ConSol Software GmbH

ConSol CM Setup and Operations Manual

Version 6.16

Contents

A - About this manual

This manual covers the use of ConSol CM in a single-server environment. It is your responsibility to:

- Set up the necessary components, as the database and application server, and install ConSol CM using a basic system configuration.
- Operate the ConSol CM system.

The further configuration of ConSol CM is carried out in the Web Admin Suite. It is described in the *ConSol CM Administrator Manual*.

> ⓘ Please contact the ConSol CM support or your ConSol CM consultant for information about setting up ConSol CM in a cluster.

A.1 Manual structure

This manual contains all you need to know to set up ConSol CM in a single-server environment.

- Introduction to ConSol CM

  This section provides an introduction to using ConSol CM for business process management and general information about this manual.

- System architecture

  This section contains information about the system architecture.

- Installing and setting up ConSol CM

  This section describes the database setup, the installation of the application server and the installation of the ConSol CM application.

- Installing and setting up CMRF and DWH

  This section explains the set up of the CMRF used for reporting with ConSol CM. It contains detailed information about the installation of the CMRF.

- Installing ConSol CM add-ons

  This section describes the setup of three ConSol CM addons: CM/Track, CM/Doc and CM/Phone.

- Operating ConSol CM

  This section describes the most common tasks for operating ConSol CM, for example the file system structure, starting and stopping the application, and logging. In addition, licenses, email, index and LDAP functionalities are covered in this section.

- Operating the DWH

  This section explains the different aspects of operating the ConSol CM Data Warehouse (DWH).

- Advanced operating topics

  This section describes some advanced operation topics, such as system fine-tuning, troubleshooting and monitoring. In addition generic topics affecting the overall system, as operating CM behind a proxy, backups, system updates and the release process, are described.

- Appendix

This section contains a list of important files, system properties, trademarks, the list of manuals and a glossary.

A.2 Variables used for standard path values

The following variables are used for paths in this manual:

- **<JBOSS_HOME>**
  Refers to the home directory of the JBoss application sever where ConSol CM is installed. Set as environment variable of the operating system.
- **<CMAS_DATADIR>**
  Refers to the data directory of ConSol CM which is defined during system setup. This is not an environment variable of the operation system but a CM system property (cmas-core-shared, data.directory).
- **<ENGINEER_HOME_DIR>**
  Refers to the home directory of the engineer. On windows systems, this is often to be found under `C:\users\<USERNAME>`, on Linux systems in `/home/<USERNAME>`.
- **<DB_HOME>**
  Refers to the home directory of the database (MySQL, Oracle, Microsoft SQL or PostgreSQL).

A.3 Variables used for names

The following variables are used for names in this manual:

- **<CM_USER>**
  Refers to the name of the database user of CM.
- **<CM_DB>**
  Refers to the name of the database used by CM.

A.4 Variables used for version numbers

The following variables are used for version numbers in this manual:

- **<JAVA_VERSION>**
  Refers to the version of the used JDK.
- **<CM_VERSION>**
  Refers to the ConSol CM version.
- **<DB_VERSION>**
  Refers to the version of the used database.
- **<AS_VERSION>**
  Refers to the version of the application server.

B - Introduction to ConSol CM

B.1 Purpose and usage of ConSol CM

ConSol CM is a process management platform suitable for a huge variety of business processes. The products CM/Ticketing, CM/Helpdesk and CM/Complaint are based on the platform and provide ready-to-use processes for handling customer inquiries, support requests and complaints. If you have purchased one of these three standard products, you can directly work with the processes or adapt them to your needs. Alternatively, you can use the platform to implement new processes which are tailored specifically to your use cases and requirements. The powerful workflow editor allows you to decide if the process flow should be rather strict or the users should have more flexibility.

ConSol CM include many features to enable communication and interaction. In addition to the Web Client, used for the internal handling, ConSol CM also comes with the customer portal CM/Track and various interfaces and add-ons which facilitate additional features as reporting or integration with other IT systems.

B.2 System components of ConSol CM

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

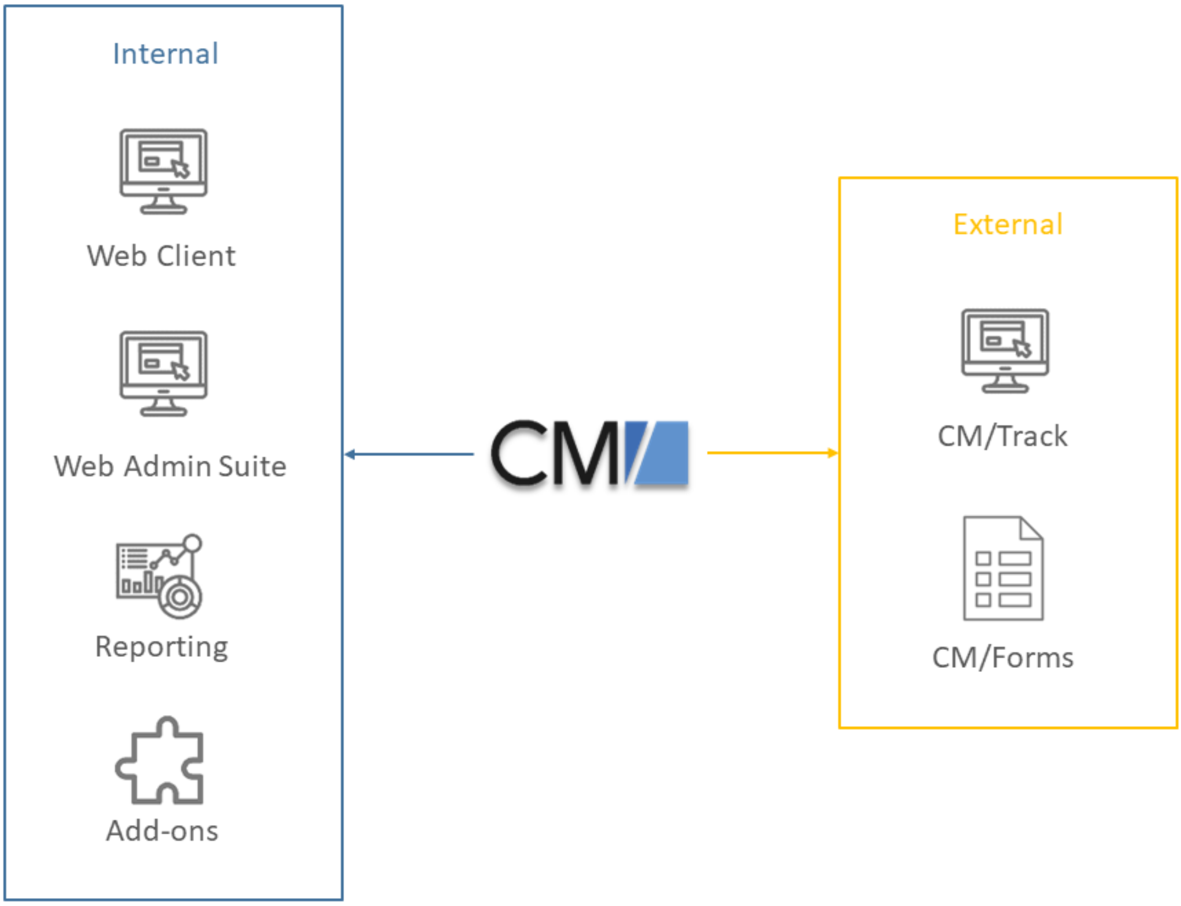


Figure 1: *ConSol CM system components*

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

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

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

B.3 Technical principles of ConSol CM

The following objects are essential when working with ConSol CM:

- **Users:**
  Users are the internal people who work with the system. They work on cases to carry out the tasks defined in the business process. A case is usually assigned to the user who takes care of it at the moment. There can be only one assignee, but other users, who perform a specific task for the case, can be added as participants. The users have one or several roles which give them access to the system and its objects.
- **Contacts:**
  Contacts are the external people whose requests are handled using ConSol CM. They always belong to a customer group which determines the used data model. You can either define simple, one-level models which contain only person data (e.g. name, email address) or complex, two-level models which contain person data (e.g. name, email address) and company data (e.g. address, industry). The permissions to access contact data depend on the customer group. There can be several customer groups and data models.
- **Cases:**
  Cases are the requests made by contacts. The users carry out the steps which are defined in the business process to handle the request. The progress, including internal and external communication, is documented in the case. When the request is solved, the case is closed. Closed cases represent a powerful archive and knowledge base. A case can have one main contact and several additional contacts. Each case has a unique number which is used to reference it throughout the system. The key data of a case are shown in its header.
- **Queues:**
  Queues contain thematically related cases which should be handled in the same way. They determine if the cases have contacts, how they are processed (workflow), whose cases are processed (customer groups), how the cases should look like (case fields) and who can work on the cases (roles). Queues often reflect the organizational structure of the company. If there are several teams working with different queues, the cases can be passed from one queue to another.
- **Workflows:**
  Workflows implement the business process. They consist of several steps, called activities, which need to be performed either manually or automatically to handle the customer requests. The activities are arranged in scopes to illustrate the status of the case. The intelligence of the process, like conditions, decisions, escalations, reminders, or automatically sent emails, is also defined in the workflow. You can implement process chains or a hierarchical process structure by linking several workflows.
- **Resources:**
  The ConSol CM add-on CM/Resource Pool allows to manage additional objects which needed in the business process. You are free to define which kinds of resources you need. The resource model defines the corresponding resources types and their data fields.

The case, contact and resource pages include sections with links to the connected objects, so that you can navigate the network easily.

The process intelligence is implemented using scripts. Scripts can be used in the following contexts:

- **Workflow:**
  The workflow can include scripts in numerous places. The most important one is the activity script which is executed together with a workflow activity. In addition, scripts can be used in triggers, activity forms and decision nodes for different purposes.
- **Actions:**
  Actions are available for contacts, resources and results of the detail search. They can include a script to be run. Contact and resource actions can either be executed manually by the user in the Web Client or automatically when a certain event occurs for the object. Search actions are always executed manually because they become available once the user has performed a detail search in the Web Client.
- **Tasks:**
  Tasks allow to execute long-running scripts which are not linked to user actions in the Web Client. They can be useful, for example, for importing data from external systems.

C - System architecture

ConSol CM is a Java EE application which is installed on an application server and accesses a relational database.

C.1 Applications

ConSol CM consists of three applications which are installed on one or several application servers, see Setup configurations.

- **ConSol CM**: core application which contains the ConSol CM server and two clients, the Web Admin Suite and the Web Client
- **CMRF**: application which fills the DWH database for reporting (optional), see Installing and setting up CMRF and DWH
- **CM/Track**: client application which provides a portal where customers can access their cases (optional)

C.2 Application servers

The ConSol CM applications are installed on one or several application servers, see Setup configurations. Supported application servers are JBoss and Wildfly. Please refer to the system requirements for details.

C.3 Databases

ConSol CM requires one database / database schema. If the CMRF application is installed, a separate database / database schema is needed for the DWH. Supported relational databases are Oracle, Microsoft SQL Server, MySQL and PostgreSQL. See Setting up the database for ConSol CM for details.

C.4 Interfaces

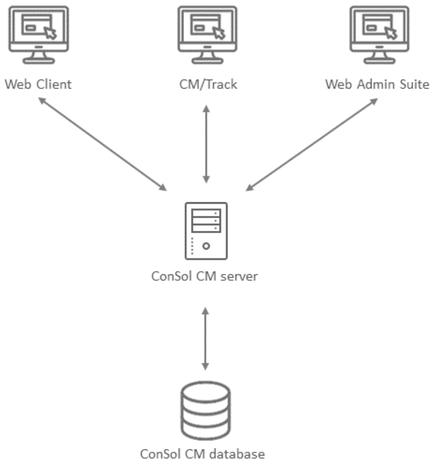
ConSol CM can connect to the following components using interfaces:

- **Email server**: The ConSol CM server acts as an email client which connects to the email server via IMAP(S) / POP3(S) and SMTP(S).
- **LDAP server**: ConSol CM can connect to an LDAP server if LDAP authentication is used, see Configuring LDAP authentication.

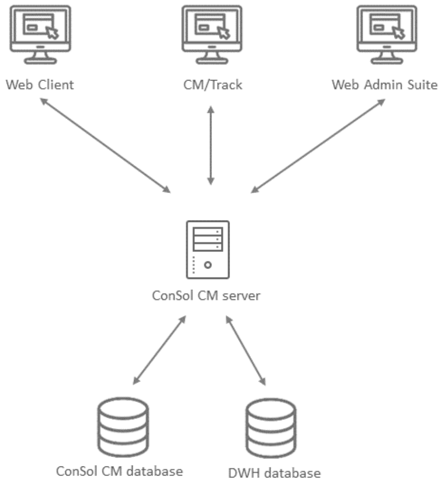
C.5 Setup configurations

There are three different ways to set up ConSol CM.

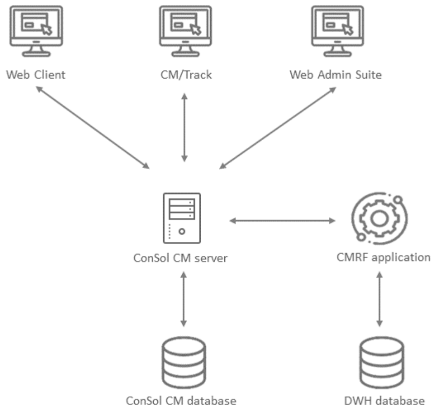
- **ConSol CM without DWH**: You need the ConSol CM application, one application server where the application is installed, and one database / database schema.



- **ConSol CM with DWH in overlay mode**: You need the ConSol CM application and the CMRF application, one application server where both applications are installed, and two databases / database schemes, one for each application.

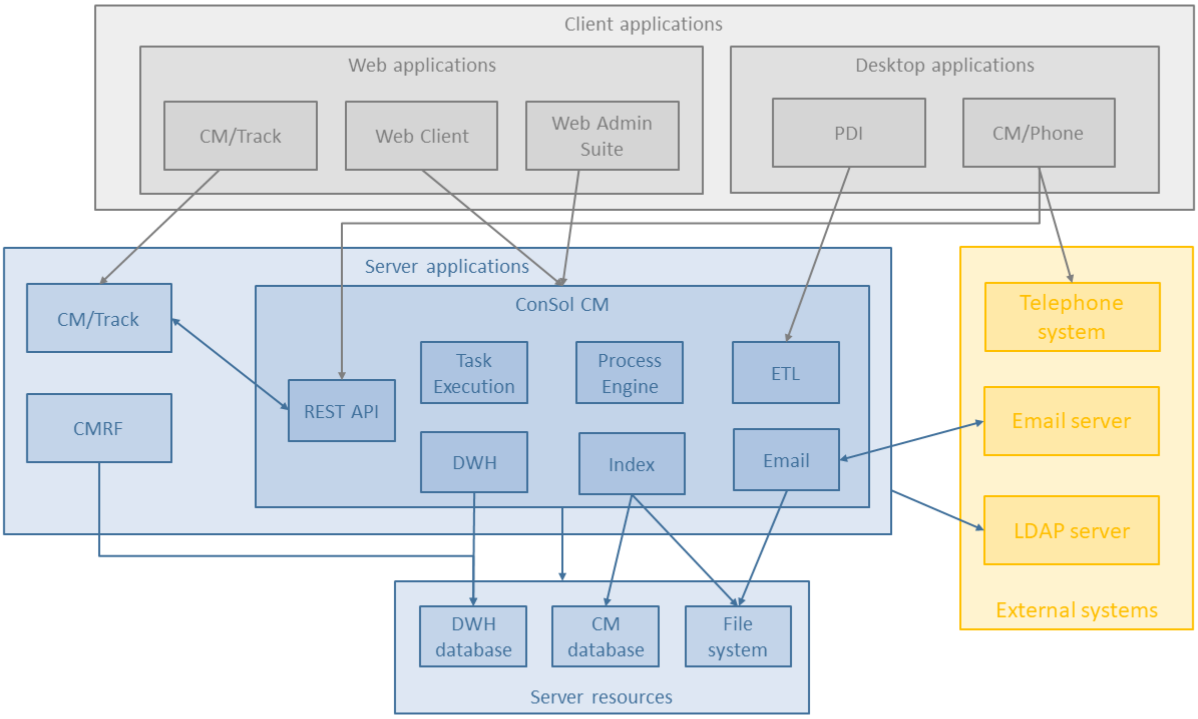


- **ConSol CM with DWH in standalone mode**: You need the ConSol CM application and the CMRF application, one application server where the ConSol CM application installed, and two databases / database schemes, one for each application. The CMRF application does not need an application server for execution.



C.6 Application architecture

The following figure shows the architecture of the ConSol CM application:



D - Installing and setting up ConSol CM

This section describes how to install and set up ConSol CM in a single-server environment. The Quick start guide provides a summary of the steps which you need to carry out. Please have a look at System architecture for an overview of the system components.

You have to proceed in four steps in order to install and set up ConSol CM:

1. Complete all required preparations, see section Preparations
2. Set up the database (or database scheme), see Setting up the database for ConSol CM.
3. Install the ConSol CM application server with ConSol CM, see Installing the application server for ConSol CM.
4. Set up ConSol CM, either via a pre-configured demo scene or with your specific scene, see Setting up the ConSol CM application.
5. Configure authentication (database, LDAP or SSO), see Authentication.

ⓘ Please contact the ConSol CM support or your ConSol CM consultant for information about setting up ConSol CM in a cluster.

D.1 Quick start guide

This section lists all steps which are required to install and configure ConSol CM and the CMRF as a single instance system.

D.1.1 ConSol CM

Please perform the following steps to install ConSol CM.

Mandatory steps:

1. Provide the required hardware, middleware and software for the desired ConSol CM version, as specified in the *ConSol CM System Requirements.*
2. Prepare the mailboxes on the email server, see Preparations.
3. Prepare the database, see Setting up the database for ConSol CM. Check the database access with the user name and password which ConSol CM should use to access the database.
4. Install ConSol CM in the application server, see Installing the application server for ConSol CM. Check that the application server lists the ConSol CM application as “deployed”.
5. Set up the ConSol CM application by providing the initial configuration, see Setting up the ConSol CM application. Check that the ConSol CM start page is accessible via web browser.
   a. If you have installed a demo scene, adapt the scene.
   b. If you have started with an empty system, you have two options:
      - Build your own CM configuration from scratch.
      - Import an existing scene.

Optional steps:

- Configure more mailboxes on the *Email configuration* page of the Web Admin Suite.
- Install and configure CM/Track, see CM/Track. CM/Track must be licensed separately.
- Install and configure CM/Archive, see CM/Archive. CM/Archive must be licensed separately.
- Install and configure CM/Phone, see CM/Phone. CM/Phone must be licensed separately.

D.1.2 CMRF

If you want to use reporting, e.g. with CM/EBIA, you have to install and configure the CMRF (ConSol CM Reporting Framework) and the DWH (Data Warehouse), see Installing and setting up CMRF and DWH.

Mandatory steps:

1. Prepare the database, see Setting up the database for CMRF / DWH.
2. Install the CMRF, see Installing the CMRF.
3. Configure and fill the DWH, see Configuring the DWH.
4. Create the required cubes and / or reports, e.g. with CM/EBIA.

D.2 Preparations

This chapter contains information about the requirements for setting up ConSol CM in a single-server environment.

- Network
- License
- Application server
- Database
- Mail server
- Indexer
- LDAP authentication (Web Client and CM/Track)
- Data Warehouse

D.2.1 Introduction

When your company has decided to use ***ConSol CM*** for managing business processes, you, as an administrator, have to prepare your IT environment for the setup of the new application. This section provides lists of all components which have to be prepared to run the CM server with its collaborating systems.

> ⓘ It is recommended to set up at least two systems:
>
> - a test system
> - a productive (live) system
>
> All the steps described in the subsequent sections have to be performed for each CM system.

The following sections provide a short overview of a complete ConSol CM system. A detailed list of supported operation systems, application servers, database systems, and other systems, as well as storage and CPU requirements is given in the current *System Requirements*.

D.2.2 Components required for a ConSol CM setup

D.2.2.1 Network

A trouble-free communication over your intranet is indispensable. Please make sure that all routes (e.g. CM server <-> database, CM server <-> email server) are configured and are not blocked by firewalls.

D.2.2.2 License

When you start installing ConSol CM, make sure you have a valid license with all required modules and for a sufficient number of users for each module. If you are interested in knowing some details about CM license files, see License.

D.2.2.3 Application server

You need:

- **Machine for the ConSol CM application server** (Linux or Windows) with
  - sufficient CPU, RAM and HD storage
  - Java JDK (Java Development Kit) from Oracle or OpenJDK
  - application server (JBoss EAP or Wildfly)

Access to log files for the ConSol CM administrator is required during the operation of the system. The default directory for log files is `<JBOSS_HOME>/standalone/log`. If you want to change the location of the log files, please also refer to section Log files.

> ⚠ The application server should always run under a user without administrator permissions / root access to the operating system.

D.2.2.4 Database

Supported database systems

The ConSol CM database is a relational database which can be operated as Oracle, Microsoft SQL Server or MySQL system. For the configuration of the database connection, please refer to section Setting up the database for ConSol CM.

- **Oracle**
  One database scheme with one database user is used by ConSol CM.
- **Microsoft SQL**
  One database scheme with one database user is used by ConSol CM.
- **MySQL**
  One database with one database user is used by ConSol CM.
- **PostgreSQL**
  One database with one database user is used by ConSol CM.

Components required for the database

You need:

- **Database server** (Oracle, Microsoft SQL or PostgreSQL, for smaller systems or test systems also MySQL)

Prepared database or a prepared schema of sufficient size. The database URL with login and password is needed for setup.

If CM does not directly connect to the database but connects via a SAN etc., please make sure that the complete access URL is at hand

ⓘ Please make sure that the database is part of your every-day backup and that the restore routine has been tested.

D.2.2.5 Mail server

Email functionalities in CM

When you work with email functionalities, please keep in mind that there are two components: the mailbox and the email address.

One mailbox can be used by one or more email addresses. ConSol CM retrieves emails using the mailbox address (server, mailbox). However, in the subsequent email scripts which process incoming emails, the mailbox name as well as the email address can be used as variables which are applied to steer further email processing.

All email functionalities and the email configuration are explained in the *ConSol CM Administrator Manual*, sections *Email configuration* and *Scripts of the type Email*.

⚠ ConSol CM can retrieve emails from **real** mailboxes only. Public folders (e.g. on a Microsoft Exchange Server) are not supported.

ⓘ ConSol CM acts towards the mail server where it fetches the emails like a regular email client, using POP, POPs, IMAP or IMAPs. You can configure as many mailboxes as you like in the Web Admin Suite. Since one mailbox can be configured for one or more email addresses, there are two different approaches:

- Use only one mailbox which has several email addresses configured on the email server. Dispatch the emails to the desired queues in the email scripts based on the email address.
- Use one mailbox for each department or team. Dispatch the emails to the desired queues in the email scripts based on the mailbox name.

Components required for email functionalities

You need:

- **Mailboxes on the incoming email server**

  ConSol CM can work with mailboxes of any email server which supports the standard protocols (POP / POPs / IMAP / IMAPs). Public folders (e.g., on a Microsoft Exchange Server) of an email server cannot be used. There might be multiple email addresses for a mailbox.

  For each mailbox configuration within CM the following data is needed:

  - Mailbox name
  - Email server name
  - Login
  - Password
  - Protocol

> Please make sure that the CM server can access the email server and contact all desired mailboxes using the given protocol.

- **URL (smtp:// <SERVER>:<PORT>) of the outgoing mail server**

  If emails should be sent to external addresses via the ConSol CM system, the SMTP server has to allow relaying.

D.2.2.6 Indexer

There has to be sufficient storage space on the file system.

Please note that the search functionality, including the search for contacts while creating a case, does not work if the indexer is not able to run error-free. Therefore, the undisturbed access to the respective file system has to be ensured at any time. This needs to be checked especially in case the file system is located on a mounted partition.

D.2.2.7 LDAP authentication (Web Client and CM/Track)

If LDAP authentication should be used, the following information is needed:

- LDAP server name and port
- If required: user name and password of the LDAP user, in order to be able to contact the LDAP server
- Base DN of the LDAP tree, where the names of the user accounts which shall work with CM can be found
- DN of the attribute which shall be used for user authentication (e.g., *sAMAccountName* or *uid*)

LDAP authentication can also be configured after the initial setup of ConSol CM. Therefore, it is not mandatory that LDAP parameters are at hand during system setup.

Please refer to Configuring LDAP authentication for a more detailed explanation.

D.2.2.8 Data Warehouse

Supported database systems

- **Oracle**
  One database scheme with one database user is required for the DWH.
- **Microsoft SQL**
  One database scheme with one database user is required for the DWH.
- **MySQL**
  One database with one database user is required for the DWH.
- **PostgreSQL**
  One database with one database user is required for the DWH.

Components required for DWH setup

The following is needed:

- **Database/Scheme**
    - The DWH needs a separate database or a separate scheme of sufficient size. During setup of CMRF the database URL with login and password are needed.
      If you do not directly connect to the DWH database but connect via a SAN etc., please make sure that the complete access URL is at hand during CM setup.
- **Application server**
    - The DWH can either run on the same application server as ConSol CM or as a standalone Java application.

D.3 Setting up the database for ConSol CM

This chapter discusses the following:

D.3.1 Supported database management systems

ConSol CM needs a relational database management system (RDBMS) for the storage of most of its operational data. Three systems are supported:

- MySQL (for smaller and for test systems)
- Oracle
- Microsoft SQL Server
- PostgreSQL

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

D.3.2 Database management tools

We recommend to install a database management tool for your database server:

- Oracle SQL Developer
- MySQL Workbench
- Microsoft SQL Server Management Studio
- pgAdmin

D.3.3 MySQL

This chapter discusses the following:

- Installation prerequisites
- General preparations
- Recommended configuration settings
- Optional configuration settings
- Post-installation tasks

To provide a database for ConSol CM or for the CMRF the same steps are required. For the CMRF database just change the database/database user name.

The recommended database character set is *utf-8*. The collation depends on the MySQL version, see UTF-8 settings.

> MySQL 5 only supports up to 3-byte Unicode characters. Use MySQL 8 if you need 4-byte characters.

D.3.3.1 Installation prerequisites

MySQL must be installed with **InnoDB** as default storage engine.

This is configured in the MySQL configuration file. The default file is:

- Linux: `my.cnf`
- Windows: `my.ini`

UTF-8 settings

For MySQL 5, the following settings are necessary in the *mysqld* section of the configuration file:

```
[mysqld]
character-set-server=utf8
collation-server=utf8_general_ci
```

For MySQL 8, the following settings are necessary in the *mysqld* section of the configuration file:

```
character-set-server=utf8mb4
collation-server=utf8mb4_0900_as_ci
```

Timezone settings on Windows

When executing MySQL on Windows, you need to manually set the time zone (see https://dev.mysql.com/doc/refman/8.0/en/time-zone-support.html#time-zone-installation for further reference).

1. Go to https://dev.mysql.com/downloads/timezones.html and download the latest timezone posix file, e.g. `timezone_2021f_posix_sql.zip`, for your MySQL version.
2. Unpack the file and open a terminal at the location where the `timezone_posix.sql` file is located.
3. Run the following command:

```
mysql -u root -p mysql < timezone_posix.sql
```

> ⓘ `mysql` has to be added to the path as an environmental variable.

4. Enter the administrator password and press the enter key.
5. Restart the MySQL server (e.g. in the task manager under *Services* -> *MySQL* -> *restart*).
6. Open the directory of the MySQL server and add the timezone setting to the `my.ini` file, `[mysqld]` section.

   ```
   default_time_zone='Europe/Berlin'
   ```

   The timezone must match the timezone of the local machine.
7. Restart the MySQL server again.

D.3.3.2 General preparations

Create a database for ConSol CM. To do so, log in to MySQL Workbench or SQL Monitor (command line) with:

```
mysql -u root -p
```

(Without `-p` if no password was set during database setup.)

Execute the following commands to create a new database (placeholder *<CM_DB>*) and a new user (placeholder *<CM_USER>*. Grant the user *<CM_USER>* all necessary rights on the database *<CM_DB>*.

> ⓘ The default password policy requires at least 8 characters and a mix of letters and numbers. Nevertheless, this is not checked when executing the commands in the command line.

MySQL 5:

```
create database <CM_DB> CHARACTER SET utf8;
grant all on <CM_DB>.* to <CM_USER>@'%' identified by '<PASSWORD>';
grant all on <CM_DB>.* to <CM_USER>@localhost identified by '<PASSWORD>';
flush privileges;
```

MySQL 8:

```
CREATE USER '<CM_USER>'@'localhost' IDENTIFIED BY '<PASSWORD>';
CREATE USER '<CM_USER>'@'%' IDENTIFIED BY '<PASSWORD>';
create database <CM_DB> CHARACTER SET utf8mb4;
grant all on <CM_DB>.* to '<CM_USER>'@'%';
grant all on <CM_DB>.* to '<CM_USER>'@localhost;
flush privileges;
```

D.3.3.3 Recommended configuration settings

To avoid problems with scene import, it is recommended to set `xa_detach_on_prepare=off` globally for the database if you use MySQL 8 in version 8.0.29 or higher.

Add the following setting to the `[mysqld]` section of the `my.cnf` / `my.ini` file of your MySQL installation.

```
[mysqld]
xa_detach_on_prepare=OFF
```

D.3.3.4 Optional configuration settings

Sometimes threads may be stuck after setup, it may not be possible to log in to CM and restart it.

In this case, disable the query cache in your MySQL configuration by editing the following file:

- Linux: `/etc/mysql/my.cnf`
- Windows: `<DB_HOME>/my.ini`

Make the following settings:

```
query_cache_type = OFF
query_cache_size = 0
```

Alternatively, you can execute the following commands:

```
SET GLOBAL query_cache_type = OFF;
SET GLOBAL query_cache_size = 0;
```

If you are planning to have attachments bigger than 1 MB, the MySQL configuration file must also be adapted:

- Linux: `/etc/mysql/my.cnf`
- Windows: `<DB_HOME>/my.ini`

In the Server section `[mysqld]`, the allowed maximum size of attachments must be entered.

The following value allows 16 MB attachments:

```
max_allowed_packet=16777216
```

To allow larger attachments, e.g. 25M MB, you might also need to increase the size of the MySQL log files:

```
innodb_log_file_size=256M
```

D.3.3.5 Post-installation tasks

After the CM-specific database objects have been created during the first successful start of the CM application (see Setting up the ConSol CM application), you can set the starting ticket number. By default, the ticket numbers are assigned starting with 1, but you can determine that they should start with another number.

