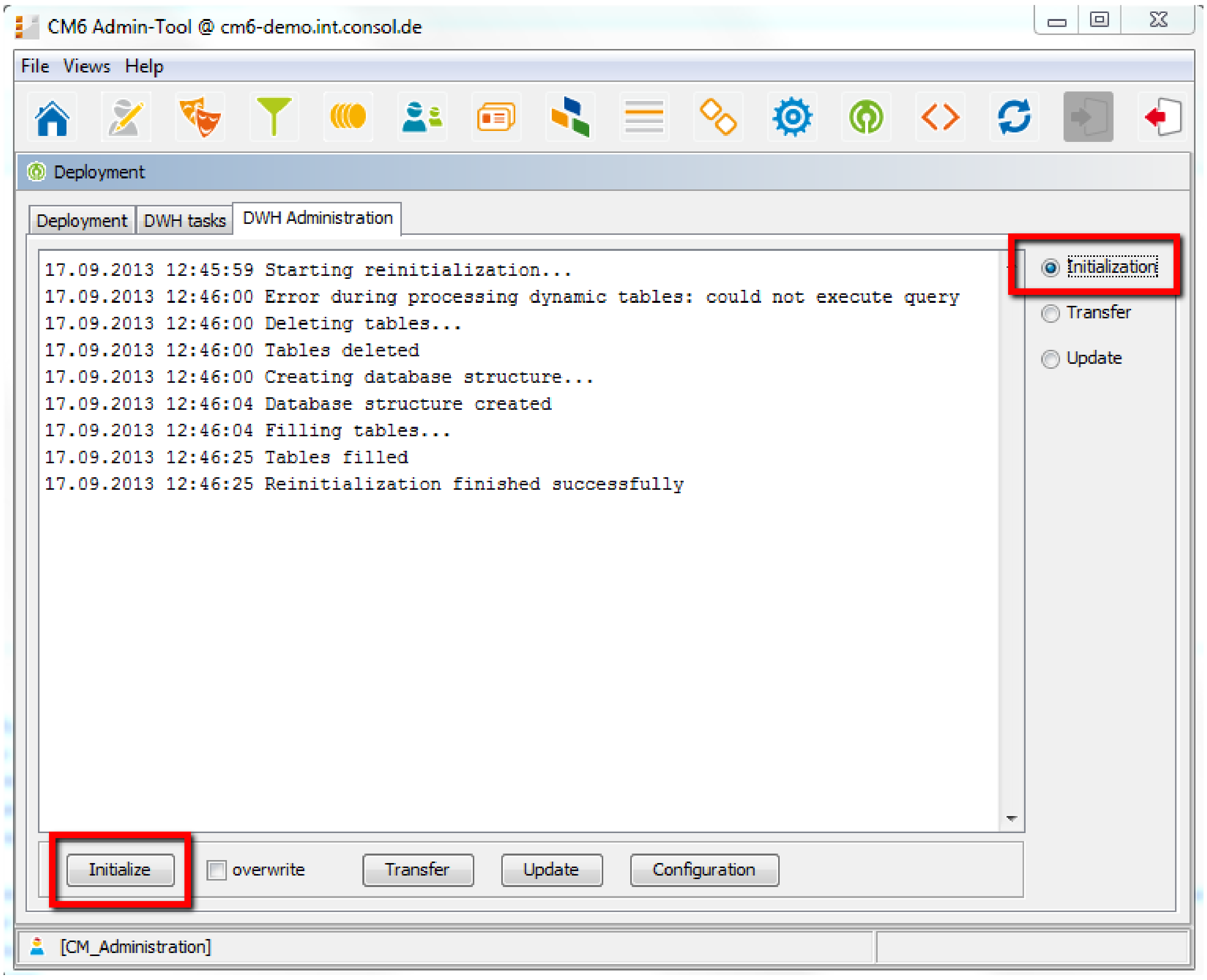
Fig. 4: ConSol*CM Admin Tool - DWH Configuration: Tab Notification

The following fields are available:

- **Protocol**
  Required - The protocol that is used to send the message, usually this is SMTP.
- **Host**
  Required - The e-mail server. You can enter a name (DNS-resolvable) or an IP address.
- **Port**
  Required - The port on the e-mail server where the mail daemon is listening.
- **User**
  Optional - User name if a user authentication is required at the e-mail server.
- **Password**
  Optional - Password of the e-mail user if a user authentication is required at the e-mail server.
- Tabs **Error/Successful/Unsuccessful**
  Here the e-mail parameters for e-mails that are sent by the system concerning the DWH 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 e-mail address for messages (maybe this is another FROM address than the one used for e-mails to customers and to engineers).
  - **To**
    The e-mail address of the recipient of the DWH messages. Initially this will be the ConSol*CM administrator e-mail address which has been entered during system set-up.
  - **Subject**
    The (e-mail) subject of the error/success/unsuccessful message.
  - **Description**
    The body (text) of the message.

30.2.3 Initialization of the DWH

When the basic configuration has been performed, the DWH initialization can be started. Press *Initialize* and follow the entries in the protocol panel. Be sure to have marked *Initialization* (radio button) in the top right corner to display initialization events.



```
17.09.2013 12:45:59 Starting reinitialization...
17.09.2013 12:46:00 Error during processing dynamic tables: could not execute query
17.09.2013 12:46:00 Deleting tables...
17.09.2013 12:46:00 Tables deleted
17.09.2013 12:46:00 Creating database structure...
17.09.2013 12:46:04 Database structure created
17.09.2013 12:46:04 Filling tables...
17.09.2013 12:46:25 Tables filled
17.09.2013 12:46:25 Reinitialization finished successfully
```

Fig. 5: ConSol*CM Admin Tool - DWH Initialization

During this step, the database structure in the DWH is created with all tables and relations. No data will be transferred yet. Depending on the amount of data, this might take some hours.

If the DWH has been in operation and has to be set-up a second time, a reinitialization has to be performed. Check the *overwrite* option in order to delete the old database structure and create a new one. Then press *Initialize*.

30.2.4 First DWH Synchronization

To fill the data warehouse with the CM data for the first time, press *Transfer*. The initial transfer is started. You can follow the log entries by opening *Transfer* in the protocol panel.

30.2.5 DWH Synchronization During System Operation

If the DWH is running in *ADMIN* mode, the DWH administrator has to start the transfer manually by clicking on *Update*. Then all data that is supposed to be transferred, i.e. all data from fields with the *reportable = true* annotation that has been added or changed since the last transfer, is transferred. When the *Update* button has been clicked, all required operations will be created as tasks and all open tasks will be listed in the *DWH tasks* panel.

If a custom field or data object group 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 and/or customers is transferred.

You can follow the log entries for the DWH operation by opening the *Update* part of the protocol panel.

Warning:

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!

30.2.6 DWH Tasks

In this list you will find entries (one entry for one 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.
- custom field or data object group field annotations have been set *reportable = true* and the check box *Automatic commit of administrative changes* has not been checked.

You can mark a task in the list and execute it manually.

If the check box *Automatic commit of administrative changes* has been checked, the tasks will be run automatically by the system.

30.2.7 DWH Troubleshooting and Repair

If any errors have occurred during initialization, transfer, or update, the log entries are displayed in the respective protocol panel.

You can also check the original log file under the following path:

- **JBoss:**

```
<JBOSS_HOME>\server\<CMRF_SERVER_NAME>\log\cmrf.log
```

- **Weblogic:**

```
<ORACLE_HOME>\Middleware\user_projects\domains\consolcm6_domain\cmrf-logs\cmrf.log
```

Please note that these are the standard paths. In ConSol*CM, *Log4J* is used. If you have configured a different path for your log files in the *log4j.xml* file, you will know where to find them.

Usually the log file and/or protocol panel entries give good hints at the initial reason for the transfer failure. If you cannot fix the problem and you have a maintenance contract with ConSol*, please contact our support team.

30.3 DWH-Related System Properties

A list of all system properties which are relevant for a specific DWH configuration can be found in section Appendix D - Important System Properties - Ordered by Area of Application, DWH.

31 CM6 Administrator Manual 6.9.4 - The Customer Portal - CM.Track

31.1 The Customer Portal: CM.Track

The portal CM.Track allows customers to log in to the ConSol*CM system. Like the 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 a REST (Representational State Transfer, see Appendix B (Glossary)) API.

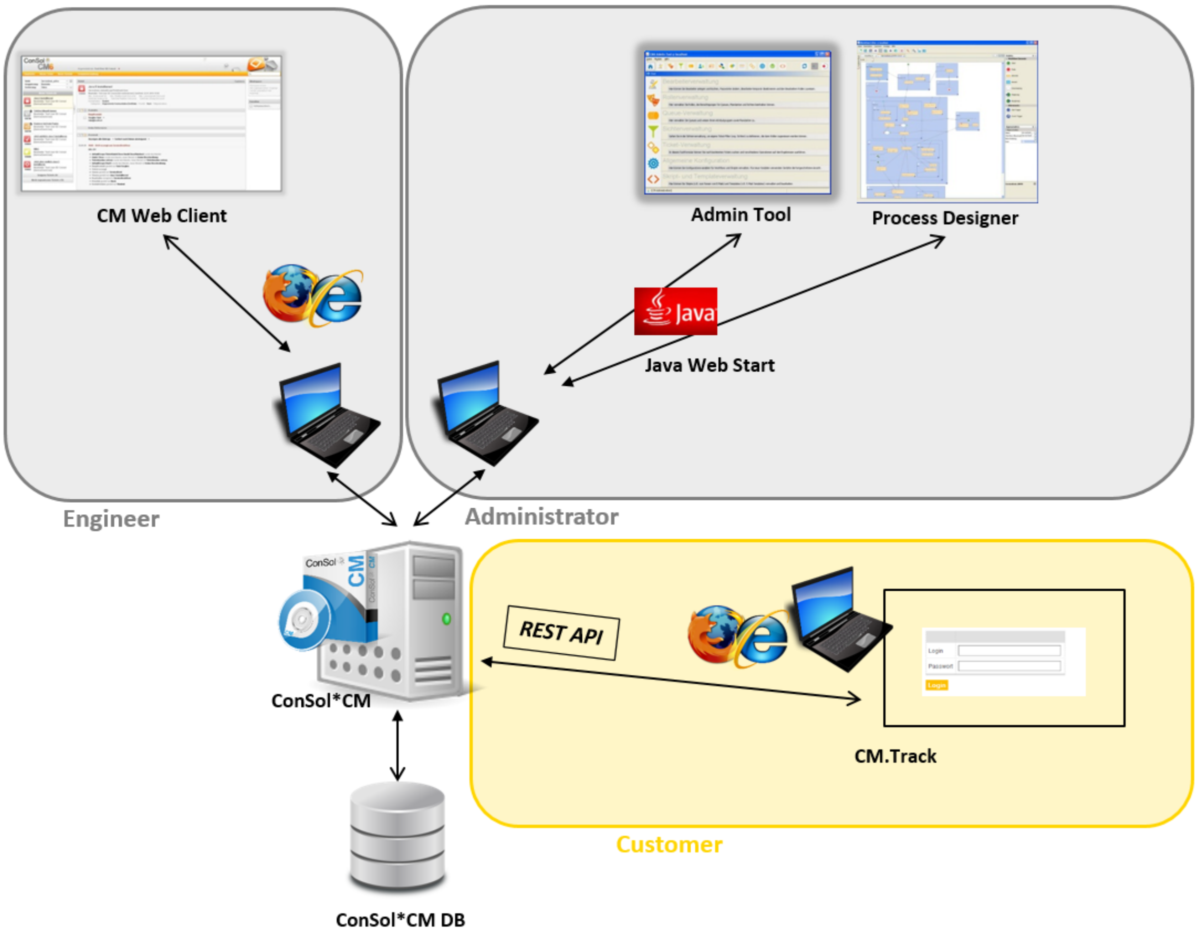


Fig. 1: ConSol*CM System Architecture with CM.Track

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 to a ticket.
- Search the FAQs for solutions.

See the following sections for topics which concern CM.Track:

- General system access to CM.Track for customers:
  See section System Access for CM.Track Users (Customers).
- Customer authentication modes:
  See section CM.Track: Authentication Modes for the Portal.
- Using the portal for FAQs:
  See section FAQs in CM.Track.

31.2 System Access for CM.Track Users (Customers)

- Precondition
- CM.Track Technical Background
- General Principle of System Access via CM.Track
- Defining the User Profiles/Access Permissions for CM.Track
- Defining the Data Object Group Fields for CM.Track Login and Password
- Granting Access to CM.Track for Customers
- Customer Login to the System
- Extended Customer Permissions to See Company Tickets

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.

Information:

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 CM licence for every customer.

31.2.1 Precondition

Technically, CM.Track is part of every default shipment of ConSol*CM, so there are no new files that have to be deployed. However, the default function set is rather rudimentary and the pages have a rather plain 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) which is 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.

31.2.2 CM.Track Technical Background

The portal CM.Track is based on the *REST API* of ConSol*CM. Please refer to the separate document *ConSol*CM REST API Documentation* for details.

31.2.3 General Principle of System Access via CM.Track

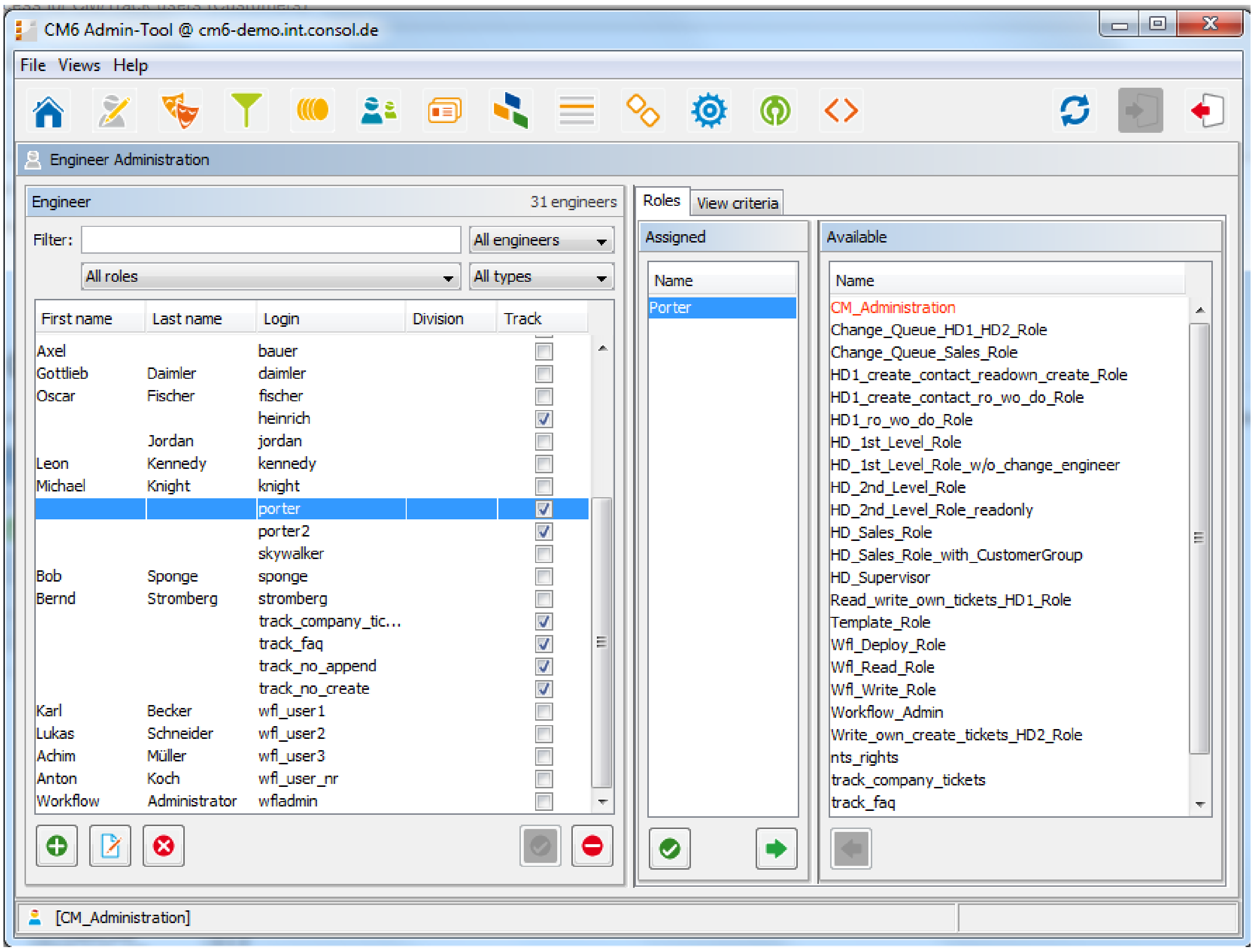
A customer who wants to or should have access to your ConSol*CM system using the portal CM.Track has to have a login and a password. Both can be initially provided by the engineer who edits the customer data using the CM Web Client, or the values can be imported automatically into the database.

The fields for the login and password of customers are data object group fields which are defined like any other data object group field and which have special annotations. If you are not familiar with data object group fields, please refer to section Data Object Group Field Management and GUI Design.

The access permissions of the customer are defined by assigning a user profile to the customer's account. The user profiles are managed by the ConSol*CM administrator using the Admin-Tool.

31.2.4 Defining the User Profiles/Access Permissions for CM. Track

As one of the first steps you have to define 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 Engineer Administration for details), but is marked as *Track*.



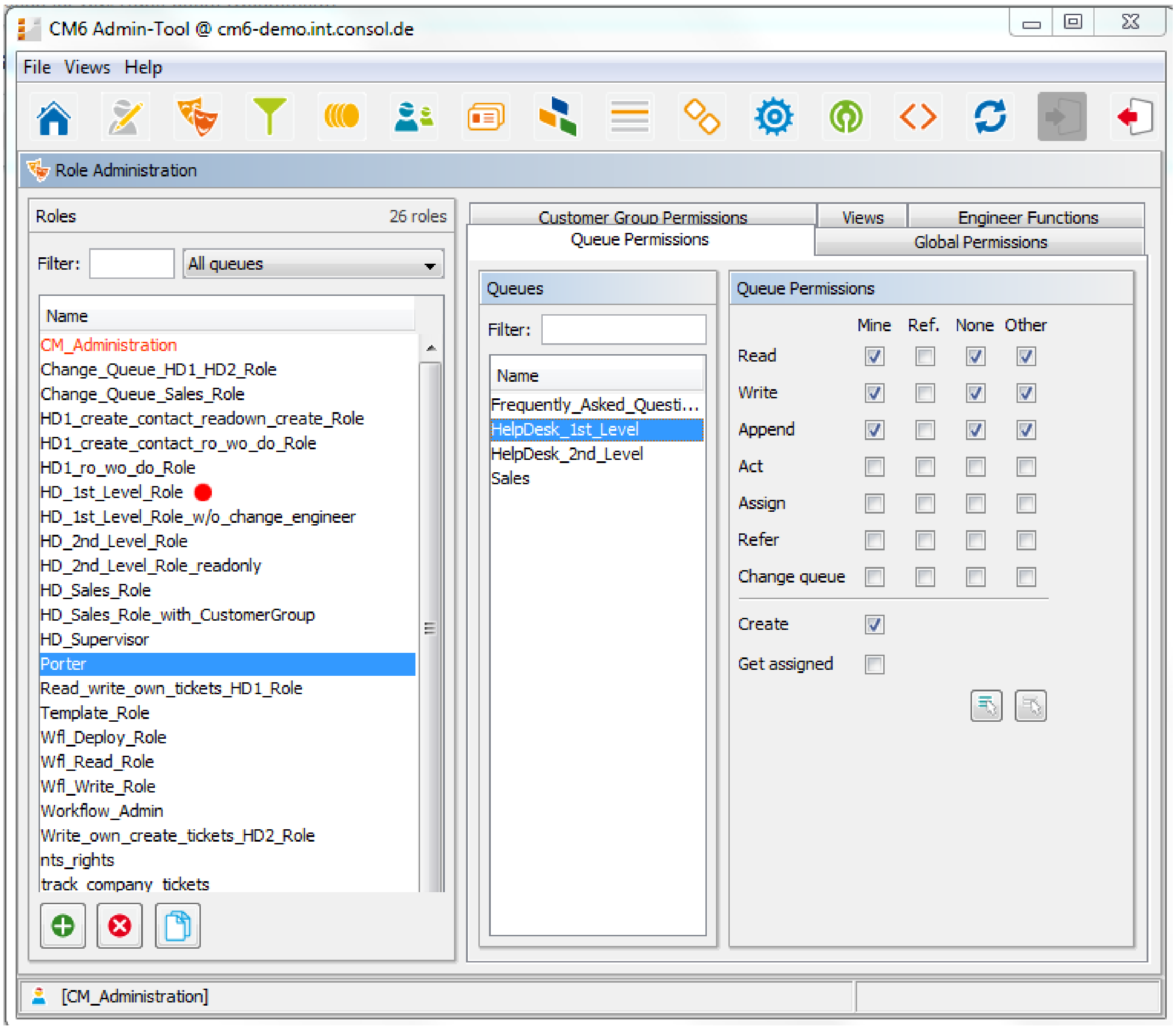
First name	Last name	Login	Division	Track
Axel		bauer		☐
Gottlieb	Daimler	daimler		☐
Oscar	Fischer	fischer		☐
		heinrich		✓
	Jordan	jordan		☐
Leon	Kennedy	kennedy		☐
Michael	Knight	knight		☐
		porter		✓
		porter2		✓
		skywalker		☐
Bob	Sponge	sponge		☐
Bernd	Stromberg	stromberg		☐
		track_company_tic...		✓
		track_faq		✓
		track_no_append		✓
		track_no_create		✓
Karl	Becker	wfl_user1		☐
Lukas	Schneider	wfl_user2		☐
Achim	Müller	wfl_user3		☐
Anton	Koch	wfl_user_nr		☐
Workflow	Administrator	wfladmin		☐

Name (Assigned)
Porter

Name (Available)
CM_Administration
Change_Queue_HD1_HD2_Role
Change_Queue_Sales_Role
HD1_create_contact_readown_create_Role
HD1_create_contact_ro_wo_do_Role
HD1_ro_wo_do_Role
HD_1st_Level_Role
HD_1st_Level_Role_w/o_change_engineer
HD_2nd_Level_Role
HD_2nd_Level_Role_readonly
HD_Sales_Role
HD_Sales_Role_with_CustomerGroup
HD_Supervisor
Read_write_own_tickets_HD1_Role
Template_Role
Wfl_Deploy_Role
Wfl_Read_Role
Wfl_Write_Role
Workflow_Admin
Write_own_create_tickets_HD2_Role
nts_rights
track_company_tickets
track_faq

Fig. 1: ConSol*CM Admin-Tool - CM.Track: User Profile Name

The user profile is assigned one or more roles to define the access permissions to queues and customer groups. For example you can set up a user profile (engineer) *Porter* that has the role *Porter*. This role has read/write/append permissions to the queue *Helpdesk_1st_Level*. For a detailed introduction to role administration, please refer to section Role Administration.


CM6 Admin-Tool @ cm6-demo.int.consol.de

File Views Help

Role Administration

Roles 26 roles

Filter: All queues

Name
CM_Administration
Change_Queue_HD1_HD2_Role
Change_Queue_Sales_Role
HD1_create_contact_readown_create_Role
HD1_create_contact_ro_wo_do_Role
HD1_ro_wo_do_Role
HD_1st_Level_Role ●
HD_1st_Level_Role_w/o_change_engineer
HD_2nd_Level_Role
HD_2nd_Level_Role_readonly
HD_Sales_Role
HD_Sales_Role_with_CustomerGroup
HD_Supervisor
Porter
Read_write_own_tickets_HD1_Role
Template_Role
Wfl_Deploy_Role
Wfl_Read_Role
Wfl_Write_Role
Workflow_Admin
Write_own_create_tickets_HD2_Role
nts_rights
track_company_tickets

Customer Group Permissions | Views | Engineer Functions | Queue Permissions | Global Permissions

Queues

Filter:

Name
Frequently_Asked_Questi...
HelpDesk_1st_Level
HelpDesk_2nd_Level
Sales

Queue Permissions

	Mine	Ref.	None	Other
Read	✓		✓	✓
Write	✓		✓	✓
Append	✓		✓	✓
Act				
Assign				
Refer				
Change queue				
Create	✓			
Get assigned				

[CM_Administration]

Fig. 2: ConSol*CM Admin-Tool - CM.Track: User Profile - Role

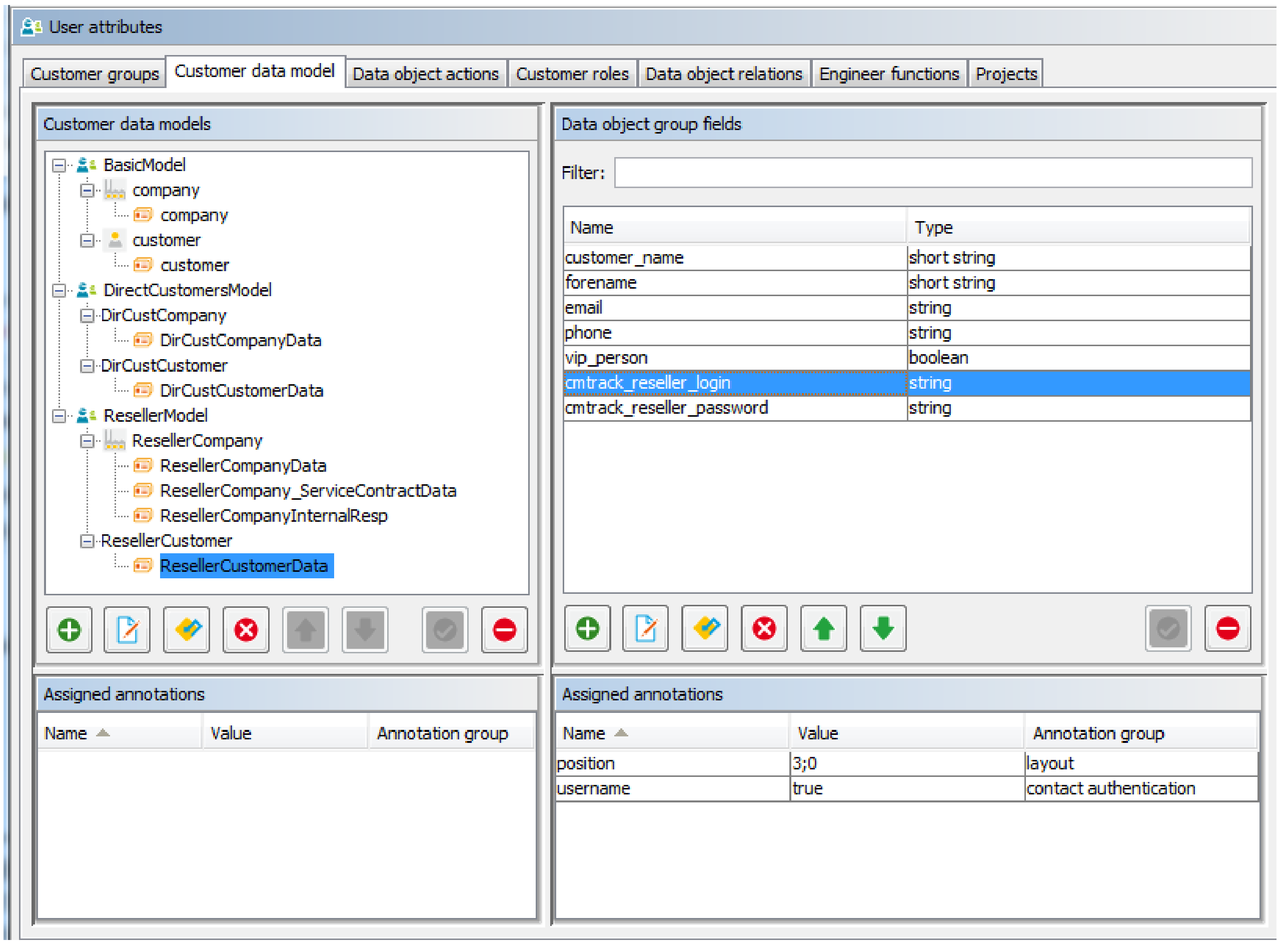
That way, a customer with the CM.Track user profile *Porter* can only see and add comments to tickets from this queue. Another user profile might have access to *Sales* tickets and/or to an *FAQ* queue.

31.2.5 Defining the Data Object Group Fields for CM.Track Login and Password

The fields for login and password for a customer are regular data object group fields on contact level. Please see section Setting up the Customer Data Model for an introduction to data object group 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!):

- One field for the login has to be created, annotation *username = true*.

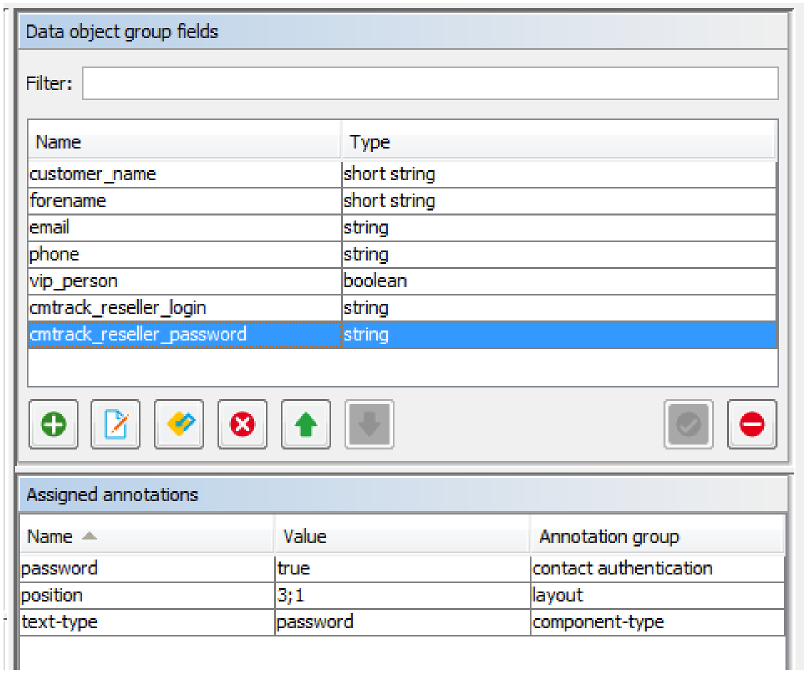


Name	Type
customer_name	short string
forename	short string
email	string
phone	string
vip_person	boolean
cmtrack_reseller_login	string
cmtrack_reseller_password	string

Name	Value	Annotation group
position	3;0	layout
username	true	contact authentication

Fig. 3: ConSol*CM Admin-Tool - CM.Track: Annotation for Login

- 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 CM Web Client, not the plain text password.