For example, execute the following statement to start the ticket numbers with 10000:

```
ALTER TABLE cmas_ticket AUTO_INCREMENT = 10000;
```

D.3.4 Oracle

This chapter discusses the following:

- Installation prerequisites
- General preparations
- Post-installation tasks

To provide a database for ConSol CM or for CMRF the same steps are required. For the CMRF database just change the database/database user name.

D.3.4.1 Installation prerequisites

An Oracle instance with database character set `AL32UTF8` is required. This can be checked with the following SQL statement:

```
SELECT * FROM NLS_DATABASE_PARAMETERS WHERE PARAMETER='NLS_CHARACTERSET';
```

The result should be:

Parameter	Value
NLS_CHARACTERSET	AL32UTF8

Initialization parameter settings

The following Oracle instance parameters must be changed to the following values or higher:

Parameter	Value
processes	20 + 200 * (number of CM connection pool) If shared with other servers, that + 200 for each CM connection pool
open_cursors	1000 **Note:** Necessary for statement caching in application server

The parameters can be changed as `sysdba` via `sqlplus` on the default instance with the following commands:

```
# sqlplus / as sysdba
> ALTER SYSTEM SET processes=420 SCOPE=SPFILE;
> ALTER SYSTEM SET open_cursors=1000 SCOPE=SPFILE;
```

Note that changes are only effective after restarting the database instance. This can be performed by:

```
> SHUTDOWN;
> STARTUP;
```

D.3.4.2 General preparations

Create a separate database user account (placeholder *<CM_USER>*) for each CM installation with the roles `CONNECT` and `RESOURCE`.

The password of this application user must not expire, as it is used in the database connection pool of the application server.

This can be created by the `sqlplus` command line interpreter via the following commands:

```
# sqlplus /nolog (start sqlplus client)
SQL> connect <SYSTEM>/<PASSWORD> --use your DBA user and password, connection
 to the default DB
  -- (or SQL>connect sys/<PASSWORD> as sysdba)
```

```
SQL> create user '<CM_USER>' identified by '<PASSWORD>';
SQL> grant connect, resource to '<CM_USER>';
SQL> grant create procedure to '<CM_USER>';
SQL> grant create view to '<CM_USER>';
```

⚠ For Oracle databases the user name is case insensitive during `CREATE USER` and `GRANT`. But it is stored in upper case in the database meta data. Therefore you must use the upper case name in `SELECT` statements on meta data as `dba_users` and `dba_role_privs`.

Tablespace for application-specific tables and indexes

The database user *<CM_USER>* needs a tablespace where he can store his tables and indexes.

In case a dedicated tablespace should be used for the ConSol CM application, the following SQL is the easiest way to create a tablespace:

```
SQL> create bigfile tablespace '<TABLESPACE_NAME>';
```

ⓘ This only works if `OMF` is enabled in the Oracle database. This should be the case for most databases.

If the default tablespace of the database (default: `USERS`) should not be used, the default tablespace of the user must be changed. This can be changed by the following command:

```
SQL> alter user '<CM_USER>' default tablespace '<TABLESPACE_NAME>';
```

For Oracle 12c, it is also required that the database user has a quota on this tablespace:

```
SQL> alter user '<CM_USER>' QUOTA UNLIMITED ON '<TABLESPACE_NAME>';
```

⚠ It is not possible to use more than one tablespace for one CM installation.

ⓘ The application-specific database objects are created by the CM application during the first startup of the application server.

Transactions recovery

The following settings must be applied for the user accessing an Oracle XA data source in order for XA recovery to operate correctly. The value *<CM_USER>* is the database user defined to connect from JBoss to Oracle.

```
sqlplus / as sysdba
#check version
SELECT * FROM V$VERSION;
#get correct spelling of CMUSER
SELECT username FROM dba_users where username = upper('<CM_USER>');

GRANT SELECT ON sys.dba_pending_transactions TO '<CM_USER>';
GRANT SELECT ON sys.pending_trans$ TO '<CM_USER>';
GRANT SELECT ON sys.dba_2pc_pending TO '<CM_USER>';
GRANT EXECUTE ON sys.dbms_xa TO '<CM_USER>';
```

D.3.4.3 Post-installation tasks

After the CM-specific database objects have been created during the first successful start of the CM application (see Setting up the ConSol CM application), you can set the starting ticket number. By default, the ticket numbers are assigned starting with 1, but you can determine that they should start with another number.

For example, execute the following statement as application Oracle user to start the ticket numbers with 10000:

```
drop sequence TICKET_SEQUENCE;
CREATE SEQUENCE TICKET_SEQUENCE" MINVALUE 10000 MAXVALUE
 9999999999999999999999999999 INCREMENT BY 1 START WITH 100000 CACHE 20 NOORDER
 NOCYCLE ;
```

D.3.5 Microsoft SQL Server

This chapter discusses the following:

- General preparations
- Post-installation tasks

To provide a database for ConSol CM or for CMRF the same steps are required. For the CMRF database just change the database/database user name.

D.3.5.1 General preparations

- **JDBC driver**

  Execute the stored procedure `sp_sqljdbc_xa_install`.

  > For Microsoft SQL Server versions prior to 2017, you need to install the XA features of the JDBC driver:
  >
  > - Download the JDBC driver from Microsoft and unpack it. Check the system requirements for the correct version.
  > - There is a directory `xa` which has several subdirectories for the XA driver DLLs. Choose the correct one according to the architecture your server is running on (e.g., `x86\sqljdbc_xa.dll`) and copy the DLL file into the `binn` directory of your Microsoft SQL Server installation, for example:
  >
  >   ```
  >   C:\Program Files (x86)\Microsoft SQL
  >    Server\MSSQL10.SQLEXPRESS\MSSQL\Binn
  >   ```
  >
  > - Then execute the `xa_install.sql` script from the above unpacked JDBC driver directory.

- **Enable XA Transactions**
  (Source: Execute the steps below Running the Microsoft DTC Service.)
  The Microsoft DTC service should be marked *Automatic* in Service Manager to make sure that it is running when the SQL Server service is started. To enable Microsoft DTC for XA transactions, you must follow these steps:

  1. Click the *Start* button, type *dcomcnfg* in the *Start Search* box, and then press *Enter* to open *Component Services*. You can also type “%windir%\system32\comexp.msc” in the *Start Search* box to open *Component Services*.
  2. Expand *Component Services*, *Computers*, *My Computer*, and then *Distributed Transaction Coordinator*.
  3. Right-click *Local DTC* and then select *Properties*.
  4. Click the *Security* tab on the *Local DTC Properties* dialog box.
  5. Select the *Enable XA Transactions* check box and then click *OK*. This will cause a Microsoft DTC service restart.

6. Click *OK* again to close the *Properties* dialog box and then close *Component Services*.
7. Stop and then restart Microsoft SQL Server to make sure that it syncs up with the Microsoft DTC changes.

- **Database and database user for CM**
    - Create a new database (placeholder *<CM_DB>*). Execute the following statement on this database:

      ```
      ALTER DATABASE <CM_DB> SET READ_COMMITTED_SNAPSHOT ON;
      ```

    - Create an application user (placeholder *<CM_USER>*) with the following authorizations:
        - Grant user *<CM_USER>* access to database medium `master` (role `public` and role `SqlJDBCXAUser`).
        - Grant user *<CM_USER>* access to database medium *<CM_DB>* (role `public` and role `db_owner`).

D.3.5.2 Post-installation tasks

After the CM-specific database objects have been created during the first successful start of the CM application (see Setting up the ConSol CM application), you can set the starting ticket number. By default, the ticket numbers are assigned starting with 1, but you can determine that they should start with another number.

For example, execute the following statement using SQL Management Studio to start the ticket numbers with 10000:

```
dbcc checkident ('cmas_ticket', reseed, 10000)
```

Performance tip for Windows operating system

When running Microsoft SQL Server on Windows, the power options have an impact on the database performance. Change the power plan from *Balanced* to *High Performance* to improve database performance.

D.3.6 PostgreSQL

This chapter discusses the following:

- Installation prerequisites
- General preparations

D.3.6.1 Installation prerequisites

Make the following settings in the postgresql.conf file in the data directory of the PostgreSQL installation:

```
max_connections=200
max_prepared_transactions = 200
jit=off
```

The setting jit=off disables JIT for all PostgreSQL databases. If this is not desired, you can make this setting for the ConSol CM and the CMRF databases individually by executing:

```
> alter database cmdatabase set jit=off;
```

D.3.6.2 General preparations

Create the database and database user:

```
CREATE DATABASE cmdatabase;
CREATE USER cmuser WITH PASSWORD 'password';
GRANT ALL PRIVILEGES ON DATABASE cmdatabase TO cmuser;
```

Connect to the database and grant access to the public schema to the database user:

```
GRANT ALL ON SCHEMA PUBLIC TO cmuser;
```

Alternatively, you can create use an own schema for the ConSol CM database:

```
CREATE SCHEMA cmschema;
GRANT ALL ON SCHEMA cmschema TO cmuser;
ALTER DATABASE cmdatabase SET search_path TO cmschema;
```

D.4 Installing the application server for ConSol CM

This chapter discusses the following:

D.4.1 General note

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

D.4.2 Supported application servers

ConSol CM version 6.16 supports two application servers:

- JBoss EAP 7.4
- Wildfly 26.1.3

The ConSol CM distribution is the same for both application servers.

D.4.3 Supported database systems

Three Relational Database Management System (RDBMS) types are supported:

- MySQL
- Oracle
- Microsoft SQL Server
- PostgreSQL

D.4.4 Supported operating systems

Two main operation systems are supported:

- Linux
- Windows

D.4.5 Installing ConSol CM

This chapter discusses the following:

- Introduction
- Prerequisites
- Installing JBoss
- Installing ConSol CM
- Configuring the database connection
- Starting and stopping the CM server
- Application server and database settings
- Autostarting the CM server

D.4.5.1 Introduction

This section explains the setup of a new ConSol CM system. If you need information about a system update, please refer to the *Release Notes* of the respective CM version.

> ⓘ Please contact the ConSol CM support or your ConSol CM consultant for information about setting up ConSol CM in a cluster.

D.4.5.2 Prerequisites

> ⓘ To extract the JBoss .zip files, about 2 GB of hard drive space are required.
>
> The final JBoss installation (without CM) requires about 750 MB.

- Installation of Java JDK 11 or 17 on the server
- JBoss EAP version 7.4 or Wildfly 26.1.3
- ConSol CM distribution, there is a specific distribution for the combination of application server and database.
- Installed database server, see section Setting up the database for ConSol CM
- In case of MySQL as database: MySQL connector 8.0.22
- Installed email server

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

D.4.5.3 Installing JBoss

Please perform the following steps. There might be slight differences depending on the systems or versions.

⚠ The application server should always run under a user without administrator permissions / root access to the operating system.

Windows

1. Extract the downloaded JBoss archive into a new folder.
2. Edit the file `<JBOSS_HOME>\bin\standalone.conf.bat` (also see standalone.conf.bat):
   - Change the Java memory options if needed:

     ```
     set "JAVA_OPTS=-Xms4g -Xmx4g -XX:MaxMetaspaceSize=512m -
      Djava.net.preferIPv4Stack=true"
     ```

   - Add the Java path (the path and Java version can be different on your system):

     ```
     set "JAVA_HOME=C:\Program Files\Java\jdk-11.0.17+8"
     ```

     Alternatively you can add *JAVA_HOME* to your operating system environment variables. In this case it is not necessary to set the line above. Refer to the Java documentation on how to do this.

   - Add `JDK_SERIAL_FILTER="maxbytes=104857600;maxdepth=128;maxarray=100000;maxrefs=300000"` (increase `maxbytes` from 10 to 100MB) or `DISABLE_JDK_SERIAL_FILTER=true` (disable serialization) to avoid problems with object (de)serialization which might occur in the context of data transfer to CMRF (only needed for JBoss EAP > 7.4.12).

Linux

1. Extract the downloaded JBoss archive into a new folder.
2. Edit the file `<JBOSS_HOME>/bin/standalone.conf` (also see standalone.conf):
   - On most Linux Systems the `JAVA_HOME` environment variable should already be set. If you wish to use the system standard Java version you do not need to specify the `JAVA_HOME` environment variable.

     If in your case the variables are not set or you wish to specify a path to a specific Java installation add the following line (the path and Java version can be different on your system):

     ```
     JAVA_HOME="/opt/java/64/jdk-11.0.17+8"
     ```

     You can use `echo $JAVA_HOME` to check if the variable is set correctly.

   - Adjust the JVM memory options, the `-Xmx` value should be at least “4G”. If you want to set the options only when no `JAVA_OPTS` have been set yet, then use the following statement:

```
## Specify options to pass to the Java VM.#
if [ "x$JAVA_OPTS" = "x" ]; then
   JAVA_OPTS="-Xms4g -Xmx4g -XX:MaxMetaspaceSize=512m -
    Djava.net.preferIPv4Stack=true"
   JAVA_OPTS="$JAVA_OPTS -Djboss.modules.system.pkgs=$JBOSS_MODULES_
    SYSTEM_PKGS -Djava.awt.headless=true"
else
   echo "JAVA_OPTS already set in environment; overriding default
    settings with values: $JAVA_OPTS"
fi
```

If you want to overwrite existing `JAVA_OPTS`, do not put the `JAVA_OPS=` statements into the `if` clause

- If the `JBOSS_HOME` environment variable is already set to a path other than your server path, add the following line at the top of `<JBOSS_HOME>/bin/standalone.conf`. Remember that <JBOSS_HOME> is the path to your JBoss installation.

```
JBOSS_HOME=<JBOSS_HOME>
```

- Add `JDK_SERIAL_FILTER="maxbytes=104857600;maxdepth=128;maxarray=100000;maxrefs=300000"` (increase `maxbytes` from 10 to 100MB) or `DISABLE_JDK_SERIAL_FILTER=true` (disable serialization) to avoid problems with object (de)serialization which might occur in the context of data transfer to CMRF (only needed for JBoss EAP > 7.4.12).

3. Change owner to a non-admin Linux user (example: *consolcm*) and make the files executable:
   - chown -R consolcm <JBOSS_HOME>/bin
   - chmod 755 -R <JBOSS_HOME>/bin

D.4.5.4 Installing ConSol CM

1. Extract the ConSol CM distribution `dist-package-distribution-<CM_VERSION>-wildfly_jboss.zip` into <JBOSS_HOME>.
2. Overwrite the 4 subfolders `bin`, `domain`, `modules` and `standalone`, i.e. confirm the overwriting of existing files.

D.4.5.5 Configuring the database connection

The database connection is configured in the file `cm6-config.properties` in the `bin` directory.

You need to set the following properties:

- **datasource.url**
  URL of the non-transactional datasource

  Example values:

  - **MySQL 5**: jdbc:mysql://localhost/cmdatabase?serverTimezone=Europe/Berlin
  - **MySQL 8**: jdbc:mysql://localhost/cmdatabase?characterEncoding=UTF-8&serverTimezone=Europe/Berlin

    - **Oracle**: jdbc:oracle:thin:@localhost:1521/SERVICE_NAME
    - **Microsoft SQL Server**: jdbc:sqlserver://localhost:1433;databaseName=cmdatabase
    - **PostgreSQL**: jdbc:postgresql://localhost:5432/cmdatabase

- **datasource.username**
  Name of the database user for ConSol CM
- **datasource.password**
  Password of the database user for ConSol CM
- **xa.datasource.url**
  URL of the transactional datasource (same URL as the non-transactional datasource)
- **xa.datasource.username**
  Name of the database user for ConSol CM
- **xa.datasource.password**
  Password of the database user for ConSol CM

The following section describes specific steps for the different database systems.

> If you use ConSol CM with CMRF, uncomment the lines with the transactional CMRF datasource and set the required values.

MySQL

MySQL Connector/J is the official JDBC driver for MySQL. The connector is not part of the CM distribution package. Therefore, you must copy it manually into the JBoss directory. You can download the connector from http://dev.mysql.com/downloads/connector/j/ (select the platform-independent download). You need version 8.0.22.

1. Extract the MySQL connector archive `mysql-connector-java-8.0.22.zip` into a temporary folder. The folder contains the JDBC driver file `mysql-connector-java-8.0.22-bin.jar`.
2. Copy the JDBC driver, e.g., `mysql-connector-java-8.0.22-bin.jar`, into the folder `<JBOSS_HOME>\modules\system\layers\base\com\mysql\jdbc\main`.

   > If you receive an error message similar to the following, then you might have to use a newer connector.

   ```
   ERROR [ontroller.management-operation] [-] JBAS014613: Operation ("add")
    failed - address: ([
      ("subsystem" => "datasources"),
      ("jdbc-driver" => "mysql-driver")
   ]) - failure description: "JBAS010441 ...
   ```

   Code example 1: *Error message for MySQL connector*

3. Edit the file `<JBOSS_HOME>\modules\system\layers\base\com\mysql\jdbc\main\module.xml` located in the same folder. Change the line highlighted in red, so it points to your version of the MySQL connector.

```
<?xml version="1.0" encoding="UTF-8"?>
<module xmlns="urn:jboss:module:1.0" name="com.mysql.jdbc">
   <resources>
      <resource-root path="mysql-connector-java-8.0.22-bin.jar"/>
   </resources>
   <dependencies>
      <module name="javax.api"/>
      <module name="javax.transaction.api"/>
   </dependencies>
</module>
```

Oracle

The Oracle JDBC driver is part of the distribution package. It can be found at `<JBOSS_HOME>/modules/system/layers/base/oracle/jdbc/main/`.

Microsoft SQL Server

The Microsoft SQL Server JDBC driver is part of the distribution package. It can be found at `<JBOSS_HOME>/modules/system/layers/base/com/microsoft/sqlserver/jdbc/main/`.

PostgreSQL

The Microsoft SQL Server JDBC driver is part of the distribution package. It can be found at `<JBOSS_HOME>/modules/system/layers/base/org/postgresql/main`.

D.4.5.6 Starting and stopping the CM server

There are two levels to consider for starting and stopping ConSol CM:

- The basic start and stop commands of ConSol CM are adapted start and stop commands of the respective application server, which can be executed on the command line.
- The integration of these commands into an environment which allows automatic system start and stop. You might want to write an *init.d* or *systemd* script for a Linux system, or configure CM as a service on a Windows system.

The following section provides information about the simple command line parameters which can be used to start and stop the system.

> ⓘ If the database system has to be restarted, stop the ConSol CM system first, wait until the database system is available again, and then start the ConSol CM application server.

JBoss on Windows

Starting JBoss on Windows

Run the following command to start the JBoss server. Alternatively you can configure JBoss to run as a service (see Installing JBoss EAP as a Windows service).

```
<JBOSS_HOME>\bin\standalone.bat -P <JBOSS_HOME>\bin\cm6-config.properties -c
 cm6-<DB>.xml
```

Replace `cm6-<DB>.xml` with the configuration file for your database:

- `cm6-mysql-5.xml` for MySQL 5
- `cm6-mysql-8.xml` for MySQL 8
- `cm6-oracle.xml` for Oracle
- `cm6-mssql.xml` for Microsoft SQL Server
- `cm6-postgresql.xml` for PostgreSQL

Shutting down JBoss on Windows

Run the following command to stop the JBoss server. Alternatively you can put it into the configuration of the Windows service.

```
<JBOSS_HOME>\bin\jboss-cli.bat --controller=localhost:9999 --connect --
 command=:shutdown
```

If you did not set a system environment variable for the Java home directory, add the path to the JDK to `<JBOSS_HOME>\bin\jboss-cli.bat`. This prevents Java warnings.

Example for JDK version 11.0.17, replace by your Java version.

```
set "JAVA_HOME=C:\Program Files\Java\jdk-11.0.17+8"
```

In case the `JBOSS_HOME` environment variable is already set in the system, you can specify the server-specific path at the beginning of `JBOSS_HOME>\bin\jboss-cli.bat`in the following syntax:

```
set JBOSS_HOME='<JBOSS_HOME>'
```

JBoss on Linux

Starting JBoss on Linux

Run the following command to start the JBoss server for a setup with ConSol CM without CMRF. Alternatively you can put it into a start script (usually *init.d* or *systemd*).

```
<JBOSS_HOME>/bin/standalone.sh -P <JBOSS_HOME>/bin/cm6-config.properties -c
 cm6-<DB>.xml
```

Replace `cm6-<DB>.xml` with the configuration file for your database:

- `cm6-mysql-5.xml` for MySQL 5
- `cm6-mysql-8.xml` for MySQL 8
- `cm6-oracle.xml` for Oracle
- `cm6-mssql.xml` for Microsoft SQL Server
- `cm6-postgresql.xml` for PostgreSQL

If JBoss starts normally (*ConSol CM Startup* appears in the `server.log` file) but *localhost:8080* cannot be reached, try setting `<CM_HOST_IP>` to `0.0.0.0`.

Shutting down JBoss on Linux

Run the following command to stop the JBoss server. Alternatively you can put it into a stop script (usually *init.d* or *systemd*).

```
<JBOSS_HOME>/bin/jboss-cli.sh --controller=localhost:9999 --connect --
 command=:shutdown
```

If you did not set a system environment variable for Java, add the path to the JDK to `<JBOSS_HOME>/bin/jboss-cli.sh`. This prevents Java warnings.

Example for JDK version 11.0.17, replace by your Java version.

```
JAVA_HOME="/opt/java/64/jdk-11.0.17+8"
```

In case the `JBOSS_HOME` environment variable is already set in the system, you can specify the server-specific path at the beginning of `<JBOSS_HOME>/bin/jboss-cli.sh` in the following syntax:

```
JBOSS_HOME='<JBOSS_HOME>'
```

General settings

- In case `-b=localhost` or `-b=127.0.0.1` is set as parameter, ConSol CM is only accessible from the same server, where JBoss is running.
- Enter the network IP or the network name to make ConSol CM accessible from outside the sever. In this case ConSol CM URL is not accessible from the inside using the localhost URL.
- If you enter `-b=0.0.0.0` the server is accessible from the outside and inside using the server URL or using the localhost URL (take care of all required security-related precautions!).
- After the server has started, open a browser window and type the server URL to start the ConSol CM setup, see section Setting up the ConSol CM application.

D.4.5.7 Application server and database settings

Java settings

The following JAVA_OPTS settings are used by default when setting up a new ConSol CM system.

Default memory configuration for JBoss 7.4 on Java 11 (Xms and Xmx should be set to the same value):

- -Xms4g -Xmx4g -XX:MaxMetaspaceSize=512m

Explanation:

- Start the application server with initial heap size of approx. 4 GB.

  ```
  -Xms4G
  ```

- The maximum heap size for the application sever is also approx. 4 GB.

```
-Xmx4G
```

- The maximum space for the permanent space where compiled Java classes, methods and other required objects are stored during operation and never de-allocated. If this space is too small, even after a full garbage collection, an out of memory error will be thrown and the JVM will crash.

```
-XX:MaxMetaspaceSize=512m
```

ⓘ These parameters are starting parameters for production systems. Depending on the size of the system it might be necessary to adjust them. The following factors are important in this context:

- number of concurrent users
- amount of cases, contacts, resources
- amount and size of emails
- amount and size of attachments
- size of the index

As an administrator, you need to check the Java heap size and the garbage collection activity using tools like jconsole or jvisualvm. If the heap size is higher than 80 - 90% of the maximum value and the garbage collection runs frequently, you need to increase the memory allocation (Xmx).

The list includes the officially supported default parameters for JBoss 6.4. They should be suitable for most installations.

- -verbose:gc
- -XX:+PrintGCDetails
- -XX:+PrintGCDateStamps
- -XX:+UseGCLogFileRotation
- -XX:NumberOfGCLogFiles=5
- -XX:GCLogFileSize=3M
- -XX:-TraceClassUnloading
- -XX:+UseCompressedOops
- -Djava.net.preferIPv4Stack=true
- -Djboss.modules.policy-permissions=true
- -Djboss.modules.system.pkgs=$JBOSS_MODULES_SYSTEM_PKGS

Depending on the individual usage of the ConSol CM system, some additional settings might be needed. The following list includes settings which are used on some systems. As these settings are not officially supported, please contact your ConSol CM consultant or the ConSol CM support before using them in your production environment.

- -server
- org.apache.tomcat.util.http.Parameters.MAX_COUNT=6000
- org.apache.coyote.http11.Http11Protocol.MAX_HEADER_SIZE=65535
- org.apache.tomcat.util.http.MimeHeaders.MAX_COUNT=200
- com.arjuna.ats.arjuna.allowMultipleLastResources=true
- -Dhttp://apache.org/xml/features/nonvalidating/load-external-dtd=false
  Prevent cross-scripting attacks
- -Djboss.platform.mbeanserver
  Enable the JMX bean server
- -Djgroups.marshalling.compatible=true
  Use old JGroups protocol instead of newer one
- -Dorg.jboss.resolver.warning=true
  Warning when an XML entity is defined as SYSTEM and the protocol is not `file://` or `vfsfile://`
- -Dserialization.jboss=false
  Disable JBoss serialization and use standard Java serialization instead
- -Dsun.rmi.dgc.client.gcInterval=3600000
- -XX:+UseParallelOldGC
  Use the old garbage collection or any better matching garbage collection depending on your requirements.
- -XX:+HeapDumpOnOutOfMemoryError
- -XX:HeapDumpPath=/some/path
  Use a path, you cannot use a file here
- -Xloggc:some/path/based/on/JbossRoot/log/gc-%t.log
- -XX:ReservedCodeCacheSize=150m
- -XX:CompressedClassSpaceSize=128m

Database settings

Database access can be a bottleneck impairing system performance. You can increase the number of database connections in the connection pool by modifying the minimum (`<min-pool-size>`) and maximum pool size (`<max-pool-size>`) in the configuration file, e.g. `<JBOSS_HOME>\standalone\configuration\cm6-<DB>.xml`.

```
...

<pool>
   <min-pool-size>5</min-pool-size>
   <max-pool-size>200</max-pool-size>
   <prefill>true</prefill>
</pool>

...
```

D.4.5.8 Autostarting the CM server

Installing JBoss EAP as a Windows service

You can install JBoss EAP as a Windows service. The service file needs to be adapted by providing the correct directory for the log files and the CM start configuration. Please also see Starting and stopping the CM server.

ⓘ If you have a maintenance contract, please contact the ConSol CM support to receive further assistance with the configuration.

Autostarting JBoss EAP on Linux systems

You can use *systemd* or *init.d* scripts to configure autostart for ConSol CM on Linux systems. The JBoss package delivered with ConSol CM contains a *systemd* script which can be used for this purpose.

ⓘ If you have a maintenance contract, please contact the ConSol CM support to receive further assistance with the configuration.

D.5 Setting up the ConSol CM application

This chapter discusses the following:

D.5.1 Introduction

After you have installed ConSol CM as an application on the application server (see section Installing the application server for ConSol CM), you can install the ConSol CM application. In this step, the database structure is built and some basic aspects are configured.

D.5.2 Starting the setup

Open the ConSol CM URL in a web browser. The default URL on JBoss is `http://localhost:8080`.

You see the ConSol CM setup dialog, which guides you through the setup.

D.5.2.1 Step 1: General

One general settings has to be made here:

- **Data directory**
  Choose a folder for application-relevant files like the CM index (for an explanation of this directory, please refer to section ConSol CM data directory.

D.5.2.2 Step 2: Database

One database-relevant setting has to be made here:

- **Recreate tables**
  When using the setup dialog during a reinstallation of ConSol CM, you can tick this checkbox to rebuild the database scheme. This deletes all data. For a new installation, you do not need to select this checkbox.

D.5.2.3 Step 3: Administrator

Some configuration parameters concerning the administrator account of the system have to be made here:

- **Login**
  The login name of the administrator account (for the Web Admin Suite).
- **Password**
  The password of this administrator account.
- **Confirm password**
  Confirm the password for the administrator account.
- **Email**
  Enter the email address for the CM administrator. This is the overall admin email is entered in the system property cmas-core-security, admin.email. For some modules, separate email addresses can be configured, see section *Email configuration* in the *ConSol CM Administrator Manual*.

  > ⓘ You can change this email address later using the Web Admin Suite.

> ⚠ It is recommended to create personalized users and grant them administrator permissions instead of using a generic administrator account. Please remove the administrator account created during the setup before using the system in production environments. Be aware that this administrator account always uses database authentication even if LDAP or SSO is used for the rest of the user accounts.

D.5.2.4 Step 4: Scene

Here, you can decide if you want to set up the system with a pre-defined environment or if you would like to start with an empty system and import a scene later.

For a detailed explanation about scenes in ConSol CM, please refer to the *ConSol CM Administrator Manual*, section *Staging*.

There are three options:

- **No scene**
  Select this value if you would like to start with an empty system. You can import a scene later using the Web Admin Suite.
- **Test and Demo Scene**
  This installs a demo environment with demo data.
- **Ticketing Scene**
  This installs the ConSol CM/Ticketing product.

D.5.2.5 Step 5: Outgoing E-mail

Optional. You can configure the SMTP server for outgoing emails here. Alternatively, you can leave this screen empty and perform the configuration later in the Web Admin Suite.

- **Outgoing E-mail connection**
  Enter the URL of the SMTP server which should be used for sending emails. See section *Email configuration* in the administrator manual for further details

> ⚠ Email passwords containing special characters might not work during the setup of ConSol CM. Please use the Web Admin Suite to set such passwords.

D.5.2.6 Step 6: Incoming E-mail

Optional. You can define the parameters for the incoming emails here. Alternatively, you can leave this screen empty and perform the configuration later in the Web Admin Suite.

- **Incoming E-mail connection**
  Define an email account (mailbox) where ConSol CM should retrieve incoming emails. See section *Email configuration* in the administrator manual for further details

> ⚠ Email passwords containing special characters might not work during the setup of ConSol CM. Please use the Web Admin Suite to set such passwords.

D.5.2.7 Step 7: Start setup

Click *Finish* to start the system setup and configuration. This will take a while (some browsers might display a timeout error, but this does not indicate any problem).

When the setup has finished you see a screen with the message *Setup has finished*.

D.5.2.8 Step 8: Check the setup

Reload the page or open the URL of the CM system in another browser tab/window to see the start page. It contains links to open the Web Client and Web Admin Suite.

Open the Web Admin Suite to add your license to the system, see License.

D.6 Authentication

D.6.1 Introduction to authentication in ConSol CM

ConSol CM provides three authentication methods to confirm the identity of its users. You can use the ConSol CM database, an LDAP server or a SSO service for authentication. Authorization is done via roles.

D.6.1.1 Concepts, terms and definitions

Concept	Other terms	Definition
authentication		Process to confirm the identity of the users
authorization		Process to determine the access permissions of the authenticated users, is done via roles
database authentication		Authentication using the ConSol CM database
LDAP authentication		Authentication using an LDAP server
SSO authentication		Authentication using a single sign-on technology, e.g. Microsoft Active Directory Federation Services or Azure AD

D.6.2 Available authentication methods

D.6.2.1 Database authentication

Availability:

- Web Client
- CM/Track

The user name and password are saved in the ConSol CM database.

If database authentication is used, you can set a password policy and configure the password reset functionality.

D.6.2.2 LDAP authentication

Availability:

- Web Client
- CM/Track

The user name is saved in the ConSol CM database. The password is saved on the LDAP server. Thus, the users or contacts cannot change it in ConSol CM.

See Configuring LDAP authentication.

D.6.2.3 SSO authentication

Availability:

- Web Client
- CM/Track

The user name is saved in the ConSol CM database. The password is saved in the active directory. Thus, the users cannot change it in ConSol CM. The credentials are retrieved from a valid Windows session.

See Configuring SSO authentication.

D.6.3 Basic tasks

D.6.3.1 Determining the authentication method

The authentication method is determined using system properties.

Determining the authentication method for the Web Client

The authentication method for the Web Client is determined using the system property cmas-core-security, engineer.authentication.method

Depending on the configured authentication method, you need to fill different fields on the *Users* page:

- **Database authentication**: Fields *Login* and *Password*
- **LDAP authentication**: Field *LDAP ID*
- **SSO authentication**: Field *Login*

Determining the authentication method for CM/Track

The authentication method for CM/Track is determined using the system property cmas-core-security, contact.authentication.method.

Depending on the configured authentication method, you need to create fields for the credentials in the contact data on the *Contact fields* page:

- **Database authentication**: User name (field with the setting *User name for CM/Track*) and password (field with the setting *Password for CM/Track*).
- **LDAP authentication**: LDAP ID (field with the setting *User name for CM/Track* and *LDAP ID for CM/Track*).
- **SSO authentication**: User name (field with the setting *User name for CM/Track*).

> ⓘ If database authentication is used, you need define whether CM/Track user names should be case-sensitive in the system property cmas-core-security, policy.track-.username.case.sensitive. Only set it to “true” if the database collation supports case-sensitive strings.

After creating the fields, they need to be filled out for each contact on the contact page of the Web Client.

Mixed authentication method

A mixed authentication method with both LDAP and database authentication is available for both the Web Client and CM/Track. This mode is configured by setting the `engineer.authentication.method` or `contact.authentication.method` properties to one of the following values:

- **LDAP,DATABASE**
  If an LDAP ID is saved in the user / contact data, the first login attempt is made using the available LDAP servers. If the login fails, an attempt to log in using the database is made, provided that a user name and password are saved in the user / contact data.
- **DATABASE,LDAP**
  If a user name and a password are saved in the user / contact data, the first login attempt is

made using the database. If the login fails, an attempt to log in using the available LDAP servers is made, provided that an LDAP ID is saved in the user/ contact data.

D.6.4 Advanced tasks

Advanced task only exist for database authentication. If LDAP or SSO authentication is used, the advanced settings need to be configured in LDAP or the SSO technology.

D.6.4.1 Setting a password policy

Setting a password policy is optional. A password policy can only be configured if database authentication is used.

The following settings can be used for the password policy:

- cmas-core-security, policy.password.pattern (String)
  RegEx pattern for the password, default value: ^(?=.*[0-9])(?=.*[A-Z])(?=.*[a-z]).{7,}$ (at least 7 characters, at least one upper case letter, one lower case letter and one number)
- cmas-core-security, policy.password.age (Integer)
  Maximum validity period, in number of days, example 183 (6 months), default value: 5500 (= 15 years, i.e., no password change enforced).
- cmas-core-security, policy.rotation.ratio (Integer)
  Number which defines the number of previous passwords which may not be identical, example and default value: 5.
- cmas-core-security, policy.username.case.sensitive (Boolean)
  Defines whether the password is case-sensitive. Example and default value: true.
  Note that this setting is affected by the MySQL collation setting and needs the correct collation to work properly with MySQL.

> ⓘ For LDAP and SSO, the password policy needs to be configured in LDAP or the SSO technology.

D.6.5 Configuring LDAP authentication

D.6.5.1 Introduction to LDAP authentication in ConSol CM

LDAP authentication is an authentication method which can be used to verify the identify of users in the Web Client and contacts in CM/Track.

Concepts, terms and definitions

Concept	Other terms	Definition
LDAP		Abbreviation of Lightweight Directory Access Protocol, protocol to manage login information for several applications
LDAPS	LDAP over SSL, secure LDAP	LDAP using SSL for communication

D.6.5.2 Basic tasks

Setting up LDAP for the Web Client

The LDAP server can be defined using the following system properties from the module `cmas-core-security`.

The placeholder {name} allows to define configurations for several different LDAP servers. It always needs to be provided, even if you only configure one LDAP server. You should use a simple string, which does not include any keywords as *internal* or *external* and does not contain special characters.

Authentication attempts against the LDAP servers are made until the first success, starting with the first server in an ascending alphabetical order.

Set the following system properties:

- engineer.authentication.method
  LDAP
- ldap.engineer.{name}.basedn
  The root path in the LDAP tree used to look up the user by the LDAP ID, e.g. `ou=accounts,dc=mycompany,dc=de`.
- ldap.engineer.{name}.password
  The password used for connecting to the LDAP server to look up the users. Only needed if the lookup cannot be done anonymously.
- ldap.engineer.{name}.providerurl
  The complete address of the LDAP server in the format `ldap[s]://host:port`. Usually, the standard port is 636 and the port for the global catalog is 3269.
- ldap.engineer.{name}.searchattr
  Search attribute to look up a user by LDAP ID, e.g. `uid`.
- ldap.engineer.{name}.userdn
  The LDAP user for connecting to the LDAP server to look up the users. Only needed if the lookup cannot be done anonymously.
- ldap.initialcontextfactory
  This is a predefined global property. The value should be `com.sun.jndi.ldap.LdapCtxFactory`.

ⓘ The LDAP IDs for the users are saved on the *Users* page of the Web Admin Suite.

Setting up LDAP for CM/Track

The LDAP server can be defined using the following system properties from the module `cmas-core-security`.

The placeholder {name} allows to define configurations for several different LDAP servers. It always needs to be provided, even if you only configure one LDAP server. You should use a simple string, which does not include any keywords as *internal* or *external* and does not contain special characters.

Authentication attempts against the LDAP servers are made until the first success, starting with the first server in an ascending alphabetical order.

Set the following system properties:

- contact.authentication.method
  LDAP
- ldap.contact.{name}.basedn
  The root path in the LDAP tree used to look up the contact by the LDAP ID, e.g. `ou=accounts,dc=mycompany,dc=de`.
- ldap.contact.{name}.password
  The password used for connecting to the LDAP server to look up the contacts. Only needed if the lookup cannot be done anonymously.
- ldap.contact.{name}.providerurl
  The complete address of the LDAP server in the format `ldap[s]://host:port`. Usually, the standard port is 636 and the port for the global catalog is 3269.
- ldap.contact.{name}.searchattr
  Search attribute to look up a contact by LDAP ID, e.g. `uid`.
- ldap.contact.{name}.userdn
  The LDAP user for connecting to the LDAP server to look up the contacts. Only needed if the lookup cannot be done anonymously.
- ldap.initialcontextfactory
  This is a predefined global property. The value should be `com.sun.jndi.ldap.LdapCtxFactory`.

These system properties might not be present initially in your CM system. Just add them manually. You do not need to restart the server after changing the above system properties. The change is applied to all cluster nodes.

ⓘ The LDAP IDs for the contacts are saved in contact fields with the user name and LDAP ID settings in the Web Client.

D.6.5.3 Advanced tasks

Setting up LDAPS

Per default, when an LDAP client accesses an LDAP server, the information is transferred in clear text. In case you want the user name and password to be transferred to the LDAP server in encrypted form, you have to set up the LDAP authentication using LDAPS.

You have to configure the CM server machine (Java) in a way that can use certificates. One way to do this for a Linux environment is described in the following section.

1. Retrieve the certificate:

   ```
   openssl s_client -connect dc2.mydomain.com:ldaps
   ```

2. The answer will contain a section which starts with `"---BEGIN CERTIFICATE "` and ends with `"END CERTIFICATE ---"`. Copy this section to a file, e.g., `/tmp/certificate2_dc2_mydomain_com.txt`
3. Import the certificate to the truststore of your machine, e.g., `/home/mydirectory/mytruststore`

   ```
   $JAVA_HOME/bin/keytool -import -alias <arbitrary> -trustcacerts -keystore
    /home/mydirectory/mytruststore -file/tmp/certificate2_dc2_mydomain_
    com.txt
   ```

   You have to enter (set) a password.
4. Enter the truststore in the ConSol CM config file in JAVA_OPTS:

   ```
   -Djavax.net.ssl.trustStore=/home/mydirectory/mytruststore -
    Djavax.net.ssl.trustStorePassword=<see above>
   ```

D.6.6 Configuring SSO authentication

D.6.6.1 Introduction to SSO authentication in ConSol CM

SSO is an authentication method which can be used to verify the identity of users in the Web Client and CM/Track. It is based on OpenID Connect and can be set up using Microsoft Active Directory Federation Services or Azure AD.

Concepts, terms and definitions

Concept	Other terms	Definition
SSO		Abbreviation of single sign-on. Authentication mechanism which allows users to access several applications after logging in once
OpenID Connect		Authentication mechanism based on the OpenID protocol, which allows clients to obtain information about authenticated user sessions

D.6.6.2 Basic tasks

Setting up SSO for the Web Client

The SSO configuration on the ConSol CM side is done using system properties from the module `cmas-core-security`. Set the following system properties:

- oidc.web.enabled.default: Indicates whether user authentication using SSO via OIDC is enabled.
- oidc.web.authority.default: Indicates the URL of the authenticating authority, e.g. ADFS. Example: https://localhost/adfs
- oidc.web.clientId.default: Indicates the client ID (application ID) of the application, as registered in ADFS or Azure AD.
- oidc.web.clientSecret.default: Indicates the secret of the client, generated using ADFS or Azure AD.
- oidc.web.redirectUri.default: Indicates the redirect URI where authentication responses can be received. This is either the OIDC endpoint on the ConSol CM server or on the load balancer. Example: http://localhost/cm-client/oidc/
- oidc.web.usernameClaim.default: Indicates the name of the claim in the ID token which is used to map the user to a login in ConSol CM. The value depends on the ADFS settings; the default values are “upn” and “unique_name”.
- oidc.web.usernameRegexp.default: Defines the regular expression used for mapping the user name claim values to ConSol CM user names.
    - **“upn” as claim**: `(.*)@.*` will transform the claim value “user1@sso.yourdomain.com” to “user1” and look up “user1” in the ConSol CM database.
    - **“unique_name” as claim**: `.*\\(.*)` will transform the claim value “SSO\user1” to “user1” and look up “user1” in the ConSol CM database.

> ⓘ The logins for the users are saved on the *Users* page of the Web Admin Suite.

Setting up SSO for CM/Track

The SSO configuration on the ConSol CM side is done using system properties from the module `cmas-core-security`.

If there are several instances of CM/Track, for example, one for internal customers and one for external customers, you can provide separate configurations for each instance. The mapping of the CM/Track instance to a configuration is done using the following property:

- domain.map.for.oidc.config.CONFIGURATION_NAME: Enables mapping multiple instances of CM/Track to a specific configuration. Replace “CONFIGURATION_NAME” with the name of the configuration (example: if your CM/Track configuration is called “trackV2customized”, the name of the property is `domain.map.for.oidc.config.trackV2customized`). You can provide several URLs separated by a comma.

The following system properties are used as a default configuration. You can copy them and replace “default” in the property name by the name of the instance to create a separate configuration for the instance. The default values are used whenever no configuration-specific property is set.

Set the following system properties:

- oidc.track.enabled.default: Indicates whether user authentication using SSO via OIDC is enabled.
- oidc.track.authority.default: Indicates the URL of the authenticating authority, e.g. ADFS. Example: https://localhost/adfs
- oidc.track.clientId.default: Indicates the client ID (application ID) of the application, as registered in ADFS or Azure AD.
- oidc.track.clientSecret.default: Indicates the secret of the client, generated using ADFS or Azure AD.
- oidc.track.redirectUri.default: Indicates the redirect URI where authentication responses can be received. This is either the OIDC endpoint on the ConSol CM server running CM/Track or on the load balancer. Example: http://localhost/track/oidc/
- oidc.track.usernameClaim.default: Indicates the name of the claim in the ID token which is used to map the user to a contact in ConSol CM. The value depends on the ADFS settings; the default values are “upn” and “unique_name”.
- oidc.track.usernameRegexp.default: Defines the regular expression used for mapping the user name claim values to ConSol CM user names.
    - **“upn” as claim**: `(.*)@.*` will transform the claim value “user1@sso.yourdomain.com” to “user1” and look up “user1” in the ConSol CM database.
    - **“unique_name” as claim**: `.*\\(.*)` will transform the claim value “SSO\user1” to “user1” and look up “user1” in the ConSol CM database.

ⓘ The logins for the contacts are saved in contact fields with the user name setting in the Web Client.

E - Installing and setting up CMRF and DWH

The ***ConSol CM Reporting Framework*** (CMRF) and ***Data Warehouse*** (DWH) are the modules which are required to provide the basis for Business Intelligence (BI) and reporting in ConSol CM. Please see System architecture for an overview of the required components.

The CMRF is a distinct Java EE application. It can either be deployed in the same application server as used for the CM application or be executed as a standalone application.

Please carry out the following steps to set up the CMRF and DWH:

1. Prepare the DWH database, see section Setting up the database for CMRF / DWH.
2. Install the CMRF, see section Installing the CMRF.
3. Configure the data warehouse, see section Configuring the DWH.

When you have installed and filled the data warehouse, you can use any reporting or business intelligence tool of your choice to retrieve data from this database.

E.1 Setting up the database for CMRF / DWH

Before you can install the CMRF and DWH, you have to set up a database (or database scheme, respectively) for the DWH. This database (scheme), i.e. the DWH, is later filled by the CMRF.

Please proceed to the section which covers the explanations for the RDBMS which is in operation in your company:

- MySQL
- Oracle
- Microsoft SQL Server
- PostgreSQL

E.1.1 MySQL

A distinct database is required.

Follow the same steps as for creating the ConSol CM database (see section MySQL). Use as database name `cmrfdatabase` and user `cmrf`.

No additional setting is necessary.

E.1.2 Oracle

A distinct database schema is required.

Follow the same steps as for creating the ConSol CM database (see section Oracle). Use as database name `cmrfdatabase` and user `cmrf`.

If you want to create views or cubes for reporting, e.g. on the *Cubes* page of the Web Admin Suite, the `cmrf` user needs permissions to create views on the CMRF database.

E.1.2.1 Transactions recovery

The following settings must be applied for the user accessing an Oracle XA data source in order for XA recovery to operate correctly. The value `cmrf` is user-defined to connect from JBoss to Oracle:

```
GRANT SELECT ON sys.dba_pending_transactions TO cmrf;
GRANT SELECT ON sys.pending_trans$ TO cmrf;
GRANT SELECT ON sys.dba_2pc_pending TO cmrf;
GRANT EXECUTE ON sys.dbms_xa TO cmrf; --If using Oracle 10g R2 (patched) or
 later
--OR
GRANT EXECUTE ON sys.dbms_system TO cmrf; --If using an unpatched Oracle
 version prior to 11g
```

E.1.3 Microsoft SQL Server

A distinct database schema is required.

Follow the same steps as for creating the ConSol CM database (see section Microsoft SQL Server). Use as database name `cmrfdatabase` and user `cmrf`.

No additional setting is necessary.

E.1.4 PostgreSQL

A distinct database is required.

Follow the same steps as for creating the ConSol CM database (see section PostgreSQL). Use as database name `cmrfdatabase` and user `cmrf`.

```
CREATE DATABASE cmrf;
CREATE USER cmrf WITH PASSWORD 'password';
GRANT ALL PRIVILEGES ON DATABASE cmrf TO cmuser;
GRANT ALL PRIVILEGES ON DATABASE cmrf TO cmrf;
```

Connect to the database and grant access to the `public` schema to the database user:

```
GRANT ALL ON SCHEMA PUBLIC TO cmuser;
GRANT ALL ON SCHEMA PUBLIC TO cmrf;
```

E.2 Installing the CMRF

The CMRF is a distinct Java EE application. It can either be deployed in the same application server as used for the CM application or be executed as a standalone application. Please refer to section System architecture for explanations on the possible setup scenarios.

E.2.1 Installation modes

Two installation modes are possible:

- Standalone mode: CMRF is executed as a standalone application.
- Overlay mode: ConSol CM and CMRF run in one application server.

E.2.2 Prerequisites

- Installation of Java JDK 11 or 17 on the server
- JBoss EAP version 7.4, or Wildfly 26.1.3
- ConSol CMRF distribution: `cmrf-application-package-app-<CM_VERSION>.jar` (standalone mode) or `cmrf-application-package-war-<CM_VERSION>.war` (overlay mode)
- Installed database server
- In case of MySQL as a database: MySQL connector

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

E.2.3 Executing CMRF as a standalone application

This chapter discusses the following:

- Configuring the database connection
- Starting the CMRF

E.2.3.1 Configuring the database connection

The CMRF is executed as a standalone Java application, without application server.

The database connection needs to be configured on both the ConSol CM and the CMRF side.

ConSol CM

The database connection is configured in the `cm6-config.properties` file in the application server where ConSol CM run:

1. Uncomment the line for the CMRF transactional datasource in the `cm6-config.properties` file located in `<JBOSS_HOME>\bin` and provide the settings for your database:
    - **cmrf.xa.datasource.url**
      URL of the transactional CMRF datasource.

      Example values:
        - **MySQL 5**: jdbc:mysql://localhost/cmdatabase?serverTimezone=Europe/Berlin
        - **MySQL 8**: jdbc:mysql://localhost/cmdatabase?characterEncoding=UTF-8&server-Timezone=Europe/Berlin
        - **Oracle**: jdbc:oracle:thin:@localhost:1521/SERVICE_NAME
        - **Microsoft SQL Server**: jdbc:sqlserver://localhost:1433;databaseName=cmdatabase
        - **PostgreSQL**: jdbc:postgresql://localhost:5432/cmdatabase
    - **cmrf.xa.datasource.username**
      Name of the database user for CMRF
    - **cmrf.xa.datasource.password**
      Password of the database user for CMRF

CMRF

The database connection is configured in the `cmrf.properties` file. This file needs to be saved in the same location from where the `cmrf-application-package-app-<CM_VERSION>.jar` is started.

1. Uncomment the line with the JDBC connection for your database system, starting with `spring.datasource.url`.
2. Enter the URL of your datasource

   Example values:
    - **MySQL**: jdbc:mysql://localhost/cmrf
    - **Oracle**: jdbc:oracle:thin:@localhost:1521/SERVICE_NAME
    - **Microsoft SQL Server**: jdbc:sqlserver://localhost;databaseName=cmrf
    - **PostgreSQL**: jdbc:postgresql://localhost/cmrf
3. Provide the username and password:

- **spring.datasource.username**
  Name of the database user for CMRF
- **spring.datasource.password**
  Password of the database user for CMRF

If needed, you can add settings to override the default logging configuration to this file.

E.2.3.2 Starting the CMRF

You need to start both the ConSol CM server and the CMRF server.

ConSol CM server

Run the following commands to start the JBoss server. Alternatively you can configure JBoss to run as a service (similar to autostart of ConSol CM, see Autostarting the CM server).

Example start command on Windows:

```
<JBOSS_HOME>\bin\standalone.bat -P <JBOSS_HOME>\bin\cm6-config.properties -c
 cm6-dwh-<DB>.xml
```

Example start command on Linux:

```
<JBOSS_HOME>/bin/standalone.sh -P <JBOSS_HOME>/bin/cm6-config.properties -c
 cm6-dwh-<DB>.xml
```

Replace `cm6-dwh-<DB>.xml` with the ConSol CM configuration file for your database:

- `cm6-dwh-mysql-5.xml` for MySQL 5
- `cm6-dwh-mysql-8.xml` for MySQL 8
- `cm6-dwh-oracle.xml` for Oracle
- `cm6-dwh-mssql.xml` for Microsoft SQL Server
- `cm6-dwh-postgresql.xml` for PostgreSQL

CMRF application

Run the following command to start the CMRF:

```
java -jar cmrf-application-package-app-<version>.jar
```

On MySQL, you need to provide the path to the My SQL driver in the start command. In the following example, the connector is saved in the same directory from which the application is started:

```
java -Dloader.path=mysql-connector-java-8.0.22.jar -jar cmrf-application-
 package-app-<version>.jar
```

E.2.4 Installing the CMRF on the same JBoss server

This chapter discusses the following:

- Installing CMRF
- Starting the server

E.2.4.1 Installing CMRF

The ConSol CM system and the CMRF run on the same JBoss application server.

1. Copy the `cmrf-application-package-war-<CM_VERSION>.war` file to `<JBOSS_HOME>/standalone/deployments`.
2. Uncomment the line for the CMRF transactional datasource in the `cm6-config.properties` file located in `<JBOSS_HOME>\bin` and provide the settings for your database:
    - **cmrf.xa.datasource.url**
      URL of the transactional CMRF datasource.

      Example values:

        - **MySQL 5**: jdbc:mysql://localhost/cmdatabase?serverTimezone=Europe/Berlin
        - **MySQL 8**: jdbc:mysql://localhost/cmdatabase?characterEncoding=UTF-8&server-Timezone=Europe/Berlin
        - **Oracle**: jdbc:oracle:thin:@localhost:1521/SERVICE_NAME
        - **Microsoft SQL Server**: jdbc:sqlserver://localhost:1433;databaseName=cmdatabase
        - **PostgreSQL**: jdbc:postgresql://localhost:5432/cmdatabase
    - **cmrf.xa.datasource.username**
      Name of the database user for CMRF
    - **cmrf.xa.datasource.password**
      Password of the database user for CMRF

E.2.4.2 Starting the server

Run the following commands to start the JBoss server. Alternatively you can configure JBoss to run as a service (similar to autostart of ConSol CM, see Autostarting the CM server).

Example start command on Windows:

```
<JBOSS_HOME>\bin\standalone.bat -P <JBOSS_HOME>\bin\cm6-config.properties -c
 cm6-dwh-<DB>.xml
```

Example start command on Linux:

```
<JBOSS_HOME>/bin/standalone.sh -P <JBOSS_HOME>/bin/cm6-config.properties -c
 cm6-dwh-<DB>.xml
```

Replace `cm6-dwh-<DB>.xml` with the ConSol CM configuration file for your database:

- `cm6-dwh-mysql-5.xml` for MySQL 5
- `cm6-dwh-mysql-8.xml` for MySQL 8

- `cm6-dwh-oracle.xml` for Oracle
- `cm6-dwh-mssql.xml` for Microsoft SQL Server
- `cm6-dwh-postgresql.xml` for PostgreSQL

E.3 Configuring the DWH

ⓘ The DWH configuration and initialization are covered in great detail in the *ConSol CM Administrator Manual* in section *DWH administration*. Please refer to this document for further information.

Two types of system properties can be set for ***ConSol CM***:

- ConSol CM system properties which have to be set on the *System properties* page of the Web Admin Suite, or when configuring the DWH on the *Administration* page. The properties belong to the module `cmas-dwh-server`, see List of system properties.
- Java system properties which have to be provided as `-D` parameters at the start of a CM system. These properties are explained in the following section.

E.3.1 Java system properties for the DWH

There exist some options which effect the CMRF functionally, which can only be provided as Java system properties during application server startup. If you want to use these system properties, they should be added to the startup scripts of the application server. In this way, these parameters are used on every restart. The variable `JAVA_OPTS` is used to configure general startup parameters in the scripts for all supported application servers. You can add the CMRF system properties here.

The following command can be used to set the Java properties:

- Windows

  ```
  set JAVA_OPTS="%JAVA_OPTS% -D<option>=<value>"
  ```

- Linux

  ```
  set JAVA_OPTS="$JAVA_OPTS -D<option>=<value>"
  ```

This works for standalone and overlay installations.

E.3.1.1 Truncation of text fields (MySQL only)

On MySQL, the values of data fields of the type “Text” (string) are truncated to avoid exceeding the row size limit of MySQL (relevant when a field group contains several text fields to be transferred). By default, only the first 512 characters of the field values are written to the DWH database.

You can modify the number of characters by adding the property cmrf.mysql.truncate.string to the start command of the CMRF, e.g.:

```
-Dcmrf.mysql.truncate.string=256
```

F - Installing ConSol CM add-ons

This section explains how to set up the optional ConSol CM add-ons:

- CM/Track
- CM/EBIA
- CM/Doc
- CM/Archive
- CM/Phone

F.1 CM/Track

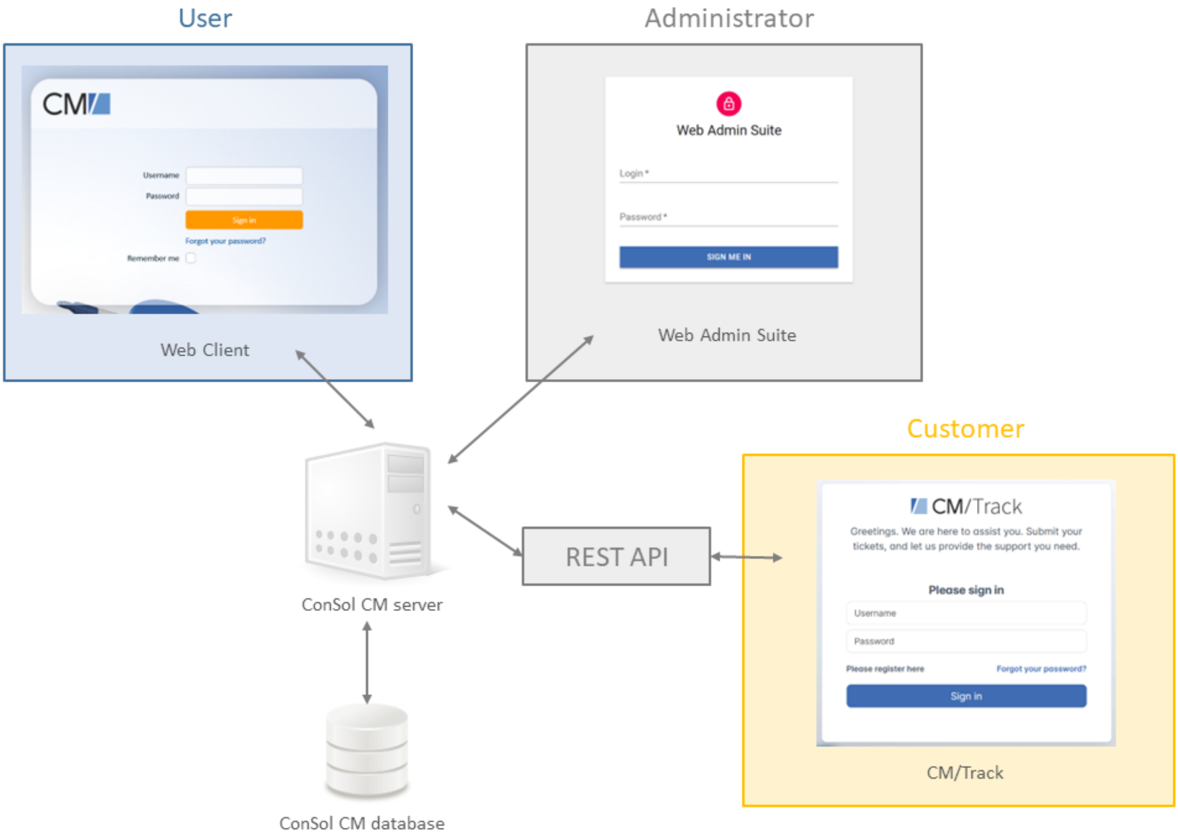
This chapter discusses the following:

F.1.1 Introduction to CM/Track

ConSol CM features a customer portal called CM/Track, an add-on which has to be purchased separately. It communicates with the ConSol CM server using a REST API.

CM/Track is a web application used by customers to track their cases. Customers who want to use CM/Track must be registered as contacts in the Web Client and granted the required access permissions.

The following figure provides and overview of the system architecture.



F.1.2 Installing CM/Track

CM/Track requires Java 11 or 17. It is a `.war` file which can be deployed on the application server where ConSol CM is running or in a separate Tomcat servlet container . The latter allows to run CM/Track as a separate portal which can be operated in another security context than CM.

If you deploy CM/Track in a separate Tomcat servlet container but on the same machine as JBoss, you need to avoid port conflicts, as both JBoss and Tomcat use the 8080 port as default.

Perform the following steps:

1. Deploy the `.war` file:
    - If the standard version of CM/Track is used, you have to deploy the standard `.war` file, usually named `cmtrack-v2-distribution-<CM_VERSION>.war` (example: `cmtrack-v2-distribution-6.16.0.0.war`).
    - In case a customer-specific `.war` file is used, you have to deploy this specific `.war` file.

    > ⚠ If you deploy CM/Track on Tomcat, you need to rename the `.war` file to `track.war`

2. By default, CM/Track will look for the CM server / REST API on `http://localhost:8888`. In order to change this default setting, you need to perform one of the following steps:
    - Set the respective system variable for the REST API in the file `standalone.conf` (Linux) or `standalone.conf.bat` (Windows)

      Linux:

      ```
      export CM_REST_URL=<PROTOCOL>://<CM_SERVER>:<CM_PORT>/restapi
      ```

      Windows:

      ```
      set CM_REST_URL=<PROTOCOL>://<CM_SERVER>:<CM_PORT>/restapi
      ```

    - Add the following Java option to the start script:

      ```
      -Djersey.cm6.url=<PROTOCOL>://<CM_SERVER>:<CM_PORT>/restapi
      ```

3. Restart the CM server during deployment.
4. Test the deployment by calling the following URL:

    ```
    <PROTOCOL>://<CM_SERVER>:<CM_PORT>/track/
    ```

The login screen should be displayed.