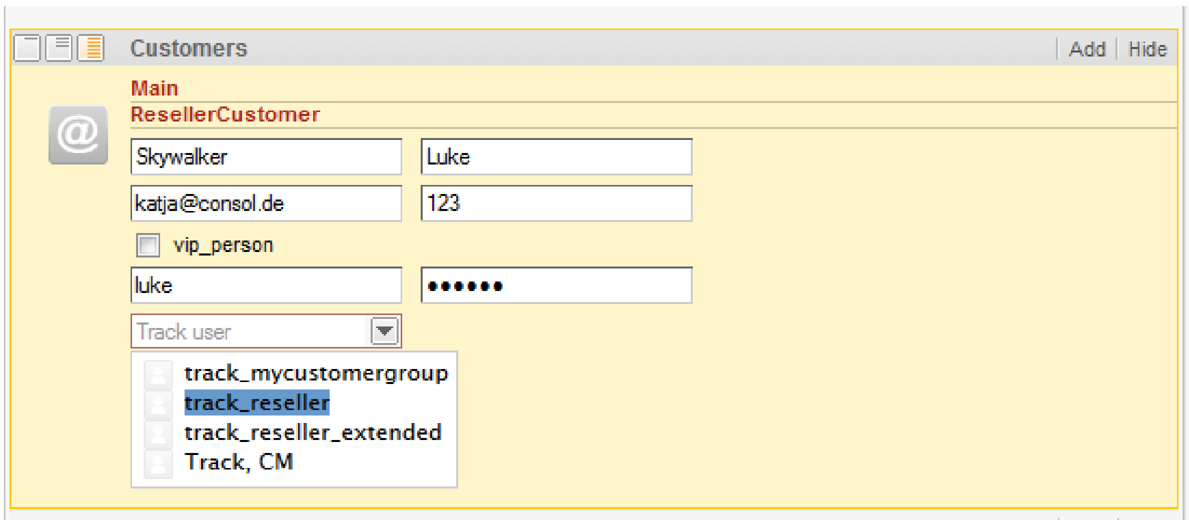
Name	Type
customer_name	short string
forename	short string
email	string
phone	string
vip_person	boolean
cmtrack_reseller_login	string
cmtrack_reseller_password	string

Name	Value	Annotation group
password	true	contact authentication
position	3;1	layout
text-type	password	component-type

Fig. 4: ConSol*CM Admin-Tool - CM.Track: Annotation for Password

31.2.6 Granting Access to CM.Track for Customers

The engineer who works with the CM 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.



Skywalker
Luke
katja@consol.de
123
vip_person
luke
••••••
Track user
track_mycustomergroup
track_reseller
track_reseller_extended
Track, CM

Fig. 5: ConSol*CM Web Client - CM.Track Users

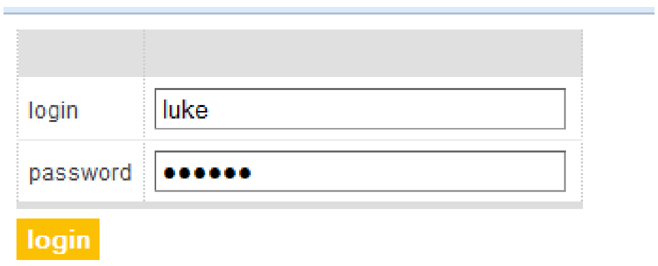
31.2.7 Customer Login to the System

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.

Technically, there are two ways for the user authentication:

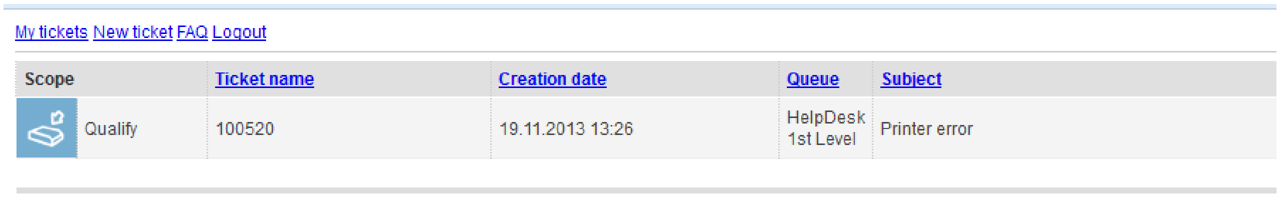
- simple authentication
- LDAP authentication

Please refer to section CM.Track: Authentication Modes for the Portal for details.



login	luke
password	••••••

Fig. 6: ConSol*CM.Track - Customer Login


My tickets New ticket FAQ Logout

Scope	Ticket name	Creation date	Queue	Subject
Qualify	100520	19.11.2013 13:26	HelpDesk 1st Level	Printer error

Fig. 7: ConSol*CM.Track - Ticket List

31.2.8 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 right *Access tickets of the own company* under *Track User Permissions*. Please refer to the section Role Administration for a detailed explanation.

31.3 CM.Track: Authentication Modes for the Portal

- Introduction to Authentication Modes in CM.Track
- Definition of the CM.Track Authentication Mode
- DATABASE Authentication Mode
    - System Property for DATABASE Authentication Mode
    - Data Object Group Fields for Contact Login
- LDAP Authentication Mode
    - System Property for LDAP Authentication Mode
    - System Properties Defining the LDAP Server(s)
    - Data Object Group Field for Contact Login
- Mixed Authentication Mode
    - System Property for Mixed Authentication Mode
    - System Properties Defining the LDAP Server(s)
    - Data Object Group Field for Contact Login
- Logging of LDAP Login Attempts in CM.Track
- Using LDAPs for Authentication

31.3.1 Introduction to Authentication Modes in CM.Track

Contacts who log in to the ConSol*CM portal (CM.Track) use their login and password. Both are data object group fields in the contact data.

There are three possible authentication modes:

- Against the ConSol*CM database.
- Against an LDAP server.
- Against an LDAP server and the ConSol*CM database.
  The order can be configured. This is called *Mixed Mode*.

31.3.2 Definition of the CM.Track Authentication Mode

The authentication mode is determined 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 Appendix C (System Properties)) and their respective system behaviors are:

- **DATABASE**
  Attempting a database login, if the unit has a database password.
- **LDAP**
  Trying authentication using the available LDAP server(s), if an LDAP ID is provided.

- **LDAP,DATABASE**
  First attempt is authentication using the available LDAP server(s), if an LDAP ID is provided. On failure trying a database login, if the unit has a database password.
- **DATABASE,LDAP**
  First attempt is a database login, if the unit has a database password. On failure trying authentication using the available LDAP server(s), if an LDAP ID is provided.

The values are case insensitive, commas and whitespace are ignored.

31.3.3 DATABASE Authentication Mode

System Property for DATABASE Authentication Mode

Set the system property *cmas-core-security*, *contact.authentication.method* to *DATABASE* (this is the default value).

Data Object Group Fields for Contact Login

Two data object group fields for the contact data are required:

- Login
- Password

Please see section System Access for CM.Track Users (Customers) for a detailed explanation.

31.3.4 LDAP Authentication Mode

System Property for LDAP Authentication Mode

Set the system property *cmas-core-security*, *contact.authentication.method* to *LDAP*.

System Properties Defining the LDAP Server(s)

The LDAP servers can be defined using the following configuration properties from the module *cmas-core-security*.

{name} is a string that you can choose to distinguish LDAP servers. It always has to be set, even if only one LDAP server is configured. You should use a simple string for the {name}, which does not contain any keywords like *internal* or *external* and which does not contain special characters.

- **ldap.initialcontextfactory**
  This is an already existing global property. If it is not set, *com.sun.jndi.ldap.LdapCtxFactory* is being used as a value.
- **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, in case 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, in case 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=consol,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*.

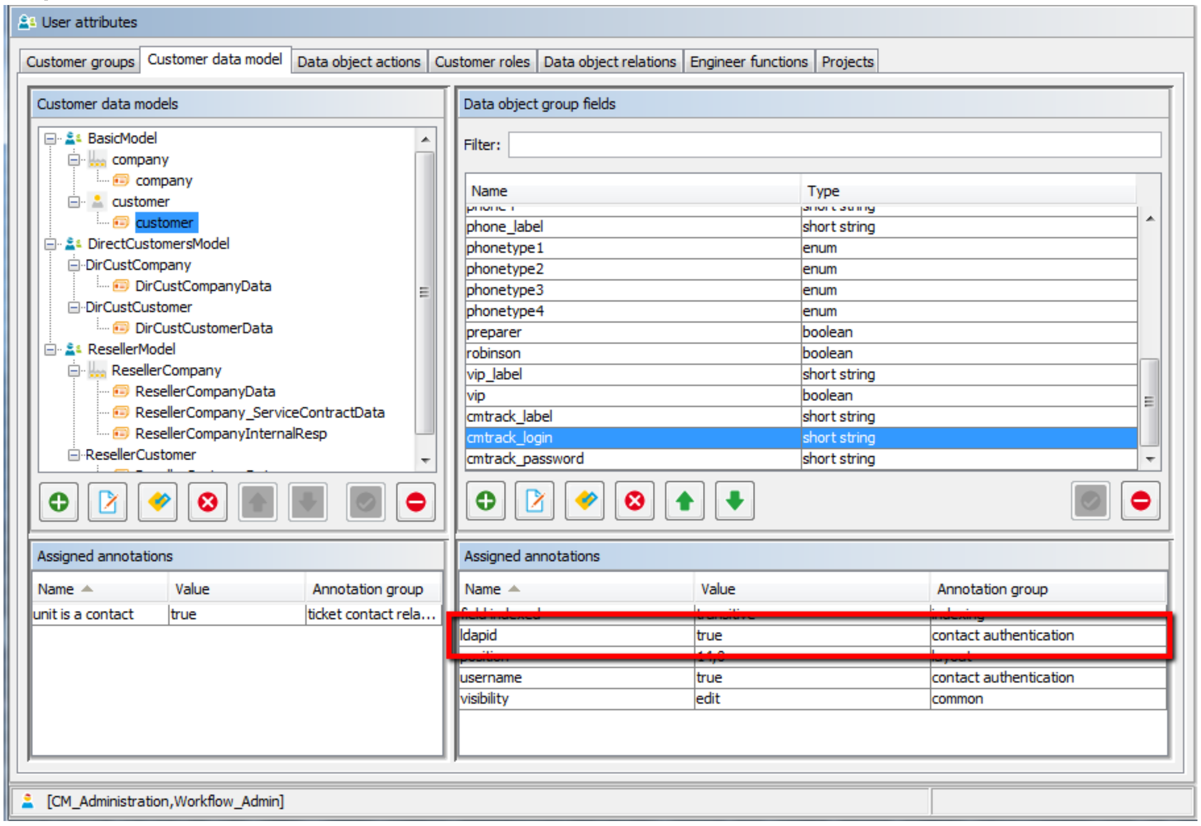
A change of any of the above configuration properties does not require a server restart and is propagated to all cluster nodes. The use of the placeholder {name} allows to define several different LDAP servers.

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).

Data Object Group Field for Contact Login

Besides the annotation *username* = *true*, the data object group field which is used for the CM.Track user name (login) has to have an additional annotation:

- **ldapid**



Customer data models:

```
BasicModel
  company
    company
  customer
    customer
DirectCustomersModel
  DirCustCompany
    DirCustCompanyData
  DirCustCustomer
    DirCustCustomerData
ResellerModel
  ResellerCompany
    ResellerCompanyData
    ResellerCompany_ServiceContractData
    ResellerCompanyInternalResp
  ResellerCustomer
```

Data object group fields:

Name	Type
phone1	short string
phone_label	short string
phonetype1	enum
phonetype2	enum
phonetype3	enum
phonetype4	enum
preparer	boolean
robinson	boolean
vip_label	short string
vip	boolean
cmtrack_label	short string
cmtrack_login	short string
cmtrack_password	short string

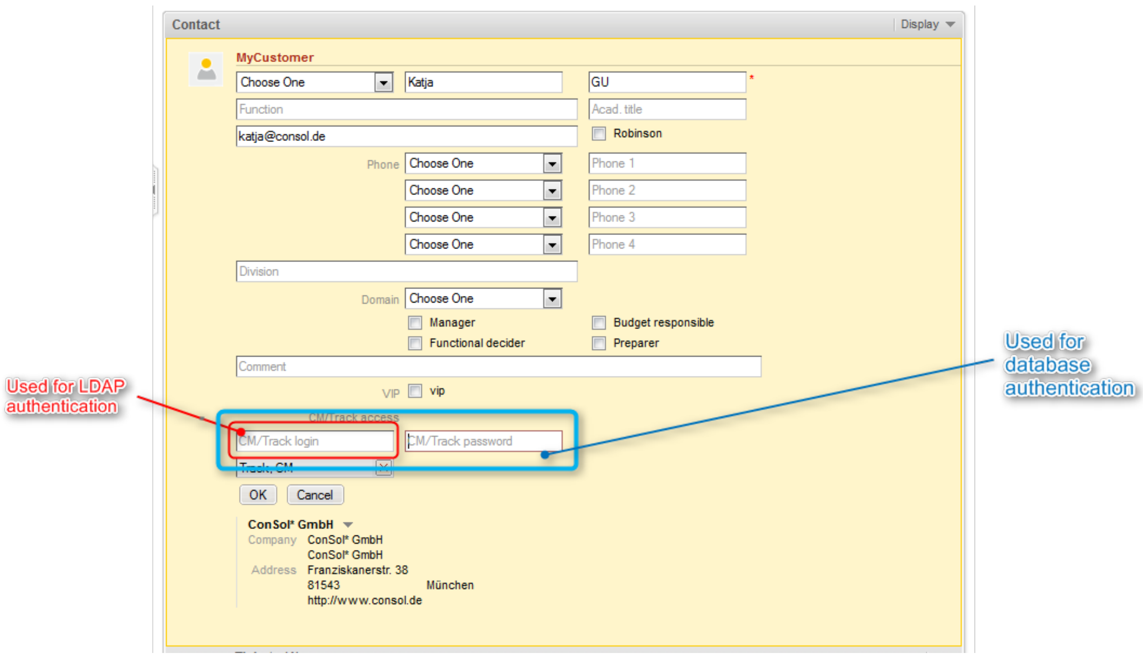
Assigned annotations (customer):

Name	Value	Annotation group
unit is a contact	true	ticket contact rela...

Assigned annotations (cmtrack_login):

Name	Value	Annotation group
fieldindexed	transitive	indexing
ldapid	true	contact authentication
position	14;0	layout
username	true	contact authentication
visibility	edit	common

Fig. 1: ConSol*CM Admin-Tool - Data Object Group Field for LDAP Authentication of CM.Track Users


Contact
Display
Used for LDAP authentication
Used for database authentication
OK
Cancel

MyCustomer

Choose One	Katja	GU
Function		Acad. title
katja@consol.de		Robinson
Phone	Choose One	Phone 1
	Choose One	Phone 2
	Choose One	Phone 3
	Choose One	Phone 4
Division		
Domain	Choose One	
	Manager	Budget responsible
	Functional decider	Preparer
Comment		
VIP	vip	
CM/Track access		
CM/Track login	CM/Track password	
Track, CM		

ConSol* GmbH
Company ConSol* GmbH
ConSol* GmbH
Address Franziskanerstr. 38
81543 München
http://www.consol.de

Fig. 2: ConSol*CM Web Client - Field (Red) for LDAP ID in Contact Data

31.3.5 Mixed Authentication Mode

System Property for Mixed Authentication Mode

Set the system property *cmas-core-security*, *contact.authentication.method* depending on the desired order of authentication instances:

- LDAP,DATABASE
- DATABASE,LDAP

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]
```

System Properties Defining the LDAP Server(s)

See respective paragraph in section *LDAP Authentication Mode*.

Data Object Group Field for Contact Login

See respective paragraph in section *LDAP Authentication Mode*.

31.3.6 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.

31.3.7 Using LDAPs for Authentication

The LDAPs authentication for CM.Track follows the same principle as using LDAPs for the authentication in the CM Web Client. Please refer to section LDAP Authentication in the CM Web Client, LDAPS.

31.4 FAQs in CM.Track

- Introduction to FAQs in CM.Track
- Configuring the ConSol*CM System to Allow FAQ Search in CM.Track
- FAQ Search in CM.Track from a Customer's Point of View
- More Complex Solutions for Managing FAQs
    - Using Two FAQ Queues: FAQ Management and Active FAQs
    - Setting Up Two (or More) Parallel FAQ Environments Using Track Users

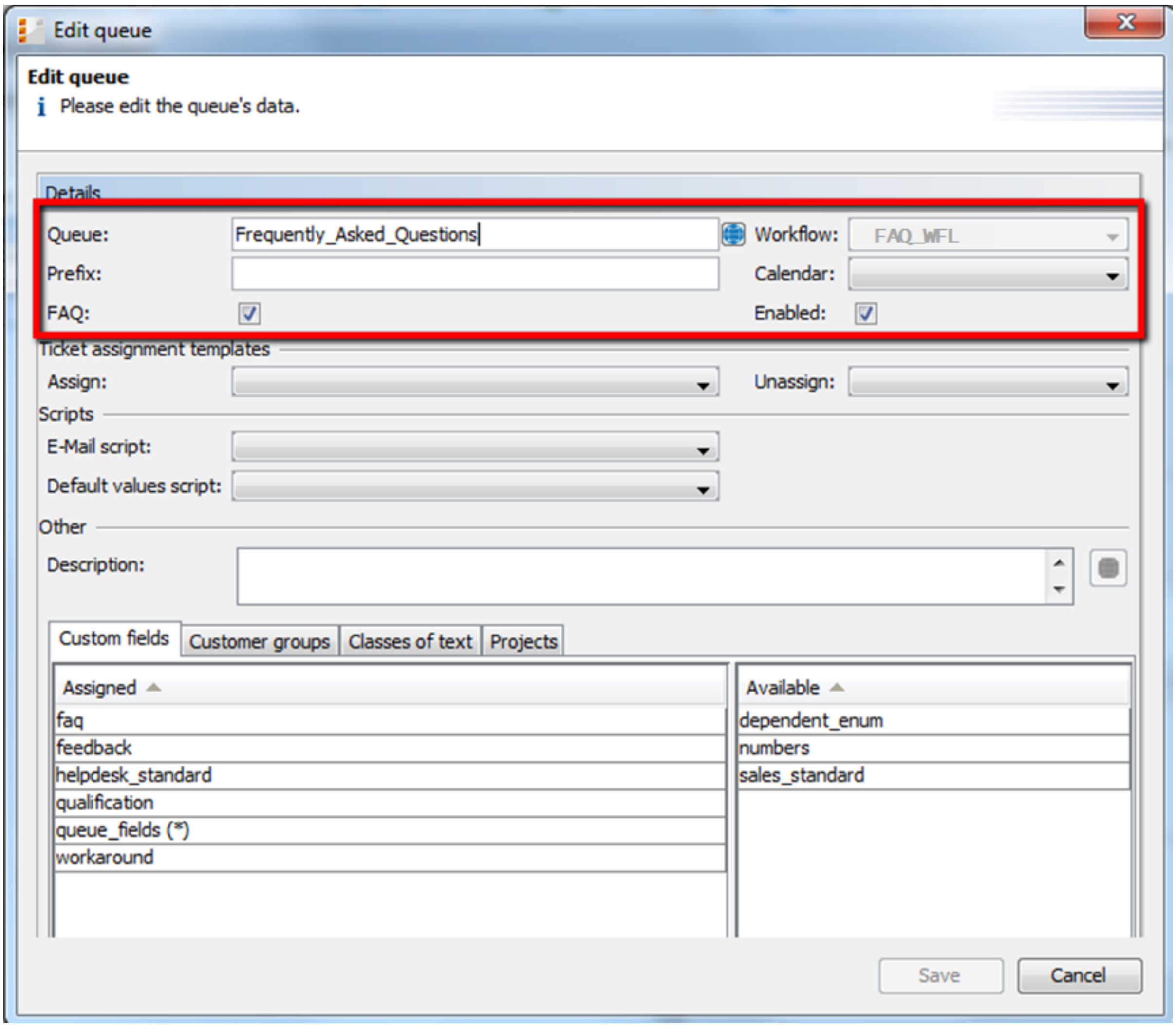
31.4.1 Introduction to FAQs in CM.Track

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 solution already. Only if they do not find any help, they will contact the service desk and/or open a new ticket. This saves time for both customer and service team. It might also be employed in other environments where you would like to offer this service.

According to ConSol*CM standard, every FAQ is treated as a ticket. The queue(s) which should be available as FAQ queue(s) via CM.Track have to be defined as special FAQ queues, because usually customers are 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 queue(s). Here, only read access has to be granted.

31.4.2 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 (check box *FAQ*).

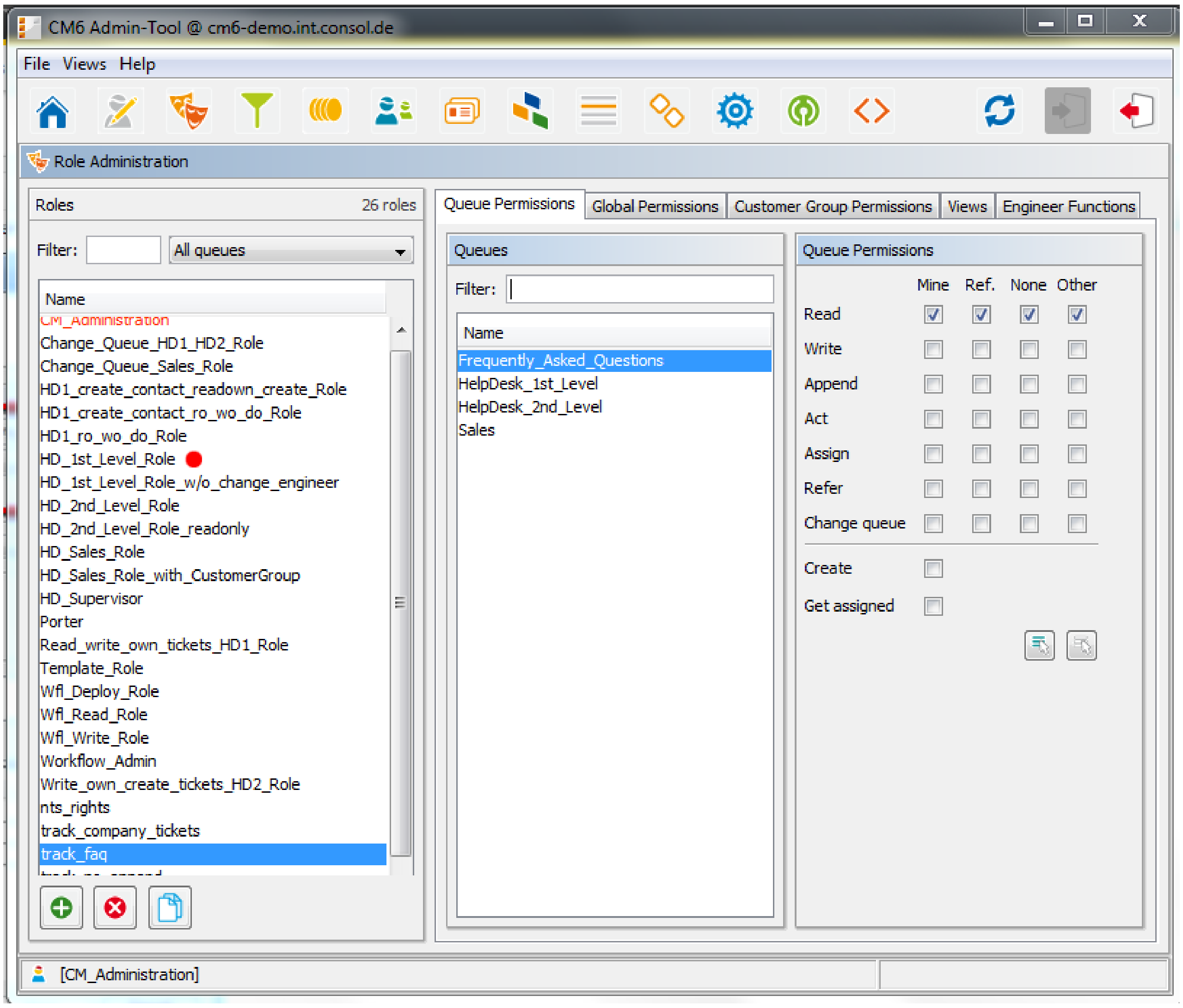


Queue: Frequently_Asked_Questions
Workflow: FAQ_WFL
Prefix:
Calendar:
FAQ: ✓
Enabled: ✓

Assigned	Available
faq	dependent_enum
feedback	numbers
helpdesk_standard	sales_standard
qualification	
queue_fields (*)	
workaround	

Fig. 1: ConSol*CM Admin-Tool - Queue Administration

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.



Roles 26 roles

Filter: All queues

Name
CM_Administration
Change_Queue_HD1_HD2_Role
Change_Queue_Sales_Role
HD1_create_contact_readown_create_Role
HD1_create_contact_ro_wo_do_Role
HD1_ro_wo_do_Role
HD_1st_Level_Role ●
HD_1st_Level_Role_w/o_change_engineer
HD_2nd_Level_Role
HD_2nd_Level_Role_readonly
HD_Sales_Role
HD_Sales_Role_with_CustomerGroup
HD_Supervisor
Porter
Read_write_own_tickets_HD1_Role
Template_Role
Wfl_Deploy_Role
Wfl_Read_Role
Wfl_Write_Role
Workflow_Admin
Write_own_create_tickets_HD2_Role
nts_rights
track_company_tickets
track_faq

Queues

Filter:

Name
Frequently_Asked_Questions
HelpDesk_1st_Level
HelpDesk_2nd_Level
Sales

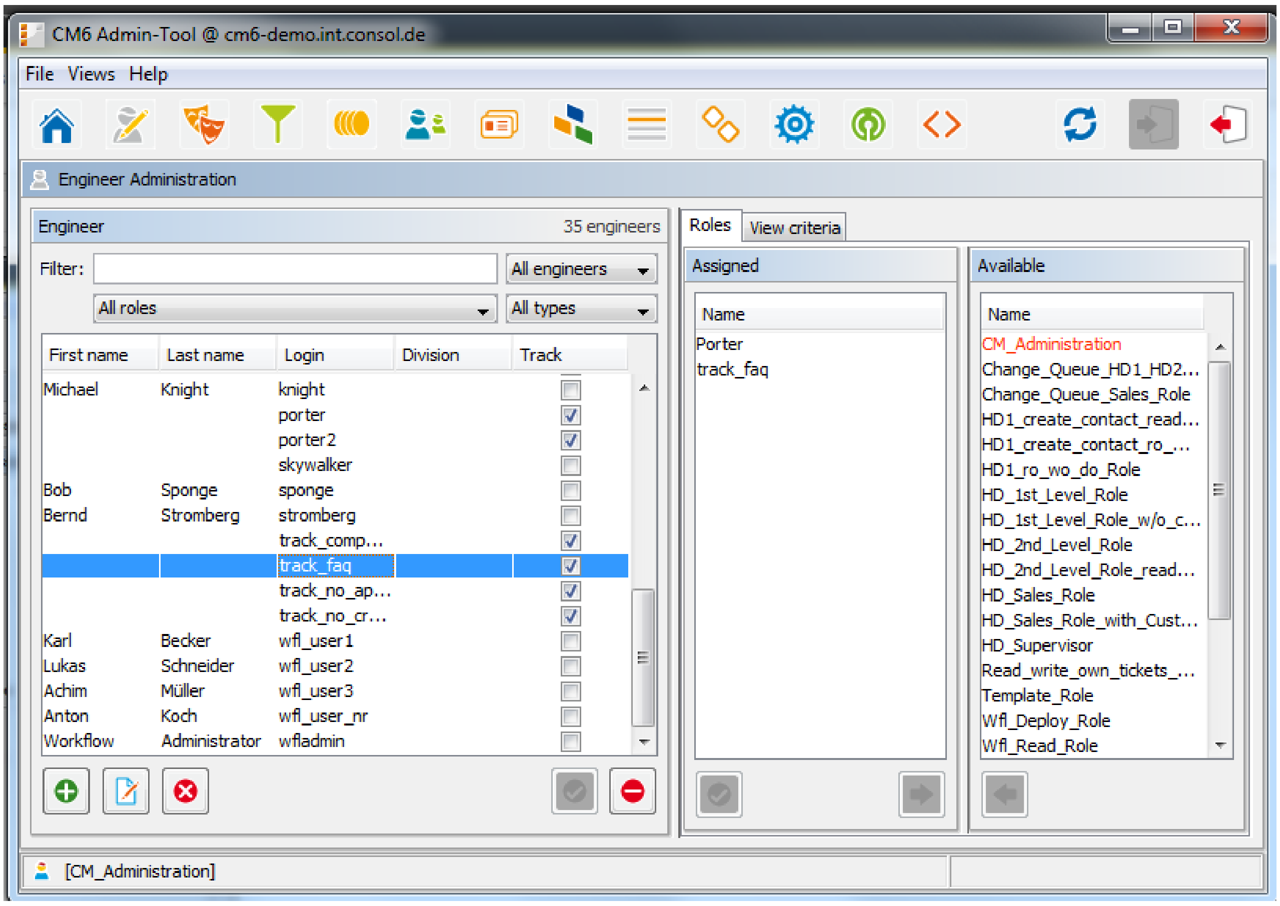
Queue Permissions

	Mine	Ref.	None	Other
Read	✓	✓	✓	✓
Write				
Append				
Act				
Assign				
Refer				
Change queue				
Create				
Get assigned				

[CM_Administration]

Fig. 2: ConSol*CM Admin-Tool - Role Administration

Then this new role has to be assigned to the user (profile) which is used as CM.Track access user (see section System Access for CM.Track Users (Customers)).



First name	Last name	Login	Division	Track
Michael	Knight	knight		☐
		porter		✓
		porter2		✓
		skywalker		☐
Bob	Sponge	sponge		☐
Bernd	Stromberg	stromberg		☐
		track_comp…		✓
		track_faq		✓
		track_no_ap…		✓
		track_no_cr…		✓
Karl	Becker	wfl_user1		☐
Lukas	Schneider	wfl_user2		☐
Achim	Müller	wfl_user3		☐
Anton	Koch	wfl_user_nr		☐
Workflow	Administrator	wfladmin		☐

Assigned:

Name
Porter
track_faq

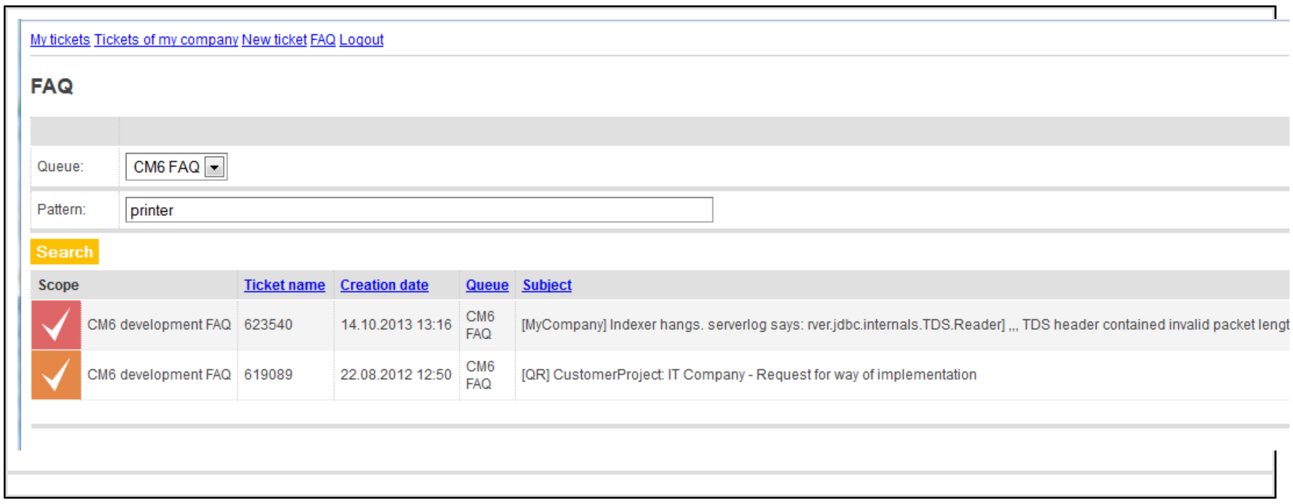
Available:

Name
CM_Administration
Change_Queue_HD1_HD2…
Change_Queue_Sales_Role
HD1_create_contact_read…
HD1_create_contact_ro_…
HD1_ro_wo_do_Role
HD_1st_Level_Role
HD_1st_Level_Role_w/o_c…
HD_2nd_Level_Role
HD_2nd_Level_Role_read…
HD_Sales_Role
HD_Sales_Role_with_Cust…
HD_Supervisor
Read_write_own_tickets_…
Template_Role
Wfl_Deploy_Role
Wfl_Read_Role

Fig. 3: ConSol*CM Admin-Tool - Engineer Administration

31.4.3 FAQ Search in CM.Track from a Customer's Point of View

A customer can search the FAQ queue using a search pattern. A list with the search results is displayed. By opening one ticket from the list, the fields of the tickets are displayed. This might be a solution as in the following example or other service information.


My tickets Tickets of my company New ticket FAQ Logout

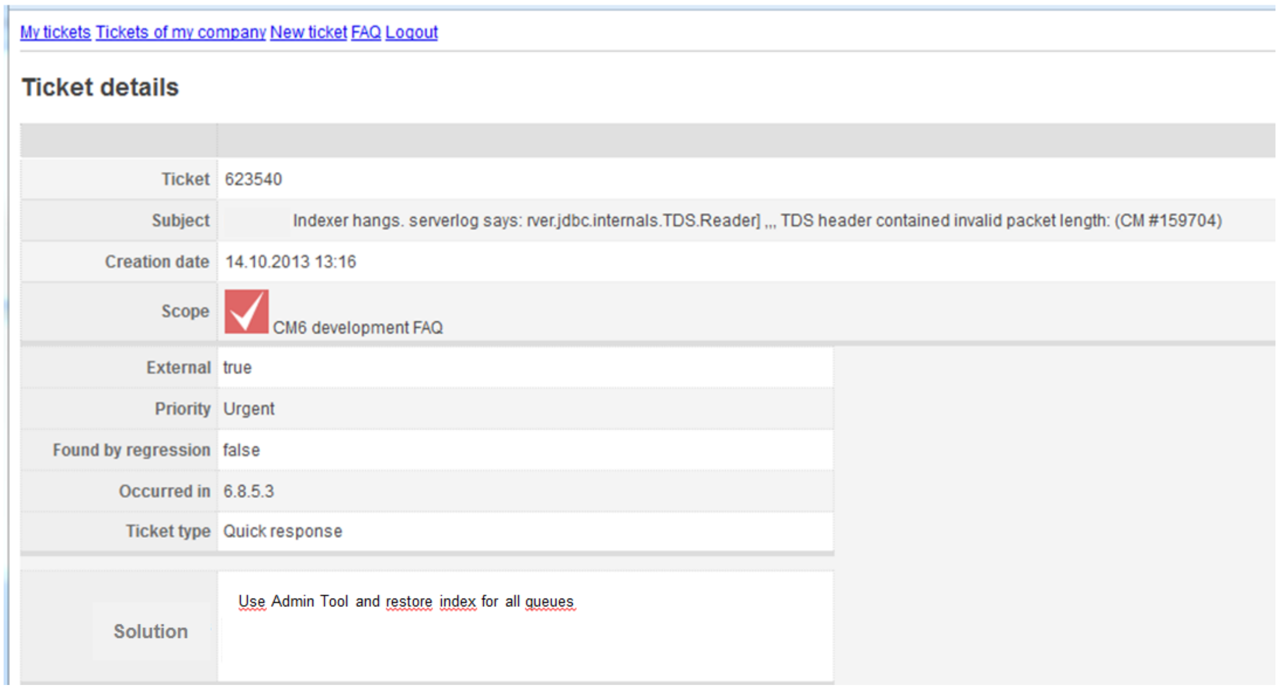
FAQ

Queue:	CM6 FAQ
Pattern:	printer

Search

Scope		Ticket name	Creation date	Queue	Subject
✓	CM6 development FAQ	623540	14.10.2013 13:16	CM6 FAQ	[MyCompany] Indexer hangs. serverlog says: rver.jdbc.internals.TDS.Reader] ,,, TDS header contained invalid packet lengt
✓	CM6 development FAQ	619089	22.08.2012 12:50	CM6 FAQ	[QR] CustomerProject: IT Company - Request for way of implementation

Fig. 4: ConSol*CM.Track - Example for FAQ Search (1)


My tickets Tickets of my company New ticket FAQ Logout

Ticket details

Ticket	623540
Subject	Indexer hangs. serverlog says: rver.jdbc.internals.TDS.Reader] ,,, TDS header contained invalid packet length: (CM #159704)
Creation date	14.10.2013 13:16
Scope	✓ CM6 development FAQ
External	true
Priority	Urgent
Found by regression	false
Occurred in	6.8.5.3
Ticket type	Quick response
Solution	Use Admin Tool and restore index for all queues

Fig. 5: ConSol*CM.Track - Example for FAQ Search (2)

31.4.4 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. Here it 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 can 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 offered 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.

32 System Overview

- System Architecture
  - Introduction to ConSol*CM System Architecture
  - Basic System Architecture
  - Components for E-Mail Interactions
  - System Architecture with Reporting Infrastructure
- Short Overview of the File Structure
  - ConSol*CM Data Directory
  - JBoss 5 Application Server File Structure
  - JBoss 7 Application Server File Structure
  - Oracle WebLogic Application Server File Structure
  - Log Files
    - Log File Types
    - Log File Structure

32.1 System Architecture

32.1.1 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 Oracle WebLogic are supported.

In this chapter, a short overview of the ConSol*CM system architecture will be provided. For a detailed description of the system, please refer to the *ConSol*CM Operations Manual*.

A detailed list of supported application servers, database systems, and other systems is given in the current *System Requirements*.

32.1.2 Basic System Architecture

ConSol*CM is a classical three-tier-architecture application. It is installed as a Java EE application in an application server. Most of the data is stored in a relational database. The clients can access the application using the web interface (Web Client or CM.Track), i.e. a web browser.

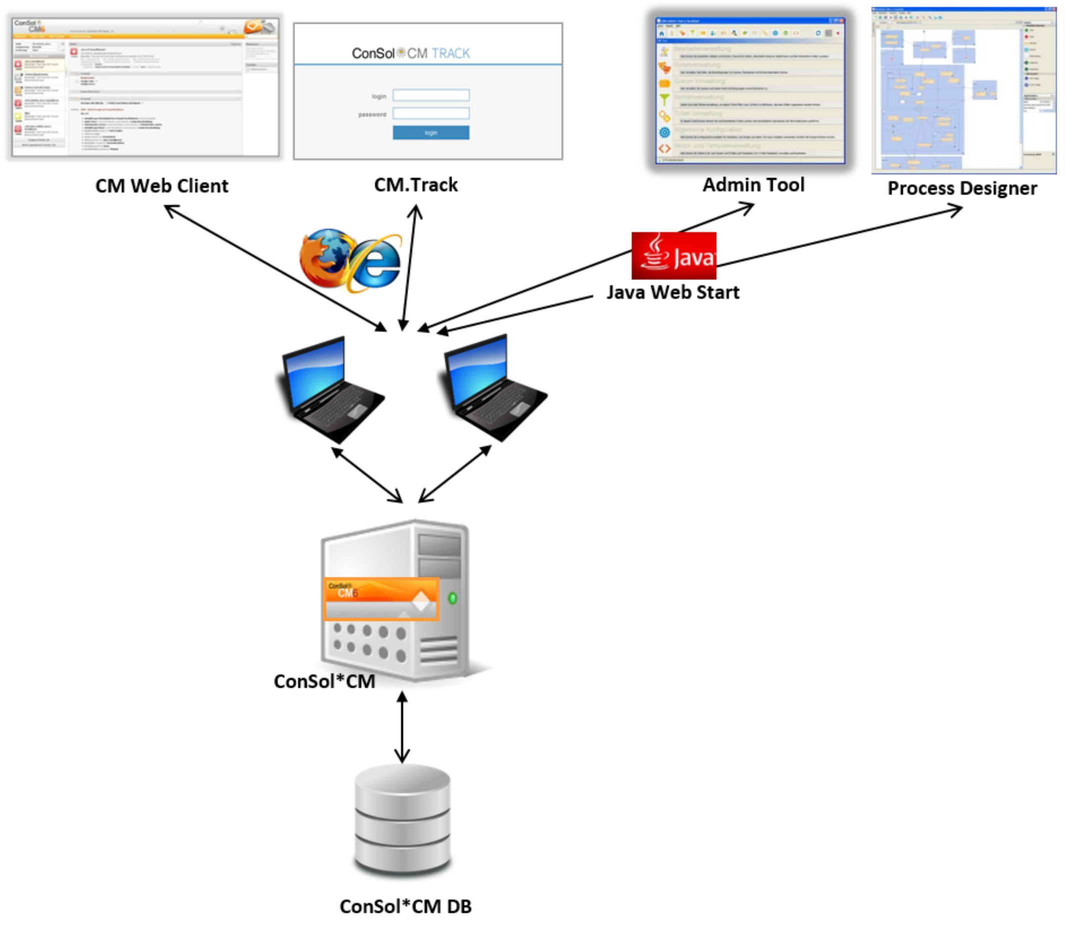


Fig. 1: ConSol*CM - Basic System Architecture

32.1.3 Components for E-Mail Interactions

ConSol*CM can retrieve e-mails from one or more e-mail servers. It acts toward such a server like a regular e-mail client, i.e. in order to establish the mail-retrieving only a network access from ConSol*CM to the e-mail server is required. POP3(S) and IMAP(S) are supported.

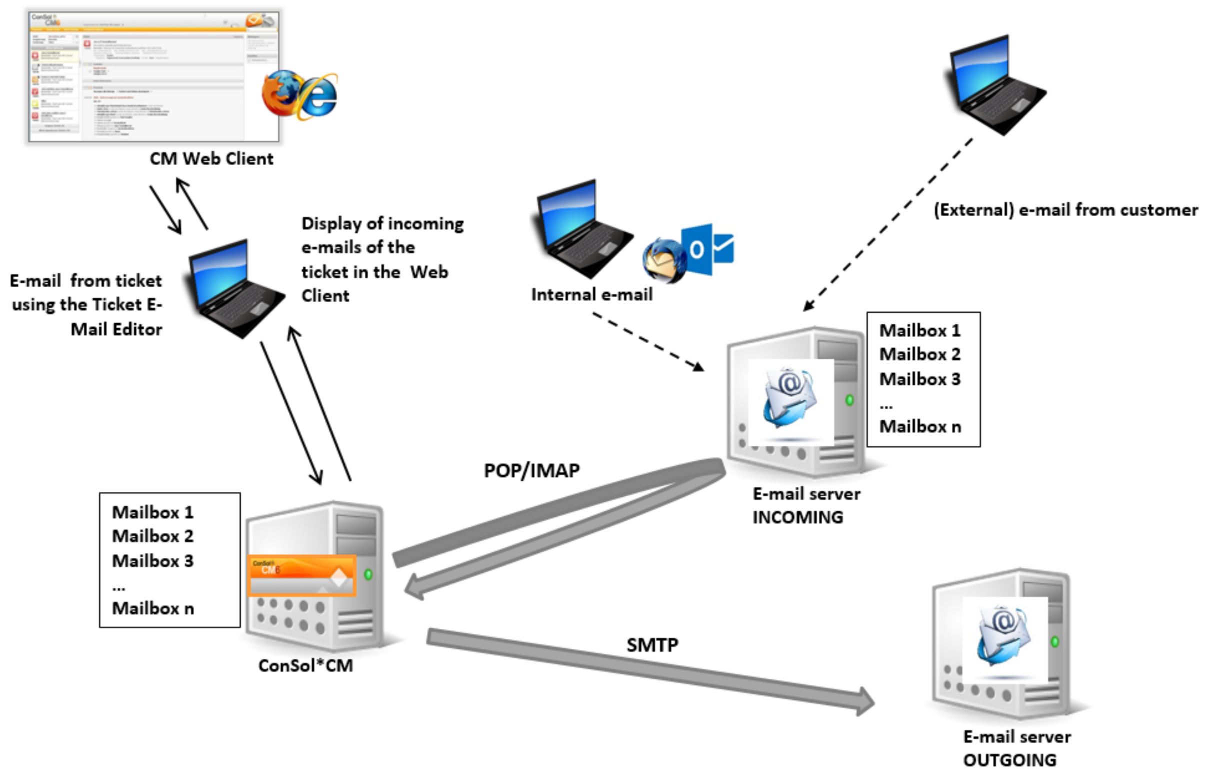


Fig. 2: ConSol*CM - E-Mail Server Interactions

32.1.4 System Architecture with Reporting Infrastructure

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). It can be installed in the same application server as ConSol*CM or in a separate application server. The synchronization of CM data with the DWH can be based on JMS (Java Messaging Service) queues or on direct messaging. For a detailed explanation, please refer to the *ConSol*CM Operations Manual*.
- **DWH** (data warehouse)
  Database (or database schema, depending on the RDBMS) that stores the integrated/pre-processed data from the ConSol*CM database.

Separate application servers for ConSol*CM and CMRF:

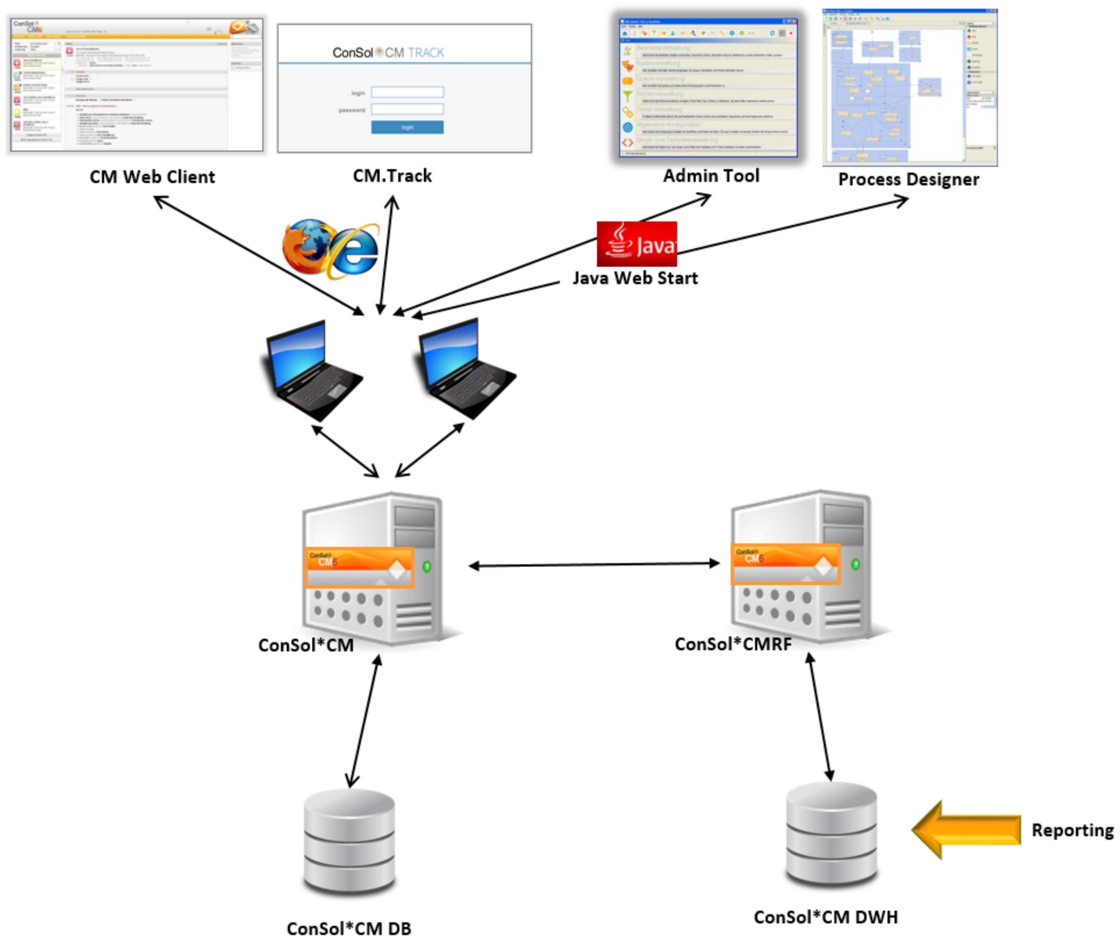


Fig. 3: ConSol*CM - Infrastructure with CMRF and DWH (2 Servers)

One application server for ConSol*CM and CMRF:

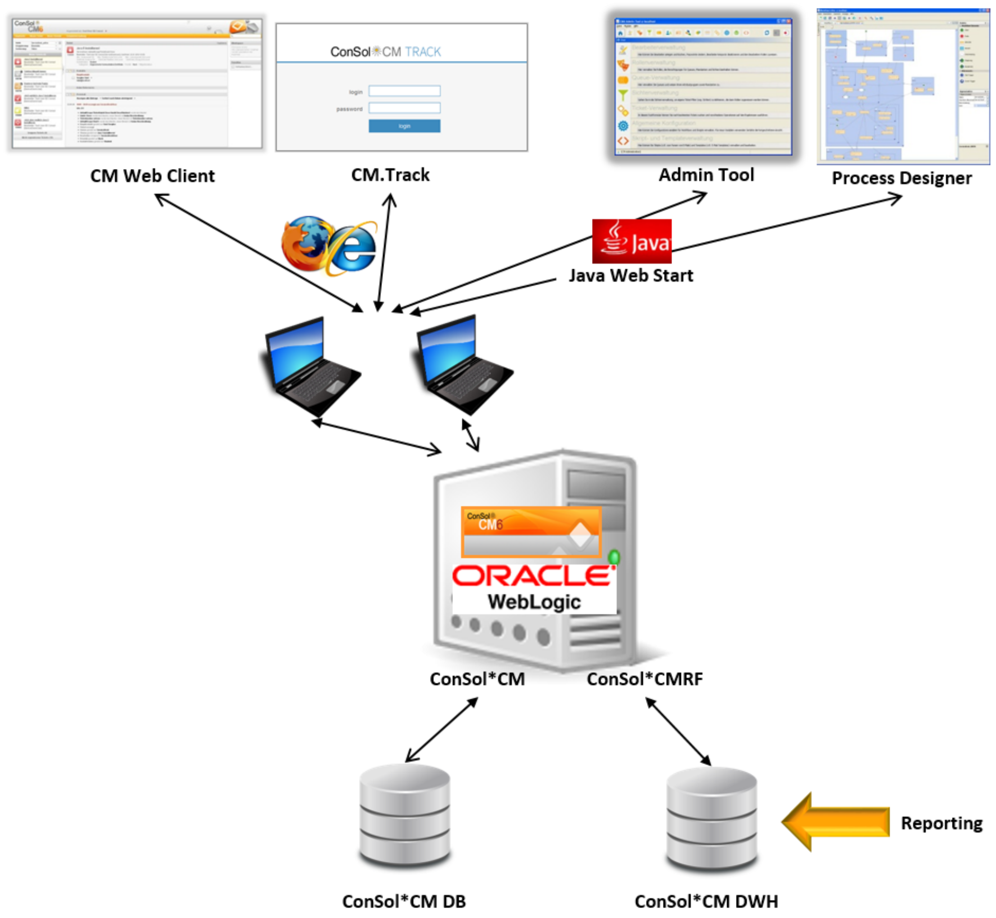


Fig. 4: ConSol*CM - Infrastructure with CMRF and DWH (1 Server)

When the DWH has been established (see section Data Warehouse (DWH) Management for details), *BI* (Business Intelligence) applications can be used to create reports, data cubes, and other reporting output formats. Please see the following example with Pentaho^{TM} BI Suite.

Separate application servers for ConSol*CM and CMRF (example: **both JBoss**):

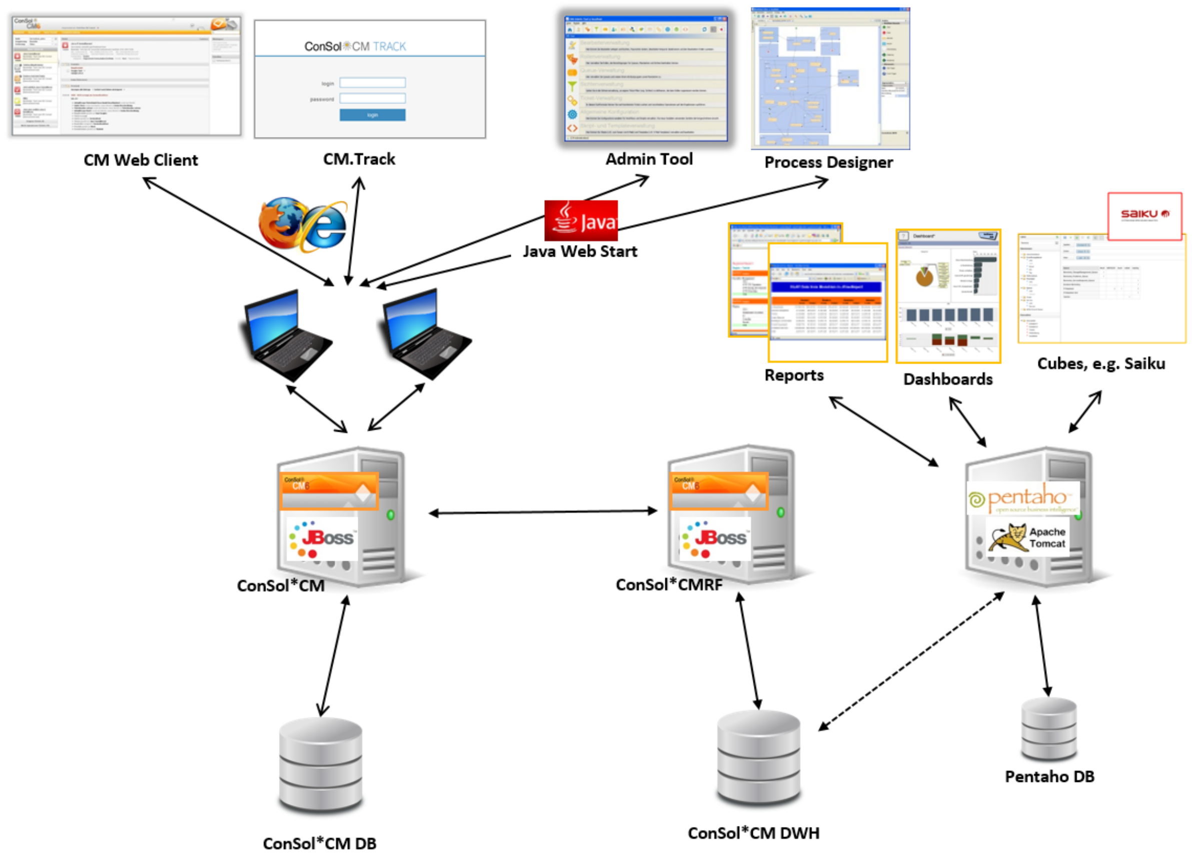


Fig. 5: ConSol*CM - Reporting Infrastructure (2 Servers, both JBoss)

One application server for ConSol*CM and CMRF (example: **WebLogic**):

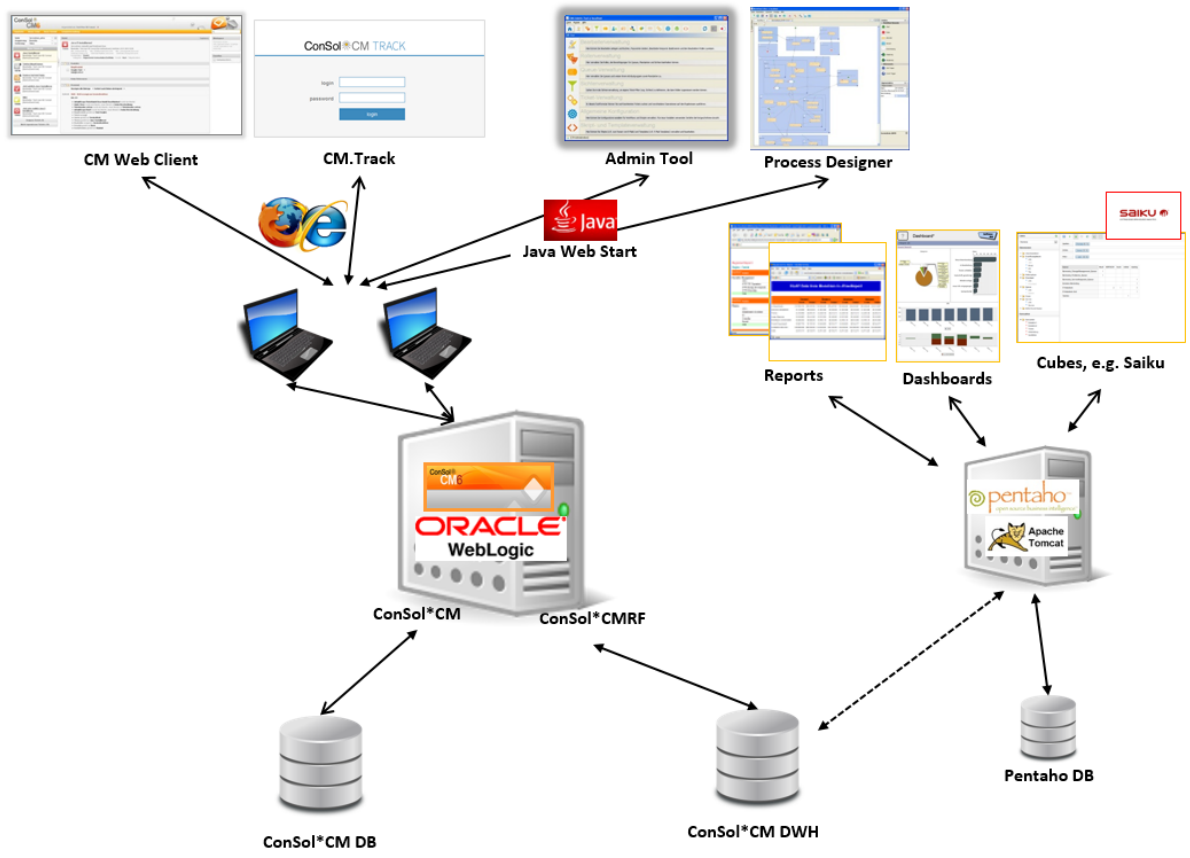


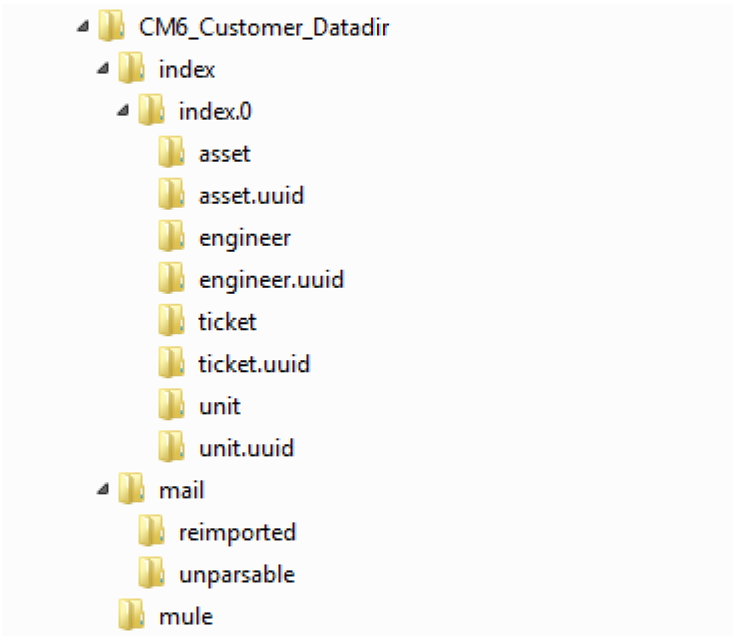
Fig. 6: ConSol*CM - Reporting Infrastructure (1 Server, WebLogic)

32.2 Short Overview of the File 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 that has been entered during system set-up.

32.2.1 ConSol*CM Data Directory

The following figure and list show an example from a Windows system:



```
CM6_Customer_Datadir
  index
    index.0
      asset
      asset.uuid
      engineer
      engineer.uuid
      ticket
      ticket.uuid
      unit
      unit.uuid
  mail
    reimported
    unparsable
  mule
```

Fig. 7: ConSol*CM - Data Directory

Example directories:

- **index**
  This is the directory where all the indexes are stored (see also section Search Configuration and Indexer Management (Tab Index)). Be sure to include it into your daily file system back-up.
  - **index.0**
    In this directory, there is a subdirectory for each required index.
- **mail**
  This directory is only relevant when the system is operated in Mule/ESB mode. In case NIMH is used, all data is stored in the database. In this directory, files that are relevant for incoming e-mails are stored.
  - **reimported**
    In this directory, e-mails are stored that had been stored in the *unparsable* directory and could then be re-imported by a manual action of the administrator.
  - **unparsable**
    In this directory, incoming e-mails that cannot be processed by the system are stored. They are listed under *E-Mail Backups* in the Admin Tool, see section Tab E-Mail Backups.

- **mule**
  This is a directory which might be used for *Mule* (internal *ESB*) data.

32.2.2 JBoss 5 Application Server File Structure

The following directories are available in a JBoss 5 installation of ConSol*CM:

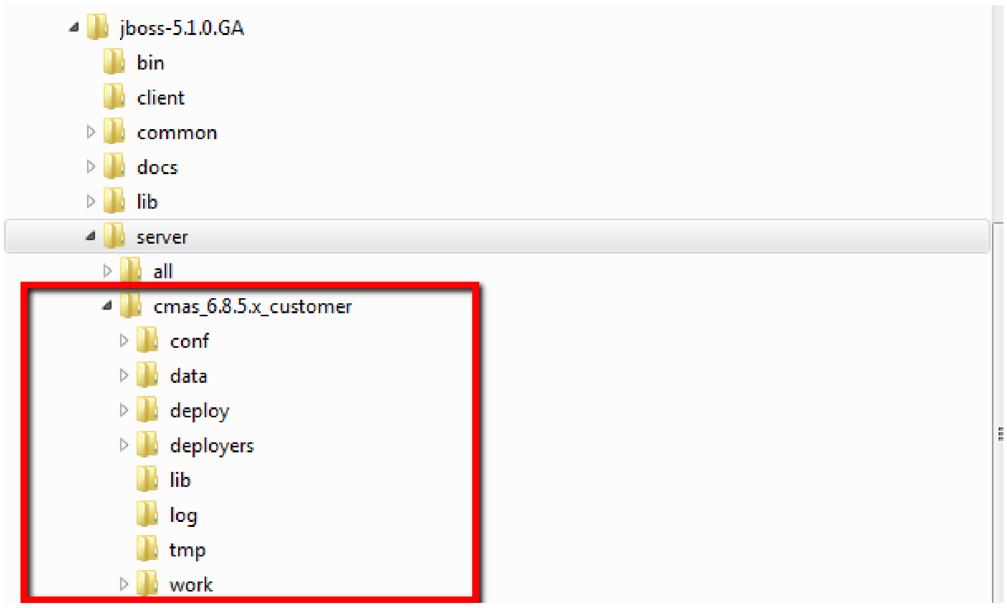


Fig. 8: ConSol*CM - File Structure in a JBoss Application Server

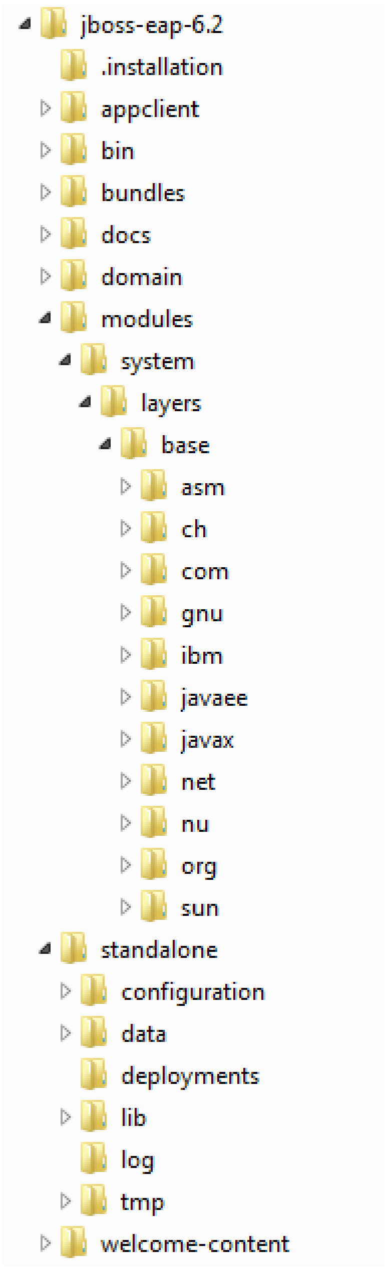
Example directories:

- **conf**
  Configuration data, e.g.:
  - **jboss-log4j**
    Log file configuration
- **data**
  Data for operation, e.g. *tx-operation* keys
- **deploy**
  Deployed data and configuration data:
  - **cm6.ear**
    Core application, *.ear* file
  - **cm-track.war**
    Application file for the portal ConSol*CM.Track
  - **cmDb-ds**
    Database connection configuration
- **deployers**
  Additional deployed application data
- **lib**
  Application-specific libraries, e.g.:
  - **mysql connector**
    In case you use MySQL as a database system.

- **log**
  Log files, see section Log Files.
- **tmp**
  Temporary data
- **work**
  Work directory with a working copy of the application server files. Can be emptied, e.g. for error analysis and/or fixing.

32.2.3 JBoss 7 Application Server File Structure

The following directories are available in a JBoss installation of ConSol*CM:



```
jboss-eap-6.2
    .installation
    appclient
    bin
    bundles
    docs
    domain
    modules
        system
            layers
                base
                    asm
                    ch
                    com
                    gnu
                    ibm
                    javaee
                    javax
                    net
                    nu
                    org
                    sun
    standalone
        configuration
        data
        deployments
        lib
        log
        tmp
    welcome-content
```

Fig. 9: ConSol*CM - File Structure in a JBoss 7 System

Example directories:

- **modules\system\layers\base**
  Subfolders contain the *JDBC* drivers:
  - com\microsoft\sqlserver\jdbc\main\sqljdbc4.jar
    (MS SQL)
  - oracle\jdbc\main\ojdbc6-11.2.0.3.jar
    (Oracle)
  - com\mysql\jdbc\main\
    (MySQL JDBC driver destination, must be installed manually)
- **standalone**
  Configuration in non-clustered environments:
  - **configuration**
    Configuration of the DB connection and logging inside the file *cm6.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. Can be emptied, e.g. for error analysis and/or fixing.

32.2.4 Oracle WebLogic Application Server File Structure

In an Oracle WebLogic environment, ConSol*CM is installed as a separate domain. ConSol*CM as well as CMRF are *managed servers*. Please see the *ConSol*CM Operations Manual* for details.

Here, only some directories are explained. If you want to administer ConSol*CM as a WebLogic application, please refer to the *ConSol*CM Operations Manual* and to general Weblogic tutorials.

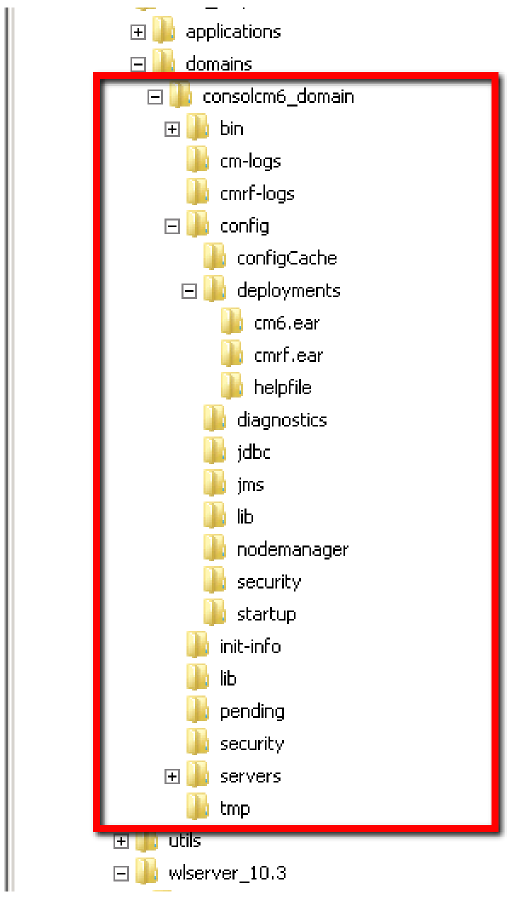


Fig. 10: ConSol*CM - File Structure in an Oracle WebLogic Application Server

Example directories:

- **bin**
  Start/stop scripts
- **cm-logs**
  All log files except for *cmrf.log*
- **cmrf-logs**
  - **cmrf.log file**
    Log messages for the CMRF (ConSol*CM Reporting Framework)
- **config**
  Configuration files
- **deployments**
  Deployed applications, i.e. here: ConSol*CM and CMRF as directories

32.2.5 Log Files

The logging behavior can be configured by editing the file(s)

- **log4j.xml**
  (in JBoss 5, where *Log4J* is used as logging framework)
- **cm6.xml** and/or **cm6-cmrf.xml**
  (in JBoss 7, where the built-in logging module of JBoss 7 is used)

Log File Types

The following log files are used:

- **boot.log**
  Messages concerning system start-up (e.g. the Java version is indicated).
- **cmrf.log**
  Messages concerning CMRF (ConSol*CM Reporting Framework), i.e. messages that concern the data transfer operations from the ConSol*CM database to the CMRF database (DWH). This is done using *JMS* (Java Messaging Service).
- **cmweb.log**
  Messages concerning the ConSol*CM Web Client.
- **ctx.log**
  Contains messages of the *Spring Framework*.
- **errors.log**
  Contains only messages that have at least the log level *ERROR*.
- **esb.log**
  Contains messages of the *Mule Framework* (Mule is the internal ESB that is used for the processing of incoming e-mails).
- **index.log**
  Messages concerning the Indexer.
- **mail.log**
  Contains messages of the e-mail subsystem.
- **operationtimes.log**
  Only used when it has been enabled. Contains times of requests in order to identify possible performance bottlenecks.
- **server.log**
  The general log file that contains all messages, as default setting at least with log level *INFO*. It is recommended to use the *DailyRollingFileAppender* in order to prevent the file system from filling up.
- **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 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.
- **tx.log**
  Contains *Spring Framework* transactions related log messages.
- **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.

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):