If you want to embed CM/Track in an iFrame located in another domain, you must allow embedding in the Java options of the application server where CM/Track runs. There are two possibilities:

- Allow embedding CM/Track as an iFrame in all domains.

  ```
  -Dhttp.headers.iframe=ALL
  ```

- Allow embedding CM/Track as an iFrame in a specific domain with a given host name and port.

  ```
  -Dhttp.headers.iframe=http://HOST:PORT
  ```

F.1.3 Configuring CM/Track

The configuration of CM/Track, including database authentication, is described in the *ConSol CM Administrator Manual*, section *CM/Track*. See Configuring LDAP authentication for LDAP authentication and Configuring SSO authentication for SSO authentication.

F.2 CM/EBIA

CM/EBIA is a ConSol CM add-on which allows to create reports directly in the Web Client.
CM/EBIA uses a licensed enterprise version the third-party application Metabase, which is integrated via SSO. The created reports are shared with the regular users as dashboard widgets.

The following figure illustrates how CM/EBIA integrates with ConSol CM:

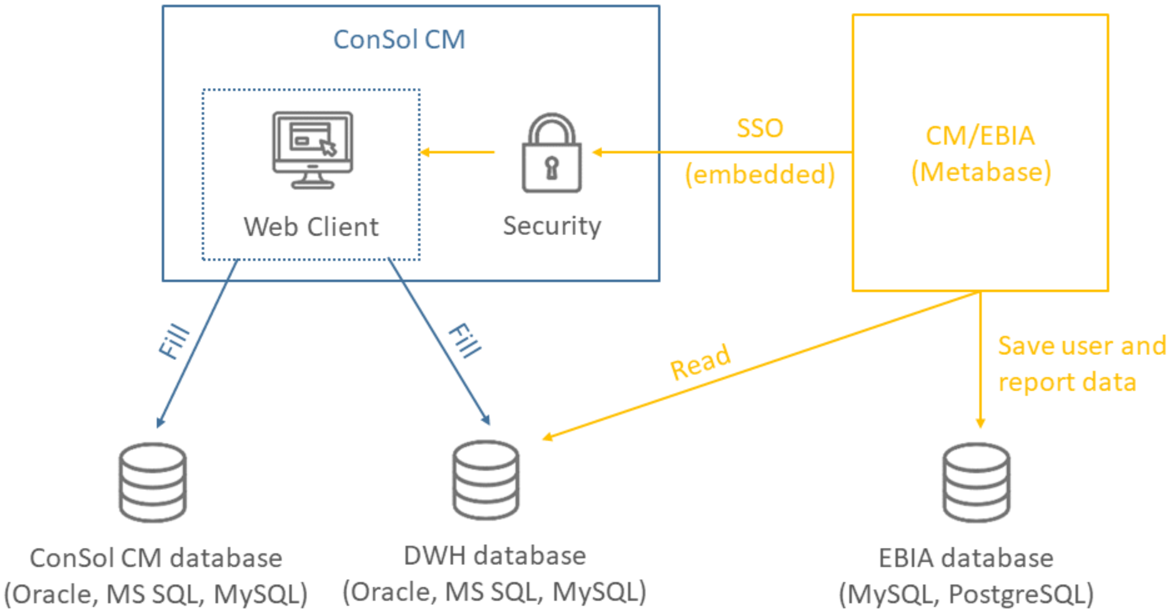


Figure 2: *Architecture of ConSol CM with CM/EBIA*

F.2.1 Requirements for CM/EBIA

The following system requirements apply for CM/EBIA:

- ConSol CM version 6.13.0.0 and up
- Java 11 or 17

ⓘ Please also see the official system requirements on the TecDoc server.

F.2.2 Installing and setting up CM/EBIA

⚠ CM/EBIA needs to be licensed separately.

The first step to use CM/EBIA is to install and set up the Metabase application:

1. Download the Metabase JAR file from https://-downloads.metabase.com/enterprise/latest/metabase.jar.
2. Start the Metabase application with the following start command:

   ```
   java -jar metabase.jar
   ```

   You can set the following variables:

   - **-DMB_SESSION_COOKIE_SAMESITE**: “None”, “Lax” or “Strict”. If Metabase is installed in a different domain than ConSol CM, you must use “None”.
   - **-DMB_JETTY_PORT**: The port of the Metabase application. The default port is 3000.
3. Open the Metabase application. It runs on port 3000 by default. Create the administrator account. You can add the database connection in this step or later.
4. Go to *Settings* -> *License and Billing*, enter the license key provided by ConSol and click *Activate*.
5. Go to *Settings* -> *General*. The value of the ConSol CM system property cmas-analytics, metabase.url needs to match the value in *Site URL*, e.g. “https://localhost:8080/analytics”.
6. Go to *Authentication* -> *JWT*. Set *JWT Authentication* to *Enabled*. Enter the URL of the ConSol CM system in *JWT Identity Provider URI*, e.g. “https://localhost:8080/analytics?auth=metabase”. Paste the value in *String used by the JWT signing key* to the ConSol CM system property cmas-analytics, metabase.secret.key.
7. Go to *Embedding*. Enter the URL of the ConSol CM system in the field *Full-app embedding*.
8. Enter the URL of the Metabase application in the ConSol CM system property cmweb-server-adapter, csrf.domain.white.list.

The next step is to add the databases. These are the databases whose data is used for creating analysis in CM/EBIA. Usually, this is the DWH database with some preconfigured cubes.

⚠ It is also possible to connect the ConSol CM production database to CM/EBIA. This is not recommended, so you must check with the ConSol CM support first.

Please proceed as follows to add a database:

1. Open the CM/EBIA administration.
2. Go to *Databases* and click *Add database*.
3. Enter the information of your database and click *Save*.

⚠ CM/EBIA uses the enterprise version of the third-party product Metabase. Therefore, the ConSol CM system must be allowed to access the internet to validate the token which gives access to the advanced Metabase features. See https://www.metabase.com/docs/latest/enterprise-guide/activating-the-enterprise-edition.html for further details.

F.2.3 Fine-tuning CM/EBIA

The following settings, which can be made in the CM/EBIA administration, have been proven useful.

Section *General*:

- In *Friendly Table and Field Names*, use the setting *Only replace underscores and dashes with spaces* to stop CM/EBIA from trying to interpret and beautify column names.

Section *Localization*:

- Do not change the settings *Report Timezone*. Use the database default instead.
- For German environments: change the date, time, decimal number format and currency.

Section *Data Model*:

- Edit the display names of your cubes and most used tables and columns to provide more user-friendly labels. This affects both the labels used for the tables and table columns when creating questions and the labels displayed to the end users.

F.3 CM/Doc

This chapter discusses the following:

F.3.1 Requirements for using CM/Doc

On the client PC or laptop, the following requirements have to be met to use CM/Doc:

- Microsoft Windows operating system
- Locally installed CM/Doc application
- Locally installed Microsoft Word / OpenOffice

> ⓘ Detailed information about the installed Microsoft Word and OpenOffice versions can be found in the current *System Requirements*.

F.3.2 Configuring CM/Doc

CM/Doc is enabled using the system property cmweb-server-adapter, cmoffice.enabled. Set this system property to “true” in the Web Admin Suite. In addition, you need to set the port to which the CM/Doc application should connect in the system property cmweb-server-adapter, cmoffice.websocket.port in the Web Admin Suite. If HTTPS is used, the engineers need to trust the URL configured for CM/Doc in their browsers.

If CM/Doc is used with OpenOffice, the engineers have to set the path to the OpenOffice installation within the CM/Doc application.

> ⓘ Please refer to the *ConSol CM Administrator Manual* for a detailed explanation about how to configure CM/Doc for your system.

F.4 CM/Archive

CM/Archive is a ConSol CM add-on which allows to archive tickets from ConSol CM. It is a separate Java web application with a MongoDB database.

The following figure illustrates how CM/Archive integrates with ConSol CM.

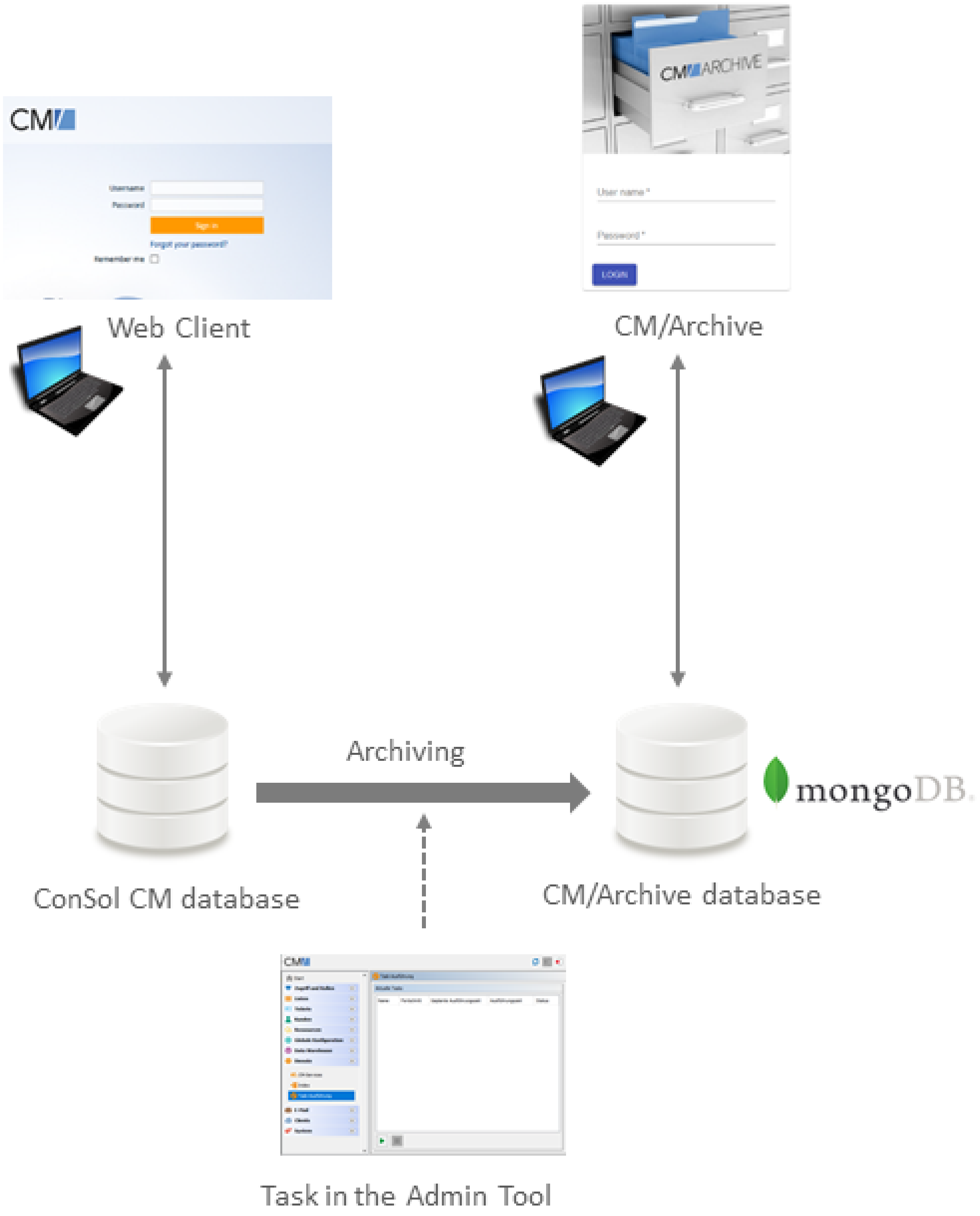


Figure 3: *Architecture of ConSol CM with CM/Archive*

F.4.1 Requirements for CM/Archive

The following system requirements apply for CM/Archive:

- ConSol CM version 6.11.2.1 and up
- MongoDB version 3.6
- Java 11 or 17

ⓘ Please also see the official system requirements on the TecDoc server.

F.4.2 Installing CM/Archive

The following steps are required to install CM/Archive:

1. Installing and setting up the MongoDB database
2. Setting the system properties for CM/Archive in the Web Admin Suite
3. Setting the properties for the CM/Archive application
4. Starting CM/Archive

Preliminary note

The CM/Archive application is delivered with three modes: production mode, demo mode and developer mode. This manual covers the installation in production mode, as the other modes should be used only for testing purposes.

F.4.2.1 Installing and setting up the MongoDB database

The first step is to install the MongoDB. You can find a detailed description in the respective MongoDB documentation for your operating system:

- Linux: Install MongoDB on Linux
- Windows: Install MongoDB on Windows

The next step is to configure the CM/Archive user with the corresponding authentication. Please proceed as follows:

1. **Start the MongoDB instance without authentication**

   Use the following command to start the MongoDB instance without authentication:

   ```
   mongod --port 27017 --dbpath /data/archive
   ```

   The *mongod* options have the following meaning:

   - --port: Indicates the database connection port. The default value is “27017”.
   - --dpath: Defines the directory where the MongoDB instance stores its data. The default value is “/data/db” on Linux and macOS, and “\data\db” on Windows.
   - --bind_ip: Use this option to provide the IP address if you want to change the default value (“localhost, 127.0.0.1”).

   Please see the mongod documentation for further information about the options.

2. **Connect to the MongoDB instance**

   Use the following command to connect to the MongoDB instance:

   ```
   mongo --host 127.0.0.1:27017
   ```

   The following output should be displayed:

```
MongoDB shell version v3.6.4
connecting to: mongodb://127.0.0.1:27017
MongoDB server version: 3.6.4
>
```

3. **Create the administrator user**

   The administrator user has to be created in the *admin* database with the role *userAdminAnyDatabase*. The administrator user has only permissions to create and manage users and roles. It cannot be used for any other operations, e.g., reading data. The following command can be used to create the administrator user:

   ```
   use admin
   db.createUser(
     {
       user: "admin",
       pwd: "consol",
       roles: [ { role: "userAdminAnyDatabase", db: "admin" } ]
     }
   )
   ```

4. **Restart the MongoDB instance with access control**

   Disconnect the MongoDB shell and restart it with the `--auth` option or, if you use a configuration file, the `security.authorization` setting. The following command can be used to restart the MongoDB instance:

   ```
   mongod --auth --port 27017 --dbpath /data/archive
   ```

   Clients that connect to this instance must now authenticate themselves as a MongoDB user. They can only perform actions as determined by their assigned roles.

5. **Connect with the administrator user**

   Use the following command to connect to the MongoDB instance with the above created administrator user:

   ```
   mongo --host 127.0.0.1:27017
   use admin
   db.auth("admin", "consol")
   ```

6. **Create the CM/Archive user**

   Create the CM/Archive user `archive` with the database `archivedb` using the `db.createUser()` command. The user needs to have the `readWrite` role, as shown in the following example:

```
use admin
db.createUser(
   {
     user: "archive",
     pwd: "consol",
     roles: [ { role: "readWrite", db: "archivedb" } ]
   }
)
```

7. **Connect with the CM/Archive user**

   Use the following command to connect to the MongoDB instance with the above created CM/Archive user:

```
mongo --host 127.0.0.1:27017
use admin
db.auth("archive", "consol")
use archivedb
```

F.4.2.2 Setting the system properties for CM/Archive in the Web Admin Suite

The system properties for CM/Archive are managed in two modules in the Web Admin Suite:

cmas-archive-core-server:

- `archive.uri`
  URL from which the CM/Archive application can be accessed.

cmas-auth-server:

- `access.token.signing.key`
  Secret shared between the authorization server and client application using OAuth2, needs to match *archive.oauth2.access.token.signing.key* in the configuration file.
- `client.archive.access.token.validity.seconds`
- `client.archive.refresh.token.validity.seconds`
- `client.archive.secret`
  Secret shared between the authorization server and CM/Archive, needs to match *archive.oauth2.client.secret* in the configuration file.

F.4.2.3 Setting the properties for the CM/Archive application

Some properties for the CM/Archive application need to be provided in a properties file which has to be saved in the same directory as the `war` file of CM/Archive.

The name of the properties file is `archive-prod.properties`. It has to contain the following settings:

```
archive.cm6.endpoint=http://127.0.0.1:8888
archive.oauth2.access.token.signing.key=94623427-5a74-11e8-a6eb-6127838b1c93
archive.oauth2.client.secret=94623428-5a74-11e8-a6eb-6127838b1c93
archive.mongodb.database.name=archivedb
archive.mongodb.uri=mongodb://archive:consol@127.0.0.1:27017/?&journal=true&w=m
 ajority&maxPoolSize=1000
```

The following list states the meaning of the properties:

- **archive.cm6.endpoint**
  This is the URL and port where the ConSol CM instance is executed.
- **archive.oauth2.access.token.signing.key**
  Secret shared between the authorization server and client application using OAuth2, needs to match the ConSol CM system property `cmas-auth-server`, `access.token.signing.key`
- **archive.oauth2.client.secret**
  Secret shared between the authorization server and CM/Archive, needs to match the ConSol CM system property `cmas-auth-server`, `client.archive.secret`
- **archive.mongodb.database.name**
  This is the name of the MongoDB database as created in step 6 above.
- **archive.mongodb.uri**
  User name and password of the CM/Archive user, and URL of the MongoDB instance

F.4.2.4 Starting CM/Archive

Start CM/Archive using the following command:

```
java -jar archive-6.16.0.0.war --spring.profiles.active=prod
```

By default, CM/Archive starts on port 8080. You can choose a different port by adding, e.g., `--server.port=8090` to the command.

You can also create a start script for CM/Archive. The following example shows a start script for Linux operating systems.

```
#!/bin/bash

ARCHIVE_WAR=${ARCHIVE_WAR:-'archive-6.11.2.9.war'}
ARCHIVE_PROPERTIES=${ARCHIVE_PROPERTIES:-'--spring.profiles.active=prod'}
ARCHIVE_PIDFILE=${ARCHIVE_PIDFILE:-'archive.pid'}
ARCHIVE_JAVA=${ARCHIVE_JAVA:-'java'}

cd "$(pwd)/$(dirname "$0")" || exit 1

start()
{
   if [ -f "$ARCHIVE_WAR" ]; then
      if [ -f "$ARCHIVE_PIDFILE" ]; then
         local pid
         pid=$(cat "$ARCHIVE_PIDFILE")
         if [ "$(ps -o pid= -p "$pid")" = "$pid" ]; then
            printf 'Archive is already running\n'
            return 0
         fi
      fi
      printf 'Starting archive...'
      "$ARCHIVE_JAVA" -jar "$ARCHIVE_WAR" $ARCHIVE_PROPERTIES > /dev/null 2>&1 &
      printf '%s' "$!" > "$ARCHIVE_PIDFILE" || return 1
      sleep 1
      if [ -f "$ARCHIVE_PIDFILE" ]; then
```

```
         local pid
         pid=$(cat "$ARCHIVE_PIDFILE")
         if [ "$(ps -o pid= -p "$pid")" = "$pid" ]; then
            printf 'done\n'
            return 0
         fi
      fi
      printf 'failed\n';
      rm -f "$ARCHIVE_PIDFILE"
      else
         printf 'Archive war file %s has not been found in %s\n' "$ARCHIVE_WAR"
          "$(pwd)"
   fi
   return 1
}

stop()
{
   if [ -f "$ARCHIVE_PIDFILE" ]; then
      local pid
      pid=$(cat "$ARCHIVE_PIDFILE")
      if [ "$(ps -o pid= -p "$pid")" = "$pid" ]; then
         printf 'Stopping archive...'
         kill "$pid" || return 1
         for i in {1..60} ; do
            if [ "$(ps -o pid= -p "$pid")" = "$pid" ]; then
               printf '.'
               sleep 1
            else
               break
            fi
         done
         if [ "$(ps -o pid= -p "$pid")" = "$pid" ]; then
            printf 'killing...'
            kill -9 "$pid" || return 1
         fi
         printf 'done\n'
         rm -f "$ARCHIVE_PIDFILE" || return 1
         return 0
      fi
   fi
   printf 'Archive is stopped\n'
}

status()
{
   if [ -f "$ARCHIVE_PIDFILE" ]; then
      local pid
      pid=$(cat "$ARCHIVE_PIDFILE")
      if [ "$(ps -o pid= -p "$pid")" = "$pid" ]; then
         printf 'Archive is running\n'
         return 0
      fi
   fi
   printf 'Archive is stopped\n'
   return 1
}

case "$1" in
   'start')
      start
   ;;
   'stop')
      stop
   ;;
   'status')
      status
   ;;
   'restart')
      stop && start
   ;;
   *)
      printf 'Usage: %s start|stop|status|restart\n' "$0"
   ;;
esac
```

Code example 2: *Start script archive.sh for CM/Archive on Linux*

Please note that execute permissions are needed for the script file:

```
chmod +x archive.sh
```

You can start, stop and restart CM/Archive with the following commands:

```
./archive.sh start|stop|status|restart
```

F.5 CM/Phone

This chapter discusses the following:

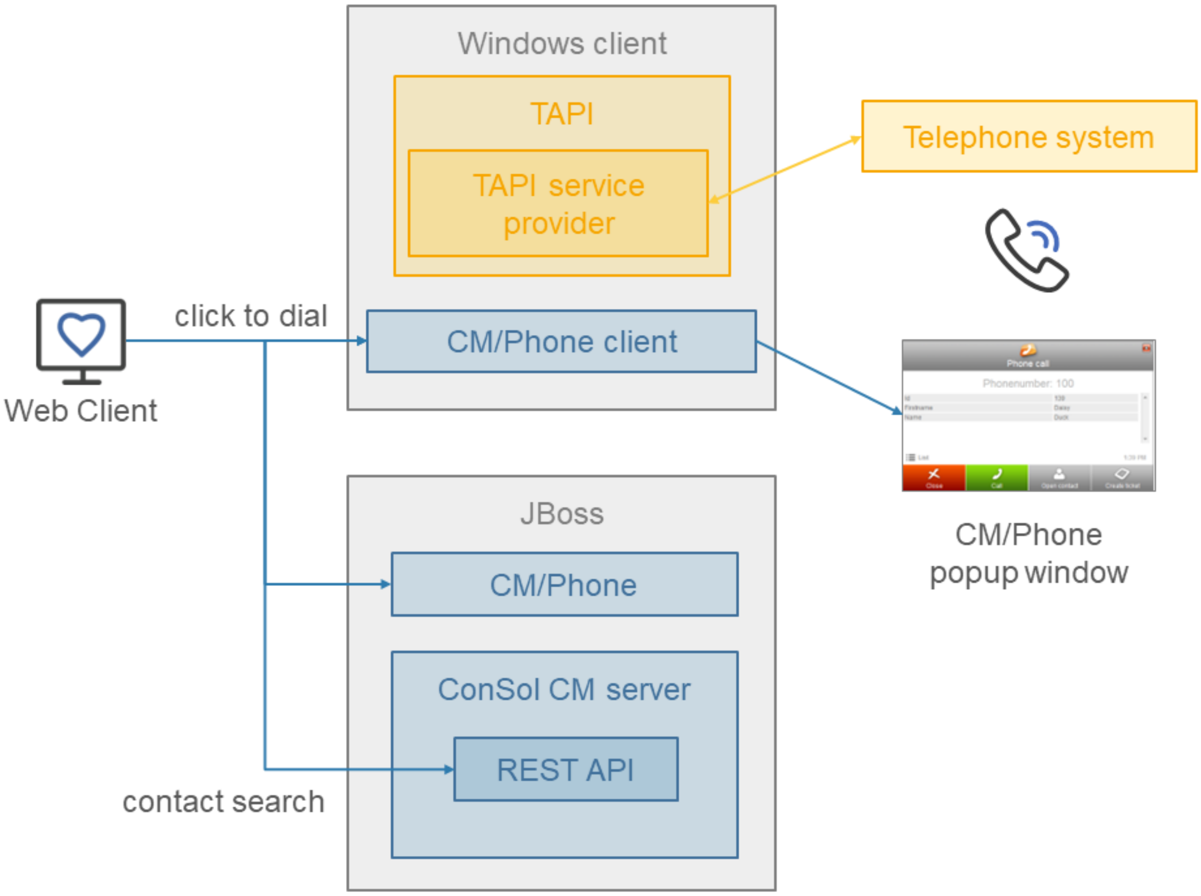
F.5.1 Introduction to CM/Phone

CM/Phone is a ConSol CM add-on which allows to integrate a telephone system using the TAPI 3 protocol, which is available on Windows operating systems. If CM/Phone is active, the users can initiate phone calls directly from the Web Client by clicking a phone number. When a customer calls, the users see the call on their screen and can choose to open the contact page or create a new case in the Web Client.

CM/Phone has to be licensed separately. Each client where CM/Phone is installed and active consumes one license of the type REST, see License.

F.5.2 Technical background of CM/Phone

The following figure illustrates how CM/Phone integrates with the telephone system and ConSol CM.



The information shown in the pop-up window opened by CM/Phone is based on HTML template files which are located in the CM/Phone folder of the ConSol CM server. The templates are loaded by the CM/Phone client application during startup. You can customize them if needed, see *ConSol CM Setup Manual*.

F.5.2.1 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 is registered, a pop-up window with the phone number of the caller is shown and a contact with this phone number is searched in ConSol CM. If one matching contact is found, its name is shown. If there are several matching contacts, the user can select the desired one from a list.

> ⓘ The users need view permissions to the customer group of the contact to see its information.

The following features are available if a matching contact has been found:

- Open the contact page
- Create a case for the contact

The following features are available if no matching contact has been found:

- Create a contact
- Create a case for the newly created contact

> ⓘ When a new contact is created, the caller's phone number is prefilled in the first contact field with the *Phone number* setting.

For missed calls, there is an option to call back.

F.5.2.2 Outgoing Calls

The users can start an outgoing call directly by clicking a phone number in a contact field with the setting *Phone number*. 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.

Usually the prefix `phone:` is set before the number for outgoing calls for interaction with TAPI. If required, you can configure another prefix (e.g. `tel:`) in the Windows Registry.

F.5.3 Installing CM/Phone on the server

Perform the following steps to install CM/Phone on the application server:

1. Copy the `cm-phone.war` folder to `<JBOSS_HOME>/standalone/deployments`

   ⓘ The application server must be configured for working with exploded .war files.

   Overwrite the parameter `${deployment.auto.deploy.scanner:false}` from the XML configuration file by adding the following line to the `cm6-config.properties` or `cm6-cmrf-config.properties` file:

   ```
   deployment.auto.deploy.scanner=true
   ```

2. Adapt the configuration as needed (see Editing the CM/Phone templates).
3. Restart the application server.

If CM/Phone was deployed correctly, you can see the welcome page of CM/Phone when accessing `https://<CM_server>:<port>/cm-phone/`

F.5.4 Installing CM/Phone on the client computers

The users who want to use the CTI features provided by CM/Phone need to install the CM/Phone application on their computer. The installation file can be downloaded from the user profile in the Web Client.

> ⚠ Administrator permissions are required for the installation because an entry for CM/Phone needs to be written to the Windows registry. Before installing a new CM/Phone version, the users need to manually uninstall the previous version.

When opening CM/Phone for the first time, the users need to select the TAPI line to be used, enter the URL of the ConSol CM system and provide their user name and password. These settings can be changed later in the *Settings* option of the CM/Phone icon in the task bar.

F.5.5 Configuring CM/Phone

You need to perform several steps in the Web Admin Suite to configure CM/Phone:

1. Configure the contact fields for the phone numbers on the *Contact fields* page.
2. Define the display name of the contacts in CM/Phone on the *Contact models* page.
3. Configure the phone number format for each customer group on the *Customer groups* page.
4. Configure the prefixes for the phone numbers on the *System properties* page.

F.5.5.1 Editing the CM/Phone templates

The layout and content of the contact details which are shown in the popup windows opened by CM/Phone are defined in HTML templates. They are saved in `<JBOSS_HOME>/standalone/deployments/cm-phone.war/templates`. The templates contain tags which are replaced with current values by the client application during a call.

> ⚠ These tags are case sensitive.

The following templates exist by default:

- CallNotification.html
- ContactData.html
- ContactList.html

They are used for the default language of the client system. To support additional languages, you need to create a folder with the name of the locale (e.g., `de` for German), which contains localized copies of the template files.

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 because at this point in the process there is no contact data available.

Available tags:

- **[phonenumber]**: Phone number of the caller.
- **[calltime]**: Time of the call.
- **[content]**: Additional information to be displayed, e.g. a loading indicator during the contact search or an error message if the search failed (wrong user name or password, etc.)

ContactData.html

This template shows the detail of the calling contact if a matching contact is found for the phone number. You can add further tags using the contact fields defined in the respective contact model.

Available tags:

- **[phonenumber]**: Phone number of the caller.
- **[calltime]**: Time of the call.
- **[contactContent]**: Display name of the contact as defined in the customer group settings in the template *CM/Phone contact details*.

- **[contact.id]**: Internal ID of the contact in ConSol CM, can be used to create links into the ConSol CM Web Client.

ContactList.html

If more than one contact is found for the phone number, a contact list is shown. It contains one row for each matching contact.

Available tags:

- **[phonenumber]**: Phone number of the caller.
- **[calltime]**: Time of the call.
- **[contactlistContent]**: List of matching contacts as defined in the customer group settings in the template *CM/Phone contact list*.

Links and buttons in 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 page in the Web Client.
- **Create ticket**: Opens the page to create a case in the Web Client.

The buttons refer to HTML links and can be customized as needed. Two types of links are available:

- **External link**: Opens the respective page in a web browser, e.g. the contact page of the Web Client.

  ```
  https://<CM URL>/cm-client/contact/[contact.id]
  ```

  You can use contact data in the URL by passing it as parameter to create links to customer-specific web applications.

- **Internal link**: Opens another page of the CM/Phone pop-up window.

  ```
  https://cmphone/<command>/?<parameter>
  ```

  Available commands:

  - **contactdata**: Used in the `ContactList.html` template. The user can select a contact to show its details (`ContactData.html` template). The contact ID is used as a parameter.

    ```
    http://cmphone/contactdata/?[contact.id]
    ```

  - **contactlist**: Used to go back from the contact details (`ContactData.html` template) to the list of contacts (`ContactList.html` template).

    ```
    http://cmphone/contactlist
    ```

  - **call**: Starts an outgoing phone call to the phone number of the contact.

```
http://cmphone/call
```

- **runcmd**: Starts a local application on the client PC. The command line of the application is used as a parameter. In addition, contact data can be passed to the application, e.g.:

```
http://cmphone/runcmd/?dbapp.exe+userid=[customer.userid]
```

- **close**: Closes the CM/Phone pop-up window.

```
http://cmphone/close
```

F.5.6 Advanced settings

F.5.6.1 Adding debug logging on the client side

You can activate debug logging on the clients using `log4net` by configuring the `CMPhone.exe.config` file in the installation path, usually `<Program Files>\ConSol\CM Phone`. If the user does not have write access to the Program Files directory, you need to set the log file path to another folder, e.g.:

```
<file value="${USERPROFILE}\phone.log" />
```

F.5.6.2 Registering CM/Phone

If the client application cannot be installed by the users themselves because of insufficient permissions, it can be distributed by a system administrator. In that case, the application needs to be registered on the client as `phone: protocol handler` by creating the appropriate Windows 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)
```

F.5.6.3 Using the REST API for testing

CM/Phone uses the REST API to request data from the ConSol CM server. You can use a REST client in the browser or tools like curl to check if the data needed by CM/Phone is present in the system, e.g. if a contact with a certain phone number exists:

```
https://<CM_SERVER_URL>:<CM_PORT>/restapi/units/?phoneNumber=[phonenumber]
```

G - Operating ConSol CM

This section provides information about common tasks concerning the operation of ConSol CM.

- File system structure
- Starting and stopping the CM server
- Log files
- License
- Index
- Email
- System monitoring

G.1 File system structure

This chapter describes the file system structure of the application server and the ConSol CM application.

G.1.1 ConSol CM data directory

Most of the data concerning the configuration and operation of ConSol CM is stored in the ConSol CM database. However, some data is saved in the file system in the data directory entered during system setup.

Important directories:

- *index*

  This is the directory where all the indexes are stored (see also section Index). Be sure to include it into your regular file system backup.

  > ⓘ Please note that you cannot just include the index directory in your daily CM backup! Rather you have to use the indexer-specific backup (and restore). Please refer to section Backup and recovery of the index.

  - **index.0**
    In this directory, there is a subdirectory for each required index.

> ⓘ In case contacts (units) could not be imported during a scene import, there might be `.tmp` files in the data directory.

G.1.2 JBoss application server

The following application server directories are relevant for ConSol CM:

- **bin**
  Settings for all environments:
  - Configuration of the database connection and logging in the file `cm6-config.properties` or `cm6-domain-config.properties`
  - Configuration of environment variables in the file `standalone.conf` or `standalone.conf.bat`
- **modules\system\layers\base**
  Subfolders contain the JDBC drivers:
  - com\microsoft\sqlserver\jdbc\main\mssql-jdbc-10.2.3.jre8.jar
    (Microsoft SQL)
  - oracle\jdbc\main\ojdbc11-21.5.0.0.jar and oracle\jdbc\main\orai18n-21.9.0.0.jar
    (Oracle)
  - com\mysql\jdbc\main\
    (MySQL JDBC driver destination, must be installed manually)
  - org\postgresql\main\postgresql-42.5.1.jar
    (PostgreSQL)
- **standalone**
  Configuration in standalone environments:
  - **configuration**
    Configuration of the database connection and logging in the files `cm6-<DB>.xml` or `cm6-dwh-<DB>.xml`
  - **data**
    Data for operation, e.g., `tx-operation` keys
  - **deployments**
    Deployed applications, for example `cm6.ear`, `cm-track.war` or `cmrf.war`
  - **log**
    Log files, see section Log files.
  - **tmp**
    Temporary data and also working copy of the application server files. When JBoss is stopped, it can be emptied, e.g. for error analysis and/or fixing.
- **domain**
  Configuration in domain (i.e. multi-server) environments
  - **configuration**
    Configuration of the database connection and logging in the files `cm6-<DB>.xml`, and `master-host.xml` and `slave-host.xml`
  - **servers/<server-name>**/log
    Log files

G.2 Starting and stopping the CM server

There are two levels to consider for starting and stopping ConSol CM:

- The basic start and stop commands of ConSol CM are adapted start and stop commands of the respective application server, which can be executed on the command line.
- The integration of these commands into an environment which allows automatic system start and stop. You might want to write an *init.d* or *systemd* script for a Linux system, or configure CM as a service on a Windows system.

The following section provides information about the simple command line parameters which can be used to start and stop the system.

> If the database system has to be restarted, stop the ConSol CM system first, wait until the database system is available again, and then start the ConSol CM application server.

G.2.1 JBoss on Windows

G.2.1.1 Starting JBoss on Windows

Run the following command to start the JBoss server. Alternatively you can configure JBoss to run as a service (see Installing JBoss EAP as a Windows service).

```
<JBOSS_HOME>\bin\standalone.bat -P <JBOSS_HOME>\bin\cm6-config.properties -c
 cm6-<DB>.xml
```

Replace `cm6-<DB>.xml` with the configuration file for your database:

- `cm6-mysql-5.xml` for MySQL 5
- `cm6-mysql-8.xml` for MySQL 8
- `cm6-oracle.xml` for Oracle
- `cm6-mssql.xml` for Microsoft SQL Server
- `cm6-postgresql.xml` for PostgreSQL

G.2.1.2 Shutting down JBoss on Windows

Run the following command to stop the JBoss server. Alternatively you can put it into the configuration of the Windows service.

```
<JBOSS_HOME>\bin\jboss-cli.bat --controller=localhost:9999 --connect --
 command=:shutdown
```

If you did not set a system environment variable for the Java home directory, add the path to the JDK to `<JBOSS_HOME>\bin\jboss-cli.bat`. This prevents Java warnings.

Example for JDK version 11.0.17, replace by your Java version.

```
set "JAVA_HOME=C:\Program Files\Java\jdk-11.0.17+8"
```

In case the JBOSS_HOME environment variable is already set in the system, you can specify the server-specific path at the beginning of JBOSS_HOME>\bin\jboss-cli.batin the following syntax:

```
set JBOSS_HOME='<JBOSS_HOME>'
```

G.2.2 JBoss on Linux

G.2.2.1 Starting JBoss on Linux

Run the following command to start the JBoss server for a setup with ConSol CM without CMRF. Alternatively you can put it into a start script (usually *init.d* or *systemd*).

```
<JBOSS_HOME>/bin/standalone.sh -P <JBOSS_HOME>/bin/cm6-config.properties -c
 cm6-<DB>.xml
```

Replace `cm6-<DB>.xml` with the configuration file for your database:

- `cm6-mysql-5.xml` for MySQL 5
- `cm6-mysql-8.xml` for MySQL 8
- `cm6-oracle.xml` for Oracle
- `cm6-mssql.xml` for Microsoft SQL Server
- `cm6-postgresql.xml` for PostgreSQL

If JBoss starts normally (*ConSol CM Startup* appears in the `server.log` file) but *localhost:8080* cannot be reached, try setting `<CM_HOST_IP>` to `0.0.0.0`.

G.2.2.2 Shutting down JBoss on Linux

Run the following command to stop the JBoss server. Alternatively you can put it into a stop script (usually *init.d* or *systemd*).

```
<JBOSS_HOME>/bin/jboss-cli.sh --controller=localhost:9999 --connect --
 command=:shutdown
```

If you did not set a system environment variable for Java, add the path to the JDK to `<JBOSS_HOME>/bin/jboss-cli.sh`. This prevents Java warnings.

Example for JDK version 11.0.17, replace by your Java version.

```
JAVA_HOME="/opt/java/64/jdk-11.0.17+8"
```

In case the `JBOSS_HOME` environment variable is already set in the system, you can specify the server-specific path at the beginning of `<JBOSS_HOME>/bin/jboss-cli.sh` in the following syntax:

```
JBOSS_HOME='<JBOSS_HOME>'
```

G.2.3 General settings

- In case `-b=localhost` or `-b=127.0.0.1` is set as parameter, ConSol CM is only accessible from the same server, where JBoss is running.
- Enter the network IP or the network name to make ConSol CM accessible from outside the sever. In this case ConSol CM URL is not accessible from the inside using the localhost URL.
- If you enter `-b=0.0.0.0` the server is accessible from the outside and inside using the server URL or using the localhost URL (take care of all required security-related precautions!).
- After the server has started, open a browser window and type the server URL to start the ConSol CM setup, see section Setting up the ConSol CM application.

G.3 Log files

G.3.1 Introduction to log files in ConSol CM

Log files are the main source of information about the activities in the system and potential problems. They are used to save log messages from the different components of ConSol CM.

G.3.1.1 Concepts, terms and definitions

Concept	Other terms	Definition
log file		Text file with the extension `.log` used to write messages about the activities in the system and errors
log level		The log level determines which messages are written in the log files and how many details are included in the log messages. The most used log levels are ERROR, INFO and DEBUG.

G.3.1.2 Purpose and usage

Log files contain information about the system activity. If an error occurs during system operation, they contain details about the error. Analyzing log files is useful for troubleshooting of errors and to reveal problems which do not appear on the user interface.

There are two ways of accessing the log files:

- Open the file on the file system. The default path is `<JBOSS_HOME>/standalone/log`.
- Open the *Log files* page of the Web Admin Suite, where you can view and download the log files.

G.3.2 Available log files

The following log files are available in ConSol CM:

- **access.log**
  Contains log messages about the access to and usage of the Web Client. Needs to be enabled explicitly, see Activating the access.log file.
- **audit.log**
  Contains log messages about login / logout operations in the Web Admin Suite, and all operations which are performed using these applications, e.g. configuration changes and workflow deployments. In addition, all changes to the page customization are logged.
- **cmrf.log**
  Contains log messages from the CMRF (ConSol CM Reporting Framework), i.e., messages regarding the data transfer operations from the ConSol CM database to the CMRF database (DWH).
- **cmweb.log**
  Contains log messages from the Web Client.
- **ctx.log**
  Contains log messages from the Spring Framework.
- **errors.log**
  Contains only log messages that have at least the log level `ERROR`.
- **index.log**
  Contains log messages regarding the index.
- **mail.log**
  Contains log messages regarding email system.
- **operationtimes.log**
  Contains log messages with timing information for requests in order to identify possible performance bottlenecks. This logging is configured using the system property cmweb-server-adapter, request.log.threshold.milliseconds.
- **operationtimes-db.log**
  Contains log messages about database access times of atomic operations in order to identify possible performance bottlenecks. This logging is configured using the system properties cmas-core-server, dao.log.threshold.milliseconds and cmas-core-server, dao.log.username.
- **operationtimes-rest.log**
  Contains log messages with operation times for REST requests. The request URL and the time needed to process the request are logged on `DEBUG` level. The logging is configured using the system properties cmas-restapi-core, request.log.threshold.milliseconds and cmas-restapi-core, request.log.username.
- **security.log**
  Contains log messages which are related to security, authorization and authentication
- **server.log**
  Contains all log messages with at least log level `INFO`.
- **session.log**
  Contains log messages about logins (session starts) and session timeouts of users.

- **sql.log**
  Contains log messages 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 error messages which are thrown by supporting libraries but are properly handled by ConSol CM (this method keeps the `server.log` clean).
- **timer-manager.log**
  Contains additional log messages written in log level `DEBUG` when workflow timers are activated or deactivated. Information about the escalation date is logged, too.
- **track_jersey.log**
  Contains log messages from CM/Track.
- **transfer.log**
  Contains log messages about the export / import of scenes.
- **tx.log**
  Contains log messages related to Spring Framework transactions.
- **unit-deletion.log**
  Contains information regarding the deletion of contacts.
- **workflow.log**
  Contains log messages about activated / reinitialized / deactivated timers, which are logged with level `INFO` and all debug output related to the workflow engine.

G.3.2.1 Structure of log files

In the default configuration, the log file entries have the following syntax:

```
Date Timestamp Loglevel [Logger] Message
```

Example for a log file entry:

```
2020-10-11 13:52:44,526 INFO [reemarker.FreeMarkerConfigurer] [-]
 ClassTemplateLoader for Spring macros added to FreeMarker configuration
```

The components of the message:

- **Date:**
  October 11th, 2020
- **Timestamp:**
  13:52:44
- **Log level:**
  INFO
- **Logger:**
  reemarker.FreeMarkerConfigurer
  Name of a Java class, not complete (only last 30 characters), the real name would be `Freemarker.FreeMarkerConfigurer`
- **Message:**
  ClassTemplateLoader for Spring macros added to FreeMarker configuration

Simple messages and messages reporting a successful operation often have only one line. When errors are reported (log level `ERROR`), you might find stack traces in the logs.

G.3.3 Basic tasks

G.3.3.1 Accessing the log files using the Web Admin Suite.

You can access the log files which are saved in the default directory on the *Log files* page in the *Operation* menu. When opening the page, the latest lines of the `server.log` file are displayed.

You can perform the following actions on the page:

- **View a log file**
  Select the desired log file in the selector on the left side. All the log files which are saved in the default directory, `<JBOSS_HOME>/standalone/log`, are available. You can click the *Full screen* icon to view the log file in full screen mode. The *Wrap text* / *Unwrap text* icon allows you to decide whether the log messages should be wrapped or not.
- **Download log files**
  Click the *Download log files* button and select the log files which you want to download. The currently displayed log file is preselected. You can select additional log files to download several files at once. If several files are selected, they are downloaded as zip package. You can click the *Download all log files* button to add all log files to the zip package.
- **Refresh log files**
  By default, the log files are refreshed every 5 seconds. If needed, you can change the selector to *Refresh every second* or *Do not refresh automatically*. You can reload the log files manually at any time by clicking the *Refresh* icon.
- **Configure display order**
  The newest lines are displayed at the top if the checkbox *Display newest lines first* is selected. Deselect his checkbox to display the oldest lines first.

G.3.3.2 Add logging to scripts

You can add log output to your scripts by using the `log` statement with the desired log level.

```
log.info "This is my log message."
```

“DEBUG” messages will only appear in the log file when the logging configuration has been changed. Check whether debugging is switched on to avoid unnecessary operations:

```
if (log.isDebugEnabled()) {
   log.debug("Processing ticket ${ticket.name}")
}
```

G.3.4 Advanced tasks

G.3.4.1 Configuring logging

The default path for the log files is `<JBOSS_HOME>/standalone/log`. The location of the log files and the logging behavior can be configured in the respective configuration files:

- **cm6-config.properties**
  (JBoss EAP / Wildfly standalone installations with or without CMRF)
- **cm6-domain-config.properties**
  (JBoss EAP cluster installation)
- **cmrf.properties**
  (CMRF as a standalone application)

See Configuring the log files for details.

G.3.4.2 Activating the access.log file

In a default installation, the following lines in the configuration file, `cm6-<DB>.xml` or `cm6-dwh-<DB>.xml` are commented out. By commenting them in, you can start access logging in your ConSol CM system.

```
<server name="default-server">
   <http-listener name="default" max-post-size="${max-post-size-
    bytes:104857600}"
      socket-binding="http"
      redirect-socket="https"
      enable-http2="true"/>
   <ajp-listener name="ajp" socket-binding="ajp"
      max-post-size="${max-post-size-bytes:104857600}"/>
   <https-listener name="https" max-post-size="${max-post-size-bytes:104857600}"
      socket-binding="https"
      ssl-context="applicationSSC"
      enable-http2="true"/>
   <host name="default-host" alias="localhost"><!--
      <access-log pattern="%h %l %u %t %r %s %b %{Referer}i %{User-Agent}i %S
       %T" directory="${jboss.server.log.dir}" prefix="access" suffix=".log"
       worker="default"/>
   --><location name="/" handler="welcome-content"/>
   <http-invoker/>
   </host>
</server>
```

G.3.5 Configuring the log files

The built-in logging module of JBoss EAP is used for the ConSol CM logging subsystem. Usually, no changes are required in a standard ConSol CM standalone installation.

The log files are saved under `<JBOSS_HOME>/standalone/log`.

The configuration of the logging subsystem is defined in one of the following files, depending on the infrastructure of your CM system:

- **ConSol CM only:**
  <JBOSS_HOME>/standalone/configuration/cm6-<DB>.xml
- **ConSol CM with CMRF/DWH**
  <JBOSS_HOME>/standalone/configuration/cm6-dwh-<DB>.xml

> ⓘ Usually, you do not need to edit the XML files.

You can modify the default configuration in the respective configuration file:

- **ConSol CM only or ConSol CM with CMRF/DWH**
  <JBOSS_HOME>/bin/cm6-config.properties
- **Standalone CMRF**
  <CMRF_HOME>/cmrf.properties

For configuring the logging behavior, the following section of the configuration file is relevant:

```
<subsystem xmlns="urn:jboss:domain:logging:3.0">
```

Within the section of this subsystem, the file handlers are defined, each in a separate subsection. By default, periodic size rotating file handlers are used. One log file is created for every day. If the maximum size of 50 MB is exceeded, a new file is created for this day. 10 files are kept as backup.

Example from the `index.log` section:

```
<periodic-size-rotating-file-handler name="INDEX_FILE" autoflush="true">
  <level name="${handler.indexfile.level:INFO}"/>
  <formatter>
    <pattern-formatter pattern="%d %-5.5p [%30.-30c] [%X{username}-%X
     {context}-%X{sessionId}] %m%n"/>
  </formatter>
  <file relative-to="jboss.server.log.dir" path="index.log"/>
  <suffix value=".yyyy.MM.dd"/>
  <max-backup-index value="${handler.indexfile.max.backup.index:10}"/>
  <rotate-size value="${handler.indexfile.rotate.size:50m}"/>
  <append value="true"/>
</periodic-size-rotating-file-handler>
```

Below the file handlers, the loggers are defined. For each logger, the target file(s) and the log level are provided.

```
<logger category="com.consol.cmas.core.index" use-parent-handlers="false">
   <level name="${logger.com.consol.cmas.core.index.level:INFO}"/>
   <handlers>
      <handler name="INDEX_FILE"/>
   </handlers>
</logger>
```

This means that the logger for classes belonging to `com.consol.cmas.core.index` writes into the file `INDEX_FILE`, which is defined in the file handler section above.

The variables in the syntax `${setting:value}`, e.g. `${handler.indexfile.level:INFO}`, indicate parameters which can be overridden in the `<JBOSS_HOME>/bin/cm6-config.properties` file.

G.3.5.1 Changing default settings

Please proceed as follows to modify logging settings:

1. Locate the logging configuration in the `cm6-<DB>.xml` file or `cm6-dwh-<DB>.xml`
2. Add entries with the desired values to the `cm6-config.properties` file.

The following example shows how to change default settings using the `cm6-config.properties` file.

Goal: You want to change the log level of the `index.log` file to DEBUG.

1. Locate the respective handler in the configuration file (example from `cm6-mysql-5.xml`).

```
<periodic-size-rotating-file-handler name="INDEX_FILE" autoflush="true">
   <level name="${handler.indexfile.level:INFO}"/>
   <formatter>
      <pattern-formatter pattern="%d %-5.5p [%30.-30c] [%X{username}-%X
       {context}-%X{sessionId}] %m%n"/>
   </formatter>
   <file relative-to="jboss.server.log.dir" path="index.log"/>
   <suffix value=".yyyy.MM.dd"/>
   <max-backup-index value="${handler.indexfile.max.backup.index:10}"/>
   <rotate-size value="${handler.indexfile.rotate.size:50m}"/>
   <append value="true"/>
</periodic-size-rotating-file-handler>
```

2. Locate the variable for the log level.

   This is `${handler.indexfile.level:INFO}`.

3. Add an entry for this variable to `cm6-config.properties`.

```
# logging overrides
handler.indexfile.level=DEBUG
```

Other changes you can make:

- Change the size of the single log files:
  → Set the value (in MB) of the parameter `rotate-size`, e.g. `handler.indexfile.rotate.size=50`.

- Change the number of copies which are saved:
  → Set the value of the parameter `max-backup-index`, e.g.
  `handler.indexfile.max.backup.index=1`.
- Debug a certain module:
  → Set the log level to “DEBUG” for the respective logger, e.g.
  `logger.com.consol.cmas.core.index.level=INFO`

> ⓘ If there is no variable for a setting which you want to modify, you can edit the `cm6-<DB>.xml` or `cm6-dwh-<DB>.xml` file directly. This is required for example if you want to change the pattern for the log file entries (`pattern-formatter` attribute).

G.4 License

G.4.1 Introduction to licenses in ConSol CM

When you purchase ConSol CM, you receive a license file valid for the purchased modules and the required number of concurrent users.

G.4.1.1 Concepts, terms and definitions

Concept	Other terms	Definition
license		Text file containing information about the purchased modules and number of users.
concurrent user license	floating license	License for a certain number of users who are logged in simultaneously.

G.4.1.2 Purpose and usage

The license file is a text file which contains entries for several modules. For each module, the number of concurrent users and the expiration date is indicated.

The license always refers to concurrent users. This means that only the number of users who are logged in simultaneously is considered. It is not checked which users are logged in. Therefore, you do not need one license for each user.

A license is consumed when the user logs in. The license is handed back to the server when the user session is terminated, either because the user logs out or because the session times out (see cmas-core-server, server.session.timeout). You can see the active sessions on the *Sessions* page of the Web Admin Suite.

G.4.2 Available settings for licenses

The license file has the following sections:

- **[ADDONS]**
  The ConSol CM add-ons, e.g. CM/EBIA, CM/Resource Pool, which can be used.
- **[CONCURRENT_USERS]**
  The number of user who can log in to the Web Client at the same time.
- **[REST_USERS]**
  The number of users who can access the REST API. This does not include the number of CM/Track users.

  > ⓘ CM/Phone consumes one REST licenses per client where CM/Phone is installed and active.

- **[TRACK]**
  The number of contacts who can log in to CM/Track at the same time.
- **[TRACK_USERS]**
  The number of user profiles for CM/Track. This is the number of users who are created as *Portal profile* on the *Users* page of the Web Admin Suite.

  > ⓘ Since several contacts can use the same user profile in CM/Track, the license numbers of [TRACK] and [TRACK_USERS] might differ considerably. For example, there might be two CM/Track profiles, but 100 contacts should be allowed to log in to CM/Track at the same time. This would mean 2 [TRACK_USERS] licenses and 100 [TRACK] licenses.

Each license section has the following data:

- **contractParty**
  The name of the customer who owns the license.
- **products**
  The ConSol CM module to which the license refers.
- **version**
  The ConSol CM version for which the license is valid.
- **expirationDate**
  The date when the license expires.
- **licenses**
  The number of licenses for the respective module.
- **enabledModules**
  Only for the module [ADDONS]. The names of the purchased add-ons, e.g. RESOURCE_POOL if CM/Resource Pool has been purchased.
- **signature**
  The license code.

G.4.3 Basic tasks

The license is managed on the *License* page of the Web Admin Suite.

G.4.3.1 Importing a new license

You can import a new license in two ways:

- Click the *Import license* button and select the license file. Click the *Save license* button.
- Open the license file in a text editor and paste the license text into the license editor in the Web Admin Suite. Click the *Save license* button.

The new license takes effect immediately. There is no downtime required.

G.4.4 Advanced tasks

G.4.4.1 Monitoring the license

You can use the monitoring endpoint to check the license status.

1. Open `<CM URL>/health/check` or `<CM URL>/health/report`.
2. Locate the section `cmas-core-server-licence`:
    - The `daysUntilNextExpiry` attribute indicates the number of days until the first license expires.
    - The `statistics` attribute lists all license pools with the number of purchased licenses and the number of currently used licenses.

G.4.5 Control of consumed licenses

You can control the number of licenses in using the system monitoring feature. In addition, each log in and log out operation is written to the log files. Alternatively, you can use database queries or MBeans to check the license usage.

> The monitoring user does not consume a license, see System monitoring.

G.4.5.1 System monitoring

The number of available and consumed licenses can be retrieved using the health check endpoints `health/check` and `health/report`, see System monitoring. The JSON output shows the total number of licenses (attribute `all licenses`) and the number of licenses which are currently in use (attribute `consumed licenses`) by license pool. To determine the number of engineers who are logged in to the Web Client, check the pool CONCURRENT_USERS.

In addition you see the number of days until the first license is going to expire (attribute `daysUntilNextExpiry`).

```
"cmas-core-server-licence" : {
   "status" : "SUCCESS",
   "name" : "licence system information",
   "attributes" : {
      "daysUntilNextExpiry" : 150,
      "statistics" : [ {
         "pool name" : "PROCESS_DESIGNER",
         "consumed licences" : 0,
         "all licences" : 5
      }, {
         "pool name" : "TRACK_USERS",
         "consumed licences" : 0,
         "all licences" : 5
      }, {
         "pool name" : "REST_USERS",
         "consumed licences" : 0,
         "all licences" : 5
      }, {
         "pool name" : "ADMINTOOL_USERS",
         "consumed licences" : 0,
         "all licences" : -1
      }, {
         "pool name" : "CONCURRENT_USERS",
         "consumed licences" : 1,
         "all licences" : 100
      }, {
         "pool name" : "TRACK",
         "consumed licences" : 0,
         "all licences" : 200
      } ]
   }
},
```

G.4.5.2 Log files

The message which is written to the log files when a user logs in or out contains the name of the client. For the Web Client, it also shows the number of licenses which are currently in use.

Example 1: User logs in to the Web Client

```
2022-12-19 11:18:25,069 INFO [ sessionTimeoutEngineerLogger] [-authentication-]
 New session for engineer: webadmin and client: WEB_CLIENT has started. Session
 id: b0a95ce1-4172-4534-a95c-e14172e534fc. (Licenses in use: 1)
```

Example 2: User logs out from the Web Client

```
2022-12-19 11:19:05,382 INFO [ sessionTimeoutEngineerLogger] [webadmin-welcome-
 b0a95ce1-4172-4534-a95c-e14172e534fc] Session of engineer: webadmin and
 client: WEB_CLIENT has ended. Reason: logout. Session id: b0a95ce1-4172-4534-
 a95c-e14172e534fc. (Licenses in use: 0)
```

G.4.5.3 Database queries

The sessions are saved in the CM database table `cmas_user_session`. The following statements return the number of licenses in use.