```
2012-11-06 14:22:12,685 INFO [e.coyote.http11.Http11Protocol] Starting Coyote HTTP/1.1 on http-
0.0.0.0-8080
```

The components of the message:

- **Date:**
  November 6th, 2012
- **Timestamp:**
  14:22:12
- **Loglevel:**
  INFO
- **Logger:**
  e.coyote.http11.Http11Protocol
  Name of a Java class, not complete (only last 30 characters), the real name would be *org.apache.coyote.http11.HttpProtocol*.
- **Message:**
  Starting Coyote HTTP/1.1 on http-0.0.0.0-8080

Simple messages and messages which concern a successful operation often comprise only one line.

When errors occur (log level *ERROR*), you might find stack traces. Please approach one of our ConSol*CM consultants or our ConSol*CM support team for help.

33 Appendix A - List of Annotations (up to Version 6.9.4)

- Alphabetical List of Field Annotations
- Alphabetical List of Group Annotations

33.1 Alphabetical List of Field Annotations

	Name	Annotation Type	Description	Values	Comment
A	accuracy	validation	For date fields, to define the level of detail displayed.	date (default)	Show date without time.
				date-time	Show date with time.
				only-time	Show only time, no date.
B	boolean-type	component-type	Definition of the layout of a boolean field.	check box (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).
C	colspan	layout	Defines how many columns are reserved for the field in the layout.	<number>	Number of columns.
	contact search result column	search-result	Identifies whether the field should be presented in the search result by default.	true	Remove the annotation if the field should not be visible by default.
				true / false	

	Name	Annotation Type	Description	Values	Comment
	contains contacts	ticket contact relation type	Used only for list field definition, indicates that it can hold unit references to units annotated as contacts.		Value type is boolean. Necessary to distinguish if the list is shown with the contact (true) or with the ticket (false).
D	dialable	phone commander	Defines a field with a phone number.	true	Used with CM. Phone only. Marks a phone number as automatically dialable for outgoing calls for the CTI system.
E	email	validation	Used for e-mail addresses to check if the format is correct, i.e. if <name>@<domain> has been entered.	true	May be used with *string* custom fields. Remove the annotation if the format should not be checked.
	enum field with ticket color	ticket display	Defines the background color of the ticket icon for ticket list and ticket.	true / false	The field has to exist within *Enum Administration* where lists, values, and colors are defined.
	enum-in-search-type	component-type	Defines whether an *enum* field used in a search	single (default) / multiple	Accepts search over multiple values if value *multiple* is set.

	Name	Annotation Type	Description	Values	Comment
			accepts search over multiple values.		
	enum-type	component-type	Layout definition of list display.	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 selection where the field is an input field used to filter the list.
F	field-group	layout	Allows grouping of fields in *view* mode. Annotation is ignored in *edit* mode.	<string>	To group fields the same *string* value has to be set in the annotation of each field. Two or more custom fields are bound when they share the same value of this annotation. The group of coupled custom fields is shown only if all of them have values set.
	field indexed	indexing	Used to indicate that a	transitive (default)	All data is displayed

	Name	Annotation Type	Description	Values	Comment
			database index will be created for this field.		(ticket and customer).
				unit	Used for customer data. Only the unit and the parent unit (i.e. company) is given as a search result, no tickets are provided.
				local	Used for customer data. Only the unit is given as a search result, no company and no tickets are displayed.
				not indexed	Field is not indexed.
	fieldsize	layout	Displayed field size within ticket layout.	<rows>;<cols>	For *string* custom fields with annotation *text-type* and value *textarea*. <rows> defines the number of displayed rows and <cols> defines the number of characters displayed per

	Name	Annotation Type	Description	Values	Comment
					row. Used only for layout purposes.
				<number>	For *enum* custom fields. Defines how many values are directly visible in the list box. Used only for layout purposes.
	format	validation	Used to check the correct format of date fields.	<date format>	The pattern for the date is based on *SimpleDateFormat*, e.g. dd.mm.yyyy. Remember to set the proper *colspan* when you want to add hours/minutes. See http://docs.oracle.com/javase/6/docs/api/java/text/SimpleDateFormat.html for date format reference.
G	groupable	cmweb-common	Enables grouping in the ticket list.	true	Used only with *enum* custom fields. Remove the annotation if you want to disable grouping.
L	label-group	layout		<string>	

	Name	Annotation Type	Description	Values	Comment
			Indicates a group of fields along with its descriptive label in *view* mode. Annotation is ignored in *edit* mode.		Indicates a group of custom fields along with its descriptive label. The annotation is used in *presentation* mode, ignored in *edit* mode. The group can have exactly one label (a custom field of type *string* with assigned additional annotation *text-type* with value *label*). The label is shown when at least one custom 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	layout	Shows custom field value as a label in *view*	true	Remove the annotation if the label

	Name	Annotation Type	Description	Values	Comment
			mode. Annotation is ignored in *edit* mode.		should not be visible in *view mode*.
	ldapid	contact authentication	Used in a data object group of type *customer*, for the data object group field which contains the LDAP ID for CM.Track authentication.		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 on 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.
	leave-trailing-zeros	common	Used for the display of fixed point numbers.	true / false	Remaining zeros of the fractional part are not cut off when value is *tr ue*.
	list-type	component-type	Disables the *ad d* and/or *delete* options for custom fields	fixed-size	It is not possible to add or delete fields /rows.

	Name	Annotation Type	Description	Values	Comment
			of type *list* or *struct*.		
				non-shrinkable	It is not possible to delete fields /rows.
				non-growable	It is not possible to add fields/rows.
M	matches	validation	Checks if input of *string* custom fields matches the given RegEx.	<string>	May be used with *string* custom fields.
	maxLength	validation	Defines the maximum length of input for *string* custom fields.	<number>	May be used with *string* custom fields.
	maxValue	validation	Defines the maximum value for *number* custom fields.	<number>	May be used with *number* custom fields, i. e. *number* and *fixed-point number*.
	minLength	validation	Defines the minimum length of input for *string* custom fields.	<number>	May be used with *string* custom fields.
	minValue	validation	Defines the minimum value for *number* custom fields.	<number>	May be used with *number* custom fields, i. e. *number* and *fixed-point number*.
N	no-history-field	performance	Indicates that a single	true / false	Annotation is active if value

	Name	Annotation Type	Description	Values	Comment
			custom field should not be historized. Overwrites the group annotation *no-history*.		is set to *true*. For fields that should be stored but not be visible in history use annotation *visib ility configuration*.
O	order-in-result	layout	Shows field as a column in the search result list at given position.	<number>	The columns are sorted in ascending order.
P	password	contact authentication	Indicates that this field will be used as a password in the authentication process.	<string>	Used for CM. Track.
	position	layout	Defines the position of a field within a grid layout.	<number>;<number>	Values define *r ow* and *column* (row;column), numbering starts at 0;0. If no values are set, the custom field will take the next free grid cell.
			Defines the position of a field within a list (struct).	0;<number>	Only the *colum n* value is used, the *row* value is ignored.
R	readonly	common	Used to indicate that the custom	true / false	Field is read only if value is set to *true*.

	Name	Annotation Type	Description	Values	Comment
			field cannot be modified.		Lack of value or any value except *false* is also treated as *true*.
	reportable	dwh	Indicates that the field is reportable and that it should be transferred to the DWH.	true / false	Field is reportable if value is set to *tr ue*.
	required	validation	Indicates that this is a required field.	true / false	Field is required if value is set to *tr ue*. 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.
	rowspan	layout	Indicates how many rows within the layout are occupied by this field.	<number>	Number of rows.
S	sortable	cmweb-common	Used to enable sorting of the ticket list.	true	Used for custom fields of type *date* or of type *enum*. Remove the annotation if you want to disable sorting.

	Name	Annotation Type	Description	Values	Comment
					For *enum* fields: Works only if order index is set for all values of the *enum* field.
	show-label-in-edit	layout	Whether custom field or data object group field should be displayed in *edit* mode with label.	true / false	Since version 6.9.4
	show-label-in-view	layout	Whether custom field or data object group field should be displayed in *view* mode with label	true / false	Since version 6.9.4
	show-tooltip	layout	Whether custom field or data object group field should be displayed with tooltip.	true / false	Since version 6.9.4
	show-watermark	layout	Whether custom field or data object group field should be displayed with watermark.	true / false	Since version 6.9.4
T	text-type	component-type	Defines the possible types of a *string* field.	text (default)	Single-line input field.

	Name	Annotation Type	Description	Values	Comment
				textarea	Multi-line input field.
				password	Input field for passwords. Password will be displayed as ******* in *view* mode.
				label	Input will be displayed as a label, i.e. the field is displayed only, no input is possible.
				url	Input will be displayed as a hyperlink in *view* mode. String has to match a specific URL pattern: "^((?:mailto\\:|(?:(?:ht|f)tps?)\\://)1\S+)(?: (?:\\\|)?(.*))?$" Example: "http://consol.de ConSol*"
	ticket-list-colspan	layout	Defines how many columns are occupied by the field in the ticket list box.	<number>	Number of columns.
	ticket-list-position	layout		<number>;<number>	Values define *row* and *column* (row;column),

	Name	Annotation Type	Description	Values	Comment
			Defines the position of the field in the ticket list box.		numbering starts at 0;0.
	ticket-list-rowspan	layout	Defines how many rows are occupied by the field in the ticket list box.	<number>	Number of rows.
U	username	contact authentification	Indicates that this field will be used as a login name in the authentication process.	true / false	Used for CM. Track.
V	visibility	common	Defines when the field is visible.	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 the field will always be visible.
	visibility configuration	visibility	Indicates the visibility of this field in history.	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	

	Name	Annotation Type	Description	Values	Comment
					Field is shown only on the 3rd level of history.

33.2 Alphabetical List of Group Annotations

	Name	Annotation Type	Description	Values	Comment
A	auto-open-group	layout	The group will be opened initially. More than one value can be entered as comma- or semicolon-separated list (can be used for the customer annotation).	ticket:create	Group is opened initially when a new ticket is created.
				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.
G	group-visibility	common	Defines the default visibility of a custom field group.	true / false	The annotation can be overwritten on field level.
N	no-history	performance	Indicates that all custom fields belonging to this group will not be historized.	true / false	Used to indicate that all custom fields that belong to this group should not be historized. Possible values are *true*

	Name	Annotation Type	Description	Values	Comment
					if this annotation should be active or *false* which is the same like removing the annotation. Use this annotation if you want to prevent 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* on field level.
R	reportable group	dwh	Indicates that all custom fields belonging to this group are reportable and should be transferred to CMRF.	true / false	A value has to be set. Annotation is active if value is set to *true*.
S	show-contact-in-ticket-list		Obsolete! Use page customization! accordionTicketList. mainCustomer	obsolete	

	Name	Annotation Type	Description	Values	Comment
			DescriptionVisible={true, false}		
	show-in-group-section	layout	Defines that a custom field group is displayed in the *Groups* section (as tab).	true / false	Without this annotation the group is shown in the non-tabbed ticket data or contact section.
	show-labels-in-edit	layout	Whether custom fields or data object group fields in this group should be displayed in *edit* mode with labels.	true / false	Since version 6.9.4
	show-labels-in-view	layout	Whether custom fields or data object group fields in this group should be displayed in *view* mode with labels.	true / false	Since version 6.9.4
	show-tooltips	layout	Whether custom fields or data object group fields in this group should be displayed with tooltips.	true / false	Since version 6.9.4
	show-watermarks	layout	Whether custom fields or data object group fields in	true / false	Since version 6.9.4

	Name	Annotation Type	Description	Values	Comment
			this group should be displayed with watermarks.		
U	unit is a contact ***deprecated***	ticket contact relation		true/false	Removed in version 6.9.0.

34 Appendix B - Glossary

	Term	Explanation
A	Access Rights	Permissions of an engineer to view or make changes to tickets in the Web Client. Access rights are always assigned to a group, never to single engineers/users.
	ACIM	Activity item - entry in the history section of a ticket (e.g. comment, e-mail, attachment, time booking entry).
	AD	Microsoft Active Directory - an LDAP-based directory service for Microsoft Windows domain networks.
	Additional customer	Customer (contact or company) besides the main customer, e.g. an employee of the company. For additional customers, customer roles can be assigned.
	Admin Tool	Graphical application to configure and manage a ConSol*CM system. Uses Java Web Start.
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 custom fields and data object group fields.
	CMDB	ConSol*CM Database - the working database of the CM system.

	Term	Explanation
	CMRF	ConSol*CM Reporting Framework - a JEE application which synchronizes data between the ConSol*CM database and the DWH.
	CM.Doc	A standard module of ConSol*CM which enables the engineer via ConSol*CM Web Client to work with MS Word documents pre-filled with ConSol*CM ticket or customer parameters.
	CM.Track	Consol*CM web portal - provides customer access to the ConSol*CM system.
	Company	A data object of type *company*. Often this is a real company or an institution, but it can also be something else, like a machine or a ship.
	Contact	A data object of type *contact*. Often this is the person who has a question or service request, but it can also be something else, like a machine or a product.
	CTI	Computer Telephony Integration - a description for any technology that allows interactions on a telephone and a computer to be integrated or coordinated.
	Customer	General term for customer objects in ConSol*CM. A customer can be a contact or a company. Technically, a customer is a data object. The respective Java class is *Unit*.
	Custom field	

	Term	Explanation
		A field where ticket data (e.g. priority, software module, etc.) can be stored.
	Custom field group	A group of custom fields where ticket data can be stored.
D	Data object	A customer, contact, or a company. Former *Unit*.
	Data object group	A group of fields where data for customers (contact or company) can be stored. Similar to custom field group for ticket data.
	Data object group field	A field where data for customers (contact or company) can be stored. Similar to custom field for ticket data.
	DWH	Data Warehouse - ConSol*CM database used for reporting and data analysis.
E	Engineer	User who has a login to the Web Client and who has to manage the tasks defined in the tickets.
	ESB	Enterprise Service Bus - a software architecture used for communication between mutually interacting software applications in a service-oriented architecture (SOA).
	ERP system	Enterprise Resource Planning - often used for this type of enterprise management software.
	ETL	Extract Transform Load - extracts data from one source (this can be a database or another source), transforms it, and loads it into a database, e.g. a data warehouse.
F	FlexCDM	

	Term	Explanation
		Flexible Customer Data Model - the customer data model which has been 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
I	IMAP	Internet Message Access Protocol - Internet standard protocol to access e-mail on a remote e-mail server. Can be used as plain IMAP or as secure IMAP (IMAPs). In the latter case the 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	KPI	Key Performance Indicator - parameter used for performance measurement for companies, projects etc.
L	LDAP	Lightweight Directory Access Protocol - application protocol to access and maintain directory information over an IP network.
	LDAPS	LDAP over SSL
M	Mailbox	Destination to which e-mail messages are delivered. Mailboxes are managed on an e-mail server. ConSol*CM can access one or more separate mailboxes to retrieve e-mails.

	Term	Explanation
	Main customer	The customer who is the main customer of a ticket. Starting with ConSol*CM version 6.9, this can be either a contact (= person) or a company.
	Mule	An open source Java-based Enterprise Service Bus (ESB).
N	NIMH	New Incoming Mail Handler - module for retrieving incoming e-mails, new in version 6.9.4.
P	PCDS	Page Customization Definition Section
	Pentaho	Pentaho^{TM} is a business intelligence (BI) suite which is available as open source version and as enterprise edition.
	POP	Post Office Protocol - Internet standard protocol to retrieve e-mails from a remote server via TCP/IP. Can be used as plain POP or as secure POP (POPs). In the latter case the 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	Comprises tickets from the same domain and makes sure that all tickets of this domain are treated in the same way. A queue always has one workflow. Access rights and other parameters are defined based on queues.
R	RDBMS	Relational Database Management System - e.g.

	Term	Explanation
		Oracle®, MS SQL Server®, MySQL.
	REST	Representational State Transfer - a method to transfer data via a network, based on HTTP.
	Role	Defines the access permissions and views of an engineer.
S	Script	Program written for a special 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.
	SMTP	Simple Message Transfer Protocol - standard protocol to send e-mails.
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. A new feature as of version 6.9.4.
	Template	Pre-formatted example concerning layout, text, and/or data, e.g. for e-mails or CM.Doc.
	Ticket	Incident, service case, or other request of an internal or external customer. A ticket is the object which runs through the process (defined by the workflow).
V	View	

	Term	Explanation
		A selection of tickets based on scopes from one or from different workflows, assigned to a role and visible in the ticket list of the ConSol*CM Web Client.
W	Workflow	Models a process that should be managed using ConSol*CM step by step.

35 Appendix C - System Properties

The lists provide explanation for all available ConSol*CM system properties. You can define properties in the Admin Tool, in the Configuration section.

- System Properties Ordered by Module
- System Properties Ordered by Property Name

35.1 System Properties Ordered by Module

Module	Property	Explanation
cmas-app-admin-tool	admin.tool.session.check. interval	*Description:* Admin-Tool inactive (ended) sessions check time interval (in seconds) *Type:* Integer *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* 30 *Since:* 6.7.5
cmas-app-admin-tool	autocomplete.enabled	*Description:* If the flag is missing or its value is *false*, then the *Auto complete address* tab is hidden in Admin-Tool. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* true *Since:* 6.9.2.0
cmas-app-admin-tool	delete.ticket.enabled	*Description: Controls if the menu entry Delete is displayed in the context menu in the AT for the ticket list in ticket administration* *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* true *Since:* 6.9.4.0
cmas-app-admin-tool	start.groovy.task.enabled	*Description: For being able to run AT scripts of type task in the AT GUI (tab Deployment), it is required to enable the 'Start task' button which is hidden by default. This is done by setting this system property to true* *Type:* Boolean *Restart required:* No

Module	Property	Explanation
		System: No *Optional:* Yes *Example value:* true *Since:* 6.9.4.0
cmas-core-cache	cache-cluster-name	*Description:* JBoss cache cluster name *Type:* String *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* 635a6de1-629a-4129-8299-2d98633310f0 *Since:* 6.4.0
cmas-core-cache	eviction.event.queue.size	*Description:* *Type:* Integer *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* 200000 *Since:* 6.4.0
cmas-core-cache	eviction.max.nodes	*Description:* *Type:* Integer *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* 100000 *Since:* 6.4.0
cmas-core-cache	eviction.wakeup.interval	*Description:* *Type:* Integer *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* 3000 *Since:* 6.4.0
cmas-core-index-common	big.task.minimum.size	*Description:* How many parts task at least should have to be handled by Indexer with low priority. *Type:* Integer *Restart required:* No *System:* Yes

Module	Property	Explanation
		Optional: No *Example value:* 15 (default) *Since:* 6.8.3
cmas-core-index-common	database.notification.enabled	*Description:* Indicates whether index update database notification channel should be used instead of jms. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* false *Since:* 6.8.4.7
cmas-core-index-common	database.notification.redelivery.delay.seconds	*Description:* In case of index update database notification channel, indicates notification redelivery delay when exception occurs. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 60 *Since:* 6.8.4.7
cmas-core-index-common	database.notification.redelivery.max.attempts	*Description:* In case of index update database notification channel, indicates maximumn redelivery attempts when exception occurs. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 60 *Since:* 6.8.4.7
cmas-core-index-common	disable.admin.task.auto.commit	*Description:* All tasks created for index update will be automatically executed right after creation. *Type:* Boolean *Restart required:* No

Module	Property	Explanation
		System: Yes *Optional:* No *Example value:* false *Since:* 6.6.1
cmas-core-index-common	index.attachment	*Description:* Describes if content of attachments is indexed. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* true *Since:* 6.4.3
cmas-core-index-common	index.history	*Description:* Describes if unit and ticket history are indexed. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* false *Since:* 6.1.0
cmas-core-index-common	index.status	*Description:* Status of the Indexer, possible values RED, YELLOW, GREEN, will be displayed in the Admin-Tool. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* GREEN *Since:* 6.6.1
cmas-core-index-common	index.task.worker.threads	*Description:* How many threads will be used to execute batch index tasks (synchronization, administrative, and repair tasks). *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 1 (default) (we

Module	Property	Explanation
		recommend to use a value not larger than 2) *Since:* 6.6.14, 6.7.3
cmas-core-index-common	index.version.current	*Description:* Holds information about current (possibly old) index version. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 1 (default) *Since:* 6.7.0
cmas-core-index-common	index.version.newest	*Description:* Holds information about which index version is considered newest. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 1 (default) *Since:* 6.7.0
cmas-core-index-common	indexed.assets.per.thread.in.memory	*Description:* How many assets should be loaded into memory at once during indexing per one thread. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 200 (default) *Since:* 6.8.0
cmas-core-index-common	indexed.engineers.per.thread.in.memory	*Description:* How many engineers should be loaded into memory at once during indexing per one thread. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 300 (default) *Since:* 6.6.14, 6.7.3

Module	Property	Explanation
cmas-core-index-common	indexed.tickets.per.thread.in.memory	*Description:* How many tickets should be loaded into memory at once during indexing per one thread. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 100 (default) *Since:* 6.6.14, 6.7.3
cmas-core-index-common	indexed.units.per.thread.in.memory	*Description:* How many units should be loaded into memory at once during indexing per one thread. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 200 (default) *Since:* 6.6.14, 6.7.3
cmas-core-index-common	synchronize.master.address	*Description:* Value of *-Dcmas.http.host.port* informing how to connect to indexing master server. Default null. Since 6.6.17 this value is configurable in set-up to designate initial indexing master server. Please note that changing this value is only allowed when all cluster nodes index changes receivers are stopped. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 127.0.0.1:80 *Since:* 6.6.0
cmas-core-index-common	synchronize.master.security.token	*Description:* The password for accessing the index snapshot via URL, e.g. for index synchronization or for back-ups. *Type:* String

Module	Property	Explanation
		Restart required: No *System:* Yes *Optional:* Yes *Example value:* token *Since:* 6.6.0
cmas-core-index-common	synchronize.master.security.user	*Description:* The user name for accessing the index snapshot via URL, e.g. for index synchronization or for back-ups. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* user *Since:* 6.6.0
cmas-core-index-common	synchronize.master.timeout.minutes	*Description:* How much time master server may constantly fail until new master gets elected with index fix procedure. Default 5. Since 6.6.17 this value is configurable in set-up where zero means that master server will never change (failover mechanism is off). *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 5 *Since:* 6.6.0
cmas-core-index-common	synchronize.megabits.per.second	*Description:* How much bandwidth can master server consume to transfer index changes to all slave servers. Default 85. Please do not use all available bandwidth to transfer index changes between hosts. This will most probably partition cluster as some subsystems will not be able to communicate. *Type:* Integer *Restart required:* No

Module	Property	Explanation
		System: Yes *Optional:* No *Example value:* 85 *Since:* 6.6.0
cmas-core-index-common	synchronize.sleep.millis	*Description:* How often each slave server polls master server for index changes. Default 1000. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 1000 *Since:* 6.6.0
cmas-core-security	admin.email	*Description:* The e-mail address of the ConSol*CM administrator. The value which you have entered during system set-up is used initially. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* myuser@consol.de *Since:* 6.0
cmas-core-security	admin.login	*Description:* The name of the ConSol*CM administrator. The value which you have entered during system set-up is used initially. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* admin *Since:* 6.0
cmas-core-security	authentication.method	*Description:* User authentication method (internal CM database or LDAP authentication). Allowed values are *LDAP* or *DATABASE*. *Type:* String

Module	Property	Explanation
		Restart required: No *System:* Yes *Optional:* No *Example value:* DATABASE *Since:* 6.0
cmas-core-security	contact.authentication.method	*Description:* Indicates contact authentication method, where possible values are *DATABASE* or *LDAP* or *LDAP,DATABASE* or *DATABASE,LDAP.* *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Since:* 6.9.3.0
cmas-core-security	contact.inherit.permissions.only.to.own.customer.group	*Description:* Indicates whether authenticated contact inherits all customer group permissions from representing engineer (false) or only permission to own customer group (true). *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Since:* 6.9.2.3
cmas-core-security	kerberos.v5.enabled	*Description:* Flag which indicates whether SSO via Kerberos is enabled. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* false (default if Kerberos has not been enabled during system set-up) *Since:* 6.2.0
cmas-core-security	kerberos.v5.username.regex	*Description:* Regular expression used for mapping Kerberos principal to CM user login. *Type:* String

Module	Property	Explanation
		Restart required: No *System:* Yes *Optional:* No *Example value:* (.*)@.* *Since:* 6.2.0
cmas-core-security	ldap.authentication	*Description:* Authentication method used when using LDAP authentication. *Type:* String *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* simple *Since:* 6.0
cmas-core-security	ldap.basedn	*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
cmas-core-security	ldap.contact.name.basedn	*Description:* Base path to search for contact DN by LDAP ID (e.g. ou=accounts,dc=consol,dc=de). *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Since:* 6.9.3.0
cmas-core-security	ldap.contact.name.password	*Description:* Password to look up contact DN by LDAP ID. If not set, anonymous account is used. *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Since:* 6.9.3.0
cmas-core-security	ldap.contact.name.providerurl	

Module	Property	Explanation
		Description: Address of the LDAP server (ldap[s]://host:port). *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Since:* 6.9.3.0
cmas-core-security	ldap.contact.name.searchattr	*Description:* Attribute to search for contact DN by LDAP ID (e.g. uid). *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Since:* 6.9.3.0
cmas-core-security	ldap.contact.name.userdn	*Description:* User DN to look up contact DN by LDAP ID. If not set, anonymous account is used. *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Since:* 6.9.3.0
cmas-core-security	ldap.initialcontextfactory	*Description:* Class name for initial context factory of LDAP implementation when using LDAP authentication. If it is not set, com.sun.jndi.ldap. LdapCtxFactory is being used as a value. *Type:* String *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* com.sun.jndi. ldap.LdapCtxFactory *Since:* 6.0
cmas-core-security	ldap.password	*Description:* Password for connecting to LDAP to look up users (when using LDAP authentication). Only needed if

Module	Property	Explanation
		look-up cannot be done anonymously. *Type:* Password *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.1.2
cmas-core-security	ldap.providerurl	*Description:* LDAP provider (when using LDAP authentication). *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* ldap://ldap.consol.de:389 *Since:* 6.0
cmas-core-security	ldap.searchattr	*Description:* Search attribute for looking up LDAP entry connected to CM6 login. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* uid *Since:* 6.0
cmas-core-security	ldap.userdn	*Description:* LDAP user for connecting to LDAP to look up users (when using LDAP authentication). Only needed if look-up cannot be done anonymously. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.1.2
cmas-core-server	attachment.allowed.types	*Description:* Comma-separated list of allowed filename extensions (if no value defined, all file extensions are allowed).

Module	Property	Explanation
		Type: String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* txt,zip,doc *Since:* 6.5.0
cmas-core-server	attachment.max.size	*Description:* Maximum attachment size in MB *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 100 *Since:* 6.4.0
cmas-core-server	config.data.version	*Description:* *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 11 *Since:* 6.0
cmas-core-server	defaultCommentClassName	*Description:* Default text class name for comments *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Example value:* *Since:* 6.3.0
cmas-core-server	defaultIncommingMailClassNam e	*Description:* Default text class name for incoming e-mails *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Example value:* *Since:* 6.3.0
cmas-core-server	defaultOutgoingMailClassName	*Description:* Default text class name for outgoing e-mails *Type:* String *Restart required:* No

Module	Property	Explanation
		System: No *Optional:* Yes *Example value:* *Since:* 6.3.0
cmas-core-server	fetchSize.strategy	*Description:* Strategy selected to set fetch size on jdbc result sets. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* FetchSizePageBasedStrategy, FetchSizeThresholdStrategy, FetchSizeFixedStrategy *Since:* 6.8.4.1
cmas-core-server	fetchSize.strategy. FetchSizeFixedStrategy.value	*Description:* Sets fetch size value if selected strategy to set fetch size is *FetchSizeFixedStrategy*. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 150 *Since:* 6.8.4.1
cmas-core-server	fetchSize.strategy. FetchSizePageBasedStrategy. limit	*Description:* Sets maximum fetch size value if selected strategy to set fetch size is *FetchSizePageBasedStrategy*. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 10000 *Since:* 6.8.4.1
cmas-core-server	fetchSize.strategy. FetchSizeThresholdStrategy. value	*Description:* Sets fetch size threshold border values if selected strategy to set fetch size is *FetchSizeThresholdStrategy*. *Type:* Integer

Module	Property	Explanation
		Restart required: No *System:* Yes *Optional:* Yes *Example value:* 150,300,600,1000 *Since:* 6.8.4.1
cmas-core-server	last.config.change	*Description:* Random UUID created during last change in config *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* 2573c7b7-2bf5-47ff-b5a2-bad31951a266 *Since:* 6.1.0, 6.2.1
cmas-core-server	ldap.certificate.basedn	*Description:* Base DN for certificates location in LDAP tree. If not provided, *cmas-core-security*, *ldap.basedn* is taken. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* ou=accounts, dc=consol,dc=de *Since:* 6.8.4
cmas-core-server	ldap.certificate.content.attribute	*Description:* LDAP attribute name used where certificate data is stored in LDAP tree. Default value is: usercertificate. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* usercertificate *Since:* 6.8.4
cmas-core-server	ldap.certificate.password	*Description:* LDAP Certificates manager password. If not set, *cmas-core-security*, *ldap.password* is taken.

Module	Property	Explanation
		Type: String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.8.4
cmas-core-server	ldap.certificate.providerurl	*Description:* LDAP Certificates provider URL. If not set, *cmas-core-security*, *ldap.providerurl* is taken. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* ldap://ldap.consol.de:389 *Since:* 6.8.4
cmas-core-server	ldap.certificate.searchattr	*Description:* LDAP attribute name used to search for certificate in LDAP tree. Default value is: mail. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* mail *Since:* 6.8.4
cmas-core-server	ldap.certificate.userdn	*Description:* LDAP Certificates manager DN. If not set, *cmas-core-security*, *ldap.userdn* is taken. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.8.4
cmas-core-server	mail.notification.engineerChange	*Description:* Flag if notification e-mail should be sent when engineer of ticket is changed. *Type:* Boolean *Restart required:* No *System:* Yes

Module	Property	Explanation
		Optional: No *Example value:* true *Since:* 6.1.0
cmas-core-server	mail.notification.sender	*Description: From* address for notification e-mails when engineer of ticket is changed. If not set, *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
cmas-core-server	mail.smtp.email	*Description:* SMTP e-mail URL for outgoing e-mails *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* smtp://mail.consol.de:25 *Since:* 6.0
cmas-core-server	mail.smtp.envelopesender	*Description:* E-mail address used as sender in SMTP envelope. If not set, the *From:* address of the e-mail is used. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* mysender@mydomain.com *Since:* 6.5.7
cmas-core-server	max.licences.perUser	*Description:* Sets maximum licenses single user can use (e.g logging in from different browsers). By default this value is not restricted. *Type:* Integer

Module	Property	Explanation
		Restart required: No *System:* Yes *Optional:* Yes *Example value:* 10 *Since:* 6.8.4.5
cmas-core-server	monitoring.engineer.login	*Description:* Login of monitoring engineer *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* nagios *Since:* 6.9.3.0
cmas-core-server	monitoring.unit.login	*Description:* Login of monitoring unit *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* nagios *Since:* 6.9.3.0
cmas-core-server	nimh.enabled	*Description:* Enables nimh service. Must be suffixed with nodeid in cluster e.g. nimh. enabled.NODEID = true *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* false *Since:* 6.9.4.0
cmas-core-server	server.session.archive.reaper. interval	*Description:* Server archived sessions' reaper interval (in seconds) *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 60 *Since:* 6.7.1
cmas-core-server	server.session.archive.timeout	

Module	Property	Explanation
		Description: Server sessions archive validity timeout (in days). After this time session info is removed from db. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 31 *Since:* 6.7.1
cmas-core-server	server.session.reaper.interval	*Description:* Server inactive (ended) sessions' reaper interval (in seconds) *Type:* Integer *Restart required:* Only Session Service *System:* Yes *Optional:* No *Example value:* 60 *Since:* 6.6.1, 6.7.1
cmas-core-server	server.session.timeout	*Description:* Server session timeout (in seconds) for connected clients. Each client can overwrite this timeout with custom value using its ID (ADMIN_TOOL, WEB_CLIENT, WORKFLOW_EDITOR, TRACK (before 6.8 please use PORTER), ETL, REST) appended to property name, e.g. server.session.timeout. ADMIN_TOOL *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 1800 *Since:* 6.6.1, 6.7.1
cmas-core-server	skip.wfl.transfer.cleanup	*Description:* If set to true skips workflow cleanup after transfer *Type:* Boolean *Restart required:* No

Module	Property	Explanation
		System: No *Optional:* Yes *Example value:* false (default) *Since:* 6.9.4.1
cmas-core-server	tickets.delete.size	*Description:* Property that defines a number of tickets deleted per transaction. By default it is set to 10. *Type:* Integer *Restart required:* Only Session Service *System:* Yes *Optional:* No *Example value:* 10 *Since:* 6.8.1
cmas-core-server	ticket.delete.timeout	*Description:* Transaction timeout (in seconds) for deleting tickets *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 60 *Since:* 6.1.3
cmas-core-server	unit.replace.batchSize	*Description:* Describes number of objects to be processed in unit replace action. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 5 *Since:* 6.8.2
cmas-core-server	unit.replace.timeout	*Description:* Transaction timeout (seconds) of unit replacement action step. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 120 *Since:* 6.8.2

Module	Property	Explanation
cmas-core-server	unused.content.remover.cluster.node.id	*Description:* Value of a *cmas.clusternode.id* designating node which will remove unused ticket attachments and unit content entries. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 1 (assuming cluster node started with -Dcmas.clusternode.id=1 parameter) *Since:* 6.9.0.0
cmas-core-server	unused.content.remover.enabled	*Description:* Flag whether unused ticket attachments and unit content entries removal should take place. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* true *Since:* 6.9.0.0
cmas-core-server	unused.content.remover.polling.minutes	*Description:* How often unused ticket attachments and unit content entries should be checked for removal. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 15 *Since:* 6.9.0.0
cmas-core-server	unused.content.remover.ttl.minutes	*Description:* Minimum interval after which unused ticket attachments and unit content entries can be removed. *Type:* Integer *Restart required:* No *System:* Yes

Module	Property	Explanation
		Optional: No *Example value:* 1440 *Since:* 6.9.0.0
cmas-core-server	warmup.executor.enabled	*Description:* Flag whether server should asynchronously warm up during startup (e.g. fill some of internal caches) *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* true *Since:* 6.9.4.2
cmas-core-shared	cluster.mode	*Description:* Flag if CMAS is running in cluster. *Type:* Boolean *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* false *Since:* 6.1.0
cmas-core-shared	data.directory	*Description:* Directory for CMAS data (e.g. index) *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* C: \Users\user\cmas *Since:* 6.0
cmas-dwh-server	autocommit.cf.changes	*Description:* *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* false *Since:* 6.7.0
cmas-dwh-server	batch-commit-interval	*Description:* Number of objects in a JMS message. Higher value means better transfer performance and bigger memory

Module	Property	Explanation
		usage. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 100 *Since:* 6.0.0
cmas-dwh-server	communication.channel	*Description:* Communication channel. Possible values are DIRECT (database communication channel, default value since 6.9.4.1), JMS (default value before 6.9.4.1). Before 6.9.4.1 it has to be manually added. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* DIRECT *Since:* 6.8.5.0
cmas-dwh-server	dwh.mode	*Description:* Current mode of 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
cmas-dwh-server	ignore-queues	*Description:* By adding a comma separated list of queue names it is configured that tickets of these queues are not transferred to the DWH. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* QueueName1,

Module	Property	Explanation
		QueueName2,QueueName3 *Since:* 6.6.19 *Removed in:* 6.8.1
cmas-dwh-server	is.cmrf.alive	*Description:* As a starting point time of sending last message to CMRF should be used. If response from CMRF is not received after value (in seconds) it should create a DWH operation status with error message that CMRF is down. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 1200 *Since:* 6.7.0
cmas-dwh-server	java.naming.factory.initial	*Description:* Factory class for DWH context factory. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* org.jnp.interfaces.NamingContextFactory *Since:* 6.0.1
cmas-dwh-server	java.naming.factory.url.pkgs	*Description:* *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* org.jboss.naming:org.jnp.interfaces *Since:* 6.0.1
cmas-dwh-server	java.naming.provider.url	*Description:* URL of naming provider *Type:* String *Restart required:* No *System:* Yes

Module	Property	Explanation
		Optional: No *Example value:* localhost *Since:* 6.0.1
cmas-dwh-server	notification.error.description	*Description:* Text for error e-mails from DWH *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* Error occurred *Since:* 6.0.1
cmas-dwh-server	notification.error.from	*Description: From* address for error e-mails from DWH *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.0.1
cmas-dwh-server	notification.error.subject	*Description:* Subject for error e-mails from DWH *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* Error occurred *Since:* 6.0.1
cmas-dwh-server	notification.error.to	*Description: To* address for error e-mails from DWH *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* myuser@consol.de *Since:* 6.0.1
cmas-dwh-server	notification.finished_successfully.description	*Description:* Text for e-mails from DWH when transfer finished successfully. *Type:* String *Restart required:* No *System:* Yes

Module	Property	Explanation
		Optional: No *Example value:* Transfer finished successfully *Since:* 6.0.1
cmas-dwh-server	notification.finished_successfully.from	*Description: From* address for e-mails from DWH when transfer finished successfully. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.0.1