```
SELECT count(auth_username) FROM cmas_user_session WHERE end_date = 0 and
 session_source = 'WEB_CLIENT';
```

Code example 3: *Web Client sessions, CM version 6.10*

```
SELECT count(*) FROM cmas_user_session WHERE end_date = 0 and session_type =
 'CUSTOMER';
```

Code example 4: *CM/Track sessions, CM version 6.10*

G.4.5.4 MBeans

The content of CM licenses can also be queried using the MBean `licenseDeployer`. Three methods are available:

- getRemainingDays
- deployLicence
- getLicenseInfo

G.5 Index

G.5.1 Introduction to the index in ConSol CM

ConSol CM stores most of its data in a relational database. In order to improve the performance of search operations, ***Apache Lucene*** indexes are used. For each data field that should serve as search criterion, an index is created. The indexes are stored on the file system, in a subdirectory of the data directory.

G.5.1.1 Concepts, terms and definitions

Concept	Other terms	Definition
indexer		ConSol CM component for index management
quick search		Simple free-text search using a search field in the upper right corner of the Web Client
detailed search		Advanced search on a specific page which allows combining several search criteria
autocomplete search		Context-dependent free-text search integrated into specific pages

G.5.1.2 Purpose and usage

The index is required to be able to search for objects in ConSol CM. The scope of the index are the basic user and case data, the entire case text, the attachments (can be configured in the system property cmas-core-index-common, index.attachment) and the data fields which have the setting *Indexed for search*. The data fields configured for the index are available as search criteria in the detailed search, and they can trigger an autocomplete search on contact and resource pages. The quick search shows the objects which contain the search term in an indexed field.

In addition, fields need to be indexed for the following purposes:

- Field is used as a table column and the users should be able to sort the table by this column
- Field is used in scripts to search for entities, e.g. using criteria objects as `TicketCriteria`, `UnitCriteria` or `ResourceCriteria`

G.5.2 Available settings for the index

The configuration of the index is done in a number of different places. The items described in the following sections need to be configured for the index to work correctly. In addition, you can fine-tune the indexer in the system properties, see Fine-tuning the indexer.

G.5.2.1 Data directory

The indexes are stored in the file system, in a subdirectory of the data directory indicated during system setup. The path of the data directory is stored in the system property cmas-core-shared, data.-directory.

The following requirements need to be fulfilled for the index to work correctly:

- The data directory is always available for the ConSol CM server, also if it was created on another server and is linked to or mounted on the application server.
- The data directory provides enough space for storing indexes.

G.5.2.2 Field settings

The data fields which should be available as search criteria need to be specified by setting *Indexed for search* for the respective field. This is done in the data field administration.

There are four possible values:

- **True:**
  An index is created for the field, so the object (case, contact or resource) can be searched by this field. Only the object itself is found.
- **True (company + persons):**
  Only relevant for contacts. An index is created for the field, so the contact can be searched by this field. For persons, only the person itself is found. For companies, the persons of the company are found as well.
- **True (company + persons + cases):**
  Only relevant for contacts. An index is created for the field, so the contact can be searched by this field. For persons, the cases of the person are also found. For companies, the cases of the company, the persons of the company, and the cases of the persons of the company are found as well.
- **False:**
  No index is created for the field, so it cannot be searched.

⚠ Including cases and persons for contact fields can have a negative impact on the indexer performance if the company or person has many cases or the company has many persons. This is due to the fact that all cases or persons have to be reindexed when there are changes to the field.

Nested fields, as used in lists or tables, all have to have the same indexing type, otherwise they cannot be searched. For example, when you work with a table, the data fields of the type *List*, *Columns*, and all data fields within the table which should be searchable, need to have the value “True” for the setting *Indexed for search*.

A **phonetic search** can be enabled for string fields. The phonetic search results do not include only exact matches but also results which sound similar even if the spelling differs from the entered search term, e.g. “Claire” and “Clare”. In order to activate the phonetic search for a data field, set *Phonetic search* to “true” for this field.

G.5.2.3 Update mode

There are two kinds of changes which require index updates:

- **Operative changes:**
  Changes to cases, contacts, resources and users
- **Administrative changes:**
  Changes to the configuration

Operative changes are always processed directly. For the processing of administrative changes, there are two modes:

- **Directly start processing:**
  Administrative changes are processed directly. Select the checkbox *Administrative changes are applied to the index automatically*.
- **Start processing manually:**
  Administrative changes are processed when the user clicks the *Apply administrative changes* button on the *Index* page. Deactivate the checkbox *Administrative changes are applied to the index automatically*.

> ⚠ Direct processing of administrative changes can have a negative impact on the indexer performance if there are changes which require reindexing lots of objects, e.g. if a data field used in all cases is added to the index.

See Technical background of the index for further details.

G.5.3 Basic tasks

There are two types of tasks related to the index:

- **Configuration-related tasks:** These tasks are performed when configuring the ConSol CM system.
- **Operation-related tasks:** These tasks are performed when operating a ConSol CM system.

G.5.3.1 Configuration-related tasks

You can perform the following configuration-related tasks on the index:

- Decide which data fields should be indexed, i.e. available for search, see Field settings.
- Decide if attachments should be indexed, see system property cmas-core-index-common, index.attachment. The default is `true`, meaning that attachments are indexed.
- Decide if administrative changes should be applied automatically, see Update mode.

G.5.3.2 Operation-related tasks

You can perform the following operation-related tasks on the index:

- **Check the current index tasks:**
  The current index tasks are displayed in the table on the *Index* page. You can see the ID, the type of task (indicates how the task was originated), the status (pending, running or paused), the creation date, the progress, and the details (which information is updated in the index).
- **Manage current index tasks:**
  You can pause a running index task by clicking the *Pause* icon in the task's row. Paused index tasks can be resumed by clicking the *Resume* icon.
- **Check the index status:**
  The index status is shown below the header buttons. In addition, it is saved to the system property cmas-core-index-common, index.status. There are three status:
  - **GREEN**
    All index tasks have run correctly. At the beginning of the synchronization process, the index status is set to GREEN. If there are any problems, it will change to YELLOW or RED.
  - **YELLOW**
    Fixable problems were identified. This status is set when there are pending administrative changes (*Administrative changes are applied to the index automatically* not selected) or a retry task was created.
  - **RED**
    Errors have occurred. The index needs to be rebuilt.
- **Rebuild the index:**
  If the index status is RED, you need to rebuild the index by clicking the *Rebuild index* button. All open index tasks are discarded and the index is rebuilt according to the options selected in the modal window:
  - **Cases**
    Determines the order in which the cases are indexed. You can add queues to the list of *Prioritized queues* and sort them to determines the indexing order. The cases from the *Remaining queues* are indexed afterwards. The options above the lists allow to decide how to handle closed cases:

        - **Index open cases by queue priorization first**: Indexes first the open cases according to the queue priorization and then the closed cases.
        - **Index open and closed cases by queue priorization**: Indexes open and closed cases according to the queue priorization.
    - **Attachments**
      Determines when to index attachments:
        - **Index cases directly with attachments**: The attachments are indexed together with the cases which they belong to. With this option, it takes longer until all cases are indexed.
        - **Index cases first, them attachments**: First the cases are indexed without attachments. The attachments are indexed afterwards. With this option, the cases are available for search more quickly, because indexing the attachments takes more time.

  Click *Start rebuild* to synchronize the index.

  > ⓘ The existing index is **not** deleted during the rebuild process. Therefore, the search features are available while the process runs. Changes made after the rebuild was started are immediately reflected in the index, because these data have a higher priority than the data which is synchronized due to a manually triggered index update.

- **Rebuild the index for a certain period:**
  You can rebuild the index for a certain period if there were known indexer problems during this time. Click the button *Rebuild index by period* to select the time range. All the changes (creation and update) which were made in the selected time period are reindexed. Deletions and field setting changes are not considered.
- **Repair the index:**
  You can repair the index by clicking the *Execute retry tasks* button. All the pending index tasks of the type *Retry tasks* are executed.
- **Apply administrative changes:**
  If the checkbox *Administrative changes are applied to the index automatically* is not selected, you need to click the button *Apply administrative changes* to update the index after configuration changes.

You can rebuild the index using the MBean method `recreateIndex` for the respective type of objects:

- **Users:**
  core.engineerIndexService
- **Resources:**
  core.resourceIndexService
- **Cases:**
  core.ticketIndexService
- **Contacts (i.e., persons and companies):**
  core.unitIndexService

G.5.4 Advanced tasks

G.5.4.1 Fine-tuning the indexer

The behavior of the indexer can be fine-tuned using system properties. The relevant properties are located in the module `cmas-core-index-common`. These properties allow to adjust parameters to improve the indexer performance, e.g. the number of threads, or whether attachments should be indexed or not.

G.5.4.2 Technical background of the index

Which changes require index updates?

There are two kind of changes relevant for the index:

- **Operative changes**
  These are changes to the runtime data:
  - **Data field values:** values of fields (case, contact and resource fields) which are configured for indexing using the *Indexed for search* setting.
  - **User data:** email, first name, last name
  - **Case data:** attachments (unless configured otherwise), creation date, assignee, history, name, queue, participants, subject, view

  The index is automatically updated when these changes are carried out.
- **Administrative changes**
  These are changes of certain configuration data:
  - scopes
  - queues
  - list values
  - user functions
  - supported locales
  - roles
  - *Indexed for search* setting of case, contact or resource fields

  The processing of administrative changes depends on the setting *Administrative changes are applied to the index automatically*:
  - If the checkbox is selected, the changes are processed automatically.
  - If the checkbox is not selected, the changes are stored in the database for deferred processing. Processing can be started by clicking the *Apply administrative changes* button.

Which is the performance impact of changes?

The performance impact of the index changes depends on the number of objects which need to be reindexed.

Change	Requires re-indexing of	Performance impact
Add or delete a system language	All cases, contacts and resources	Critical

Change	Requires re-indexing of	Performance impact
Change internal or localized name of a queue	All cases residing in that queue	High
Add, remove or change value of the *Indexed for search* setting for a data field	All cases, contacts or resources which have this field set	High
Change internal name, localized name or order of a list value	All cases, contacts or resources which have the changed list value set	High
Add or remove the *Rich text display in view mode* setting to / from an indexed data field of the type "Text (long)"	All cases, contacts or resources which have this field set	Medium
Change internal name, localized name or order of a scope	All cases currently residing in the changed scope	Medium
Add or remove the *Phone number* setting of a contact field	All contacts which have the data field	Medium
Change phone numbers configuration of a customer group	All contacts of the customer group which have a phone number field	Medium
Change internal or localized name of a user function	All cases which have at least one participant in that function	Low
Change login, first name or last name of a user	All cases which the user is assigned to (as primary or referenced user)	Low
Change internal name or localized name of a resource type or resource category	All resources of that resource type / all resources which belong to the resource category	Low

How are index updates processed?

Two services are involved in index updates:

- **Index changes notifier:**
  Stores the entities which require an index update in the `cmas_index_update_serialized` table.
- **Index changes receiver:**
  Polls the `cmas_index_update_serialized` table for new entries and creates index tasks in the tables `cmas_index_update_task` and `cmas_index_update_part`. Administrative tasks are directly created in the database table `cmas_index_administrative_task`. The indexing server then executes the tasks from these tables and updates the Lucene files accordingly. If the execution of an update task fails, a new update task with the type *Retry task* is created and the index status is set to YELLOW. You need to execute the task by clicking the *Execute retry tasks* button.

> ⓘ On clustered systems, the index is built on the master indexing server. The slave indexing servers download the index files from the master indexing server.

⚠ Do **not** stop the *Index changes notifier* service. If it is stopped and there is a change which requires an index update, the index status is set to RED and a rebuild is necessary. You can stop the *Index changes receiver*. After restart it will pick up all the missing changes from the respective table

G.5.5 Operating the indexer

This section describes the use of the indexer in a single-server environment. Please contact the ConSol CM support or your ConSol CM consultant for information about the indexer in a cluster.

G.5.5.1 Indexer directory structure

The data directory is set using the system property cmas-core-shared, data.directory. The structure and content of the index directory are described in ConSol CM data directory.

G.5.5.2 Monitoring the indexer

The indexer status is displayed on the *Index* page of the Web Admin Suite. Alternatively, you can check the system property cmas-core-index-common, index.status. The following status can occur:

- **GREEN**: The index is fine. No action is required.
- **YELLOW**: There are solvable problems with the index, either pending administrative changes or pending retry tasks. You need to perform the required action in the Web Admin Suite.
- **RED**: There are errors. You need to rebuild the index in the Web Admin Suite.

In addition, you need to monitor the file system where the index is located to ensure that there are enough space and RAM available, that the file system is available (if it is mounted on another server), and that the synchronization between the master and slave nodes works correctly (if you have a clustered environment).

In addition, the following problems can indicate that the indexer is not running smoothly:

- The search in the Web Client does not work correctly. Newly created objects are not found or the search does not work at all.
- Incoming emails are not processed (processing requires a search for case and contact).

G.5.5.3 Backup and recovery of the index

All ConSol CM indices are stored on the file system. Please follow the following procedure to create an index backup, in order to avoid inconsistent states:

1. Back up the index using the following HTTP request. Replace the variables with the correct values for you system (URL, and name and password of the admin user):

   ```
   wget http://${indexing.master.host:port}/index/snapshot --user ${admin} --
    password ${password} -Obackup.jar
   ```

   This will download the full index into a `backup.jar` file. The received full snapshot will have the timestamp of the moment when the command was executed.

Please perform the following steps to restore the index:

1. Stop the ConSol CM server.
2. Clean the directories inside the `index` folder.
3. Unpack the `backup.jar` file to the `index` folder.

4. Restart the ConSol CM server.
5. Rebuild the index for the missing time period (between the creation of the backup and the restore) using the Web Admin Suite.

G.5.5.4 Fine-tuning the index

If the search is slow or you notice delays, you might increase the number of indexer threads in the CM system property cmas-core-index-common, index.task.worker.threads. The default value is “1”. Do not set the value to a number higher than the number of CPU cores of the machine.

G.6 Email

Sending and receiving emails is one of the core functionalities of ***ConSol CM***. The application interacts with one or more mail servers to fetch emails and send emails.

> ⓘ This section describes emailing in a single-server environment. Please contact the ConSol CM support or your ConSol CM consultant for information about emailing in a cluster.

G.6.1 Relevant components

In order to run smoothly, the following modules of ConSol CM have to be taken into consideration on a server:

- email has to be fetched
- email has to be sent

> ⓘ The email configuration is explained in detail in the *ConSol CM Administrator Manual*, section *Email*.

G.6.1.1 Fetching emails

The following items are relevant for fetching emails:

- **Configured accounts**
  The accounts from which emails should be fetched must be configured in the *Accounts* tab of the *Email configuration* page in the *Email* menu of the Web Admin Suite.
- **Service**
  The NIMH service must be active, see *Services* page in the *Operation* menu of the Web Admin Suite.
- **Scripts**
  The email scripts for NIMH, e.g. `NIMHCreateTicket.groovy`, must exist, see *Scripts* page in the *Business logic* menu of the Web Admin Suite.
- **Data**
  Emails which cannot be processed are stored in the database table `cmas_nimh_archived_mail`. Emails, which cannot be saved to the database, e.g., due to their size, are saved in the file system (subfolder `/mail/unstorable` of the ConSol CM data directory) instead.

> ⓘ ConSol CM only fetches emails which are marked as “unread” on the mail server.

G.6.1.2 Sending emails

The following items are relevant for fetching emails:

- **Configured account**
  The account which should be used to send emails must be configured in the *Settings* tab of the *Email configuration* page in the *Email* menu of the Web Admin Suite.

G.6.2 Fine-tuning

G.6.2.1 Changing administrator email addresses

You can change the administrator email address in the *Addresses* tab of the *Email configuration* page in the *Email* menu of the Web Admin Suite. This updates the system property cmas-core-security, admin.email.

G.6.2.2 Changing email parameters

The basic email parameters are changed on the *Email configuration* page of the Web Admin Suite. Advanced parameters are managed in system properties. They can be changed on the *System properties* page.

The following properties regarding email attachments are modified frequently:

- **Define the maximum attachment size**: Set the maximum size in MB for attachments at cases, contacts, resources and incoming emails using cmas-core-server, attachment.max.size.

> ⓘ Before allowing large attachments, please consider:
>
> - Large attachments increase the database size significantly.
> - If attachments are indexed (CM system property cmas-core-index-common, index.attachment is set to “true”), the indexing operation takes long for large attachments.

- **Define the allowed file types**: Provide a comma-separated list of all allowed file extensions in cmas-core-server, attachment.allowed.types.
- **Decide whether emails should be deleted from the mail server**: ConSol CM acts as a regular email client. In the system property cmas-nimh, mailbox.default.task.delete.read.messages you can define whether emails should be deleted from the mailbox on the email server after CM has fetched them. By default this property is set to “false”, so that emails are not deleted. Set this property to “true” to delete the emails which have been fetched by ConSol CM.

> ⓘ If you do not want ConSol CM to delete emails from the email server, you need to monitor the mailboxes to avoid a data overflow and server or performance problems. This is especially important because when fetching new emails, CM first has to retrieve a list of all emails in the mailbox to filter “SEEN” and new emails in the second step. When the first check takes longer than the polling interval (system property cmas-nimh, queue.task.interval.seconds), there might be a deadlock situation and no emails can be fetched anymore. Therefore, you should not keep too many “SEEN” messages in the mailbox.

- **Change the email polling interval**: Define how often emails are polled using cmas-nimh, queue.task.interval.seconds. You might want to set a shorter interval if emails received by the mail server are not transferred to ConSol CM in the required time.

G.6.3 Monitoring

Errors in the email system are logged to the `mail.log` file. Received emails, which cannot be processed are shown on the *Failed incoming emails* page of the Web Admin Suite. Emails which could not be saved to the database, e.g., due to their size, are saved in the file system (subfolder

`/mail/unstorable` of the ConSol CM data directory) instead.

When errors occur, the administrator receives an email notification. The address which should receive the notification is defined in the CM system property cmas-nimh-extension, mail.error.to.address.

If incoming emails are not processed correctly, please also check if the indexer is working correctly, see Operating the indexer.

G.7 System monitoring

G.7.1 Monitoring information provided by ConSol CM

Monitoring information can be retrieved using the health check endpoint. The monitoring reports contain information about the configuration of ConSol CM and the operational status of the system components.

Two reports are provided:

- full report: `<CM URL>/health/report`
- short report: `<CM URL>/health/check`

The information is retrieved in JSON format. Some sections contain a status (SUCCESS, WARNING) with a description.

The following information is retrieved:

- **Workflow engine**
  number of job execution threads, number of jobs (active, running, locked, delayed, under retry), transaction timeout, locking limit, timestamp of the latest job execution
- **Database**
  name, version and hibernate dialect, status and speed of test connection, number of available and used connections (only available for JBoss application servers)
- **Database performance**

  This check needs to be executed manually. It creates a temporary table, generates rows in this table and performs select operations in order to test the performance of the database.

  The check is configured using the following URL parameters:

  - **cmas-core-dao-performance.enabled**
    Determines whether the performance check is enabled. The default value is false. You need to set this parameter to “true” to activate the performance check.
  - **cmas-core-dao-performance.cleanData**
    Determines whether the table should be cleared before each run. The default value is “true”.
  - **cmas-core-dao-performance.threadsNumber**
    Determines the number of threads used for the performance check. The default value is “10”.
  - **cmas-core-dao-performance.rowsPerThread**
    Determines how many rows are generated by each thread. The default value is “5000”.
  - **cmas-core-dao-performance.selectsPerThread**
    Determines how many selects are performed by each thread. The default value is “100”.

  The syntax for using the URL parameters is:

  ```
  <CM URL>/health/report?cmas-core-dao-performance.enabled=true&cmas-core-
   dao-performance.cleanData=true
  ```

  After enabling the check, you need to wait a couple of minutes and then retrieve the report again.

- **Hardware**
  total and available memory and disk space, number of processors, CPU usage, system load, time since last boot (in seconds)
- **Server information**
  ConSol CM version, Java version, JavaMail version, status of the SMTP connection, cluster node ID (if the system runs in a cluster)
- **Server events**
  number of cluster members, timestamp and response to test calls
- **Email system**
  mailbox ID, number of queued emails, timestamp of the last connection and processed email, ID of the last processed email
- **DWH information**
  mode, status, date of last transfer, live mode status, date of last live message, statistics
- **License information**
  number of available and currently used licenses for each license pool, days until the first license expires
- **Authentication protection**
  protected endpoints, number of failed attempts in the last hour, configuration

G.7.2 Using third-party tools for monitoring

You can use the monitoring tools of your choice to control your ConSol CM system. Monitoring should cover the application server, the database and the file system.

If you want to perform end-to-end tests, you can create a dedicated monitoring user which does not consume a license. This user can access the respective client once and act according to his permissions. The session created is marked as a monitoring session.

The login of the monitoring user in the system properties.

Web Client:

1. Create the user in the Web Admin Suite.
2. Assign him the required roles to grant permissions.
3. Enter the login of the user in the system property cmas.core.server, monitoring.engineer.login.

CM/Track:

1. Create the contact in the Web Client.
2. Assign him the required CM/Track user profile to grant permissions.
3. Enter the login of the contact in the system property cmas.core.server, monitoring.unit.login

H - Operating the DWH

This chapter discusses the following:

H.1 Introduction

The ***ConSol CM Reporting Framework*** (CMRF) and ***Data Warehouse*** (DWH) are the modules which are required to provide the basis for Business Intelligence (BI) and reporting in ConSol CM. Please see System architecture for an overview of the required components. The use of the DWH has several advantages, most importantly, that you avoid performance decreases because you do not need to use the production database for reporting, and that the database schema and table structure are optimized for reporting queries.

The installation and setup of the CMRF and DWH is described in the section Installing and setting up CMRF and DWH. You can find detailed information about the administration of the DWH in the *ConSol CM Administrator Manual*.

The current manual contains information about the technical background of the synchronisation, see DWH synchronization, and a section about troubleshooting, see FAQs and tips for troubleshooting.

H.2 Required components

The ConSol CM database and the DWH are two separate databases or database schemes. The CMRF is responsible for transferring the data.

The following components are required to operate ConSol CM with a DWH:

- ConSol CM server is running and writes write data into the ConSol CM database.
- The CMRF is running and has access to both the CM and the DWH database.
- The DWH mode is set to *Admin* or *Live* mode. The mode determines the handling of DWH updates and tasks, see Open DWH tasks.

H.3 Open DWH tasks

Open DWH tasks are saved in the `cmas_dwh_task` table of the ConSol CM database. You can view and execute them on the *Tasks* page of the *DWH* menu in the Web Admin Suite.

Tasks are created for configuration changes, e.g. setting *Transfer to DWH* to “true” for an entity (data field or relation), or creating a new cube. If the DWH runs in LIVE mode and the checkbox *Administrative changes are applied automatically* is selected on the *Tasks* page, the tasks are executed automatically. Otherwise, you need to start them manually on the *Tasks* page. This setting is also saved in the ConSol CM system property cmas-dwh-server, autocommit.cf.changes.

H.4 Monitoring

You can see the status of the DWH on the *Monitor* page in the Web Admin Suite. In addition, the log files and some tables contain useful information:

- **hlp_parameter** (DWH table): `dwh_status` indicates the status, which should be 4, see Tables in the DWH database for details.
- **int_parameter** (DWH table): `last-success-live-id` indicates the epoch time stamp in milliseconds of the last successful live update of a transfer or update operation.
- **cmas_dwh_synchronization** (CM table): `dwh_status` indicates the status on the ConSol CM side, `cmrf_status` indicates the status on the DWH side, see DWH synchronization for details.

H.4.1 Log files

The DWH-specific log messages are written to the `cmrf.log` file. There are several ways to access it:

- Open it on the file system:

```
<JBOSS_HOME>/standalone/log/cmrf.log
```

- View or download it on the *Log files* page of the Web Admin Suite.
- View log entries which refer to specific DWH operations by clicking the *View log* button of the operation on the *Administration* page of the *DWH* menu in the Web Admin Suite.

In addition, `server.log` file entries related to `dwh-transfer-service`, `dwh-live-service`, `dwh-log-service` are relevant for the DWH.

See Log files for details about configuring the logging.

H.5 DWH synchronization

This chapter discusses the following:

H.5.1 Introduction

The transfer of data from the ConSol CM database to the DWH database is called synchronization. The synchronization process can be started by a manual action performed by an administrator in the Web Admin Suite, or be triggered automatically when the DWH is running in live mode. You can check the status of a synchronization action on the *Administration* page in the *DWH* menu of the Web Admin Suite. The *Monitor* page provides additional information about the status and content of the DWH.

H.5.2 Synchronization mode

The synchronization mode is determined when activating the data warehouse on the *Administration* page of the *DWH* menu in the Web Admin Suite. The following values are available for the DWH mode:

- **Off**: The data warehouse is disabled.
- **Admin**: The data warehouse is enabled, but the data is not updated automatically. The administrator needs to create an update action to start synchronization.
- **Live**: The data warehouse is enabled and the data is updated automatically. This means that the required update actions are created and executed automatically. Therefore, the data is always up-to-date without administrator actions.

If you run the DWH in LIVE mode, you can choose whether tasks, which are created after configuration changes, should be executed automatically or manually.

H.5.3 Components involved in transfer

H.5.3.1 Tables in the CM database

- **cmas_dwh_operation_status**
  The DWH log service writes into this table to document the DWH status. These are the info and error log entries which you can also find in the log file `cmrf.log`.
- **cmas_dwh_ser_sync_object**
  This table is only used in LIVE mode. CM inserts an entry into this table for a transaction which was performed in the CM database. The entry contains a serialized package which needs to be sent to CMRF. The DWH live service reads from this table.
- **cmas_dwh_synchronization**
  When a DWH operation is performed using the Web Admin Suite, an entry is created in this table. The column `type` contains one of the possible values: INITIALIZATION, REINITIALIZATION, TRANSFER, UPDATE, TASK. The DWH transfer service reads from this table.
- **cmas_dwh_task**
  An entry in this database is created when a setting which is relevant for DWH transfer has been changed, i.e. *Transfer to DWH* and *Field value omitted from DWH history*. Tasks listed in this table are visible on the *Tasks* page under the *DWH* menu in the Web Admin Suite. When a task is started (automatically or manually), a new entry in `cmas_dwh_synchronization` is created. When the data has been processed successfully, the entry from `cmas_dwh_task` is deleted.

H.5.3.2 Tables in the DWH database

- **INT_CONTROL_QUEUE**
  INT_CONTROL_QUEUE is used to control the processing in CMRF. Two actions are available: *pause* and *resume*. Only the JMX components write into INT_CONTROL_QUEUE. Only the CMRF service reads and deletes entries from INT_CONTROL_QUEUE. It is done during processing of other data (e.g. transfer). The CMRF service pauses processing after *pause* action and continues after *resume* action.
- **INT_LIVE_QUEUE**
  The DWH live service writes serialized packages into this table. CMRF reads from this table to process the LIVE data.
- **INT_TRANSFER_QUEUE**
  The DWH transfer service writes serialized packages into this table. CMRF reads from this table to process the TRANSFER data.
- **INT_LOG_QUEUE**
  This table is used for log entries written by the CMRF. Based on those entries, the DWH log service creates entries in the table `cmas_dwh_operation_status`. See section DWH log service.
- **hlp_parameter**
  This table is not directly involved in transfer but provides some useful information, most importantly:
  - **parameter_namelast_data_transfer**: The timestamp of the end of the last DHW update or transfer in the local time of the CMRF server.
  - **parameter_namedwh_status**: The possible values are explained in the following table.

dwh_ status (ID)	Description of the value	Explanation
0	DWH uninitialized	This is the status after application startup, when the DWH sees that the database has not been initialized. This is a theoretical value, because when the DWH is uninitialized, it does not yet exist.
1	DWH waiting for initial transfer	This status is set during (re)initialization. After the start of the DWH application, the DWH sees that there has not been a data transfer yet. When transfer messages are sent, the DWH switches to the status 2.
2	DWH: data transfer in progress	The DWH enters this status when the transfer action is executed. The CMRF is working on transfer messages, and the DWH ignores all incoming non-transfer messages (there may have been problems with the transmission). The DWH remains in this status until either a fatal error occurs (subsequent status will be 3) or all transfer messages have been handled (subsequent status will be either 3 or 4, depending on whether they were errors).
3	DWH: data transfer finished unsuccessfully	All data transfer messages have been handled, but, there were some errors.
4	DWH operational	The DWH is operational and will handle all incoming update messages from CM. This status is reached when all data transfers or updates have finished successfully.

- **hlp_transfer_error**:
  Keeps a copy of the error message from exceptions of the CMRF live mode since the last REINITIALIZE. Here you can look up the `object_uid` for which the exception occurred.
- **int_parameter**:
  This table show technical details about the data transfer applied to the DWH database.
  - **live-serial-number**: The last applied serial number during live mode.
  - **live-error**: Boolean. If it is “false”, the live mode did not abort.
  - **last-success-id**: Epoch time stamp in milliseconds of the last successful live mode update
  - **last-success-live-id**: Epoch time stamp in milliseconds of the last successful live update of a transfer or update operation. Note: This value can be lower than `last-success-id`, when a transfer or update task is being executed.

H.5.3.3 CM processes and services

The following processes and services are relevant for the data transfer from the ConSol CM database to the DWH . The details are explained in the following sections.

- DWH transfer service
- CMRF service
- DWH log service
- DWH live service

H.5.4 Manual operations

When the administrator clicks one of the buttons of the *Administration* page, the corresponding action is created in the ConSol CM table `cmas_dwh_synchronization` and shown on the page. The actions are executed according to the timestamp of their creation in ascending order (oldest actions first).

The sequence for the initial setup of the DWH is:

1. Initialize
2. Transfer
3. Update

If you do not follow the required order (e.g. perform an update without having performed a transfer first) or try to execute several actions of the same type (e.g. perform an transfer while another transfer is still running), warning messages are shown. If you have created an action by accident, you can remove it by clicking the *Delete* icon while the action is still in state NEW.

During the regular operation of the DWH, i.e. after the initial data transfer, the only actions which you need to execute are updates when the DWH is running in admin mode.

H.5.4.1 Initialize

The initialization operation creates a database structure with all static tables and relations. This step is only performed once during the initial setup of the data warehouse.

It creates a new entry in `cmas_dwh_synchronization` with the type 'INITIALIZATION' and both the DWH status and the CMRF status 'NEW'.

If you want to reinitialize the data warehouse, i.e. rebuild it from scratch after an error, you must select the *Delete existing data* checkbox. This drops the whole database structure and creates it again. In this case, the entry in `cmas_dwh_synchronization` is created with the type 'REINITIALIZATION':

The (re)initialization is processed in another thread, by the DWH transfer service.

H.5.4.2 Transfer

The transfer operation creates the dynamic tables and fills both the static and the dynamic tables with data from ConSol CM. This step is only performed once during the initial setup of the data warehouse. If the data warehouse tables are not empty, all existing data and all custom tables are removed during the transfer.

It creates new entry in `cmas_dwh_synchronization` with the type 'TRANSFER' and both the DWH status and the CMRF status 'NEW'. In addition, the entry contains the transfer settings provided in the WAS dialog.

The transfer is processed in another thread by the DWH transfer service.

> ⚠ If the DWH already contains data, it will be removed. This is not done not very efficiently though. If you need to recreate the DWH, you should use the option to initialize the DWH with delete existing data instead.

H.5.4.3 Update

The update operation adapts the DWH structure to configuration changes and transfers data which has been added or changed since the last update or transfer operation. Updates are required during the regular operation of the data warehouse. If the DWH is running in live mode, the required update actions are created and executed automatically. If the DWH is running in admin mode, you need to create the update action manually.

The update creates a new entry in `cmas_dwh_synchronization` with the type 'UPDATE' and both the DWH status and the CMRF status 'NEW'. In addition, the entry contains the update settings provided in the WAS dialog.

The update is processed in another thread by the DWH transfer service.

> You can start the update using the MBean `consol.cmas.global.dwh.synchronizationService` with the command `update`.
>
> The following command line shows an example command with Twiddle.
>
> ```
> $TWIDDLE_STANDALONE_HOME/bin/twiddle.sh -s service:jmx:remoting-
>  jmx://127.0.0.1:9999 invoke
>  consol.cmas:type=admin,topic=global,name=dwh.synchronizationService
>  update
> ```
>
> The CM package size, the CMRF cache size, and the retry counts cannot be set using the MBean. The default values will be used.
>
> Please see also section DWH synchronization.

H.5.4.4 Tasks

Tasks are used to transfer the data which is not transferred with the update operation because it is outside the time period covered by the update. Tasks are created automatically when needed because certain configuration changes were performed. They are saved in the ConSol CM table `cmas_dwh_task` and shown on the *Tasks* page of the DWH menu. If the DWH is running in admin mode, tasks need to be executed manually. In Live mode, they can be executed automatically if *Administrative changes are applied automatically* is selected.

When a task is executed, a new entry in `cmas_dwh_synchronization` is created with the type 'TASK' and both the DWH status and the CMRF status 'NEW'.

The tasks are processed in another thread by the DWH transfer service. Their execution status is visible on *Administration* page.

H.5.4.5 Enabling live mode

If the live mode is enabled, the required update actions are created automatically. If *Administrative changes are applied automatically* is selected on the *Tasks* page, the tasks are also executed automatically.

When you enable the live mode, the CM system property cmas-dwh-server, live.start is created and set to the current date. If an update is needed, a new entry in `cmas_dwh_synchronization` is created with the type 'UPDATE' and both the DWH status and the CMRF status 'NEW'. In addition, the `to_date` is set to the current date + the value of the property cmas-dwh-server, time.buffer and the retry count is set to 3.

The update is processed in another thread by the DWH transfer service.

H.5.5 Automatic operations

Automatic operations are performed by CM and by DWH services.

H.5.5.1 DWH transfer service

The DWH transfer service works in the same way in admin and in live mode. It reads entries from the CM table `cmas_dwh_synchronization` with `dwh_status` = “NEW” or “ACTIVE” and executes the respective jobs in the order of the timestamp of the entry (in ascending order).

The value of `dwh_status` is changed during processing from “NEW” to “ACTIVE” to “SUCCESS” or “ERROR”). It is set to “ACTIVE” when the entry is processed (only one entry can be active). If the processing is finished successfully, it is set to “SUCCESS”. Otherwise, it is set to “ERROR”.

During the active period, the DWH transfer service inserts serialized packages into the `INT_TRANSFER_QUEUE`. The package size can be set in the field *CM package size* when starting the transfer on the *Administration* page.

> ⓘ If this value is not set, e.g. because the job was created via the JMX operations, the CM system property cmas-dwh-server, batch-commit-interval is used (default 1000).

Processing of one entry:

1. The entry is read from `cmas_dwh_synchronization` (`dwh_status` = “ACTIVE” or “NEW”)
2. If `dwh_status` is “NEW”, it is set to “ACTIVE”
3. Data is sent to CMRF in packages.
4. Each package has a serial number.
   a. Sending of one package:
      i. A portion of data is read from the CM database (table `cmas_dwh_synchronization`).
      ii. The package is created.
      iii. The serialized package is inserted into the `INT_TRANSFER_QUEUE`. The value of the next serial number (`next_serial_number`) is incremented. The progress of the processing (`position`) is updated. These operations are done in one transaction.
5. Processing in CM is finished, `dwh_status` is updated to “SUCCESS” or “ERROR”.

Error handling:

After an exception, the processing will be continued from the point where it was stopped (`position`) if:

- the DWH transfer service is stopped
- a database failure is detected
- the exception is recoverable (CM system property cmas-dwh-server, recoverable.exceptions, see also FAQ about recoverable exceptions)
- retry is configured (`retry_count` > 0, 1 is used if not set)
- it is an error (e.g. OutOfMemoryError)

Otherwise the processing will not be continued (`dwh_status` = “ERROR”).

H.5.5.2 CMRF service

CMRF reads and processes TRANSFER data (types *INITIALIZATION*, *REINITIALIZATION*, *TRANSFER*, *UPDATE*, *TASK*), LIVE data and CONTROL data (*pause* and *resume*). TRANSFER data has a higher priority than LIVE data. LIVE data is processed only if there is no TRANSFER data available.

- TRANSFER data is read from `INT_TRANSFER_QUEUE`.
- LIVE data is read from `INT_LIVE_QUEUE`.
- CONTROL data is read from `INT_CONTROL_QUEUE`.

TRANSFER data

After the DWH transfer service has written entries into `INT_TRANSFER _QUEUE`, the CMRF service can process these entries. The processing of each initialization / reinitialization / transfer / update / task is split into multiple transactions. The current state of the processing is saved in the database at the end of each transaction.

The performance and memory usage can be influenced by the cache size parameter. The cache size can be set in the field *CMRF cache size* when starting the transfer on the *Administration* page.

> ⓘ If this value is not set, e.g. because the job was created via the JMX operations, the default value 10000 is used.

After restart the state of the processing is loaded from the database and the processing is continued.

A package (BLOB entry in the `data` column) of the `INT_TRANSFER_QUEUE` table contains the complete new / modified object (e.g. case, user, contact, resource).

Error handling:

After an exception the processing will be continued from the point where it was stopped if:

- a database failure is detected - in this case processing is paused
- the exception is recoverable (system property cmas-dwh-server, recoverable.exceptions)
- it is an error (e.g. OutOfMemoryError)
- retry is configured (`cmrf_retry_count` > 0, 1 is used if not set)

Otherwise, the processing will be not continued but the remaining packages are extracted from the table and discarded.

LIVE data

When the DWH live service has written entries into `INT_LIVE _QUEUE`, the CMRF service can process these entries.

Error handling:

The entry which caused the abort is logged in `hlp_transfer_errors` together with the error message.

CONTROL data

Pause and *resume* entries written by JMX into `INT_CONTROL_QUEUE` are read by the CMRF service. They pause and resume the CMRF service.

H.5.5.3 DWH log service

The DWH log service works in the same way in admin and live mode. It is responsible for providing log information. The service reads serialized packages from `INT_LOG_QUEUE` and writes them into the ConSol CM table `cmas_dwh_log`. This information is available on the *Administration* page of the Web Admin Suite by clicking the *Show logs* icon for an action. The log messages are retrieved from the `operation_message` column of the `cmas_dwh_log` table.

One iteration (transaction):

1. Log entries are read and deleted from the `INT_LOG_QUEUE`.
2. A new entry in `cmas_dwh_log` is created for each log entry.
3. The CM system property `last.success.live.timestamp` is updated for each LIVE log
4. `cmas_dwh_synchronization.cmrf_status` is updated for each log entry except for LIVE log.

> ⓘ If you need to find out more details about the possible cause of an error, you can increase the log level in the CM properties file `cm6-cmrf-config.properties` or `cmrf-config.properties`. See Configuring the log files for details.

H.5.5.4 DWH live service

The DWH live service controls the just-in-time DWH updates, i.e. it is responsible for sending information about changes to data field values in CM to CMRF. It is only active if the synchronization mode is set to *Live* in the DWH configuration (i.e. if the CM system property cmas-dwh-server, live.start is set) and if all data older than the date of the property `live.start` were already sent to CMRF. If the live mode is not enabled and there is no data in `cmas_dwh_ser_sync_object`, the property `live.start` is deleted.

If the live mode is enabled and CMRF needs to be informed about changes done in a CM transaction, CM writes one entry for each transaction into the table `cmas_dwh_ser_sync_object`. The entry is written at the end of the transaction, before the commit operation is executed. Each entry contains a serialized package which needs to be sent to CMRF. The DWH live service reads the entries from the table and inserts the respective serialized packages into the `INT_LIVE_QUEUE` in the DWH.

One iteration (transaction):

1. Packages are read from `cmas_dwh_ser_sync_object` (max. 100, in order of creation).
2. The property `live.serial.number` is read.
3. A serial number is set for each package.
4. The packages are deleted from `cmas_dwh_ser_sync_object`.
5. The property `live.serial.number` is updated.
6. The serialized packages are inserted into `INT_LIVE_QUEUE`.

The packages from `INT_LIVE_QUEUE` will be processed by DWH transfer service.

H.5.6 Operations via JMX

Some DWH-related operations can be executed via JMX.

ConSol CM:

- Start transfer and update. The CM package size, CMRF cache size and retry counts cannot be set using the MBean. The default values will be used. A new entry in `cmas_dwh_synchronization` is created