cmas-dwh-server	notification.finished_successfully.subject	*Description:* Subject for e-mails from DWH when transfer finished successfully. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* Transfer finished successfully *Since:* 6.0.1
cmas-dwh-server	notification.finished_successfully.to	*Description: To* address for e-mails from DWH when transfer finished successfully. *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* myuser@consol.de *Since:* 6.0.1
cmas-dwh-server	notification.finished_unsuccessfully.description	*Description:* Text for e-mails from DWH when transfer finished unsuccessfully. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* Transfer finished unsuccessfully *Since:* 6.0.1

Module	Property	Explanation
cmas-dwh-server	notification. finished_unsuccessfully.from	*Description: From* address for e-mails from DWH when transfer finished unsuccessfully. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.0.1
cmas-dwh-server	notification. finished_unsuccessfully.subject	*Description:* Subject for e-mails from DWH when transfer finished unsuccessfully. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* Transfer finished unsuccessfully *Since:* 6.0.1
cmas-dwh-server	notification. finished_unsuccessfully.to	*Description: To* address for e-mails from DWH when transfer finished unsuccessfully. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* myuser@consol.de *Since:* 6.0.1
cmas-dwh-server	notification.host	*Description:* Mail (SMTP) server hostname for sending DWH e-mails *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* mail.consol.de *Since:* 6.1.0
cmas-dwh-server	notification.password	*Description:* Password for sending DWH e-mails (optional) *Type:* String *Restart required:* No

Module	Property	Explanation
		System: Yes *Optional:* Yes *Since:* 6.1.0
cmas-dwh-server	notification.port	*Description:* SMTP port for sending DWH e-mails *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 25 *Since:* 6.1.0
cmas-dwh-server	notification.protocol	*Description:* The protocol used for sending e-mails from DWH. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* pop3\
cmas-dwh-server	notification.username	*Description:* (SMTP) User name for sending DWH e-mails *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* myuser *Since:* 6.1.0
cmas-dwh-server	skip-ticket	*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
cmas-dwh-server	skip-ticket-history	*Description:* History of ticket is not transferred during transfer /update. *Type:* Boolean *Restart required:* No

Module	Property	Explanation
		System: Yes *Optional:* No *Example value:* false *Since:* 6.6.19 *Removed in:* 6.8.1
cmas-dwh-server	skip-unit	*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
cmas-dwh-server	skip-unit-history	*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
cmas-dwh-server	split.history	*Description:* Changes the SQL that fetches the history for the tickets during DWH transfer not to 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
cmas-dwh-server	unit.transfer.order	*Description:* Define in which order data object groups should be transferred to the DWH. *Type:* String *Restart required:* No

Module	Property	Explanation
		System: Yes *Optional:* Yes *Example value: company; customer* *Since:* 6.6.19 *Removed in:* 6.8.1
cmas-esb-core	esb.directory	*Description:* Directory used by ESB (Mule) *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* C:\Users\user\cmas\mule *Since:* 6.0
cmas-esb-mail	mail.attachments.validation.info.sender	*Description:* Sets *From* header of attachments type error notification e-mail. As a default the e-mail address of the administrator which you have entered during system set-up is used. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* admin@consolcm.com *Since:* 6.7.5
cmas-esb-mail	mail.attachments.validation.info.subject	*Description:* Sets subject of attachments type error notification e-mail. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* E-mail was not processed because its attachments were rejected!!! *Since:* 6.7.5
cmas-esb-mail	mail.callname.pattern	

Module	Property	Explanation
		Description: Regular expression for subject of incoming e-mails. Available as TICKET_NAME_PATTERN_FORMAT in incoming e-mail scripts. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* .*?Ticket\s+\((\S+)\).* *Since:* 6.0
cmas-esb-mail	mail.cluster.node.id	*Description:* Only the node whose mail.cluster.node.id equals cmas.clusternode.id will start the Mule ESB mail services. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* unspecified *Since:* 6.6.5
cmas-esb-mail	mail.db.archive	*Description:* If property is set to *true*, incoming e-mails are archived in the database. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* false (default) *Since:* 6.8.5.5
cmas-esb-mail	mail.delete.read	*Description:* Determines whether CM deletes messages fetched via IMAP(S). Setting value to *true* will cause deletion of messages after fetching. Default is to not delete messages fetched via IMAP(S). Note: Messages fetched via POP3(S) will always be deleted. *Type:* Boolean *Restart required:* No

Module	Property	Explanation
		System: Yes *Optional:* No *Example value:* true *Since:* 6.7.3
cmas-esb-mail	mail.encryption	*Description:* If property is set to *true*, the encrypt check box in the Ticket E-Mail Editor is checked by default. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* true (default = false) *Since:* 6.8.4.0
cmas-esb-mail	mail.incoming.uri	*Description:* URL for incoming e-mails *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* pop3://cm-incoming-user:password@localhost:10110 *Since:* 6.0 This value should not be edited here using the system properties pop-up window, but the mailboxes should be configured using the tab E-mail. Using this standard feature all entries are controlled - i. e. for each mailbox which is added, CM establishes a test connection during mailbox set-up. That way it is not possible to enter wrong values.

Module	Property	Explanation
cmas-esb-mail	mail.max.restarts	*Description:* Maximum number of mail service restarts before giving up *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 3 *Since:* 6.0
cmas-esb-mail	mail.mime.strict	*Description:* If set to *false*, e-mail addresses are not parsed for strict MIME compliance. Default is *true*, which means check for strict MIME compliance. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* false *Since:* 6.6.17, 6.7.3
cmas-esb-mail	mail.mule.service	*Description:* *From* address for e-mails sent by Mule service *Type:* EMail *Restart required:* No *System:* Yes *Optional:* No *Example value:* myuser@consol.de *Since:* 6.0
cmas-esb-mail	mail.polling.interval	*Description:* Mail polling interval in ms *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 60000 *Since:* 6.0
cmas-esb-mail	mail.process.error	*Description:* *To* address for error e-mails from Mule. As a default the e-mail address of the administrator which you have

Module	Property	Explanation
		entered during system set-up is used. *Type:* EMail *Restart required:* No *System:* Yes *Optional:* No *Example value:* myuser@consol.de *Since:* 6.0
cmas-esb-mail	mail.process.retry.attempts	*Description:* Number of retries when processing e-mail *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 3 *Since:* 6.0.2
cmas-esb-mail	mail.process.timeout	*Description:* Mail processing timeout in seconds *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 60 *Since:* 6.1.3
cmas-esb-mail	mail.redelivery.retry.count	*Description:* Indicates the number of retries of re-delivering an e-mail from the CM system. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 3 *Since:* 6.1.0
cmas-nimh	filesystem.polling.threads.number	*Description:* Number of threads started for db mails' queue polling. Default: 1 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes

Module	Property	Explanation
		Example value: 10 *Since:* 6.4.0
cmas-nimh	filesystem.polling.threads. shutdown.timeout.seconds	*Description:* Waiting time after the shutdown signal. When the timeout reached, thread will be terminated. Default: 60 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	filesystem.polling.threads. watchdog.interval.seconds	*Description:* Watchdog thread interval. Default: 30 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	filesystem.task.enabled	*Description:* With this property service thread related to given poller can be disabled. Default: true *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* true *Since:* 6.4.0
cmas-nimh	filesystem.task.interval.seconds	*Description:* Default interval for polling mailboxes. Default: 60 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	filesystem.task.polling.folder	*Description:* Polling folder location which will be scanned

Module	Property	Explanation
		for emails in the format of eml files. Default: "mail" subdir of cmas data directory *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Example value:* c://cmas//mail *Since:* 6.4.0
cmas-nimh	filesystem.task.timeout.seconds	*Description:* After this time (of inactivity) the service thread is considered as damaged and automatically restarted. Default: 120 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	filesystem.task.transaction.timeout.seconds	*Description:* Default transaction timeout for mail fetching transactions. Should be correlated with number of messages fetched at once. Default: 60 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	mailbox.1.connection.host	host (server) for first configured mailbox. Will overwrite the default parameter *mailbox.default.connection.host*
cmas-nimh	mailbox.1.connection.password	password for first configured mailbox. Will overwrite the default parameter *mailbox.default.connection.password*
cmas-nimh	mailbox.1.connection.port	

Module	Property	Explanation
		port for first configured mailbox. Will overwrite the default parameter *mailbox.default.connection.port*
cmas-nimh	mailbox.1.connection.protocol	protocol (e.g., IMAP or POP3) for first configured mailbox. Will overwrite the default parameter *mailbox.default.connection.protocol*
cmas-nimh	mailbox.1.connection.username	username for first configured mailbox. Will overwrite the default parameter *mailbox.default.connection.username*
cmas-nimh	mailbox.2.connection.host	host (server) for second configured mailbox. Will overwrite the default parameter *mailbox.default.connection.host*
cmas-nimh	mailbox.2.connection.password	password for second configured mailbox. Will overwrite the default parameter *mailbox.default.connection.password*
cmas-nimh	mailbox.2.connection.port	port for second configured mailbox. Will overwrite the default parameter *mailbox.default.connection.port*
cmas-nimh	mailbox.2.connection.protocol	protocol (e.g., IMAP or POP3) for second configured mailbox. Will overwrite the default parameter *mailbox.default.connection.protocol*
cmas-nimh	mailbox.2.connection.username	username 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.

Module	Property	Explanation
cmas-nimh	mailbox.default.connection.host	*Description:* Host (server name) of a given mailbox from which the poller reads e-mails. *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Example value:* 10.10.1.157 *Since:* 6.4.0
cmas-nimh	mailbox.default.connection. password	*Description:* Password for given mailbox from which the poller reads e-mails. *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Example value:* consol *Since:* 6.4.0
cmas-nimh	mailbox.default.connection.port	*Description:* Port for a given mailbox from which the poller reads e-mails. *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Example value:* 143 *Since:* 6.4.0
cmas-nimh	mailbox.default.connection. protocol	*Description:* Poller's protocol e.g: IMAP or POP3. No default value *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Example value:* imap *Since:* 6.4.0
cmas-nimh	mailbox.default.connection. username	*Description:* Username for a given mailbox from which the poller reads e-mails. *Type:* String *Restart required:* No *System:* No

Module	Property	Explanation
		Optional: Yes *Example value:* username *Since:* 6.4.0
cmas-nimh	mailbox.default.session.mail. debug	*Description:* Example javax.mail property - allows for more detailed javax.mail session debugging *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* true *Since:* 6.4.0
cmas-nimh	mailbox.default.session.mail. imap.timeout	*Description:* Example javax.mail property *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 120 *Since:* 6.4.0
cmas-nimh	mailbox.default.session.mail. mime.address.strict	*Description:* Example javax.mail property - counterpart of the old mule mail.mime.strict, allows to set not so strict mail header parsing *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* true *Since:* 6.4.0
cmas-nimh	mailbox.default.session.mail. pop3.timeout	*Description:* Example javax.mail property. *Type:* *Restart required:* *System:* *Optional:* *Example value:* *Since:* 6.4.0
cmas-nimh		

Module	Property	Explanation
	mailbox.default.task.delete.read.messages	*Description:* This defines whether messages should be removed from the mailbox after processing. For IMAP protocol messages are marked as SEEN by default. For POP3 protocol, when flag is set to true the message is removed, otherwise remains on server and will result in infinite reads. Default: false. *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* false *Since:* 6.4.0
cmas-nimh	mailbox.default.task.enabled	*Description:* With this property service thread related to given poller can be disabled. Default: true *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* false *Since:* 6.4.0
cmas-nimh	mailbox.default.task.interval.seconds	*Description:* Default interval for polling mailboxes. Default: 60 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	mailbox.default.task.max.message.size	*Description:* Maximum size of e-mail messages in bytes. Messages with a size greater than the value of this property can be processed from the Admin-Tool. Default set to 10Mb : 10485760

Module	Property	Explanation
		Type: Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 10485760 *Since:* 6.4.0
cmas-nimh	mailbox.default.task.max. messages.per.run	*Description:* Number of messages fetched at once from mailbox. Must be correlated with transaction timeout. Default set to: 20 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	mailbox.default.task.timeout. seconds	*Description:* After this time (of inactivity) the service thread is considered as damaged and automatically restarted. Default: 120 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	mailbox.default.task.transaction. timeout.seconds	*Description:* Default transaction timeout for mail fetching transactions. Should be correlated with number of messages fetched at once. Default: 60 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh		

Module	Property	Explanation
	mailbox.polling.threads.mail.log.enabled	*Description:* Enables mail logging which is especially crucial in cluster environment (used as semaphore there) *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Example value:* true (default) *Since:* 6.9.4.1
cmas-nimh	mailbox.polling.threads.number	*Description:* Number of threads for accessing mailboxes. Default: 1 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 1 *Since:* 6.4.0
cmas-nimh	queue.polling.threads.number	*Description:* Number of threads started for mails' queue polling. Default: 1 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 1 *Since:* 6.4.0
cmas-nimh	queue.polling.threads.shutdown.timeout.seconds	*Description:* Waiting time after the shutdown signal. When the timeout is reached, the thread will be terminated. Default: 60 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	queue.polling.threads.watchdog.interval.seconds	*Description:* Watchdog thread interval. Default: 30 *Type:* Integer

Module	Property	Explanation
		Restart required: No *System:* No *Optional:* Yes *Example value:* 30 *Since:* 6.4.0
cmas-nimh	queue.task.error.pause.seconds	*Description:* Maximum number of seconds, the queue poller waits after infrastructure (e.g. database) error. Default 180 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 180 *Since:* 6.4.0
cmas-nimh	queue.task.interval.seconds	*Description:* Main mails' queue polling thread interval. Default: 15 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 15 *Since:* 6.4.0
cmas-nimh	queue.task.max.retries	*Description:* Maximum number of e-mail processing retries after an exception. When reached, the e-mail is moved to the e-mail archive. This e-mail can be rescheduled again using NIMH API (or the Admin-Tool). *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 10 *Since:* 6.4.0
cmas-nimh	queue.task.timeout.seconds	*Description:* After this time (of inactivity) the service thread is considered as damaged and automatically restarted. Default:

Module	Property	Explanation
		600 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 600 *Since:* 6.4.0
cmas-nimh	queue.task.transaction.timeout.seconds	*Description:* Transaction timeout for mail processing in the pipe. Default: 60 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh-extension	mail.attachments.validation.info.sender	*Description:* Sets FROM header of attachments type error notification mail *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* admin@mail.com *Since:* 6.7.5 This is an equivalent to the old *cmas-esb-mail, mail.attachments.validation.info.sender*
cmas-nimh-extension	mail.attachments.validation.info.subject	*Description:* Sets subject of attachments type error notification mail *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* Mail was not

Module	Property	Explanation
		processed because its attachments were rejected!!! *Since:* 6.7.5 This is an equivalent to the old *cmas-esb-mail, mail.attachments. validation.info.subject*
cmas-nimh-extension	mail.db.archive	*Description:* If property is set to true, incoming emails are archived in the database *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* false (default) *Since:* 6.8.5.5 This is an equivalent to the old *cmas-esb-mail, mail.db.archive*
cmas-nimh-extension	mail.error.from.address	This is an equivalent to the old *cmas-esb-mail, mail.mule.service*
cmas-nimh-extension	mail.error.to.address	This is an equivalent to the old *cmas-esb-mail, mail.process.error*
cmas-nimh-extension	mail.mule.service	*Description:* From address for mails sent by Mule service *Type:* EMail

Module	Property	Explanation
		Restart required: No *System:* Yes *Optional:* No *Example value:* myuser@consol.de *Since:* 6.4.0
cmas-nimh-extension	mail.on.error	*Description:* When set true an error mail is sent to the above configured address in case email message could not be processed. Default: false *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* false *Since:* 6.4.0
cmas-nimh-extension	mail.process.error	*Description:* To address for error mails from Mule *Type:* EMail *Restart required:* No *System:* Yes *Optional:* No *Example value:* myuser@consol.de *Since:* 6.4.0
cmas-nimh-extension	mail.ticketname.pattern	*Description:* *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* .*?Ticket\s+\((\S+)\).* *Since:* 6.4.0
cmas-setup-hibernate	hibernate.dialect	*Description:* The dialect used by hibernate. Usually set during initial set-up (depending on the database system). *Type:* String *Restart required:* No *System:* Yes

Module	Property	Explanation
		Optional: No *Example value:* org.hibernate. dialect.MySQL5InnoDBDialect *Since:* 6.0
cmas-setup-hibernate	cmas.dropSchemaBeforeSetup	*Description:* Flag if schema is to be (was) dropped during setup *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* true *Since:* 6.0
cmas-setup-manager	initialized	*Description:* Flag if CMAS is initialized. If this value is missing or not *true*, set-up will be performed. *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 *fals e*, the ConSol*CM server will perform the system set-up at the next start, i.e. all data of the existing system is lost, including system properties!!!
cmas-setup-scene	scene	*Description:* Scene file which was imported during set-up (can be empty). *Type:* String *Restart required:* No *System:* Yes *Optional:* No

Module	Property	Explanation
		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
cmas-workflow-engine	jobExecutor.adminMail	*Description:* E-mail 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
cmas-workflow-engine	jobExecutor.idleInterval.seconds	*Description:* Determines how often job executor thread will look for new jobs to execute. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 5 (default) *Since:* 6.8.0
cmas-workflow-engine	jobExecutor.jobMaxRetries	*Description:* *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 5 (default) *Since:* 6.8.0
cmas-workflow-engine	jobExecutor. jobMaxRetriesReachedSubject	*Description:* *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* Job maximum

Module	Property	Explanation
		retries reached. Job was removed!!! (default) *Since:* 6.8.0
cmas-workflow-engine	jobExecutor.lockTimeout. seconds	*Description:* How long the job can be locked (marked for execution) by job executor. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 360 (default) *Since:* 6.8.0
cmas-workflow-engine	jobExecutor.lockingLimit	*Description:* Number of jobs locked at once (marked for execution) by job executor thread *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 10 (default) *Since:* 6.8.0
cmas-workflow-engine	jobExecutor.mailFrom	*Description:* E-mail which will be set as *From* header during admin notifications. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* jobexecutor@consol.de *Since:* 6.8.0
cmas-workflow-engine	jobExecutor.maxInactivityInterval. minutes	*Description:* Number of minutes of allowed job executor inactivity (e.g. when it is blocked by long timer execution). After this time executors threads are restarted. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes. Default value is set to 30 minutes

Module	Property	Explanation
		Example value: 15 (default) *Since:* 6.9.2.0
cmas-workflow-engine	jobExecutor.threads	*Description:* Number of job execution threads *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 1 (default) *Since:* 6.8.0
cmas-workflow-engine	jobExecutor.timerRetryInterval.seconds	*Description:* Determines how long job executor thread will wait after job execution error. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 10 (default) *Since:* 6.8.0
cmas-workflow-engine	jobExecutor.txTimeout.seconds	*Description:* Transaction timeout used for job execution *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 60 (default) *Since:* 6.8.0
cmweb-server-adapter	automatic.booking.enabled	*Description:* If enabled, time spend on creating comment /email will be measured and automatic time booking will be added. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* true *Since:* 6.9.4.2
cmweb-server-adapter	checkUserOnlineIntervalInSeconds	*Description:* The interval in seconds to check which users are online (default 180sec =

Module	Property	Explanation
		3min). *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 180 *Since:* 6.0
cmweb-server-adapter	cmoffice.enabled	*Description:* Flag if CM.Doc (former CM/Office) is enabled. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* false *Since:* 6.4.0
cmweb-server-adapter	commentRequiredForTicketCreation	*Description:* Flag if comment is a required field for ticket creation. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* true (default) *Since:* 6.2.0
cmweb-server-adapter	customizationVersion	*Description:* *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* cd58453e-f3cc-4538-8030-d15e8796a4a7 *Since:* 6.5.0
cmweb-server-adapter	data.optimization	*Description:* Defines optimization to be applied on response data. So far, 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

Module	Property	Explanation
		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 property data.optimization.*nodeId* to override 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
cmweb-server-adapter	defaultContentEntryClassName	*Description:* Default text class for new acims *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* default_class *Since:* 6.3.0
cmweb-server-adapter	defaultNumberOfCustomFieldsColumns	*Description:* Default number of columns for custom fields *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 3 *Since:* 6.2.0
cmweb-server-adapter	favoritesSizeLimit	*Description:* Maximum number of items in favorites list *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 10 *Since:* 6.0

Module	Property	Explanation
cmweb-server-adapter	globalSearchResultSizeLimit	*Description:* Maximum number of items in global (Q&E) search result *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 10 *Since:* 6.0
cmweb-server-adapter	helpFilePath	*Description:* URL for online help. If not empty, *Help* button is displayed in Web Client. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* http://www. consol.de *Since:* 6.2.1
cmweb-server-adapter	hideTicketSubject	*Description:* If set to *true*, ticket subject is hidden. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* false *Since:* 6.2.1
cmweb-server-adapter	mail.from	*Description:* Use this address if set instead of engineer e-mail address during e-mail conversation. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.1.2
cmweb-server-adapter	mail.reply.to	*Description:* When set, Web Client will display reply-to field on e-mail send, prefilled with this value. *Type:* String

Module	Property	Explanation
		Restart required: No *System:* Yes *Optional:* Yes *Since:* 6.0.1 Please see also section Queue Administration. When you set the REPLY TO address in the outgoing e-mail script, the *mail.reply.to* system property must not be set (because it would overwrite the configured value)! That means when you use one outgoing e-mail script for a queue you have to define outgoing e-mail scripts for all queues because the *mail.reply.to* property can no longer be used.
cmweb-server-adapter	mailTemplateAboveQuotedText	*Description:* Indicates behavior of e-mail template in the Ticket E-Mail Editor when another e-mail is quoted, i.e. forwarded or replied to. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* false *Since:* 6.2.4
cmweb-server-adapter	maxSizePerPagemapInMegaBytes	*Description:* Maximum size (in MB) for each Wicket pagemap *Type:* Integer *Restart required:* No *System:* Yes

Module	Property	Explanation
		Optional: No *Example value:* 15 *Since:* 6.3.5
cmweb-server-adapter	pagemapLockDurationInSeconds	*Description:* Number of seconds to pass before 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
cmweb-server-adapter	postActivityExecutionScriptName	*Description:* Defines the name for the script which should be executed after every workflow activity, see section Default Workflow Activity Script. 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
cmweb-server-adapter	queuesExcludedFromGS	*Description:* Comma-separated list of queue names which are excluded from global search. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.0
cmweb-server-adapter	rememberMeLifetimeInMinutes	*Description:* Lifetime for *rememb er me* in minutes *Type:* Integer *Restart required:* Yes *System:* Yes *Optional:* No

Module	Property	Explanation
		Example value: 1440 *Since:* 6.0
cmweb-server-adapter	request.scope.transaction	*Description:* It allows to disable request scope transaction. By default one transaction is used per request. Setting this property to *false* there will cause one transaction per service method invocation. *Type:* Boolean *Restart required:* Yes *System:* Yes *Optional:* Yes *Example value:* true *Since:* 6.8.1
cmweb-server-adapter	searchPageSize	*Description:* Default page size for search results *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 20 *Since:* 6.0
cmweb-server-adapter	searchPageSizeOptions	*Description:* Options for page size for search results *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* 10|20|30|40|50|75|100 *Since:* 6.0
cmweb-server-adapter	serverPoolingInterval	*Description:* *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 5 *Since:* 6.1.0
cmweb-server-adapter	supportEmail	*Description:* *Type:* String

Module	Property	Explanation
		Restart required: No *System:* Yes *Optional:* Yes *Since:* 6.0
cmweb-server-adapter	themeOverlay	*Description:* Name of used theme overlay *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* kyoEUR *Since:* 6.0
cmweb-server-adapter	ticketListRefreshIntervalInSeconds	*Description:* Refresh interval for ticket list (in seconds) *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 180 *Since:* 6.0
cmweb-server-adapter	ticketListSizeLimit	*Description:* Maximum number of tickets in ticket list *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 100 *Since:* 6.0
cmweb-server-adapter	unitIndexSearchResultSizeLimit	*Description:* Maximum number of units in unit search result (e.g. when searching for contact) *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 5 *Since:* 6.0
cmweb-server-adapter	urlLogoutPath	*Description:* URL which is used when user logs out. (If no value is set, logout leads to login-mask.)

Module	Property	Explanation
		Type: String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* http://intranet.consol.de *Since:* 6.3.1
cmweb-server-adapter	webSessionTimeoutInMinutes	*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
cmweb-server-adapter	wicketAjaxRequestHeaderFilterEnabled	*Description:* This enables filter for Wicket AJAX requests, coming from stale pages with Wicket 1.4 scripting (CM6 pre-6.8.0), after update to CM6 post-6.8.0. *Type:* Boolean *Restart required:* Yes *System:* Yes *Optional:* Yes *Example value:* false *Since:* 6.8.1
cmas-workflow-jbpm	fetchLock.interval	*Description:* *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 5000 *Removed in:* 6.8.0
cmas-workflow-jbpm	fetchLock.timeout	*Description:* *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No

Module	Property	Explanation
		Example value: 15000 *Removed in:* 6.8.0
cmas-workflow-jbpm	jobExecutor.idleInterval	*Description:* *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 45000 *Removed in:* 6.8.0 *Replaced by:* jobExecutor. idleInterval.seconds
cmas-workflow-jbpm	jobExecutor. jobExecuteRetryNumber	*Description:* *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 5 *Removed in:* 6.8.0 *Replaced by:* jobExecutor. jobMaxRetries
cmas-workflow-jbpm	jobExecutor.timerRetryInterval	*Description:* *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 10000 *Removed in:* 6.8.0 *Replaced by:* jobExecutor. timerRetryInterval.seconds
cmas-workflow-jbpm	mail.sender.address	*Description: From* address for e-mails from the workflow engine *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* myuser@conso. de *Removed in:* 6.8.0 *Replaced by:* jobExecutor. mailFrom
cmas-workflow-jbpm	outdated.lock.age	

Module	Property	Explanation
		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
cmas-workflow-jbpm	refreshTimeInCaseOfConcurrentRememberMeRequests	*Description:* It sets the refresh time (in seconds) after which page will be reloaded in case of concurrent *remember me* requests. This feature prevents one user from occupying many licenses. Please increase that time if sessions are still occupying. *Type:* Integer *Restart required:* Yes *System:* Yes *Optional:* Yes *Example value:* 5 *Since:* 6.8.2

35.2 System Properties Ordered by Property Name

Module	Property	Explanation
cmas-app-admin-tool	admin.tool.session.check.interval	*Description:* Admin-Tool inactive (ended) sessions check time interval (in seconds) *Type:* Integer *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* 30 *Since:* 6.7.5
cmas-app-admin-tool	autocomplete.enabled	*Description:* If the flag is missing or its value is *false*, then the *Auto complete address* tab is hidden in Admin-Tool. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* true *Since:* 6.9.2.0
cmas-app-admin-tool	delete.ticket.enabled	*Description: Controls if the menu entry Delete is displayed in the context menu in the AT for the ticket list in ticket administration* *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* true *Since:* 6.9.4.0
cmas-app-admin-tool	start.groovy.task.enabled	*Description: For being able to run AT scripts of type task in the AT GUI (tab Deployment), it is required to enable the 'Start task' button which is hidden by default. This is done by setting this system property to true* *Type:* Boolean *Restart required:* No

Module	Property	Explanation
		System: No *Optional:* Yes *Example value:* true *Since:* 6.9.4.0
cmas-core-cache	cache-cluster-name	*Description:* JBoss cache cluster name *Type:* String *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* 635a6de1-629a-4129-8299-2d98633310f0 *Since:* 6.4.0
cmas-core-cache	eviction.event.queue.size	*Description:* *Type:* Integer *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* 200000 *Since:* 6.4.0
cmas-core-cache	eviction.max.nodes	*Description:* *Type:* Integer *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* 100000 *Since:* 6.4.0
cmas-core-cache	eviction.wakeup.interval	*Description:* *Type:* Integer *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* 3000 *Since:* 6.4.0
cmas-core-index-common	big.task.minimum.size	*Description:* How many parts task at least should have to be handled by Indexer with low priority. *Type:* Integer *Restart required:* No *System:* Yes

Module	Property	Explanation
		Optional: No *Example value:* 15 (default) *Since:* 6.8.3
cmas-core-index-common	database.notification.enabled	*Description:* Indicates whether index update database notification channel should be used instead of jms. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* false *Since:* 6.8.4.7
cmas-core-index-common	database.notification.redelivery.delay.seconds	*Description:* In case of index update database notification channel, indicates notification redelivery delay when exception occurs. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 60 *Since:* 6.8.4.7
cmas-core-index-common	database.notification.redelivery.max.attempts	*Description:* In case of index update database notification channel, indicates maximumn redelivery attempts when exception occurs. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 60 *Since:* 6.8.4.7
cmas-core-index-common	disable.admin.task.auto.commit	*Description:* All tasks created for index update will be automatically executed right after creation. *Type:* Boolean *Restart required:* No

Module	Property	Explanation
		System: Yes *Optional:* No *Example value:* false *Since:* 6.6.1
cmas-core-index-common	index.attachment	*Description:* Describes if content of attachments is indexed. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* true *Since:* 6.4.3
cmas-core-index-common	index.history	*Description:* Describes if unit and ticket history are indexed. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* false *Since:* 6.1.0
cmas-core-index-common	index.status	*Description:* Status of the Indexer, possible values RED, YELLOW, GREEN, will be displayed in the Admin-Tool. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* GREEN *Since:* 6.6.1
cmas-core-index-common	index.task.worker.threads	*Description:* How many threads will be used to execute batch index tasks (synchronization, administrative, and repair tasks). *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 1 (default) (we

Module	Property	Explanation
		recommend to use a value not larger than 2) *Since:* 6.6.14, 6.7.3
cmas-core-index-common	index.version.current	*Description:* Holds information about current (possibly old) index version. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 1 (default) *Since:* 6.7.0
cmas-core-index-common	index.version.newest	*Description:* Holds information about which index version is considered newest. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 1 (default) *Since:* 6.7.0
cmas-core-index-common	indexed.assets.per.thread.in.memory	*Description:* How many assets should be loaded into memory at once during indexing per one thread. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 200 (default) *Since:* 6.8.0
cmas-core-index-common	indexed.engineers.per.thread.in.memory	*Description:* How many engineers should be loaded into memory at once during indexing per one thread. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 300 (default) *Since:* 6.6.14, 6.7.3

Module	Property	Explanation
cmas-core-index-common	indexed.tickets.per.thread.in.memory	*Description:* How many tickets should be loaded into memory at once during indexing per one thread. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 100 (default) *Since:* 6.6.14, 6.7.3
cmas-core-index-common	indexed.units.per.thread.in.memory	*Description:* How many units should be loaded into memory at once during indexing per one thread. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 200 (default) *Since:* 6.6.14, 6.7.3
cmas-core-index-common	synchronize.master.address	*Description:* Value of *-Dcmas.http.host.port* informing how to connect to indexing master server. Default null. Since 6.6.17 this value is configurable in set-up to designate initial indexing master server. Please note that changing this value is only allowed when all cluster nodes index changes receivers are stopped. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 127.0.0.1:80 *Since:* 6.6.0
cmas-core-index-common	synchronize.master.security.token	*Description:* The password for accessing the index snapshot via URL, e.g. for index synchronization or for back-ups. *Type:* String

Module	Property	Explanation
		Restart required: No *System:* Yes *Optional:* Yes *Example value:* token *Since:* 6.6.0
cmas-core-index-common	synchronize.master.security.user	*Description:* The user name for accessing the index snapshot via URL, e.g. for index synchronization or for back-ups. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* user *Since:* 6.6.0
cmas-core-index-common	synchronize.master.timeout.minutes	*Description:* How much time master server may constantly fail until new master gets elected with index fix procedure. Default 5. Since 6.6.17 this value is configurable in set-up where zero means that master server will never change (failover mechanism is off). *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 5 *Since:* 6.6.0
cmas-core-index-common	synchronize.megabits.per.second	*Description:* How much bandwidth can master server consume to transfer index changes to all slave servers. Default 85. Please do not use all available bandwidth to transfer index changes between hosts. This will most probably partition cluster as some subsystems will not be able to communicate. *Type:* Integer *Restart required:* No

Module	Property	Explanation
		System: Yes *Optional:* No *Example value:* 85 *Since:* 6.6.0
cmas-core-index-common	synchronize.sleep.millis	*Description:* How often each slave server polls master server for index changes. Default 1000. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 1000 *Since:* 6.6.0
cmas-core-security	admin.email	*Description:* The e-mail address of the ConSol*CM administrator. The value which you have entered during system set-up is used initially. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* myuser@consol.de *Since:* 6.0
cmas-core-security	admin.login	*Description:* The name of the ConSol*CM administrator. The value which you have entered during system set-up is used initially. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* admin *Since:* 6.0
cmas-core-security	authentication.method	*Description:* User authentication method (internal CM database or LDAP authentication). Allowed values are *LDAP* or *DATABASE*. *Type:* String

Module	Property	Explanation
		Restart required: No *System:* Yes *Optional:* No *Example value:* DATABASE *Since:* 6.0
cmas-core-security	contact.authentication.method	*Description:* Indicates contact authentication method, where possible values are *DATABASE* or *LDAP* or *LDAP,DATABASE* or *DATABASE,LDAP.* *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Since:* 6.9.3.0
cmas-core-security	contact.inherit.permissions.only.to.own.customer.group	*Description:* Indicates whether authenticated contact inherits all customer group permissions from representing engineer (false) or only permission to own customer group (true). *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Since:* 6.9.2.3
cmas-core-security	kerberos.v5.enabled	*Description:* Flag which indicates whether SSO via Kerberos is enabled. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* false (default if Kerberos has not been enabled during system set-up) *Since:* 6.2.0
cmas-core-security	kerberos.v5.username.regex	*Description:* Regular expression used for mapping Kerberos principal to CM user login. *Type:* String

Module	Property	Explanation
		Restart required: No *System:* Yes *Optional:* No *Example value:* (.*)@.* *Since:* 6.2.0
cmas-core-security	ldap.authentication	*Description:* Authentication method used when using LDAP authentication. *Type:* String *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* simple *Since:* 6.0
cmas-core-security	ldap.basedn	*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
cmas-core-security	ldap.contact.name.basedn	*Description:* Base path to search for contact DN by LDAP ID (e.g. ou=accounts,dc=consol,dc=de). *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Since:* 6.9.3.0
cmas-core-security	ldap.contact.name.password	*Description:* Password to look up contact DN by LDAP ID. If not set, anonymous account is used. *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Since:* 6.9.3.0
cmas-core-security	ldap.contact.name.providerurl	

Module	Property	Explanation
		Description: Address of the LDAP server (ldap[s]://host:port). *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Since:* 6.9.3.0
cmas-core-security	ldap.contact.name.searchattr	*Description:* Attribute to search for contact DN by LDAP ID (e.g. uid). *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Since:* 6.9.3.0
cmas-core-security	ldap.contact.name.userdn	*Description:* User DN to look up contact DN by LDAP ID. If not set, anonymous account is used. *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Since:* 6.9.3.0
cmas-core-security	ldap.initialcontextfactory	*Description:* Class name for initial context factory of LDAP implementation when using LDAP authentication. If it is not set, com.sun.jndi.ldap. LdapCtxFactory is being used as a value. *Type:* String *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* com.sun.jndi. ldap.LdapCtxFactory *Since:* 6.0
cmas-core-security	ldap.password	*Description:* Password for connecting to LDAP to look up users (when using LDAP authentication). Only needed if

Module	Property	Explanation
		look-up cannot be done anonymously. *Type:* Password *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.1.2
cmas-core-security	ldap.providerurl	*Description:* LDAP provider (when using LDAP authentication). *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* ldap://ldap.consol.de:389 *Since:* 6.0
cmas-core-security	ldap.searchattr	*Description:* Search attribute for looking up LDAP entry connected to CM6 login. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* uid *Since:* 6.0
cmas-core-security	ldap.userdn	*Description:* LDAP user for connecting to LDAP to look up users (when using LDAP authentication). Only needed if look-up cannot be done anonymously. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.1.2
cmas-core-server	attachment.allowed.types	*Description:* Comma-separated list of allowed filename extensions (if no value defined, all file extensions are allowed).

Module	Property	Explanation
		Type: String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* txt,zip,doc *Since:* 6.5.0
cmas-core-server	attachment.max.size	*Description:* Maximum attachment size in MB *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 100 *Since:* 6.4.0
cmas-core-server	config.data.version	*Description:* *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 11 *Since:* 6.0
cmas-core-server	defaultCommentClassName	*Description:* Default text class name for comments *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Example value:* *Since:* 6.3.0
cmas-core-server	defaultIncommingMailClassName	*Description:* Default text class name for incoming e-mails *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Example value:* *Since:* 6.3.0
cmas-core-server	defaultOutgoingMailClassName	*Description:* Default text class name for outgoing e-mails *Type:* String *Restart required:* No

Module	Property	Explanation
		System: No *Optional:* Yes *Example value:* *Since:* 6.3.0
cmas-core-server	fetchSize.strategy	*Description:* Strategy selected to set fetch size on jdbc result sets. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* FetchSizePageBasedStrategy, FetchSizeThresholdStrategy, FetchSizeFixedStrategy *Since:* 6.8.4.1
cmas-core-server	fetchSize.strategy. FetchSizeFixedStrategy.value	*Description:* Sets fetch size value if selected strategy to set fetch size is *FetchSizeFixedStrat egy*. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 150 *Since:* 6.8.4.1
cmas-core-server	fetchSize.strategy. FetchSizePageBasedStrategy. limit	*Description:* Sets maximum fetch size value if selected strategy to set fetch size is *FetchSizePageB asedStrategy*. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 10000 *Since:* 6.8.4.1
cmas-core-server	fetchSize.strategy. FetchSizeThresholdStrategy. value	*Description:* Sets fetch size threshold border values if selected strategy to set fetch size is *FetchSizeThresholdStrate gy*. *Type:* Integer

Module	Property	Explanation
		Restart required: No *System:* Yes *Optional:* Yes *Example value:* 150,300,600,1000 *Since:* 6.8.4.1
cmas-core-server	last.config.change	*Description:* Random UUID created during last change in config *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* 2573c7b7-2bf5-47ff-b5a2-bad31951a266 *Since:* 6.1.0, 6.2.1
cmas-core-server	ldap.certificate.basedn	*Description:* Base DN for certificates location in LDAP tree. If not provided, *cmas-core-security*, *ldap.basedn* is taken. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* ou=accounts, dc=consol,dc=de *Since:* 6.8.4
cmas-core-server	ldap.certificate.content.attribute	*Description:* LDAP attribute name used where certificate data is stored in LDAP tree. Default value is: usercertificate. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* usercertificate *Since:* 6.8.4
cmas-core-server	ldap.certificate.password	*Description:* LDAP Certificates manager password. If not set, *cmas-core-security*, *ldap.password* is taken.

Module	Property	Explanation
		Type: String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.8.4
cmas-core-server	ldap.certificate.providerurl	*Description:* LDAP Certificates provider URL. If not set, *cmas-core-security*, *ldap.providerurl* is taken. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* ldap://myserver.consol.de:389 *Since:* 6.8.4
cmas-core-server	ldap.certificate.searchattr	*Description:* LDAP attribute name used to search for certificate in LDAP tree. Default value is: mail. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* mail *Since:* 6.8.4
cmas-core-server	ldap.certificate.userdn	*Description:* LDAP Certificates manager DN. If not set, *cmas-core-security*, *ldap.userdn* is taken. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.8.4
cmas-core-server	mail.notification.engineerChange	*Description:* Flag if notification e-mail should be sent when engineer of ticket is changed. *Type:* Boolean *Restart required:* No *System:* Yes

Module	Property	Explanation
		Optional: No *Example value:* true *Since:* 6.1.0
cmas-core-server	mail.notification.sender	*Description: From* address for notification e-mails when engineer of ticket is changed. If not set, *cmas-core-security*, *admi n.email* is used instead. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* cm6notification@cm6installation *Since:* 6.6.3
cmas-core-server	mail.smtp.email	*Description:* SMTP e-mail URL for outgoing e-mails *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* smtp://myserver. consol.de:25 *Since:* 6.0
cmas-core-server	mail.smtp.envelopesender	*Description:* E-mail address used as sender in SMTP envelope. If not set, the *From:* address of the e-mail is used. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* mysender@mydomain.com *Since:* 6.5.7
cmas-core-server	max.licences.perUser	*Description:* Sets maximum licenses single user can use (e.g logging in from different browsers). By default this value is not restricted. *Type:* Integer

Module	Property	Explanation
		Restart required: No *System:* Yes *Optional:* Yes *Example value:* 10 *Since:* 6.8.4.5
cmas-core-server	monitoring.engineer.login	*Description:* Login of monitoring engineer *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* nagios *Since:* 6.9.3.0
cmas-core-server	monitoring.unit.login	*Description:* Login of monitoring unit *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* nagios *Since:* 6.9.3.0
cmas-core-server	nimh.enabled	*Description:* Enables nimh service. Must be suffixed with nodeid in cluster e.g. nimh.enabled.NODEID = true *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* false *Since:* 6.9.4.0
cmas-core-server	server.session.archive.reaper.interval	*Description:* Server archived sessions' reaper interval (in seconds) *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 60 *Since:* 6.7.1
cmas-core-server	server.session.archive.timeout	

Module	Property	Explanation
		Description: Server sessions archive validity timeout (in days). After this time session info is removed from db. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 31 *Since:* 6.7.1
cmas-core-server	server.session.reaper.interval	*Description:* Server inactive (ended) sessions' reaper interval (in seconds) *Type:* Integer *Restart required:* Only Session Service *System:* Yes *Optional:* No *Example value:* 60 *Since:* 6.6.1, 6.7.1
cmas-core-server	server.session.timeout	*Description:* Server session timeout (in seconds) for connected clients. Each client can overwrite this timeout with custom value using its ID (ADMIN_TOOL, WEB_CLIENT, WORKFLOW_EDITOR, TRACK (before 6.8 please use PORTER), ETL, REST) appended to property name, e.g. server.session.timeout.ADMIN_TOOL *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 1800 *Since:* 6.6.1, 6.7.1
cmas-core-server	skip.wfl.transfer.cleanup	*Description:* If set to true skips workflow cleanup after transfer *Type:* Boolean *Restart required:* No

Module	Property	Explanation
		System: No *Optional:* Yes *Example value:* false (default) *Since:* 6.9.4.1
cmas-core-server	tickets.delete.size	*Description:* Property that defines a number of tickets deleted per transaction. By default it is set to 10. *Type:* Integer *Restart required:* Only Session Service *System:* Yes *Optional:* No *Example value:* 10 *Since:* 6.8.1
cmas-core-server	ticket.delete.timeout	*Description:* Transaction timeout (in seconds) for deleting tickets *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 60 *Since:* 6.1.3
cmas-core-server	unit.replace.batchSize	*Description:* Describes number of objects to be processed in unit replace action. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 5 *Since:* 6.8.2
cmas-core-server	unit.replace.timeout	*Description:* Transaction timeout (seconds) of unit replacement action step. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 120 *Since:* 6.8.2

Module	Property	Explanation
cmas-core-server	unused.content.remover.cluster.node.id	*Description:* Value of a *cmas.clusternode.id* designating node which will remove unused ticket attachments and unit content entries. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 1 (assuming cluster node started with -Dcmas.clusternode.id=1 parameter) *Since:* 6.9.0.0
cmas-core-server	unused.content.remover.enabled	*Description:* Flag whether unused ticket attachments and unit content entries removal should take place. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* true *Since:* 6.9.0.0
cmas-core-server	unused.content.remover.polling.minutes	*Description:* How often unused ticket attachments and unit content entries should be checked for removal. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 15 *Since:* 6.9.0.0
cmas-core-server	unused.content.remover.ttl.minutes	*Description:* Minimum interval after which unused ticket attachments and unit content entries can be removed. *Type:* Integer *Restart required:* No *System:* Yes

Module	Property	Explanation
		Optional: No *Example value:* 1440 *Since:* 6.9.0.0
cmas-core-server	warmup.executor.enabled	*Description:* Flag whether server should asynchronously warm up during startup (e.g. fill some of internal caches) *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* true *Since:* 6.9.4.2
cmas-core-shared	cluster.mode	*Description:* Flag if CMAS is running in cluster. *Type:* Boolean *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* false *Since:* 6.1.0
cmas-core-shared	data.directory	*Description:* Directory for CMAS data (e.g. index) *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* C:\Users\user\cmas *Since:* 6.0
cmas-dwh-server	autocommit.cf.changes	*Description:* *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* false *Since:* 6.7.0
cmas-dwh-server	batch-commit-interval	*Description:* Number of objects in a JMS message. Higher value means better transfer performance and bigger memory

Module	Property	Explanation
		usage. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 100 *Since:* 6.0.0
cmas-dwh-server	communication.channel	*Description:* Communication channel. Possible values are DIRECT (database communication channel, default value since 6.9.4.1), JMS (default value before 6.9.4.1). Before 6.9.4.1 it has to be manually added. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* DIRECT *Since:* 6.8.5.0
cmas-dwh-server	dwh.mode	*Description:* Current mode of 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
cmas-dwh-server	ignore-queues	*Description:* By adding a comma separated list of queue names it is configured that tickets of these queues are not transferred to the DWH. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* QueueName1,

Module	Property	Explanation
		QueueName2,QueueName3 *Since:* 6.6.19 *Removed in:* 6.8.1
cmas-dwh-server	is.cmrf.alive	*Description:* As a starting point time of sending last message to CMRF should be used. If response from CMRF is not received after value (in seconds) it should create a DWH operation status with error message that CMRF is down. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 1200 *Since:* 6.7.0
cmas-dwh-server	java.naming.factory.initial	*Description:* Factory class for DWH context factory. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* org.jnp.interfaces.NamingContextFactory *Since:* 6.0.1
cmas-dwh-server	java.naming.factory.url.pkgs	*Description:* *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* org.jboss.naming:org.jnp.interfaces *Since:* 6.0.1
cmas-dwh-server	java.naming.provider.url	*Description:* URL of naming provider *Type:* String *Restart required:* No *System:* Yes

Module	Property	Explanation
		Optional: No *Example value:* localhost *Since:* 6.0.1
cmas-dwh-server	notification.error.description	*Description:* Text for error e-mails from DWH *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* Error occurred *Since:* 6.0.1
cmas-dwh-server	notification.error.from	*Description: From* address for error e-mails from DWH *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.0.1
cmas-dwh-server	notification.error.subject	*Description:* Subject for error e-mails from DWH *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* Error occurred *Since:* 6.0.1
cmas-dwh-server	notification.error.to	*Description: To* address for error e-mails from DWH *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* myuser@consol.de *Since:* 6.0.1
cmas-dwh-server	notification.finished_successfully.description	*Description:* Text for e-mails from DWH when transfer finished successfully. *Type:* String *Restart required:* No *System:* Yes

Module	Property	Explanation
		Optional: No *Example value:* Transfer finished successfully *Since:* 6.0.1
cmas-dwh-server	notification.finished_successfully.from	*Description: From* address for e-mails from DWH when transfer finished successfully. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.0.1
cmas-dwh-server	notification.finished_successfully.subject	*Description:* Subject for e-mails from DWH when transfer finished successfully. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* Transfer finished successfully *Since:* 6.0.1
cmas-dwh-server	notification.finished_successfully.to	*Description: To* address for e-mails from DWH when transfer finished successfully. *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* myuser@consol.de *Since:* 6.0.1
cmas-dwh-server	notification.finished_unsuccessfully.description	*Description:* Text for e-mails from DWH when transfer finished unsuccessfully. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* Transfer finished unsuccessfully *Since:* 6.0.1

Module	Property	Explanation
cmas-dwh-server	notification. finished_unsuccessfully.from	*Description: From* address for e-mails from DWH when transfer finished unsuccessfully. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.0.1
cmas-dwh-server	notification. finished_unsuccessfully.subject	*Description:* Subject for e-mails from DWH when transfer finished unsuccessfully. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* Transfer finished unsuccessfully *Since:* 6.0.1
cmas-dwh-server	notification. finished_unsuccessfully.to	*Description: To* address for e-mails from DWH when transfer finished unsuccessfully. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* myuser@consol. de *Since:* 6.0.1
cmas-dwh-server	notification.host	*Description:* Mail (SMTP) server hostname for sending DWH e-mails *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* myserver.consol. de *Since:* 6.1.0
cmas-dwh-server	notification.password	*Description:* Password for sending DWH e-mails (optional) *Type:* String

Module	Property	Explanation
		Restart required: No *System:* Yes *Optional:* Yes *Since:* 6.1.0
cmas-dwh-server	notification.port	*Description:* SMTP port for sending DWH e-mails *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 25 *Since:* 6.1.0
cmas-dwh-server	notification.protocol	*Description:* The protocol used for sending e-mails from DWH. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* pop3\
cmas-dwh-server	notification.username	*Description:* (SMTP) User name for sending DWH e-mails *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* maz *Since:* 6.1.0
cmas-dwh-server	skip-ticket	*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
cmas-dwh-server	skip-ticket-history	*Description:* History of ticket is not transferred during transfer /update. *Type:* Boolean

Module	Property	Explanation
		Restart required: No *System:* Yes *Optional:* No *Example value:* false *Since:* 6.6.19 *Removed in:* 6.8.1
cmas-dwh-server	skip-unit	*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
cmas-dwh-server	skip-unit-history	*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
cmas-dwh-server	split.history	*Description:* Changes the SQL that fetches the history for the tickets during DWH transfer not to 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
cmas-dwh-server	unit.transfer.order	*Description:* Define in which order data object groups should be transferred to the DWH. *Type:* String

Module	Property	Explanation
		Restart required: No *System:* Yes *Optional:* Yes *Example value: company; customer* *Since:* 6.6.19 *Removed in:* 6.8.1
cmas-esb-core	esb.directory	*Description:* Directory used by ESB (Mule) *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* C:\Users\user\cmas\mule *Since:* 6.0
cmas-esb-mail	mail.attachments.validation.info.sender	*Description:* Sets *From* header of attachments type error notification e-mail. As a default the e-mail address of the administrator which you have entered during system set-up is used. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* admin@consolcm.com *Since:* 6.7.5
cmas-esb-mail	mail.attachments.validation.info.subject	*Description:* Sets subject of attachments type error notification e-mail. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* E-mail was not processed because its attachments were rejected!!! *Since:* 6.7.5

Module	Property	Explanation
cmas-esb-mail	mail.callname.pattern	*Description:* Regular expression for subject of incoming e-mails. Available as TICKET_NAME_PATTERN_FORMAT in incoming e-mail scripts. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* .*?Ticket\s+\((\S+)\).* *Since:* 6.0
cmas-esb-mail	mail.cluster.node.id	*Description:* Only the node whose mail.cluster.node.id equals cmas.clusternode.id will start the Mule ESB mail services. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* unspecified *Since:* 6.6.5
cmas-esb-mail	mail.db.archive	*Description:* If property is set to *true*, incoming e-mails are archived in the database. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* false (default) *Since:* 6.8.5.5
cmas-esb-mail	mail.delete.read	*Description:* Determines whether CM deletes messages fetched via IMAP(S). Setting value to *true* will cause deletion of messages after fetching. Default is to not delete messages fetched via IMAP(S). Note: Messages fetched via POP3(S) will always be deleted. *Type:* Boolean *Restart required:* No

Module	Property	Explanation
		System: Yes *Optional:* No *Example value:* true *Since:* 6.7.3
cmas-esb-mail	mail.encryption	*Description:* If property is set to *tr ue*, the encrypt check box in the Ticket E-Mail Editor is checked by default. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* true (default = false) *Since:* 6.8.4.0
cmas-esb-mail	mail.incoming.uri	*Description:* URL for incoming e-mails *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* pop3://cm-incoming-user:password@localhost:10110 *Since:* 6.0 This value should not be edited here using the system properties pop-up window, but the mailboxes should be configured using the tab E-mail. Using this standard feature all entries are controlled - i. e. for each mailbox which is added, CM establishes a test connection during mailbox set-up. That way it is not possible to enter wrong values.

Module	Property	Explanation
cmas-esb-mail	mail.max.restarts	*Description:* Maximum number of mail service restarts before giving up *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 3 *Since:* 6.0
cmas-esb-mail	mail.mime.strict	*Description:* If set to *false*, e-mail addresses are not parsed for strict MIME compliance. Default is *true*, which means check for strict MIME compliance. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* false *Since:* 6.6.17, 6.7.3
cmas-esb-mail	mail.mule.service	*Description: From* address for e-mails sent by Mule service *Type:* EMail *Restart required:* No *System:* Yes *Optional:* No *Example value:* myuser@consol.de *Since:* 6.0
cmas-esb-mail	mail.polling.interval	*Description:* Mail polling interval in ms *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 60000 *Since:* 6.0
cmas-esb-mail	mail.process.error	*Description: To* address for error e-mails from Mule. As a default the e-mail address of the administrator which you have

Module	Property	Explanation
		entered during system set-up is used. *Type:* EMail *Restart required:* No *System:* Yes *Optional:* No *Example value:* myuser@consol.de *Since:* 6.0
cmas-esb-mail	mail.process.retry.attempts	*Description:* Number of retries when processing e-mail *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 3 *Since:* 6.0.2
cmas-esb-mail	mail.process.timeout	*Description:* Mail processing timeout in seconds *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 60 *Since:* 6.1.3
cmas-esb-mail	mail.redelivery.retry.count	*Description:* Indicates the number of retries of re-delivering an e-mail from the CM system. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 3 *Since:* 6.1.0
cmas-nimh	filesystem.polling.threads.number	*Description:* Number of threads started for db mails' queue polling. Default: 1 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes

Module	Property	Explanation
		Example value: 10 *Since:* 6.4.0
cmas-nimh	filesystem.polling.threads. shutdown.timeout.seconds	*Description:* Waiting time after the shutdown signal. When the timeout reached, thread will be terminated. Default: 60 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	filesystem.polling.threads. watchdog.interval.seconds	*Description:* Watchdog thread interval. Default: 30 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	filesystem.task.enabled	*Description:* With this property service thread related to given poller can be disabled. Default: true *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* true *Since:* 6.4.0
cmas-nimh	filesystem.task.interval.seconds	*Description:* Default interval for polling mailboxes. Default: 60 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	filesystem.task.polling.folder	*Description:* Polling folder location which will be scanned

Module	Property	Explanation
		for emails in the format of eml files. Default: "mail" subdir of cmas data directory *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Example value:* c://cmas//mail *Since:* 6.4.0
cmas-nimh	filesystem.task.timeout.seconds	*Description:* After this time (of inactivity) the service thread is considered as damaged and automatically restarted. Default: 120 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	filesystem.task.transaction.timeout.seconds	*Description:* Default transaction timeout for mail fetching transactions. Should be correlated with number of messages fetched at once. Default: 60 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	mailbox.1.connection.host	host (server) for first configured mailbox. Will overwrite the default parameter *mailbox.default.connection.host*
cmas-nimh	mailbox.1.connection.password	password for first configured mailbox. Will overwrite the default parameter *mailbox.default.connection.password*
cmas-nimh	mailbox.1.connection.port	

Module	Property	Explanation
		port for first configured mailbox. Will overwrite the default parameter *mailbox.default.connection.port*
cmas-nimh	mailbox.1.connection.protocol	protocol (e.g., IMAP or POP3) for first configured mailbox. Will overwrite the default parameter *mailbox.default.connection.protocol*
cmas-nimh	mailbox.1.connection.username	username for first configured mailbox. Will overwrite the default parameter *mailbox.default.connection.username*
cmas-nimh	mailbox.2.connection.host	host (server) for second configured mailbox. Will overwrite the default parameter *mailbox.default.connection.host*
cmas-nimh	mailbox.2.connection.password	password for second configured mailbox. Will overwrite the default parameter *mailbox.default.connection.password*
cmas-nimh	mailbox.2.connection.port	port for second configured mailbox. Will overwrite the default parameter *mailbox.default.connection.port*
cmas-nimh	mailbox.2.connection.protocol	protocol (e.g., IMAP or POP3) for second configured mailbox. Will overwrite the default parameter *mailbox.default.connection.protocol*
cmas-nimh	mailbox.2.connection.username	username 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.

Module	Property	Explanation
cmas-nimh	mailbox.default.connection.host	*Description:* Host (server name) of a given mailbox from which the poller reads e-mails. *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Example value:* 10.10.1.157 *Since:* 6.4.0
cmas-nimh	mailbox.default.connection.password	*Description:* Password for given mailbox from which the poller reads e-mails. *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Example value:* consol *Since:* 6.4.0
cmas-nimh	mailbox.default.connection.port	*Description:* Port for a given mailbox from which the poller reads e-mails. *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Example value:* 143 *Since:* 6.4.0
cmas-nimh	mailbox.default.connection.protocol	*Description:* Poller's protocol e.g: IMAP or POP3. No default value *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Example value:* imap *Since:* 6.4.0
cmas-nimh	mailbox.default.connection.username	*Description:* Username for a given mailbox from which the poller reads e-mails. *Type:* String *Restart required:* No *System:* No

Module	Property	Explanation
		Optional: Yes *Example value:* username *Since:* 6.4.0
cmas-nimh	mailbox.default.session.mail. debug	*Description:* Example javax.mail property - allows for more detailed javax.mail session debugging *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* true *Since:* 6.4.0
cmas-nimh	mailbox.default.session.mail. imap.timeout	*Description:* Example javax.mail property *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 120 *Since:* 6.4.0
cmas-nimh	mailbox.default.session.mail. mime.address.strict	*Description:* Example javax.mail property - counterpart of the old mule mail.mime.strict, allows to set not so strict mail header parsing *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* true *Since:* 6.4.0
cmas-nimh	mailbox.default.session.mail. pop3.timeout	*Description:* Example javax.mail property. *Type:* *Restart required:* *System:* *Optional:* *Example value:* *Since:* 6.4.0
cmas-nimh		

Module	Property	Explanation
	mailbox.default.task.delete.read.messages	*Description:* This defines whether messages should be removed from the mailbox after processing. For IMAP protocol messages are marked as SEEN by default. For POP3 protocol, when flag is set to true the message is removed, otherwise remains on server and will result in infinite reads. Default: false. *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* false *Since:* 6.4.0
cmas-nimh	mailbox.default.task.enabled	*Description:* With this property service thread related to given poller can be disabled. Default: true *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* false *Since:* 6.4.0
cmas-nimh	mailbox.default.task.interval.seconds	*Description:* Default interval for polling mailboxes. Default: 60 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	mailbox.default.task.max.message.size	*Description:* Maximum size of e-mail messages in bytes. Messages with a size greater than the value of this property can be processed from the Admin-Tool. Default set to 10Mb : 10485760

Module	Property	Explanation
		Type: Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 10485760 *Since:* 6.4.0
cmas-nimh	mailbox.default.task.max. messages.per.run	*Description:* Number of messages fetched at once from mailbox. Must be correlated with transaction timeout. Default set to: 20 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	mailbox.default.task.timeout. seconds	*Description:* After this time (of inactivity) the service thread is considered as damaged and automatically restarted. Default: 120 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	mailbox.default.task.transaction. timeout.seconds	*Description:* Default transaction timeout for mail fetching transactions. Should be correlated with number of messages fetched at once. Default: 60 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh		

Module	Property	Explanation
	mailbox.polling.threads.mail.log.enabled	*Description:* Enables mail logging which is especially crucial in cluster environment (used as semaphore there) *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Example value:* true (default) *Since:* 6.9.4.1
cmas-nimh	mailbox.polling.threads.number	*Description:* Number of threads for accessing mailboxes. Default: 1 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 1 *Since:* 6.4.0
cmas-nimh	queue.polling.threads.number	*Description:* Number of threads started for mails' queue polling. Default: 1 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 1 *Since:* 6.4.0
cmas-nimh	queue.polling.threads.shutdown.timeout.seconds	*Description:* Waiting time after the shutdown signal. When the timeout is reached, the thread will be terminated. Default: 60 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	queue.polling.threads.watchdog.interval.seconds	*Description:* Watchdog thread interval. Default: 30 *Type:* Integer

Module	Property	Explanation
		Restart required: No *System:* No *Optional:* Yes *Example value:* 30 *Since:* 6.4.0
cmas-nimh	queue.task.error.pause.seconds	*Description:* Maximum number of seconds, the queue poller waits after infrastructure (e.g. database) error. Default 180 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 180 *Since:* 6.4.0
cmas-nimh	queue.task.interval.seconds	*Description:* Main mails' queue polling thread interval. Default: 15 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 15 *Since:* 6.4.0
cmas-nimh	queue.task.max.retries	*Description:* Maximum number of e-mail processing retries after an exception. When reached, the e-mail is moved to the e-mail archive. This e-mail can be rescheduled again using NIMH API (or the Admin-Tool). *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 10 *Since:* 6.4.0
cmas-nimh	queue.task.timeout.seconds	*Description:* After this time (of inactivity) the service thread is considered as damaged and automatically restarted. Default:

Module	Property	Explanation
		600 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 600 *Since:* 6.4.0
cmas-nimh	queue.task.transaction.timeout.seconds	*Description:* Transaction timeout for mail processing in the pipe. Default: 60 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh-extension	mail.attachments.validation.info.sender	*Description:* Sets FROM header of attachments type error notification mail *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* admin@mail.com *Since:* 6.7.5 This is an equivalent to the old *cmas-esb-mail, mail.attachments.validation.info.sender*
cmas-nimh-extension	mail.attachments.validation.info.subject	*Description:* Sets subject of attachments type error notification mail *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* Mail was not

Module	Property	Explanation
		processed because its attachments were rejected!!! *Since:* 6.7.5 This is an equivalent to the old *cmas-esb-mail, mail.attachments. validation.info.subject*
cmas-nimh-extension	mail.db.archive	*Description:* If property is set to true, incoming emails are archived in the database *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* false (default) *Since:* 6.8.5.5 This is an equivalent to the old *cmas-esb-mail, mail.db.archive*
cmas-nimh-extension	mail.error.from.address	This is an equivalent to the old *cmas-esb-mail, mail.mule.service*
cmas-nimh-extension	mail.error.to.address	This is an equivalent to the old *cmas-esb-mail, mail.process.error*
cmas-nimh-extension	mail.mule.service	*Description:* From address for mails sent by Mule service *Type:* EMail

Module	Property	Explanation
		Restart required: No *System:* Yes *Optional:* No *Example value:* myuser@consol.de *Since:* 6.4.0
cmas-nimh-extension	mail.on.error	*Description:* When set true an error mail is sent to the above configured address in case email message could not be processed. Default: false *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* false *Since:* 6.4.0
cmas-nimh-extension	mail.process.error	*Description:* To address for error mails from Mule *Type:* EMail *Restart required:* No *System:* Yes *Optional:* No *Example value:* myuser@consol.de *Since:* 6.4.0
cmas-nimh-extension	mail.ticketname.pattern	*Description:* *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* .*?Ticket\s+\((\S+)\).* *Since:* 6.4.0
cmas-setup-hibernate	hibernate.dialect	*Description:* The dialect used by hibernate. Usually set during initial set-up (depending on the database system). *Type:* String *Restart required:* No *System:* Yes

Module	Property	Explanation
		Optional: No *Example value:* org.hibernate.dialect.MySQL5InnoDBDialect *Since:* 6.0
cmas-setup-hibernate	cmas.dropSchemaBeforeSetup	*Description:* Flag if schema is to be (was) dropped during setup *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* true *Since:* 6.0
cmas-setup-manager	initialized	*Description:* Flag if CMAS is initialized. If this value is missing or not *true*, set-up will be performed. *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 set-up at the next start, i.e. all data of the existing system is lost, including system properties!!!
cmas-setup-scene	scene	*Description:* Scene file which was imported during set-up (can be empty). *Type:* String *Restart required:* No *System:* Yes *Optional:* No

Module	Property	Explanation
		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
cmas-workflow-engine	jobExecutor.adminMail	*Description:* E-mail 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
cmas-workflow-engine	jobExecutor.idleInterval.seconds	*Description:* Determines how often job executor thread will look for new jobs to execute. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 5 (default) *Since:* 6.8.0
cmas-workflow-engine	jobExecutor.jobMaxRetries	*Description:* *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 5 (default) *Since:* 6.8.0
cmas-workflow-engine	jobExecutor.jobMaxRetriesReachedSubject	*Description:* *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* Job maximum

Module	Property	Explanation
		retries reached. Job was removed!!! (default) *Since:* 6.8.0
cmas-workflow-engine	jobExecutor.lockTimeout. seconds	*Description:* How long the job can be locked (marked for execution) by job executor. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 360 (default) *Since:* 6.8.0
cmas-workflow-engine	jobExecutor.lockingLimit	*Description:* Number of jobs locked at once (marked for execution) by job executor thread *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 10 (default) *Since:* 6.8.0
cmas-workflow-engine	jobExecutor.mailFrom	*Description:* E-mail which will be set as *From* header during admin notifications. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* jobexecutor@consol.de *Since:* 6.8.0
cmas-workflow-engine	jobExecutor.maxInactivityInterval. minutes	*Description:* Number of minutes of allowed job executor inactivity (e.g. when it is blocked by long timer execution). After this time executors threads are restarted. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes. Default value is set to 30 minutes

Module	Property	Explanation
		Example value: 15 (default) *Since:* 6.9.2.0
cmas-workflow-engine	jobExecutor.threads	*Description:* Number of job execution threads *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 1 (default) *Since:* 6.8.0
cmas-workflow-engine	jobExecutor.timerRetryInterval.seconds	*Description:* Determines how long job executor thread will wait after job execution error. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 10 (default) *Since:* 6.8.0
cmas-workflow-engine	jobExecutor.txTimeout.seconds	*Description:* Transaction timeout used for job execution *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 60 (default) *Since:* 6.8.0
cmweb-server-adapter	automatic.booking.enabled	*Description:* If enabled, time spend on creating comment /email will be measured and automatic time booking will be added. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* true *Since:* 6.9.4.2
cmweb-server-adapter	checkUserOnlineIntervalInSeconds	*Description:* The interval in seconds to check which users are online (default 180sec =

Module	Property	Explanation
		3min). *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 180 *Since:* 6.0
cmweb-server-adapter	cmoffice.enabled	*Description:* Flag if CM.Doc (former CM/Office) is enabled. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* false *Since:* 6.4.0
cmweb-server-adapter	commentRequiredForTicketCreation	*Description:* Flag if comment is a required field for ticket creation. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* true (default) *Since:* 6.2.0
cmweb-server-adapter	customizationVersion	*Description:* *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* cd58453e-f3cc-4538-8030-d15e8796a4a7 *Since:* 6.5.0
cmweb-server-adapter	data.optimization	*Description:* Defines optimization to be applied on response data. So far, 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

Module	Property	Explanation
		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 property data.optimization.*nodeId* to override 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
cmweb-server-adapter	defaultContentEntryClassName	*Description:* Default text class for new acims *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* default_class *Since:* 6.3.0
cmweb-server-adapter	defaultNumberOfCustomFieldsColumns	*Description:* Default number of columns for custom fields *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 3 *Since:* 6.2.0
cmweb-server-adapter	favoritesSizeLimit	*Description:* Maximum number of items in favorites list *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 10 *Since:* 6.0

Module	Property	Explanation
cmweb-server-adapter	globalSearchResultSizeLimit	*Description:* Maximum number of items in global (Q&E) search result *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 10 *Since:* 6.0
cmweb-server-adapter	helpFilePath	*Description:* URL for online help. If not empty, *Help* button is displayed in Web Client. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* http://www.consol.de *Since:* 6.2.1
cmweb-server-adapter	hideTicketSubject	*Description:* If set to *true*, ticket subject is hidden. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* false *Since:* 6.2.1
cmweb-server-adapter	mail.from	*Description:* Use this address if set instead of engineer e-mail address during e-mail conversation. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.1.2
cmweb-server-adapter	mail.reply.to	*Description:* When set, Web Client will display reply-to field on e-mail send, prefilled with this value. *Type:* String

Module	Property	Explanation
		Restart required: No *System:* Yes *Optional:* Yes *Since:* 6.0.1 ⛔ Please see also section Queue Administration. When you set the REPLY TO address in the outgoing e-mail script, the *mail.reply.to* system property must not be set (because it would overwrite the configured value)! That means when you use one outgoing e-mail script for a queue you have to define outgoing e-mail scripts for all queues because the *mail.reply.to* property can no longer be used.
cmweb-server-adapter	mailTemplateAboveQuotedText	*Description:* Indicates behavior of e-mail template in the Ticket E-Mail Editor when another e-mail is quoted, i.e. forwarded or replied to. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* false *Since:* 6.2.4
cmweb-server-adapter	maxSizePerPagemapInMegaBytes	*Description:* Maximum size (in MB) for each Wicket pagemap *Type:* Integer *Restart required:* No *System:* Yes

Module	Property	Explanation
		Optional: No *Example value:* 15 *Since:* 6.3.5
cmweb-server-adapter	pagemapLockDurationInSeconds	*Description:* Number of seconds to pass before 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
cmweb-server-adapter	postActivityExecutionScriptName	*Description:* Defines the name for the script which should be executed after every workflow activity, see section Default Workflow Activity Script. 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
cmweb-server-adapter	queuesExcludedFromGS	*Description:* Comma-separated list of queue names which are excluded from global search. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.0
cmweb-server-adapter	rememberMeLifetimeInMinutes	*Description:* Lifetime for *rememb er me* in minutes *Type:* Integer *Restart required:* Yes *System:* Yes *Optional:* No

Module	Property	Explanation
		Example value: 1440 *Since:* 6.0
cmweb-server-adapter	request.scope.transaction	*Description:* It allows to disable request scope transaction. By default one transaction is used per request. Setting this property to *false* there will cause one transaction per service method invocation. *Type:* Boolean *Restart required:* Yes *System:* Yes *Optional:* Yes *Example value:* true *Since:* 6.8.1
cmweb-server-adapter	searchPageSize	*Description:* Default page size for search results *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 20 *Since:* 6.0
cmweb-server-adapter	searchPageSizeOptions	*Description:* Options for page size for search results *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* 10|20|30|40|50|75|100 *Since:* 6.0
cmweb-server-adapter	serverPoolingInterval	*Description:* *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 5 *Since:* 6.1.0
cmweb-server-adapter	supportEmail	*Description:* *Type:* String

Module	Property	Explanation
		Restart required: No *System:* Yes *Optional:* Yes *Since:* 6.0
cmweb-server-adapter	themeOverlay	*Description:* Name of used theme overlay *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* kyoEUR *Since:* 6.0
cmweb-server-adapter	ticketListRefreshIntervalInSeconds	*Description:* Refresh interval for ticket list (in seconds) *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 180 *Since:* 6.0
cmweb-server-adapter	ticketListSizeLimit	*Description:* Maximum number of tickets in ticket list *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 100 *Since:* 6.0
cmweb-server-adapter	unitIndexSearchResultSizeLimit	*Description:* Maximum number of units in unit search result (e.g. when searching for contact) *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 5 *Since:* 6.0
cmweb-server-adapter	urlLogoutPath	*Description:* URL which is used when user logs out. (If no value is set, logout leads to login-mask.)

Module	Property	Explanation
		Type: String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* http://intranet.consol.de *Since:* 6.3.1
cmweb-server-adapter	webSessionTimeoutInMinutes	*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
cmweb-server-adapter	wicketAjaxRequestHeaderFilterEnabled	*Description:* This enables filter for Wicket AJAX requests, coming from stale pages with Wicket 1.4 scripting (CM6 pre-6.8.0), after update to CM6 post-6.8.0. *Type:* Boolean *Restart required:* Yes *System:* Yes *Optional:* Yes *Example value:* false *Since:* 6.8.1
cmas-workflow-jbpm	fetchLock.interval	*Description:* *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 5000 *Removed in:* 6.8.0
cmas-workflow-jbpm	fetchLock.timeout	*Description:* *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No

Module	Property	Explanation
		Example value: 15000 *Removed in:* 6.8.0
cmas-workflow-jbpm	jobExecutor.idleInterval	*Description:* *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 45000 *Removed in:* 6.8.0 *Replaced by:* jobExecutor. idleInterval.seconds
cmas-workflow-jbpm	jobExecutor. jobExecuteRetryNumber	*Description:* *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 5 *Removed in:* 6.8.0 *Replaced by:* jobExecutor. jobMaxRetries
cmas-workflow-jbpm	jobExecutor.timerRetryInterval	*Description:* *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 10000 *Removed in:* 6.8.0 *Replaced by:* jobExecutor. timerRetryInterval.seconds
cmas-workflow-jbpm	mail.sender.address	*Description: From* address for e-mails 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
cmas-workflow-jbpm	outdated.lock.age	

Module	Property	Explanation
		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
cmas-workflow-jbpm	refreshTimeInCaseOfConcurrentRememberMeRequests	*Description:* It sets the refresh time (in seconds) after which page will be reloaded in case of concurrent *remember me* requests. This feature prevents one user from occupying many licenses. Please increase that time if sessions are still occupying. *Type:* Integer *Restart required:* Yes *System:* Yes *Optional:* Yes *Example value:* 5 *Since:* 6.8.2

36 Appendix D (Important System Properties, Ordered by Area of Application)

- CMRF & DWH Configuration
- Indexer & Search Configuration
    - Indexer
    - Search Results
- LDAP Configuration
    - LDAP Configuration (If LDAP Is Used as Authentication Mode in the CM Web Client)
    - LDAP Configuration (If LDAP Is Used as Authentication Mode in CM.Track)
- E-Mail Configuration
    - Outgoing E-Mail
    - Incoming E-mail
        - Settings for ESB/Mule
        - Settings for NIMH
        - Mapping of Former Mule and New NIMH Properties
        - Attachments for Incoming E-Mails
    - E-Mail Encryption (Outgoing and Incoming)
- Activity Interval Configuration

36.1 CMRF & DWH Configuration

Module	Property	Explanation
cmas-dwh-server	autocommit.cf.changes	*Description:* *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* false *Since:* 6.7.0
cmas-dwh-server	batch-commit-interval	*Description:* Number of objects in a JMS message. Higher value means better transfer performance and bigger memory usage. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 100 *Since:* 6.0.0
cmas-dwh-server	communication.channel	*Description:* Communication channel. Possible values are *DIRECT* (database communication channel, default value since 6.9.4.1), *JMS* (default value before 6.9.4.1). Before 6.9.4.1 it has to be manually added. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* DIRECT *Since:* 6.8.5.0
cmas-dwh-server	dwh.mode	*Description:* Current mode of DWH data transfer. Possible values are *OFF*, *ADMIN*, *LIVE*. *Type:* String *Restart required:* No *System:* Yes *Optional:* No

Module	Property	Explanation
		Example value: OFF *Since:* 6.0.1
cmas-dwh-server	ignore-queues	*Description:* By adding a comma separated list of queue names it is configured that tickets of these queues are 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
cmas-dwh-server	is.cmrf.alive	*Description:* As a starting point time of sending last message to CMRF should be used. If response from CMRF is not received after value (in seconds) it should create a DWH operation status with error message that CMRF is down. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 1200 *Since:* 6.7.0