```
consol.cmas:type=admin,topic=global,name=dwh.synchronizationService
```

CMRF:

- Pause and resume processing in CMRF. A new entry in `INT_CONTROL_QUEUE` is created

```
consol.cmrf:name=cmrf.control
```

H.6 FAQs and tips for troubleshooting

This chapter discusses the following:

H.6.1 Can I remove DWH actions?

DWH actions can only be removed if the operation has not been started yet, i.e. have the status NEW. In this case, a Delete icon is shown in the row of the action on the *Administration* page. This is useful to remove an action, if you have created it accidentally.

The action entries on the *Administration* page are never removed, but you can filter them to locate a certain entry more easily.

On the *Tasks* page, the action entries are automatically removed once the processing has finished successfully.

H.6.2 In which order are tasks executed in live mode?

Tasks are created automatically in the `cmas_dwh_task` table, when there are configuration changes affecting the DWH, e.g. the setting *Transfer to the DWH* was applied to a data field. On the *Tasks* page, you can choose whether you want to apply administrative changes automatically or not. When a task is executed, a new entry is created in the `cmas_dwh_synchronization` table. The DWH transfer service reads the data from the table and writes chunks into `INT_TRANSFER_QUEUE` in the DWH. The CMRF service reads the data from `INT_TRANSFER_QUEUE` and `INT_LIVE_QUEUE`. Transfer data has a higher priority than the live data. Therefore, the new data which results from the configuration change is processed first, then the live data processing is continued.

H.6.3 How can I reduce the transfer and update duration?

When starting a transfer or update operation on the *Administration* page, you can provide appropriate advanced settings to improve performance.

The following settings are available:

- **CM package size**: Number of transferred objects per transaction from CM to the DWH control tables. Higher values mean better performance and bigger memory usage.
- **CM retry count**: Number of retries performed before the action is finished with the CM status *Error*.
- **Language**: Select one or several languages for localized names of entities in the DWH. This adds localizations to various DWH tables which are not localized by default.

> ⚠ Adding additional localizations may have a significant performance impact. Only select languages here if they are really needed for reporting. Consider that some entities, such as enum values, are localized by default.

- **CMRF cache size**: Number of transferred objects per transaction from the DWH control tables to the DWH tables.
- **CMRF retry count**: Number of retries performed before the action is finished with the CMRF status *Error*.
- **CMRF thread count**: Number of threads used to transfer the history data. The default value is 1, it can be increased to improve the transfer and update performance.

You can either modify these settings manually or choose one of the predefined configurations:

- **Default**: default settings providing a balance between safety and velocity
- **Safe**: settings to avoid out of memory errors, the action will take longer
- **Fast**: settings for fast execution, might cause out of memory errors
- **Last**: same settings as the last execution

H.6.4 How can I increase the log level of CMRF?

The log level is determined in the CMRF configuration file, `<JBOSS_CMRF_HOME>/standalone/configuration/cmrf-<DB>.xml`:

```
<logger category="com.consol.cmrf">
   <level name="${logger.com.consol.cmrf.level:INFO}"/>
   <handlers>
      <handler name="CMRF_FILE"/>
      <handler name="ERROR_FILE"/>
   </handlers>
</logger>
```

It can be changed in the CM/CMRF properties file `cm6-cmrf-config.properties` or `cmrf-config.properties`. For example, you can change the level from the default INFO to DEBUG to get more information for troubleshooting. Add an entry `logger.com.consol.cmrf.level=DEBUG` to the properties file. Please see Configuring the log files for details.

H.6.5 Why are updates missing although the live mode is active?

The latest changes to objects in ConSol CM are not visible instantaneously in the DWH because the live mode runs asynchronously. It can take several minutes or longer depending on the update volume until the updates are applied to the DWH database. In addition, the CMRF only checks for updates in certain intervals.

It you need live reports you must implement them using the production database. Live reports can be implemented in CM by using the detail search, case list or dashboard widgets.

H.6.6 How do I know if there are performance problems?

The following kinds of log entries indicate that the execution of SQL queries is especially slow:

- Check the `server.log` for slow CM queries:

```
2022-10-14 10:40:56,303 INFO [Thread-timerFactory]
 [rvice.impl.TransferServiceImpl] Still waiting on messages...
2022-10-14 10:55:56,394 INFO [Thread-timerFactory]
 [rvice.impl.TransferServiceImpl] Still waiting on messages...
```

- Check the `cmrf.log` for slow insert operations in the DWH:

```
2022-03-20 10:23:46,757 INFO [rvice.impl.TransferServiceImpl] Still
 processing unit history(estimated progress 0%, estimated remaining time
 40:23:10)...
2022-03-20 10:24:46,757 INFO [rvice.impl.TransferServiceImpl] Still
 processing unit history(estimated progress 0%, estimated remaining time
 41:23:10)...
```

Special case Oracle

SQL performance problems due to stale or missing table statistics can occur for large data transfers when the tables size is significantly extended or during the initial transfer when the DWH tables are empty. The update time can also be significantly degraded if the DWH is not updated regularly.

You can significantly reduce the duration of the actions by forcing an update of the table statistics on the whole CM and CMRF schemes. The following example command creates statistics for the schema `CM6`:

```
exec DBMS_STATS.GATHER_SCHEMA_STATS('CM6', OPTIONS => 'GATHER')
```

With the standard Oracles statistics setting, this creates histograms statistics which are needed for the specific DWH update SQLs, which allow the optimizer to find a better execution plan. You may need to repeat several times, as the different objects have different SQLs. If the data volume is very large, this can still take several hours.

H.6.7 How can I avoid that the transfer stops after an error?

Usually, the transfer stops when an error occurs. You can avoid this by defining recoverable exceptions which are ignored so that the transfer continues. You can provide a comma-separated list of the Java classes of the exceptions which should be ignored. Use + at the end of the class name to ignore all exceptions which inherit from the class.

Example: `java.lang.Exception+` . In this case, all exceptions will be ignored, because `java.lang.Exception` is the parent class of all exceptions in Java.

A regular expression can be provided as optional parameter. The regular expression is applied to the message of the exception as a filter.

> The following characters need to be escaped in the regular expression:
>
> - , -> \,
> - : -> \:
> - \ -> \\

Example: java.sql.SQLRecoverableException,java.lang.RuntimeException+:.*T.{1\,2}T.*

Please note that the value for recoverable exceptions has to be set at two locations:

- **CM**: in the system property cmas-dwh-server, recoverable.exceptions
- **CMRF**: in the start command of the application server

```
-
 Dcmrf.recoverable.exceptions=java.sql.SQLRecoverableException,java.lang.R
 untimeException+:.*T.{1\,2}T.*
```

In addition, you can also set a large number of retries.

H.6.8 What should I do when the live mode was aborted?

When the live mode was aborted, you will the an error message *An error occurred. Update is needed. Live packages will be processed again after update.* in the log files. In this case, you need to perform an update. All changes since the last successful live message will be retransmitted. This date is displayed on the *Monitor* page of the Web Admin Suite in the *Status* section (*Last Live message date*) and also available as epoch timestamp in the system property `last.success.live.timestamp` of the module `cmas-dwh-server`.

H.6.9 How should problems after a database connection failure be handled?

If the connection to the DWH database has been interrupted, it can happen that the live mode or DWH tasks do not work correctly anymore.

Please perform the following steps to restart the live mode:

- Restart the CMRF by either restarting the application server or restarting the service via MBean.
- Perform a DWH update to reenable the live mode.

If a DWH task does not update the DWH tables anymore, i.e. the CM progress works, but the CMRF progress is stuck and the task cannot be resumed on the Tasks page, you need to perform the following step to resume task processing

- Restart the CMRF by either restarting the application server or restarting the service via MBean.

H.6.10 What should I do after recovering the DWH from a backup?

When the DWH database had to be recovered from a full database backup, the log files will probably show errors and like *An error occurred. Update is needed. Live packages will be processed again after update.* and many warnings like *Update is needed. Live packages will be processed again after update.('from' date must be equal or lesser than X, 'to' date must be equal or greater than Y.*

If you want to restart the live mode without reinitializing the DWH and performing a full transfer, you need to perform the following step:

1. Execute the following command:

   ```
   update cmas_dwh_synchronization set cmrf_status='ERROR' where to_date > X;
   ```

2. Start an update on the *Administration* page.

H.6.11 What should I do if the DWH database size increases faster than CM database on Oracle?

When you initialize the DWH with the option to delete existing data, all existing DWH tables are deleted and recreated. Nevertheless, by default, Oracle does not delete tables but keeps them in the recycle bin. To free the data space you must purge the deleted tables from the recycle bin.

H.6.12 Why is data which was deleted on scene import still present in the DWH?

When you import a scene with the option to delete existing data, the data is deleted from the ConSol CM database but not from the DWH database. It is neither removed by the live mode nor by regular updates. If you want to remove the old data from the DWH database you need to reinitialize the DWH with the option to delete existing data and start a transfer afterwards.

H.6.13 Why did reports stop working?

DWH reports, which have been working before, now throw SQL errors or fields are missing or empty. In this case, you need to check if there were changes to the contact data model which affected the table structure in the DWH database, see chapter *ConSol CM DWH Dynamic Tables / Custom Data Models* of the *ConSol CM DWH Manual* in chapter *ConSol CM DWH Dynamic Tables / Custom Data Models*.

Additionally DWH tables structures may change due to updates of ConSol CM, see *ConSol CM Release Notes*.

I - Advanced operating topics

This section provides information regarding the advanced aspects of operating ConSol CM as well as topics affecting the overall CM system.

- Troubleshooting
- Data backup and recovery
- Operating ConSol CM behind a proxy
- Groovy sandbox
- System updates
- Development and release process

I.1 Troubleshooting

I.1.1 General checklist

The following checklist helps you to determine possible causes for errors in ConSol CM. Please keep in mind that there are a lot of collaborating systems (e.g., mail server, database server) which can be the source of the error, since CM is integrated into the IT landscape and is no standalone system.

1. Check the log files. The `server.log` file reports everything which happens in CM and should be the first entry point for troubleshooting. Maybe the reason for the error becomes obvious from this file. Search for `ERROR` in the log file.
2. Check the application server:
   a. Is the application server up and running? With enough RAM and CPU? Check the application server settings, see General settings.
   b. Does the application server have enough disc space? The Indexer needs enough disc space to be able to work correctly, see Monitoring the indexer.
3. Check the database server:
   a. Is the database server of the CM database up and running?
   b. Can CM connect without errors and without timeout to the database server?
4. Check the mail server:
   a. Is the mail server up and running?
   b. Is the email fetched by CM?
5. Check the LDAP server:
   a. Is the LDAP server up and running?
   b. Has there be a change in directory structure on the LDAP server?

I.1.2 Creating a support ticket

Please prepare the following data to create a support ticket:

- Description of the actions which lead to the error and background information:
  - Did you make changes to the ConSol CM configuration using the Web Admin Suite?
  - What did the end user do in the Web Client or CM/Track before the problem occurred?
- Log files:

  You can download the log files using the *Log files* page of the Web Admin Suite.
- Version and infrastructure details:
  - ConSol CM version
  - Server information: operating system, application server, database, Java version
  - Client information: operating system, web browser, Java version
  - Java version

I.2 Data backup and recovery

ConSol CM is a Java EE application with a classic three-tier architecture. Therefore, in general all standard mechanisms for data backup and restore of a Java EE application server with a database connection apply. Some specifics about ConSol CM are explained in this section.

You backup and recovery strategy needs to cover the following:

- **ConSol CM database**

  Back up your database using standard mechanisms.

- **DWH database**

  Large DWH databases should be backed up using standard mechanisms. Small DWH databases can be recreated from scratch by using the data transfer feature on the DWH *Administration* page of the Web Admin Suite.

  > ⚠ Be aware that reporting is not possible while the data transfer runs. Depending on the size of the DWH, the initial transfer operation required to recreate the database can take several days or even weeks.

- **ConSol CM configuration**

  Export a scenario containing all configuration data using the *Export* page of the Web Admin Suite and save it on a share which is included in the backup system. The ConSol CM system properties are not included in the scenario. You can either take a screenshot of the Web Admin Suite values or export the table `cmas_configuration` of the ConSol CM database.

- **ConSol CM index**

  All ConSol CM indices are stored on the file system. You need to follow the instructions described in Backup and recovery of the index

I.3 Operating ConSol CM behind a proxy

I.3.1 Introduction

You can operate ConSol CM behind a proxy server, e.g. Apache HTTPD. This might be required by security policies concerning system access over the internet. Two common scenarios are:

- The Web Client and CM/Track should be available through the proxy, but external access to the other clients should be blocked.
- All clients should be available through the proxy.

When using a proxy server, all URLs which are not white-listed explicitly in the proxy configuration are blocked. Therefore, you can decide for each client if it should be available externally via proxy or internally only.

> ⓘ ConSol CM comes with an example configuration for Apache HTTPD proxies. The example configuration has been created for Ubuntu 18. Please ask your consultant or the ConSol CM support for the files. The following section shows how to adapt this configuration. If another Linux distribution is used, additional changes might be needed.

I.3.2 Preconditions

The following preconditions apply for the example configuration:

- Apache HTTPD version 2.4
- The JBoss bind address (`-b` parameter) has been set to “127.0.0.1”, so it will only allow connections from the local machine.
- Apache HTTPD and JBoss reside on the same machine (applicable for most ConSol CM installations except for clustered environments).

I.3.3 Content of the example configuration

The example configuration consists of two folders, one for Linux and one for Windows. Each folder contains three files.

Linux:

- **cm.http.apache.conf**: Virtual host configuration for HTTP
- **cm.https.apache.conf**: Virtual host configuration for HTTPS
- **envvars**: Environment variables

Windows:

- **http-vhosts.conf**: Virtual host configuration for HTTP
- **https-vhosts.conf**: Virtual host configuration for HTTPS
- **httpd.conf**: Parameters and main configuration

The `envvars` / `httpd.conf` files contain the parameters which are used in the virtual host configuration. The following parameters are available:

- **SERVER_NAME**: The host name and port that the server uses to identify itself. Used to set the main address of the website, e.g. `cm6.apacheproxy.consol.pl`. Accepts port numbers.

- **SERVER_ALIAS**: Alternate names for a host. Used to add additional addresses, e.g. `www.cm6.apacheproxy.consol.pl`. Accepts wildcards, e.g. `*.cm6.apacheproxy.consol.pl`. Does not accept port numbers.
- **SERVER_ADMIN**: Email address of the administrator included in error messages to the client.
- **ERROR_LOG**: The name of the file to which the server logs the errors it encounters. If the file path is not absolute, it is assumed to be relative to the server root, e.g. `"/logs/error.log"`. Must be specified with quotes.
- **CUSTOM_LOG**: The name of the file to which the server logs the requests. If the file path is not absolute, it is assumed to be relative to the server root, e.g. `"/logs/error.log combined"`. Must be specified with quotes and the keyword “combined”.
- **SRVROOT**: Root path of the server, e.g. `"C:\\consol\apache\Apache24"` without the last `\`. Must be specified with quotes. Only for the Windows configuration.
- **CM6_BACKEND_SERVER**: Server host and port of the JBoss with the cm6.ear, e.g. `cm6.consol.pl:8009`. Connection via AJP. Used for the Web Client, Admin Tool, Process Designer and webhook interface.
- **WAS_BACKEND_SERVER**: Server host and port of the Web Admin Suite. Connection via AJP. Used for the Web Admin Suite.
- **ATMOSPHERE_BACKEND_SERVER**: Server host and port of Atmosphere. Connection via WS. Used for the notifications in the Web Client.
- **TRACK_BACKEND_SERVER**: Server host, port and path without the last `\`. There are two options:
  - CM/Track deployed in the same JBoss as ConSol CM: `track.consol.pl:8080/track`
  - CM/Track deployed on Tomcat: The path to backend server is different. You can register CM/Track as root path, e.g. `track.consol.pl:8080`.

  Connection via AJP. Used for CM/Track.
- **TRACK_PROXY**: Request name for CM/Track, e.g. `cm6.apache.consol.pl/track`. You need to specify the same request name like in the backend server, e.g. if the backend server is `track.consol.pl:8080/track`, set the `TRACK_PROXY` parameter to `track` (without `/`). Used for CM/Track.
- **TRACK_BACKEND_SERVER2**: Same as `TRACK_BACKEND_SERVER`. Uncomment to use a second CM/Track server. Copy to use more CM/Track servers. Connection via AJP.
- **TRACK_PROXY2**: Same as `TRACK_PROXY`. Uncomment to use a second CM/Track server. Copy to use more CM/Track servers.
- **ARCHIVE_BACKEND_SERVER**: Server host and port of CM/Archive. Connection by HTTP. Used for CM/Archive.
- **EBIA_BACKEND_SERVER**: Server host and port of CM/EBIA. Connection by HTTP. Used for CM/EBIA.
- **PATH_TO_CERTIFICATE_FILE**: Path to a file with certificate data in PEM format. Only needed for the HTTPS configuration.
- **PATH_TO_CERTIFICATE_KEY**: Path to the PEM-encoded private key file for the server. Only needed for the HTTPS configuration.

⚠ If you specify all parameters, the respective clients are available through the proxy. If this is not desired, you must comment out the respective parameters in the configuration file and in the virtual host configuration file for HTTP or HTTPS. The only exception are the Admin Tool and the Process Designer which are commented out in the virtual host configuration, so they are not available automatically when setting CM6_BACKEND_SERVER. If you want to expose the Admin Tool and Process Designer through the proxy, you need to comment in the respective lines.

I.3.4 Installing and configuring Apache HTTPD on Windows

1. Download and unzip Apache 2.4.x, e.g. from the Apache Haus.
2. Download and unzip the proxy distribution file provided by ConSol.
3. Copy the `httpd.conf` file from `proxy-distribution\windows` to `<APACHE_PROXY_HOME>\httpd-2.4.x\Apache24\conf\`
4. Copy the appropriate virtual host configuration file to `<APACHE_PROXY_HOME>\httpd-2.4.x\Apache24\conf\extra\`. Choose `http-vhosts.conf` for the HTTP configuration or `https-vhosts.conf` for the HTTPS configuration.
5. Open the `<APACHE_PROXY_HOME>\httpd-2.4.x\Apache24\conf\httpd.conf` file and fill out the properties in the *CM6 ENVIRONMENT VARIABLES* section. The paths must be provided in quotes.
6. Choose the HTTPS or HTTP configuration by specifying the respective virtual host file in `VHOST_FILE_NAME`. By default, the *https* section is commented in, so the HTTPS configuration is enabled. For HTTP configuration, comment out the *https* section and comment in the *http* section. For HTTPS, the SSL module is loaded by default and you need to set `PATH_TO_CERTYFIKATE_FILE` and `PATH_TO_CERTYFIKATE_KEY`.
7. Comment out all unneeded parameters in the `httpd.conf` and `https-vhosts.conf` files. For example, if the Web Admin Suite should not be available for external access, comment out the settings referring to the WAS backend server.
8. Remove the example definition of `VirtualHost _default_:${SSL_PORT}>` in the `<APACHE_PROXY_HOME>\httpd-2.4.x\Apache24\conf\extra\httpd-ssl.conf` file.
9. Start the Apache server:

```
<APACHE_PROXY_HOME>\Apache24\bin\httpd.exe
```

I.3.4.1 Installing Apache HTTPD as a Windows service

On Windows machines, you might want to run Apache HTTPD as a service. Execute the following command as an administrator:

```
<APACHE_PROXY_HOME>\Apache24\bin>httpd.exe -k install
```

Make sure that the start mode of this service is “automatic” (same as the JBoss service).

I.3.5 Installing and configuring Apache HTTPD on Linux

1. Install Apache 2.4.x, e.g. using ***apt-get***:

   ```
   sudo apt-get update
   sudo apt-get install apache2
   ```

2. Download and unzip the proxy distribution file provided by ConSol.
3. Copy the `envvars` file from `proxy-distribution/linux` to `/etc/apache2`
4. Copy the appropriate virtual host configuration file to `/etc/apache2/sites-available/`. Choose `cm.http.apache.conf` for the HTTP configuration or `cm.https.apache.conf` for the HTTPS configuration.
5. Rename the virtual host configuration file using the pattern: `<server name>.conf`, e.g. `cm.https.apache.conf` → `cm6.consol.conf`
6. Load all needed modules for a HTTP configuration:

   ```
   sudo a2enmod headers
   sudo a2enmod proxy
   sudo a2enmod proxy_ajp
   sudo a2enmod proxy_http
   sudo a2enmod proxy_wstunnel
   sudo a2enmod rewrite
   ```

   For HTTPS, also load the SSL module:

   ```
   sudo a2enmod ssl
   ```

7. Open the `/etc/apache2/envvars` file and fill out the properties in the *CM6 ENVIRONMENT VARIABLES* section. The paths must be provided in quotes.
8. Choose the HTTPS or HTTP configuration. By default, the *https* section is commented in, so the HTTPS configuration is enabled. For HTTP configuration, comment out the *https* section. For HTTPS you need to set `PATH_TO_CERTYFIKATE_FILE` and `PATH_TO_CERTYFIKATE_KEY`.
9. Comment out all unneeded parameters in the `envvars` and `cm.https.apache.conf` files. For example, if the Web Admin Suite should not be available for external access, comment out the settings referring to the WAS backend server.
10. Enable the virtual host:

    ```
    a2ensite cm6.consol.conf
    ```

11. Restart the Apache server:

    ```
    apache2ctl stop && apache2ctl start
    ```

or

```
sudo systemctl restart apache2.service
```

I.3.6 Advanced settings

On Windows, you can change the default thread stack size in the `<APACHE_PROXY_HOME>\httpd-2.4.x\Apache24\conf\extra\httpd-mpm.conf` file:

```
# WinNT MPM
# ThreadsPerChild: constant number of worker threads in the server process
# MaxConnectionsPerChild: maximum number of connections a server process serves
<IfModule mpm_winnt_module>
   ThreadsPerChild 150
   MaxConnectionsPerChild 0
   ThreadStackSize 8242880
</IfModule>
```

If there is a firewall between Apache and JBoss, you can set `disablereuse=on` to force `mod_proxy` to close connections to its backend after using it.

By default, AJP uses port `8009`. In order to change this port, add the variable `jboss.socket.binding.ajp.port` to your configuration file, e.g. `cm6-config.properties`, and specify the desired port:

```
# port overrides
jboss.socket.binding.ajp.port=8009
```

I.3.7 Additional settings for CM/Track and CM/EBIA

CM/Track:

- Enter the server proxy URL without any path in the property cmas-restapi-core, csrf.domain.white.list, e.g. “https://myserver.consol.de”.

CM/EBIA:

- Log in to the CM/EBIA administration as an administrator and enter the EBIA proxy URL, e.g. “https://myserver.consol.de/ebia”, in the following settings:
  - Admin -> General -> Site URL
  - Admin -> Embedding in other applications -> Embedding the entire CM/EBIA app
- Enter the EBIA proxy URL in the property cmas-analytics, metabase.url, e.g. “https://myserver.consol.de/ebia”

I.4 Groovy sandbox

Groovy scripts are now executed in a sandbox for security reasons. The sandbox can be configured using Java system properties which can be set in the start scripts of the application server, see Starting and stopping the CM server.

The following Java system properties are available:

- **cm6.groovy.sandbox.enabled**
  Determines if the sandbox is enabled ("true", default value) or disabled ("false").

  ```
  -Dcm6.groovy.sandbox.enabled=false
  ```

- **cm6.groovy.sandbox.blacklists**
  Determines which predefined list of patterns is used for blocking methods:

  - **command**: blocks the execution of shell commands
  - **filesystem**: blocks access to the file system

  The default value is "command". Both lists of patterns block bypassing the sandbox. You can configure both lists of patterns as a comma-separated list.

  ```
  -Dcm6.groovy.sandbox.blacklists=command,filesystem
  ```

- **cm6.groovy.sandbox.whitelist.regex**
  Optional. Regular expression for whitelisted API calls. Takes precedence over the predefined blacklist.

- **cm6.groovy.sandbox.blacklist.regex**
  Optional. Regular expression for blacklisted API calls. Takes precedence over the custom whitelist and the predefined blacklist.

- **cm6.groovy.sandbox.cache.size**
  Determines the number of results of pattern matching which are cached (only for predefined backlists). The default value is 10000.

  ```
  -Dcm6.groovy.sandbox.cache.size=1000
  ```

- **cm6.groovy.sandbox.statistics.invocations.threshold**
  Determines the number of method invocations which needs to be exceeded for a warning to be written to the log files. The default value is 100000.

  ```
  -Dcm6.groovy.sandbox.statistics.invocations.threshold=1000
  ```

- **cm6.groovy.sandbox.statistics.details.enabled**
  Determines if the logging of additional details about method execution is enabled ("true") or disabled ("false", default value). The statistics shows the most frequently invoked and most time-consuming methods. By default, this feature is disabled as it might impact performance.