cmas-dwh-server	java.naming.factory.initial	*Description:* Factory class for DWH context factory. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* org.jnp.interfaces.NamingContextFactory *Since:* 6.0.1
cmas-dwh-server	java.naming.factory.url.pkgs	*Description:* *Type:* String *Restart required:* No

Module	Property	Explanation
		System: Yes *Optional:* No *Example value:* org.jboss. naming:org.jnp.interfaces *Since:* 6.0.1
cmas-dwh-server	java.naming.provider.url	*Description:* URL of naming provider. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* localhost *Since:* 6.0.1
cmas-dwh-server	notification.error.description	*Description:* Text for error e-mails from DWH *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* Error occurred *Since:* 6.0.1
cmas-dwh-server	notification.error.from	*Description:* FROM-address for error e-mails from DWH *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.0.1
cmas-dwh-server	notification.error.subject	*Description:* Subject for error e-mails from DWH *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* Error occurred *Since:* 6.0.1
cmas-dwh-server	notification.error.to	*Description:* TO-address for error e-mails from DWH *Type:* String *Restart required:* No *System:* Yes

Module	Property	Explanation
		Optional: No *Example value:* myuser@consol.de *Since:* 6.0.1
cmas-dwh-server	notification.finished_successfully.description	*Description:* Text for e-mails from DWH when transfer finished successfully. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* Transfer finished successfully *Since:* 6.0.1
cmas-dwh-server	notification.finished_successfully.from	*Description:* FROM-address for e-mails from DWH when transfer finished successfully. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.0.1
cmas-dwh-server	notification.finished_successfully.subject	*Description:* Subject for e-mails from DWH when transfer finished successfully. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* Transfer finished successfully *Since:* 6.0.1
cmas-dwh-server	notification.finished_successfully.to	*Description:* TO-address for e-mails from DWH when transfer finished successfully. *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* myuser@consol.de *Since:* 6.0.1

Module	Property	Explanation
cmas-dwh-server	notification. finished_unsuccessfully. description	*Description:* Text for e-mails from DWH when transfer finished unsuccessfully. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* Transfer finished unsuccessfully *Since:* 6.0.1
cmas-dwh-server	notification. finished_unsuccessfully.from	*Description:* FROM-address for e-mails from DWH when transfer finished unsuccessfully. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.0.1
cmas-dwh-server	notification. finished_unsuccessfully.subject	*Description:* Subject for e-mails from DWH when transfer finished unsuccessfully. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* Transfer finished unsuccessfully *Since:* 6.0.1
cmas-dwh-server	notification. finished_unsuccessfully.to	*Description:* TO-address for e-mails from DWH when transfer finished unsuccessfully. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* myuser@consol.de *Since:* 6.0.1
cmas-dwh-server	notification.host	*Description:* E-mail (SMTP) server hostname for sending DWH e-mails.

Module	Property	Explanation
		Type: String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* mail.consol.de *Since:* 6.1.0
cmas-dwh-server	notification.password	*Description:* Password for sending DWH e-mails (optional). *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.1.0
cmas-dwh-server	notification.port	*Description:* SMTP port for sending DWH e-mails. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 25 *Since:* 6.1.0
cmas-dwh-server	notification.protocol	*Description:* The protocol used for sending e-mails from DWH. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* pop3\
cmas-dwh-server	notification.username	*Description:* (SMTP) User name for sending DWH e-mails. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* myuser *Since:* 6.1.0
cmas-dwh-server	skip-ticket	*Description:* Tickets are not transferred during transfer /update. *Type:* Boolean *Restart required:* No

Module	Property	Explanation
		System: Yes *Optional:* No *Example value:* false *Since:* 6.6.19 *Removed in:* 6.8.1
cmas-dwh-server	skip-ticket-history	*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
cmas-dwh-server	skip-unit	*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
cmas-dwh-server	skip-unit-history	*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
cmas-dwh-server	split.history	*Description:* Changes the SQL that fetches the history for the tickets during DWH transfer, not to all tickets at once but only for one ticket per SQL. *Type:* Boolean

Module	Property	Explanation
		Restart required: No *System:* Yes *Optional:* Yes *Example value:* false *Since:* 6.8.0
cmas-dwh-server	unit.transfer.order	*Description:* Define in which order data object 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

36.2 Indexer & Search Configuration

36.2.1 Indexer

Module	Property	Explanation
cmas-core-index-common	big.task.minimum.size	*Description:* How many parts a task at least should have to be handled by the Indexer with low priority. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 15 (default) *Since:* 6.8.3
cmas-core-index-common	database.notification.enabled	*Description:* Indicates whether index update database notification channel should be used instead of JMS. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* false *Since:* 6.8.4.7
cmas-core-index-common	database.notification.redelivery.delay.seconds	*Description:* In case of index update database notification channel, indicates notification redelivery delay when exception occurs. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 60 *Since:* 6.8.4.7
cmas-core-index-common	database.notification.redelivery.max.attempts	*Description:* In case of index update database notification channel, indicates maximum

Module	Property	Explanation
		redelivery attempts when exception occurs. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 60 *Since:* 6.8.4.7
cmas-core-index-common	disable.admin.task.auto.commit	*Description:* All tasks created for index update will be executed automatically right after creation. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* false *Since:* 6.6.1
cmas-core-index-common	index.attachment	*Description:* Describes if content of attachments is indexed. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* true *Since:* 6.4.3
cmas-core-index-common	index.history	*Description:* Describes if unit and ticket history are indexed. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* false *Since:* 6.1.0
cmas-core-index-common	index.status	*Description:* Status of the Indexer (possible values RED, YELLOW, GREEN) will be displayed in the Admin Tool. *Type:* String *Restart required:* No *System:* Yes *Optional:* No

Module	Property	Explanation
		Example value: GREEN *Since:* 6.6.1
cmas-core-index-common	index.task.worker.threads	*Description:* How many threads will be used to execute batch index tasks (synchronization, administrative, and repair tasks). *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 1 (default) (we recommend to use a value not larger than 2) *Since:* 6.6.14, 6.7.3
cmas-core-index-common	index.version.current	*Description:* Holds information about current (possibly old) index version. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 1 (default) *Since:* 6.7.0
cmas-core-index-common	index.version.newest	*Description:* Holds information about which index version is considered newest. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 1 (default) *Since:* 6.7.0
cmas-core-index-common	indexed.assets.per.thread.in.memory	*Description:* How many assets should be loaded into memory at once during indexing per one thread. *Type:* Integer *Restart required:* No *System:* Yes

Module	Property	Explanation
		Optional: No *Example value:* 200 (default) *Since:* 6.8.0
cmas-core-index-common	indexed.engineers.per.thread.in.memory	*Description:* How many engineers should be loaded into memory at once during indexing per one thread. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 300 (default) *Since:* 6.6.14, 6.7.3
cmas-core-index-common	indexed.tickets.per.thread.in.memory	*Description:* How many tickets should be loaded into memory at once during indexing per one thread. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 100 (default) *Since:* 6.6.14, 6.7.3
cmas-core-index-common	indexed.units.per.thread.in.memory	*Description:* How many units should be loaded into memory at once during indexing per one thread. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 200 (default) *Since:* 6.6.14, 6.7.3
cmas-core-index-common	synchronize.master.address	*Description:* Value of *-Dcmas.http.host.port* informing how to connect to the indexing master server. Default null. Since 6.6.17 this value is configurable in set-up to designate the initial indexing master server. Please note that changing this value is

Module	Property	Explanation
		only allowed when all cluster nodes index changes receivers are stopped. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* 127.0.0.1:80 *Since:* 6.6.0
cmas-core-index-common	synchronize.master.security.token	*Description:* The password for accessing the index snapshot via URL, e.g. for index synchronization or for back-ups. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* token *Since:* 6.6.0
cmas-core-index-common	synchronize.master.security.user	*Description:* The user name for accessing the index snapshot via URL, e.g. for index synchronization or for back-ups. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* user *Since:* 6.6.0
cmas-core-index-common	synchronize.master.timeout.minutes	*Description:* How much time master server may constantly fail until new master gets elected with index fix procedure. Default 5. Since 6.6.17 this value is configurable in set-up where zero means that master server will never change (fail-over mechanism is off). *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No

Module	Property	Explanation
		Example value: 5 *Since:* 6.6.0
cmas-core-index-common	synchronize.megabits.per.second	*Description:* How much bandwidth can master server consume to transfer index changes to all slave servers. Default 85. Please do not use all available bandwidth to transfer index changes between hosts. This will most probably partition the cluster as some subsystems will not be able to communicate. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 85 *Since:* 6.6.0
cmas-core-index-common	synchronize.sleep.millis	*Description:* How often each slave server polls the master server for index changes. Default 1000. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 1000 *Since:* 6.6.0

36.2.2 Search Results

Module	Property	Explanation
cmweb-server-adapter	globalSearchResultSizeLimit	*Description:* Maximum number of items in global search result (Quick Search). *Type:* Integer *Restart required:* No *System:* Yes

Module	Property	Explanation
		Optional: No *Example value:* 10 *Since:* 6.0
cmweb-server-adapter	unitIndexSearchResultSizeLimit	*Description:* Maximum number of units in unit search result (e.g. when searching for contacts). *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 5 *Since:* 6.0
cmweb-server-adapter	searchPageSize	*Description:* Default page size for search results. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 20 *Since:* 6.0
cmweb-server-adapter	searchPageSizeOptions	*Description:* Options for page size for search results. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* 10|20|30|40|50|75|100 *Since:* 6.0

36.3 LDAP Configuration

36.3.1 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*:

Module	Property	Explanation
cmas-core-security	authentication.method	*Description:* User authentication method (internal CM 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

Module	Property	Explanation
cmas-core-security	ldap.authentication	*Description:* Authentication method used when using LDAP authentication. *Type:* String *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* simple *Since:* 6.0
cmas-core-security	ldap.basedn	*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
cmas-core-security	ldap.initialcontextfactory	

Module	Property	Explanation
		Description: Class name for initial context factory of LDAP implementation when using LDAP authentication. If it is not set, *com.sun.jndi.ldap. LdapCtxFactory* is being used as a value. *Type:* String *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* com.sun.jndi. ldap.LdapCtxFactory *Since:* 6.0
cmas-core-security	ldap.password	*Description:* Password for connecting to LDAP to look up users (when using LDAP authentication). Only needed if look-up cannot be done anonymously. *Type:* Password *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.1.2
cmas-core-security	ldap.providerurl	*Description:* LDAP provider (when using LDAP authentication). *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* ldap://ldap. consol.de:389 *Since:* 6.0
cmas-core-security	ldap.searchattr	*Description:* Search attribute for looking up LDAP entry connected to CM login. *Type:* String *Restart required:* No *System:* Yes *Optional:* No

Module	Property	Explanation
		Example value: uid *Since:* 6.0
cmas-core-security	ldap.userdn	*Description:* LDAP user for connecting to LDAP to look up users (when using LDAP authentication). Only needed if look-up cannot be done anonymously. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.1.2

36.3.2 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*:

Module	Property	Explanation
cmas-core-security	contact.authentication.method	*Description:* Indicates contact authentication method, where possible values are *DATABASE* or *LDAP* or *LDAP,DATABASE* or *DATABASE,LDAP.* *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Since:* 6.9.3.0

Module	Property	Explanation
cmas-core-security	ldap.contact.name.basedn	*Description:* Base path to search for contact DN by LDAP ID (e.g. ou=accounts,dc=consol,dc=de)*.* *Type:* String *Restart required:* No

Module	Property	Explanation
		System: No *Optional:* Yes *Since:* 6.9.3.0
cmas-core-security	ldap.contact.name.password	*Description:* Password to look up contact DN by LDAP ID. If not set, anonymous account is used*.* *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Since:* 6.9.3.0
cmas-core-security	ldap.contact.name.providerurl	*Description:* Address of the LDAP server (ldap[s]://host:port)*.* *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Since:* 6.9.3.0
cmas-core-security	ldap.contact.name.searchattr	*Description:* Attribute to search for contact DN by LDAP ID (e.g. uid)*.* *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Since:* 6.9.3.0
cmas-core-security	ldap.contact.name.userdn	*Description:* User DN to look up contact DN by LDAP ID. If not set, anonymous account is used*.* *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Since:* 6.9.3.0
cmas-core-security	ldap.initialcontextfactory	*Description:* Class name for initial context factory of LDAP implementation when using LDAP authentication. If it is not set, *com.sun.jndi.ldap.LdapCtxFactory* is being used as a value.

Module	Property	Explanation
		Type: String *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* com.sun.jndi.ldap.LdapCtxFactory *Since:* 6.0

36.4 E-Mail Configuration

36.4.1 Outgoing E-Mail

Independent of incoming e-mail mode (ESB/Mule and NIMH).

Module	**Property**	**Explanation**
cmas-core-server	mail.smtp.email	*Description:* SMTP e-mail URL for outgoing e-mails. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* smtp://mail.consol.de:25 *Since:* 6.0
cmas-core-server	mail.smtp.envelopesender	*Description:* E-mail address used as sender in SMTP envelope. If not set, the FROM-address of the e-mail is used. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* mysender@mydomain.com *Since:* 6.5.7

Module	**Property**	**Explanation**
cmweb-server-adapter	mail.from	*Description:* Use this address if set instead of engineer e-mail address during e-mail conversation. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.1.2
cmweb-server-adapter	mail.reply.to	*Description:* When set, Web Client will display REPLY-TO-

Module	Property	Explanation
		field on e-mail sent, prefilled with this value. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.0.1 Please see also section Queue Administration. When you set the REPLY-TO-address in the outgoing e-mail script, the *mail.reply.to* system property must not be set (because it would overwrite the configured value)! That means when you use one outgoing e-mail script for a queue you have to define outgoing e-mail scripts for all queues because the *mail.reply.to* property can no longer be used.
cmweb-server-adapter	mailTemplateAboveQuotedText	*Description:* Indicates behavior of e-mail template in the Ticket E-Mail Editor when another e-mail is quoted, i.e. forwarded or replied to. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* false *Since:* 6.2.4

Module	Property	Explanation
cmas-workflow-jbpm	mail.sender.address	*Description:* FROM-address for e-mails 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

36.4.2 Incoming E-mail

Settings for ESB/Mule

Module	Property	Explanation
cmas-esb-core	esb.directory	*Description:* Directory used by ESB (Mule). *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* C:\Users\user\cmas\mule *Since:* 6.0
cmas-esb-mail	mail.attachments.validation.info.sender	*Description:* Sets FROM-header of attachments type error notification e-mail. As a default the e-mail address of the administrator which you have entered during system set-up is used. *Type:* String *Restart required:* No *System:* Yes *Optional:* No

Module	Property	Explanation
		Example value: admin@consolcm.com *Since:* 6.7.5
cmas-esb-mail	mail.attachments.validation.info.subject	*Description:* Sets subject of attachments type error notification e-mail. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* E-mail was not processed because its attachments were rejected!!! *Since:* 6.7.5
cmas-esb-mail	mail.callname.pattern	*Description:* Regular expression for subject of incoming e-mails. Available as TICKET_NAME_PATTERN_FORMAT in incoming e-mail scripts. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* .*?Ticket\s+\((\S+)\).* *Since:* 6.0
cmas-esb-mail	mail.cluster.node.id	*Description:* Only the node whose mail.cluster.node.id equals cmas.clusternode.id will start the Mule ESB mail services. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* unspecified *Since:* 6.6.5
cmas-esb-mail	mail.db.archive	*Description:* If property is set to *true*, incoming e-mails are archived in the database. *Type:* Boolean *Restart required:* No

Module	Property	Explanation
		System: Yes *Optional:* Yes *Example value:* false (default) *Since:* 6.8.5.5
cmas-esb-mail	mail.delete.read	*Description:* Determines whether CM deletes messages fetched via IMAP(S). Setting value to *true* will cause deletion of messages after fetching. Default is to not delete messages fetched via IMAP(S). Note: Messages fetched via POP3(S) will always be deleted. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* true *Since:* 6.7.3
cmas-esb-mail	mail.encryption	*Description:* If property is set to *tr ue*, the encrypt check box in the Ticket E-Mail Editor is checked by default. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* true (default = false) *Since:* 6.8.4.0
cmas-esb-mail	mail.incoming.uri	*Description:* URL for incoming e-mails. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* pop3://cm-incoming-user:password@localhost:10110 *Since:* 6.0

Module	Property	Explanation
		This value should not be edited here using the system properties pop-up window, but the mailboxes should be configured using the tab E-mail. Using this standard feature all entries are controlled - i. e. for each mailbox which is added, CM establishes a test connection during mailbox set-up. That way it is not possible to enter wrong values.
cmas-esb-mail	mail.max.restarts	*Description:* Maximum number of e-mail service restarts before giving up. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 3 *Since:* 6.0
cmas-esb-mail	mail.mime.strict	*Description:* If set to *false*, e-mail addresses are not parsed for strict MIME compliance. Default is *true*, which means check for strict MIME compliance. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* false *Since:* 6.6.17, 6.7.3
cmas-esb-mail	mail.mule.service	*Description:* FROM-address for e-mails sent by Mule service. *Type:* EMail *Restart required:* No

Module	Property	Explanation
		System: Yes *Optional:* No *Example value:* myuser@consol.de *Since:* 6.0
cmas-esb-mail	mail.polling.interval	*Description:* Mail polling interval in ms. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 60000 *Since:* 6.0
cmas-esb-mail	mail.process.error	*Description:* TO-address for error e-mails from Mule. As a default the e-mail address of the administrator which you have entered during system set-up is used. *Type:* EMail *Restart required:* No *System:* Yes *Optional:* No *Example value:* myuser@consol.de *Since:* 6.0
cmas-esb-mail	mail.process.retry.attempts	*Description:* Number of retries when processing e-mail. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 3 *Since:* 6.0.2
cmas-esb-mail	mail.process.timeout	*Description:* Mail processing timeout in seconds. *Type:* Integer *Restart required:* No *System:* Yes

Module	Property	Explanation
		Optional: No *Example value:* 60 *Since:* 6.1.3
cmas-esb-mail	mail.redelivery.retry.count	*Description:* Indicates the number of retries of re-delivering an e-mail from the CM system. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 3 *Since:* 6.1.0

Settings for NIMH

Those settings apply if NIMH is enabled (and therefore ESB/Mule is disabled):

Module	Property	Explanation
cmas-core-server	nimh.enabled	*Description:* Enables *nimh* service. Must be suffixed with nodeid in cluster, e.g. *nimh. enabled.NODEID = true* *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* true *Since:* 6.9.4.0

Module	Property	Explanation
cmas-nimh	filesystem.polling.threads. number	*Description:* Number of threads started for db e-mails' queue polling. Default: 1 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 10 *Since:* 6.4.0
cmas-nimh		

Module	Property	Explanation
	filesystem.polling.threads. shutdown.timeout.seconds	*Description:* Waiting time after the shutdown signal. When the timeout is reached, thread will be terminated. Default: 60 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	filesystem.polling.threads. watchdog.interval.seconds	*Description:* Watchdog thread interval. Default: 30 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	filesystem.task.enabled	*Description:* With this property service thread related to given poller can be disabled. Default: true *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* true *Since:* 6.4.0
cmas-nimh	filesystem.task.interval.seconds	*Description:* Default interval for polling mailboxes. Default: 60 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	filesystem.task.polling.folder	*Description:* Polling folder location which will be scanned for e-mails in the format of *eml* files. Default: "mail" subdir of

Module	Property	Explanation
		cmas data directory. *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Example value:* c://cmas//mail *Since:* 6.4.0
cmas-nimh	filesystem.task.timeout.seconds	*Description:* After this time (of inactivity) the service thread is considered as damaged and automatically restarted. Default: 120 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	filesystem.task.transaction.timeout.seconds	*Description:* Default transaction timeout for e-mail fetching transactions. Should be correlated with number of messages fetched at once. Default: 60 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	mailbox.1.connection.host	Host (server) for first configured mailbox. Will overwrite the default parameter *mailbox.default.connection.host*.
cmas-nimh	mailbox.1.connection.password	Password for first configured mailbox. Will overwrite the default parameter *mailbox.default.connection.password*.
cmas-nimh	mailbox.1.connection.port	

Module	Property	Explanation
		Port for first configured mailbox. Will overwrite the default parameter *mailbox.default.connection.port*.
cmas-nimh	mailbox.1.connection.protocol	Protocol (e.g. IMAP or POP3) for first configured mailbox. Will overwrite the default parameter *mailbox.default.connection.protocol*.
cmas-nimh	mailbox.1.connection.username	User name for first configured mailbox. Will overwrite the default parameter *mailbox.default.connection.username*.
cmas-nimh	mailbox.2.connection.host	Host (server) for second configured mailbox. Will overwrite the default parameter *mailbox.default.connection.host*.
cmas-nimh	mailbox.2.connection.password	Password for second configured mailbox. Will overwrite the default parameter *mailbox.default.connection.password*.
cmas-nimh	mailbox.2.connection.port	Port for second configured mailbox. Will overwrite the default parameter *mailbox.default.connection.port*.
cmas-nimh	mailbox.2.connection.protocol	Protocol (e.g., IMAP or POP3) for second configured mailbox. Will overwrite the default parameter *mailbox.default.connection.protocol*.
cmas-nimh	mailbox.2.connection.username	User name for second configured mailbox. Will overwrite the default parameter *mailbox.default.connection.username*.

Module	Property	Explanation
For all NIMH-related mailbox properties, the following principle is used: A defaut property is defined (e.g. *mailbox.default.connection.port*). If no mailbox-specific value is configured, this default value will be used.		
cmas-nimh	mailbox.default.connection.host	*Description:* Host (server name) of a given mailbox from which the poller reads e-mails. *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Example value:* 10.10.1.157 *Since:* 6.4.0
cmas-nimh	mailbox.default.connection. password	*Description:* Password for given mailbox from which the poller reads e-mails. *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Example value:* consol *Since:* 6.4.0
cmas-nimh	mailbox.default.connection.port	*Description:* Port for a given mailbox from which the poller reads e-mails. *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Example value:* 143 *Since:* 6.4.0
cmas-nimh	mailbox.default.connection. protocol	*Description:* Poller's protocol, e.g: IMAP or POP3. No default value. *Type:* String *Restart required:* No *System:* No

Module	Property	Explanation
		Optional: Yes *Example value:* imap *Since:* 6.4.0
cmas-nimh	mailbox.default.connection. username	*Description:* User name for a given mailbox from which the poller reads e-mails. *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Example value:* username *Since:* 6.4.0
cmas-nimh	mailbox.default.session.mail. debug	*Description:* Example *javax.mail* property. Allows for more detailed javax.mail session debugging. *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* true *Since:* 6.4.0
cmas-nimh	mailbox.default.session.mail. imap.timeout	*Description:* Example *javax.mail* property. *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 120 *Since:* 6.4.0
cmas-nimh	mailbox.default.session.mail. mime.address.strict	*Description:* Example *javax.mail* property. Counterpart of the old Mule *mail.mime.strict*, allows to set not so strict mail header parsing. *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* true *Since:* 6.4.0

Module	Property	Explanation
cmas-nimh	mailbox.default.session.mail.pop3.timeout	*Description:* Example *javax.mail* property. *Type:* *Restart required:* *System:* *Optional:* *Example value:* *Since:* 6.4.0
cmas-nimh	mailbox.default.task.delete.read.messages	*Description:* This defines whether messages should be removed from the mailbox after processing. For IMAP protocol messages are marked as *SEEN* by default. For POP3 protocol, if flag is set to true, the message is removed, otherwise remains on server and will result in infinite reads. Default: false. *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* false *Since:* 6.4.0
cmas-nimh	mailbox.default.task.enabled	*Description:* With this property service thread related to given poller can be disabled. Default: *tr ue*. *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* false *Since:* 6.4.0
cmas-nimh	mailbox.default.task.interval.seconds	*Description:* Default interval for polling mailboxes. Default: 60 seconds *Type:* Integer *Restart required:* No *System:* No

Module	Property	Explanation
		Optional: Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	mailbox.default.task.max.message.size	*Description:* Maximum size of e-mail messages in bytes. Messages with a size larger than the value of this property cannot be processed by the Admin Tool. Default set to 10 MB: 10485760 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 10485760 *Since:* 6.4.0
cmas-nimh	mailbox.default.task.max.messages.per.run	*Description:* Number of messages fetched at once from mailbox. Must be correlated with transaction timeout. Default set to: 20 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	mailbox.default.task.timeout.seconds	*Description:* After this time (of inactivity) the service thread is considered as damaged and automatically restarted. Default: 120 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	mailbox.default.task.transaction.timeout.seconds	*Description:* Default transaction timeout for e-mail fetching transactions. Should be correlated with number of

Module	Property	Explanation
		messages fetched at once. Default: 60 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	mailbox.polling.threads.mail.log.enabled	*Description:* Enables e-mail logging which is especially crucial in cluster environments (used as semaphore there). *Type:* String *Restart required:* No *System:* No *Optional:* Yes *Example value:* true (default) *Since:* 6.9.4.1
cmas-nimh	mailbox.polling.threads.number	*Description:* Number of threads for accessing mailboxes. Default: 1 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 1 *Since:* 6.4.0
cmas-nimh	queue.polling.threads.number	*Description:* Number of threads started for e-mails' queue polling. Default: 1 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 1 *Since:* 6.4.0
cmas-nimh	queue.polling.threads.shutdown.timeout.seconds	*Description:* Waiting time after the shutdown signal. When the timeout is reached, the thread will be terminated. Default: 60 *Type:* Integer

Module	Property	Explanation
		Restart required: No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh	queue.polling.threads.watchdog.interval.seconds	*Description:* Watchdog thread interval. Default: 30 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 30 *Since:* 6.4.0
cmas-nimh	queue.task.error.pause.seconds	*Description:* Maximum number of seconds, the queue poller waits after infrastructure (e.g. database) error. Default 180 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 180 *Since:* 6.4.0
cmas-nimh	queue.task.interval.seconds	*Description:* Main e-mails' queue polling thread interval. Default: 15 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 15 *Since:* 6.4.0
cmas-nimh	queue.task.max.retries	*Description:* Maximum number of e-mail processing retries after an exception. When reached, the e-mail is moved to the e-mail archive. This e-mail can be rescheduled again using the NIMH API (or the Admin Tool). *Type:* Integer *Restart required:* No

Module	Property	Explanation
		System: No *Optional:* Yes *Example value:* 10 *Since:* 6.4.0
cmas-nimh	queue.task.timeout.seconds	*Description:* After this time (of inactivity) the service thread is considered as damaged and automatically restarted. Default: 600 seconds *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 600 *Since:* 6.4.0
cmas-nimh	queue.task.transaction.timeout.seconds	*Description:* Transaction timeout for e-mail processing in the pipe. Default: 60 *Type:* Integer *Restart required:* No *System:* No *Optional:* Yes *Example value:* 60 *Since:* 6.4.0
cmas-nimh-extension	mail.attachments.validation.info.sender	*Description:* Sets FROM-header of attachments type error notification e-mail. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* admin@mail.com *Since:* 6.7.5 This is an equivalent to the old *cmas-esb-mail*, *mail.attachments.validation.info.sender*
cmas-nimh-extension		

Module	Property	Explanation
	mail.attachments.validation.info.subject	*Description:* Sets subject of attachments type error notification e-mail. *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* Mail was not processed because its attachments were rejected!!! *Since:* 6.7.5 This is an equivalent to the old *cmas-esb-mail*, *mail.attachments.validation.info.subject*
cmas-nimh-extension	mail.db.archive	*Description:* If property is set to *true*, incoming emails are archived in the database *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* false (default) *Since:* 6.8.5.5 This is an equivalent to the old *cmas-esb-mail*, *mail.db.archive*
cmas-nimh-extension	mail.error.from.address	This is an equivalent to the old *cmas-esb-mail*, *mail.mule.service*
cmas-nimh-extension	mail.error.to.address	

Module	Property	Explanation
		This is an equivalent to the old *cmas-esb-mail*, *mail.process.error*
cmas-nimh-extension	mail.mule.service	*Description:* FROM-address for e-mails sent by Mule service. *Type:* EMail *Restart required:* No *System:* Yes *Optional:* No *Example value:* myuser@consol.de *Since:* 6.4.0
cmas-nimh-extension	mail.on.error	*Description:* If set to *true* an error e-mail is sent to the above configured address in case the e-mail message could not be processed. Default: false *Type:* Boolean *Restart required:* No *System:* No *Optional:* Yes *Example value:* false *Since:* 6.4.0
cmas-nimh-extension	mail.process.error	*Description:* TO-address for error e-mails from Mule. *Type:* EMail *Restart required:* No *System:* Yes *Optional:* No *Example value:* myuser@consol.de *Since:* 6.4.0
cmas-nimh-extension	mail.ticketname.pattern	*Description:* *Type:* String *Restart required:* No *System:* Yes *Optional:* No *Example value:* .*?Ticket\s+\((\S+)\).* *Since:* 6.4.0

Mapping of Former Mule and New NIMH Properties

Mule property	NIMH property
cmas-esb-mail, mail.delete.read	cmas-nimh, mailbox.default.task.delete.read.messages
cmas-esb-mail, mail.polling.interval	cmas-nimh, mailbox.default.task.interval.seconds
cmas-esb-mail, mail.process.retry.attempts	cmas-nimh, queue.task.max.retries
cmas-esb-mail, mail.mime.strict	cmas-nimh, mailbox.default.session.mail.mime.address.strict
cmas-esb-mail, mail.encryption	cmas-core-server, mail.encryption (moved to core server properties)
cmas-esb-mail, mail.callname.pattern	cmas-nimh-extension, mail.ticketname.pattern
cmas-esb-mail, mail.attachments.validation.info.sender	cmas-nimh-extension, mail.attachments.validation.info.sender
cmas-esb-mail, mail.attachments.validation.info.subject	cmas-nimh-extension, mail.attachments.validation.info.subject
cmas-esb-mail, mail.db.archive	cmas-nimh-extension, mail.db.archive (seems not legit now)
cmas-esb-mail, mail.mule.service	cmas-nimh-extension, mail.error.from.address
cmas-esb-mail, mail.process.error	cmas-nimh-extension, mail.error.to.address

Attachments for Incoming E-Mails

These settings apply to Mule/ESB and NIMH.

Module	Property	Explanation
cmas-core-server	attachment.allowed.types	*Description:* Comma-separated list of allowed filename extensions (if no value defined, all file extensions are allowed). *Type:* String *Restart required:* No *System:* Yes

Module	Property	Explanation
		Optional: Yes *Example value:* txt,zip,doc *Since:* 6.5.0
cmas-core-server	attachment.max.size	*Description:* Maximum attachment size in MB. *Type:* Integer *Restart required:* No *System:* Yes *Optional:* No *Example value:* 100 *Since:* 6.4.0

36.4.3 E-Mail Encryption (Outgoing and Incoming)

These settings only apply if e-mail encryption is active (*true*). This is valid for ESB Mail and NIMH.

Module	Property	Explanation
cmas-core-server	mail.encryption	*Description:* If property is set to *tr ue*, the encrypt check box in the Ticket E-Mail Editor is checked by default. *Type:* Boolean *Restart required:* No *System:* Yes *Optional:* No *Example value:* true (default = false) *Since:* 6.8.4.0

In case certificates are stored in an LDAP directory, the following settings have to be made:

Module	Property	Explanation
cmas-core-server	ldap.certificate.basedn	*Description:* Base DN for certificates location in LDAP tree. If not provided, *cmas-core-security*, *ldap.basedn* is taken. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes

Module	Property	Explanation
		Example value: ou=accounts, dc=consol,dc=de *Since:* 6.8.4
cmas-core-server	ldap.certificate.content.attribute	*Description:* LDAP attribute name used where certificate data is stored in LDAP tree. Default value is *usercertificate*. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* usercertificate *Since:* 6.8.4
cmas-core-server	ldap.certificate.password	*Description:* LDAP certificates manager password. If not set, *cm as-core-security*, *ldap.password* is taken. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.8.4
cmas-core-server	ldap.certificate.providerurl	*Description:* LDAP certificates provider URL. If not set, *cmas-core-security*, *ldap.providerurl* is taken. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Example value:* ldap://ldap. consol.de:389 *Since:* 6.8.4
cmas-core-server	ldap.certificate.searchattr	*Description:* LDAP attribute name used to search for certificate in LDAP tree. Default value is *mail*. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes

Module	Property	Explanation
		Example value: mail *Since:* 6.8.4
cmas-core-server	ldap.certificate.userdn	*Description:* LDAP certificates manager DN. If not set, *cmas-core-security*, *ldap.userdn* is taken. *Type:* String *Restart required:* No *System:* Yes *Optional:* Yes *Since:* 6.8.4

36.5 Activity Interval Configuration

Module	Property	Explanation
cmas-app-admin-tool	admin.tool.session.check.interval	*Description:* Admin Tool inactive (ended) sessions check time interval (in seconds). *Type:* Integer *Restart required:* Yes *System:* Yes *Optional:* No *Example value:* 30 *Since:* 6.7.5
cmas-core-server	server.session.timeout	*Description:* Server session timeout (in seconds) for connected clients. Each client can overwrite this timeout with custom value using its ID (ADMIN_TOOL, WEB_CLIENT, WORKFLOW_EDITOR, TRACK (before 6.8 please use PORTER), ETL, REST) appended to property name, e.g. server.session.timeout.ADMIN_TOOL *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 incativity.

37 Appendix E - Trademarks

- 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® 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
- 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® 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® 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
- 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
- MuleSoft^{TM} and Mule ESB^{TM} are among the trademarks of MuleSoft, Inc. See Mule Soft web 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

Index

A

Access, system access for customers in CM.Track 528
Access rights, to customer groups 105, 137
acimSection 245
Action Framework 147
Activity form 198
Administrator e-mail 313
Adminmanual Consolcm 6 9 4 10, 21, 31, 32, 41, 54, 64, 73, 74, 76, 82, 96, 106, 121, 132, 140, 147, 167, 177, 186, 187, 207, 219, 228, 231, 280, 300, 301, 309, 313, 315, 317, 332, 335, 338, 342, 348, 354, 356, 368, 377, 387, 389, 413, 419, 421, 450, 467, 479, 481, 487, 495, 513, 525, 528, 534, 539, 545, 559, 578, 585, 706, 753
Admin Tool, for customer data model 97
Annotation 201
Annotations 559
Annotations, field annotations 560
Annotations, group annotations 574
Authentication, in Web Client 480
Authentication, LDAP 481
Authentication mode, in CM.Track 534
autocomplete (for page customization) 255

B

Business calendar 342

C

Calendar 342
Certificates, client 329
Certificates, server 329
Chart Widget 280
Classes of text 348
CM.Doc 450
CM.Phone 496
CM.Track 526
CM.Track, authentication modes 534
CM.Track, extended customer permissions 533
CM.Track, FAQs 539
CM.Track users 528

CMRF (ConSol*CM Reporting Framework) 514
Company, as main customer 88
Company page 85
Configuration, advanced 311
Configuration, basic 310
Configuration, classes of text 348
Configuration, CM services 315
Configuration, e-mail 317
Configuration, general 313
Configuration, index 356
Configuration, ticket history 354
contactCreatePage 241
contactEditPage 243
Contact page 87
CTI 496
Customer 19
Customer actions 89
Customer data model 80, 96
Customer group 79
Customer group, access rights to customer groups 105, 137
Customer group, in quick search 95
Customer group, new, in Admin-Tool 103
Customer group, selector 83
Customer group, Web Client 83
Customer relations 80, 90
Customer roles 169
Custom field 80, 188
Custom field annotation 201
Custom field group 190

D

Dashboard, Web Client Dashboard 228, 280
Data directory 552
Data field 80
Data field group 80
Data object 79, 99
Data object actions 148
Data object actions, in Admin-Tool 150
Data object action scripts 159
Data object condition scripts 166
Data object definition 80
Data object group fields 107
Data object groups 108
Data warehouse (DWH) management 514

Deactivation, of company or contact 91
Default value scripts 397
Dependent enum scripts 402
Deployment 371
Doc Template Manager, in CM.Doc 454
DWH configuration 517
DWH live service 315
DWH log service 315
DWH transfer service 316

E

E-mail (architecture in CM) 547
E-mail backups 332
E-mail configuration 320
E-mail encryption 327
E-mail scripts 405
E-mail server 321
E-Mail Templates 422, 423
Engineer 19, 33
Engineer functions 51, 171
Enum administration 208
esb_mail_preprocessorService 340
esb_mail_scriptService 340
esb_mail_SuccessService 340
ESB/Mule Mail 323
ESB service 316
ESB services 338

F

FAQs (in CM.Track) 539
File structure 552
FlexCDM 76
Freemarker 122

G

Global permissions 46
Glossary 578
Groovy Task 383

I

Include, in Template Manager 436
Index changes notifier 316
Index changes receiver 316
Indexer 356

J

Job Executor 316

K

Kerberos 487
Kerberos v5 authentication provider 316

L

LDAP authentication in CM.Track
LDAP authentication in Web Client 481
LDAP ID 485
LDAPS authentication in CM.Track 538
LDAPS authentication in Web Client 485
Library of Markers 432
Licence 335
Locales 313
Log files 556
Login 22

M

mailTemplate 267
Main menu 28
Main role 38
MLA administration 220
MS Word Templates, for CM.Doc 454
Multi Level Attribute (MLA) 220

N

NIMH, New Incoming Mail Handler 323
NIMH Service 316

O

officeTemplatePage 245

P

Page customization 232
Projects 175
Projects (for time booking) 469

Q

Queue 19, 65
Queue permissions 44

R

Relations, between customers 90
Relations, customers 141
Relations, customers, in Admin-Tool 142
Relations, customers, in Web Client 145
Remote client pooling 316
Representation permissions 47
Rest API service 316
Role 42

S

Scenario 369
Scripts (for default values) 397
Scripts (for dependent enums) 402
Scripts (for use in workflows) 410
Scripts (in the Admin-Tool) 389
Scripts (of type e-mail) 405
Search, detail search 83
Search, quick search 83
Search configuration 356
searchDetailPage 241
Server session service 316
SSO (single sign-on) 487
System architecture 546
System properties 585

T

Table Widget 280
Task, of Task Execution Framework 380
Task Definition 378
Task Descriptor 379
Task Execution Framework 378
Task Executor 378
TaskExecutorService 316
Task Script 385
TEF, Task Execution Framework 378
Template Library 426
Template Manager 421
Template permissions 47
Templates, E-Mail Templates 422
Templates, for display of customer data 121
Templates, MS Word templates for CM.Doc 454
Templates, Text Templates 420
Templates, Ticket Text Templates 423
Templates (contact format templates) 415
Templates (for e-mails, in Template Manager) 423
Templates (in Admin-Tool) 413
Templates (system templates) 415
Templates (ticket assignment templates) 416
Template types, for definition of customer data display 124
templateViewPage 244
Text Block, in Template Manager 438
Text classes 348
Text Templates 420
Ticket 16
Ticket administration 302
ticketCreatePage 243
ticketEditPage 240
Ticket filter, on company page 92
Ticket filter, on contact page 94
Ticket history 354
Ticket Text Templates 423
Time booking 468
Time booking (on projects) 469
timeBookingSection 274
Track user permissions 47

U

unitSearch 275

unitSearchHeader 276
Unused content remover 316
User attributes 97
User attributes administration 168
User profile, for CM.Track 529
userProfilePage 244

V

View criterion, dynamic 59
View criterion, static 58
viewDiscriminatorsSection 277
Views 50, 55

W

Web Client Dashboard 228, 280
welcomePage 240
Widget, for Web Client Dashboard 280
Word Templates , MS Word Templates in CM.Doc 454
Workflow 17
Workflow Include, in Template Manager 436
Workflow permissions 46