  ```
  -Dcm6.groovy.sandbox.statistics.details.enabled=true
  ```

I.4.1 Syntax to whitelist a method

The following example shows a piece of code which causes an exception in the default configuration, because all method invocations on `freemarker.template.Template` are blocked by default:

```
import freemarker.template.Template

def onInitialize(taskDescriptor) {}
def onExecute(taskDescriptor) {
   Template template = new Template('template', '${firstname} ${lastname}\n',
    null)
   template.process([lastname: 'Smith', firstname: 'John'], new
    java.io.OutputStreamWriter(System.out))
}
def onError(taskDescriptor) {}
def onCancel(taskDescriptor) {}
```

This causes the following exception:

```
com.consol.cmas.common.util.security.groovy.sandbox.GroovySandboxException:
 Method <init> in class freemarker.template.Template cannot be executed in
 sandbox mode
```

You can whitelist the affected method using the following syntax:

```
-Dcm6.groovy.sandbox.whitelist.regex=freemarker[.]template[.]Template#.*
```

I.5 System updates

The following process applies to regular installations, i.e. systems which have been customized using the capabilities of the Web Admin Suite and Web Client. These systems can be installed and updated by deploying the standard EAR and WAR files included in the ConSol CM distribution.

> ⚠ If your system requires modified EAR and WAR files, you cannot update your system by yourself, but need assistance by a ConSol CM consultant.

Please perform the following steps to install an update:

1. Read the release notes for all versions between your current version and the target version to learn about the included new features, changes and bugfixes. Most importantly, you need to check the *Breaking changes and mandatory actions* section to find out if there are any configuration changes which you must perform to have a working system after the update.
2. Obtain the installation files from the ConSol CM support. You need the following files:
    - **CM .ear file**: dist-package-ear-<CM_VERSION>.ear, e.g. `dist-package-ear-6.16.0.1.ear`
    - **CMRF .war file** (only if the DWH is used and the CMRF is installed on the same JBoss server): cmrf-application-package-war-<CM_VERSION>.war, e.g. `cmrf-application-package-war-6.16.0.1.war`
    - **CM/Track .war file** (only if the CM/Track is used): cmtrack-v2-distribution-<CM_VERSION.war>, e.g., `cmtrack-v2-distribution-6.16.0.1.war`
    - **ETL .zip file** (only if the ETL is used): etl-package-distribution-<CM_VERSION>-kettle.zip, e.g., `etl-package-distribution-6.16.0.1-kettle.zip`
3. Shut down the ConSol CM server.
4. Perform the mandatory steps described in the release notes, if any.
5. Deploy the new EAR and WAR files in the appropriate directory of the application server.
6. Restart the ConSol CM server.
7. If the CMRF is installed as a standalone application, replace the installation file and restart the CMRF.

I.6 Development and release process

Developing and running a ConSol CM system involves at least two ConSol CM server installations because the production system should not be used for developing and testing configuration changes. The following types of system are commonly used:

- **Development system (dev)**

  System used to develop new features and adapt the configuration to new requirements.

- **Test system (test)**

  System used to test the changes made in the development system.

- **Staging system (stage)**

  System used for final tests of the changes, especially for changes involving external interfaces and interactions with other systems.

- **Production system (prod)**

  System used by the end users for their daily work.

The system type can be indicated in the system property cmas-core-shared, system.flavour to show a visual indication of the system type in the Web Admin Suite and enable additional features for development systems.

Each system is completely independent, on its own machine with its own database and application server.

The configuration is transferred from one system to another using a scene. The scene is created on the *Export* page of the Web Admin Suite of the source system, and imported on the target system using the *Import* page, see section *Staging* of the *ConSol CM Administrator Manual*.

The following actions are performed during the development cycle:

1. Transfer the configuration of the production system to the development system.
2. Implement your changes on the development system.
3. When development is finished, transfer the configuration of the development system to the test system.
4. Test your changes on the test system. If the tests reveal the need for further changes, repeat the steps 2 to 4.
5. Once the changes have been tested positive and approved, transfer the configuration of the test system to the stage system.
6. Test the interaction of the ConSol CM system with external systems, e.g. via ETL transformations, data imports or webhook interfaces.
7. Transfer the configuration of the staging system to the production system to release the new configuration.

Please follow these best practices:

- All systems need to run the same ConSol CM version. In addition, the database, application server and Java versions should match.
- Always perform all changes on the development system and use scenes to transfer them to the other systems. Performing changes on stage or production systems might lead to undesired side effects, even for small changes.

Example: A new user is created directly on the production system. If the user is created later on the development system, he will have a different transfer key, which causes the user to be duplicated during the next scene import.

- Scene exports should not contain runtime data except for test systems.
- Each system needs its own mailboxes. You must ensure that only the production system sends emails to real persons who do not belong to the team which develops and tests the changes.
- Before releasing changes, you should test them on the staging system under real conditions, e.g. with the real amount of data and number of concurrent users. If possible, transfer a snapshot of the runtime data from the production system to the stage system using ETL.

J - Appendix

This section contains several appendices:

- List of important files
- List of system properties
- Email addresses
- List of manuals
- Legal notice
- Trademarks
- Glossary

J.1 List of important files

The following files are important for the ConSol CM installation and are explained in this section in some more detail.

- cm6-config.properties
  Used to configure ConSol CM with or without CMRF
- cmrf.properties
  Used to configure CMRF in standalone mode
- cm6.xml
  Contains the configuration of ConSol CM without CMRF, does not need to be edited
- cm6-dwh.xml
  Contains the configuration of ConSol CM with CMRF, does not need to be edited
- standalone.conf
  Used to configure environment variables in JBoss on Linux
- standalone.conf.bat
  Used to configure environment variables in JBoss on Windows

J.1.1 cm6-config.properties

The `cm6-config.properties` file is used to configure the database connection and logging. The configuration of the database connection is mandatory. You can configure logging parameters to override the default settings from `cm6.xml` (see Configuring the log files).

The following example shows the configuration file with the default settings.

```
# Used to override configuration properties in the following files
 ../standalone/configuration/ cm6-<db>.xml

# Example urls for database connection
# mysql 5: jdbc:mysql://localhost/cmdatabase?serverTimezone=Europe/Berlin
# mysql 8: jdbc:mysql://localhost/cmdatabase?characterEncoding=UTF-
 8&serverTimezone=Europe/Berlin
# oracle: jdbc:oracle:thin:@localhost:1521/SERVICE_NAME
# mssql: jdbc:sqlserver://localhost:1433;databaseName=cmdatabase

# non-transactional cmas data source url, credentials
datasource.url=jdbc:mysql://localhost/cmdatabase?serverTimezone=Europe/Berlin
datasource.username=cmuser
datasource.password=consol

# transactional cmas data source url, credentials
xa.datasource.url=jdbc:mysql://localhost/cmdatabase?serverTimezone=Europe/Berli
 n
xa.datasource.username=cmuser
xa.datasource.password=consol

# transactional cmrf data source url, credentials
# uncomment and fill only for cm6 with dwh usage
#
 cmrf.xa.datasource.url=jdbc:mysql://localhost/cmrfdatabase?serverTimezone=Euro
 pe/Berlin
# cmrf.xa.datasource.username=cmrf
# cmrf.xa.datasource.password=consol

# container http settings
# set max-post-size to the largest value (in bytes) of binary http uploads
 expected on this system. Typically the CM setting attachment.max.size drives
 this property.
max-post-size-bytes=104857600

# enforce that cookies are only accepted via https which makes unencrypted
 logins impossible except via localhost. Strongly recommended for prod systems.
enforce.secure.cookies=false
```

J.1.2 cmrf.properties

The `cmrf.properties` file is used to configure the database connection of CMRF when running ConSol CM with CMRF in standalone mode. The configuration of the database connection is mandatory. Optionally, you can configure logging parameters to override the default settings.

The following example shows the configuration file with the default settings. You need to uncomment the line with your database and provide the correct URL, user name and password.

```
#Configuration file example (this file should be put in the cmrf.jar launch
 location)

# Datasource URL (uncomment and alter the suitable one):

## MySQL
#spring.datasource.url=jdbc:mysql://localhost/cmrf

## SQL Server
#spring.datasource.url=jdbc:sqlserver://localhost;databaseName=cmrf

## Oracle
#spring.datasource.url=jdbc:oracle:thin:@localhost:1521/SERVICE_NAME

## PostgreSQL
#spring.datasource.url=jdbc:postgresql://localhost/cmrf


# Datasource credentials:
spring.datasource.username=cmrf
spring.datasource.password=consol
```

J.1.3 cm6.xml

This file is used for ConSol CM installations without CMRF. The ConSol CM distribution includes a configuration file for each database:

- `cm6-mysql-5.xml` for MySQL 5
- `cm6-mysql-8.xml` for MySQL 8
- `cm6-oracle.xml` for Oracle
- `cm6-mssql.xml` for Microsoft SQL Server
- `cm6-postgresql.xml` for PostgreSQL

This file contains the configuration and default settings for running ConSol CM. You do **not** need to edit this file. All the installation-specific settings are made in the properties file `cm6-config.properties` (see cm6-config.properties). If you want to override default settings, you might need to refer to the configuration file for looking up the names of variables (see Configuring the log files).

J.1.4 cm6-dwh.xml

This file is used for ConSol CM installations with CMRF. The ConSol CM distribution includes a configuration file for each database:

- `cm6-dwh-mysql-5.xml` for MySQL 5
- `cm6-dwh-mysql-8.xml` for MySQL 8
- `cm6-dwh-oracle.xml` for Oracle
- `cm6-dwh-mssql.xml` for Microsoft SQL Server
- `cm6-dwh-postgresql.xml` for PostgreSQL

This file contains the configuration and default settings for running ConSol CM with CMRF. You do **not** need to edit this file. All the installation-specific settings are made in the properties file `cm6-config.properties` (see cm6-config.properties). If you want to override default settings, you might need to refer to the configuration file for looking up the names of variables (see Configuring the log files).

J.1.5 standalone.conf

`standalone.conf` is a JBoss configuration file for the environment variables on Linux. The content of the file depends on the used application server. You should always use the file from the JBoss distribution which you are installing.

You need to edit `standalone.conf` to enter the following information:

- Java memory options
- Adjust serialization settings (only needed for JBoss > 7.4.12)

J.1.6 standalone.conf.bat

`standalone.conf.bat` is a JBoss configuration file for the environment variables on Windows. The content of the file depends on the used application server. You should always use the file from the JBoss distribution which you are installing.

You need to edit `standalone.conf.bat` to enter the following information:

- Java memory options
- Adjust serialization settings (only needed for JBoss > 7.4.12)

J.2 List of system properties

The following list contains all system properties. If a system property is not present in your ConSol CM system, you can add it manually. Placeholders within the property name are indicated by the < and > characters.

access.token.signing.key

- **Module**: cmas-auth-server
- **Description**: Determines the secret shared between the authorization server and all client applications using OAuth2, e.g. CM/Archive. The value of this property needs to match the value of *archive.oauth2.access.token.signing.key* in the configuration file of CM/Archive.
- **Type**: password
- **Restart required**: yes
- **System**: no
- **Optional**: no
- **Example value**: my_password
- **Since**: 6.11.2.0

admin.email

- **Module**: cmas-core-security
- **Description**: The email address of the ConSol CM administrator. The value which you entered during system setup is used initially.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Since**: 6.0

admin.login

- **Module**: cmas-core-security
- **Description**: The login name of the ConSol CM administrator. The value which you entered during system setup is used initially.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: admin
- **Since**: 6.0

admin.tool.consumed.licences.check.interval

- **Module**: cmas-app-admin-tool
- **Description**: Sets the interval (in seconds) to monitor the number of consumed licenses.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 30 (default value)
- **Since**: 6.11.0.0
- **Removed in**: 6.16.0.0

admin.tool.consumed.licences.pool.name

- **Module**: cmas-app-admin-tool
- **Description**: Sets the license pool name to monitor the number of consumed licenses.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: CONCURRENT_USERS (default value)
- **Since**: 6.11.0.0
- **Removed in**: 6.16.0.0

admin.tool.session.check.interval

- **Module**: cmas-app-admin-tool
- **Description**: Configures the time interval (in seconds) in which the system checks for inactive (ended) Admin Tool sessions.
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: 30
- **Since**: 6.7.5
- **Removed in**: 6.16.0.0

archive.enabled

- **Module**: cmas-archive-core-server
- **Description**: Determines if GDPR-compliant deletions or anonymizations, which delete case or contact data from the ConSol CM database, are applied to CM/Archive as well. By default, this property is set to “false”, meaning that no data is deleted from CM/Archive. If you set this property to “true”, GDPR-compliant deletions or anonymizations are also performed in CM/Archive. This affects both already archived cases and newly archived cases.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.11.2.5

archive.uri

- **Module**: cmas-archive-core-server
- **Description**: Sets the URL where the CM/Archive application is running.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: <server_url>:8090
- **Since**: 6.11.2.0

attachment.allowed.types

- **Module**: cmas-core-server
- **Description**: Comma-separated list of allowed attachment file types. By default, only images, PDF and text files are allowed. If no value is provided, all files are allowed (not recommended). The property mail.attachments.validation.handling defines how the system reacts on an unallowed attachment.

> ⓘ The property is not considered for attachments added with CM/Doc, i.e. it is not required to allow docx as a file extension for attachments in order to use CM/Doc.

- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: jpg,jpeg,gif,bmp,png,pdf,txt,csv
- **Since**: 6.5.0, modified in 6.13.0.8, 6.15.3.0

attachment.max.size

- **Module**: cmas-core-server
- **Description**: Maximum attachment size, in MB. This property controls the size of attachments at cases, contacts and resources. It also controls the size of incoming (not outgoing) email attachments. The value of this property needs to be aligned with the respective setting in the application server configuration (adapt the setting `max-post-size-bytes` in `cm6-config.properties`, `cm6-cmrf-config.properties` and `cm6-domain-config.properties`). In addition you might need to increase the value of `cmas-nimh`, `mailbox.default.task.max.message.size`. The value of this property is displayed in the Web Client when adding a new attachment.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 100 (default value)
- **Since**: 6.4.0

attachment.type.hints

- **Module**: cmas-core-server
- **Description**: Allows you to assign MIME types to not yet officially supported file extensions, so these file extensions can be detected correctly.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 7z:application/x-7z-compressed, zip:application/zip, wav:audio/x-wav
- **Since**: 6.11.2.0

attachment.upload.timeout

- **Module**: cmweb-server-adapter
- **Description**: Defines the transaction timeout in minutes for adding attachments to a case, contact or resource. Counts the time for the upload of all attachments of one transaction. When the timeout occurs, all files which have been temporarily stored on the server are deleted and no file is uploaded.
- **Type**: Integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 3
- **Since**: 6.10.5.3

authentication.method

- **Module**: cmas-core-security
- **Description**: Sets the user authentication method (internal database or LDAP authentication). Allowed values are LDAP or DATABASE.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: DATABASE
- **Since**: 6.0
- **Removed in**: 6.14.0.0

autocommit.cf.changes

- **Module**: cmas-dwh-server
- **Description**: Defines whether DWH tasks which result from configurational changes on case fields are executed automatically without manual interaction in the Web Admin Suite. Can be also set in the Web Admin Suite on the DWH configuration page. The default and recommended value is “false”.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false (default value)
- **Since**: 6.7.0

autocomplete.enabled

- **Module**: cmas-app-admin-tool
- **Description**: Controls if the *Autocomplete address* navigation item is displayed in the Admin Tool. It is hidden, if this system property is missing or its value is “false”.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: true
- **Since**: 6.9.2.0
- **Removed in**: 6.16.0.0

automatic.booking.enabled

- **Module**: cmweb-server-adapter
- **Description**: If enabled, time spent on creating comments or emails is measured and automatic time bookings are added.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: true
- **Since**: 6.9.4.2

batch-commit-interval

- **Module**: cmas-dwh-server
- **Description**: Number of objects sent in one DWH message. Larger values mean better transfer performance at the cost of higher memory usage. This property is only used if the package size of a DWH operation is not set. This can only happen when the command is directly addressed to the Java MBean `consol.cmas.global.dwh.synchronizationService`, e.g. using the `update()` method. When a DWH operation is started using the Web Admin Suite, there is always a value for the package size. If not explicitly set, the default value of 1000 is used as value for `batch.commit.interval`.
- **Default value**: 1000
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 100
- **Since**: 6.0.0

big.task.minimum.size

- **Module**: cmas-core-index-common
- **Description**: Indicates the minimum size of index task (in parts, each part has 100 entities) to qualify this task as a big one. Big tasks have a lower priority than runtime tasks.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 15 (default value)
- **Since**: 6.8.3

brute.force.admin.auth.failures.notify.amount

- **Module**: cmas-core-security
- **Description**: Defines the number of unsuccessful authentication attempts after which the administrator will be notified. The default value is 100.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 100
- **Since**: 6.14.0

brute.force.admin.auth.failures.notify.period

- **Module**: cmas-core-security
- **Description**: Defines the period in minutes for checking for failed authentication attempts to inform the administrator. If more failed authentication attempts than defined in `brute.force.admin.auth.failures.notify.amount` occur in this timeframe, the administrator is notified about this via email. The default value is 60.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 60
- **Since**: 6.14.0

brute.force.attempts.to.block

- **Module**: cmas-core-security
- **Description**: Defines the number of unsuccessful authentication attempts after which an account will be blocked. The default value is 7.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 7
- **Since**: 6.14.0

brute.force.auto.unblock.active

- **Module**: cmas-core-security
- **Description**: Determines if the mechanism to automatically unblock an account after a certain time period is enabled. The default value is “true”.

- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.14.0

brute.force.auto.unblock.period

- **Module**: cmas-core-security
- **Description**: Defines the period in minutes after which an account is automatically unblocked. The default value is 30.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 30
- **Since**: 6.14.0

brute.force.endpoint.blocking.active

- **Module**: cmas-core-security
- **Description**: Determines if the authentication protection feature is enabled for webhooks. The default value is “true”.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.14.0

brute.force.engineer.blocking.active

- **Module**: cmas-core-security
- **Description**: Determines if the authentication protection feature is enabled for users. The default value is “true”.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no

- **Example value**: true
- **Since**: 6.14.0

brute.force.mail.notification.active

- **Module**: cmas-core-security
- **Description**: Determines if users and contacts who were blocked receive an email with a link to unblock their accounts. The administrator receives emails with information about blocked accounts and the unblock codes for these accounts. In addition, the administrator receives alerts if the number of failed attempts exceeds the value set in `brute.force.admin.auth.failures.notify.amount`. The default value is “false”.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.14.0

brute.force.period.between.attempts.to.block

- **Module**: cmas-core-security
- **Description**: Defines the period in minutes for checking for failed authentication attempts. If more failed authentication attempts than defined in `brute.force.attempts.to.block` occur in this timeframe, the user or endpoint is blocked. The default value is “1”.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1
- **Since**: 6.14.0

brute.force.unblock.code.expiration.period

- **Module**: cmas-core-security
- **Description**: Defines the period in hours after which the unblock code will expire. The default value is “24”.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 24
- **Since**: 6.14.0

brute.force.unit.blocking.active

- **Module**: cmas-core-security
- **Description**: Determines if the authentication protection feature is enabled for contacts / REST users. The default value is “true”.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.14.0

cache-cluster-name

- **Module**: cmas-core-cache
- **Description**: Unique name of a cluster consisting of several CM nodes. This value is used to identify the nodes belonging to the same cluster during network communication. Please do not change this value manually unless advised by ConSol.
- **Type**: string
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: 635a6de1-629a-4129-8299-2d98633310f0
- **Since**: 6.4.0

cache.logging.listener.enabled

- **Module**: cmas-core-cache
- **Description**: Determines whether cache events are logged to `infinispan.log` for the standalone mode and `jgroups-infinispan.log` for the cluster mode.
- **Type**: boolean
- **Restart required**: yes
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.12.0.5

calendar.csv.dateFormat

- **Module**: cmas-core-server
- **Description**: Format of the date given in the CSV file containing the list of holidays.
- **Type**: string

- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: dd/MM/yyyy
- **Since**: 6.9.3.2

calendar.csv.separator

- **Module**: cmas-core-server
- **Description**: Separator used in the CSV file containing the list of holidays.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: ,
- **Since**: 6.9.3.2

checkUserOnlineIntervalInSeconds

- **Module**: cmweb-server-adapter
- **Description**: The interval in seconds to check which users are online (default 180sec = 3min).
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 180
- **Since**: 6.0
- **Removed in**: 6.5 / 6.11.0.1

client.archive.access.token.validity.seconds

- **Module**: cmas-auth-server
- **Description**: Determines the validity period in seconds of the access token required for using CM/Archive.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 43200 (default value)
- **Since**: 6.11.2.0

client.archive.enabled

- **Module**: cmas-auth-server
- **Description**: Determines if CM/Archive is enabled (“true”) or disabled (“false”).
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: false
- **Since**: 6.11.2.0
- **Removed in**: 6.11.2.2

client.archive.refresh.token.validity.seconds

- **Module**: cmas-auth-server
- **Description**: Determines the validity period in seconds of the refresh token required for obtaining new access tokens for CM/Archive.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 2592000 (default value)
- **Since**: 6.11.2.0

client.archive.secret

- **Module**: cmas-auth-server
- **Description**: Determines the secret shared between the authorization server and CM/Archive. The value of the property needs to match the value of *archive.oauth2.client.secret* in the configuration file of CM/Archive.
- **Type**: password
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: my_password
- **Since**: 6.11.2.0

client.was.access.token.validity.seconds

- **Module**: cmas-auth-server
- **Description**: Determines the validity period in seconds of the access token required for using the Web Admin Suite.

- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 43200 (default value)
- **Since**: 6.12.0.0

client.was.refresh.token.validity.seconds

- **Module**: cmas-auth-server
- **Description**: Determines the validity period in seconds of the refresh token required for obtaining new access tokens for the Web Admin Suite.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 2592000 (default value)
- **Since**: 6.12.0.0

client.was.secret

- **Module**: cmas-auth-server
- **Description**: Determines the secret shared between the authorization server and the Web Admin Suite.
- **Type**: password
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: my_password
- **Since**: 6.12.0.0
- **Removed in**: 6.12.0.1

cluster.mode

- **Module**: cmas-core-shared
- **Description**: Specifies whether ConSol CM is running in a cluster.
- **Type**: boolean
- **Restart required**: yes
- **System**: yes
- **Optional**: no

- **Example value**: false
- **Since**: 6.1.0

cluster.unicast

- **Module**: cmas-core-shared
- **Description**: Flag to activate the jgroups unicast mode for ConSol CM clusters (as opposed to the default multicast mode causing problems in some data center environments). If set to "true", remember to set the JVM start parameters: `jgroups.bind.port`, `jgroups.bind.address` and `jgroups.initial_hosts`.
- **Type**: boolean
- **Restart required**: yes
- **System**: yes
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.11.0.0
- **Removed in**: 6.12.0.0

cmas.dropSchemaBeforeSetup

- **Module**: cmas-setup-hibernate
- **Description**: Determines if an existing database schema should should be dropped during setup. This value is set on the setup page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.0

cmoffice.enabled

- **Module**: cmweb-server-adapter
- **Description**: Determines if CM/Doc is enabled.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.4.0

cmoffice.oo.path.NUMBER

- **Module**: cmweb-server-adapter
- **Description**: Possible location of the OpenOffice installation. The properties are numbered starting with 0.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: c:\Program Files (x86)\LibreOffice 3.6\program
- **Since**: 6.10.1.0
- **Removed in**: 6.12.0.0

cmoffice.strict.versioning.enabled

- **Module**: cmweb-server-adapter
- **Description**: Controls if the SAVE operation in Microsoft Word / OpenOffice documents creates a new attachment (“true”) or overwrites the existing attachment (“false”). This concerns the behavior within one session using the text editing program. If the program is stopped, the overwrite mechanism will not work anymore.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true
- **Since**: 6.10.5.4

cmoffice.websocket.port

- **Module**: cmweb-server-adapter
- **Description**: Determines the port on which the CM/Doc application should be started and connected to.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 30333
- **Since**: 6.12.0.0

comment.authors.disabled

- **Module**: cmas-restapi-core
- **Description**: Disables the display of the content author via REST API.

- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.11.0

commentRequiredForTicketCreation

- **Module**: cmweb-server-adapter
- **Description**: Determines if the comment is a required field for case creation.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true (default value)
- **Since**: 6.2.0

communication.channel

- **Module**: cmas-dwh-server
- **Description**: Communication channel. Only possible value since CM version 6.11.0.0: DIRECT
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: DIRECT
- **Since**: 6.8.5.0
- **Removed in**: 6.11.0.0 (DIRECT mode is the only available mode and is set automatically)

config.data.version

- **Module**: cmas-core-server
- **Description**: The internal version number of the current system configuration. This property is maintained internally, please do not change it unless advised by ConSol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 11
- **Since**: 6.0

config.import.global.transaction.enabled

- **Module**: cmas-core-server
- **Description**: Determines whether the configuration (without localizations) should be imported within a single transaction.
- **Type**: Boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.11.1.0

connection.release.mode

- **Module**: cmas-setup-hibernate
- **Description**: Describes the JEE connection handling strategy for transactions. If set to “AFTER_TRANSACTION”, the connection will be cached during the transaction and released at the end. If set to “AFTER_STATEMENT”, the connection will be released to the pool after each statement execution. Please do not change the default here unless advised by ConSol.
- **Type**: string
- **Restart required**: yes
- **System**: no
- **Optional**: yes
- **Example value**: AFTER_STATEMENT (default for JEE environment)
- **Since**: 6.0

contact.authentication.method

- **Module**: cmas-core-security
- **Description**: Sets the authentication method for contacts accessing REST services, possible values are DATABASE or LDAP or LDAP,DATABASE or DATABASE,LDAP.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: LDAP
- **Since**: 6.9.3.0

contact.inherit.permissions.only.to.own.customer.group

- **Module**: cmas-core-security
- **Description**: Determines whether an authenticated contact via REST inherits all customer group permissions from the CM/Track user profile (“false”) or only has permissions to his own customer group (“true”).
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.9.2.3

csrf.domain.allow.none

- **Module**: cmas-restapi-core
- **Description**: Determines whether empty `Origin/Referer` headers are accepted. By default, the property is set to “false”, so that REST requests without `Origin/Referer` headers are blocked.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false (default value)
- **Since**: 6.11.2.2

csrf.domain.white.list

- **Module**: cmas-restapi-core
- **Description**: The list of domains (format protocol://domain:port, separated with “|”) which are allowed in `Origin/Referer` headers and will not be blocked by the CSRF filter. By default, the property is empty so that cross-site requests are blocked.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: https://example.com:80 | http://www.consol.de:8080
- **Since**: 6.11.2.2

csrf.domain.white.list

- **Module**: cmweb-server-adapter
- **Description**: The list of domains (separated with "|") which are allowed and will not be blocked by the CSRF (cross-site request forgery) filter, e.g.: "example.com | consol.de"
- **Type**: String
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: example.com | consol.de
- **Since**: 6.10.7.0

csrf.request.filter.enabled

- **Module**: cmas-restapi-core
- **Description**: Determines whether the CSRF request filter (Origin/Referer) for the REST API is enabled.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.11.2.2

csrf.request.filter.enabled

- **Module**: cmweb-server-adapter
- **Description**: It allows to disable the CSRF request filter, which is enabled by default.
- **Type**: Boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true
- **Since**: 6.10.7.0

customfield.content.file.max.size

- **Module**: cmas-core-server
- **Description**: Determines the maximum size in MB of the images which can be added as content of rich text fields.
- **Type**: integer
- **Restart required**: no

- **System**: yes
- **Optional**: yes
- **Example value**: 10
- **Since**: 6.11.2.2

customizationVersion

- **Module**: cmweb-server-adapter
- **Description**: UID representing the latest web customization version. Used only internally, please do not change the value.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: cd58453e-f3cc-4538-8030-d15e8796a4a7
- **Since**: 6.5.0

dao.log.threshold.milliseconds

- **Module**: cmas-core-server
- **Description**: Used to configure database operation times logging. DAO methods whose execution takes longer than the time set in this property (in milliseconds) are logged.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 500 (default value)
- **Since**: 6.11.1.0

dao.log.username

- **Module**: cmas-core-server
- **Description**: Used to configure database operation times logging. The execution of DAO methods which are related to the user name stated in this property is logged. Only one user name can be provided. The value is empty by default.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: webadmin
- **Since**: 6.11.1.0

data.directory

- **Module**: cmas-core-shared
- **Description**: Global directory for ConSol CM data (e.g., index).
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: C:\Users\user\cmas
- **Since**: 6.0

data.optimization

- **Module**: cmweb-server-adapter
- **Description**: Defines optimization to be applied on HTML response data. The following values are supported (for setting more than one value, separate values by '|'): MINIFICATION and COMPRESSION. MINIFICATION minifies HTML data by e.g. stripping whitespaces and comments. COMPRESSION applies gzip compression to HTTP response. (Note: If you are running in cluster mode and want to test different configurations in parallel, you can set different values for each cluster node by specifying the property `data.optimization.nodeId` to override the default property.)
- **Type**: string
- **Restart required**: COMPRESSION can be switched on/off without restart, MINIFICATION requires restart.
- **System**: yes
- **Optional**: yes
- **Example value**: MINIFICATION|COMPRESSION

database.notification.enabled

- **Module**: cmas-core-index-common
- **Description**: Indicates whether the database should be used as a notification channel for index updates instead of JMS. The default value is “true.”
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.8.4.7
- **Removed in**: 6.12.0.0

database.notification.redelivery.delay.seconds

- **Module**: cmas-core-index-common
- **Description**: If the database is used as a notification channel for index updates, this property indicates the delay for notification redelivery when an exception occurs.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 60
- **Since**: 6.8.4.7

database.notification.redelivery.max.attempts

- **Module**: cmas-core-index-common
- **Description**: If the database is used as a notification channel for index updates, this property indicates maximum redelivery attempts when an exception occurs.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 60
- **Since**: 6.8.4.7

defaultAttachmentEntryClassName

- **Module**: cmweb-server-adapter
- **Description**: The default text class used to classify an attachment if no other text class was set explicitly.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: DefaultTextElement
- **Since**: 6.9.2.0

defaultCommentClassName

- **Module**: cmas-core-server
- **Description**: Name of the default text class for comments.
- **Type**: string
- **Restart required**: no

- **System**: no
- **Optional**: yes
- **Example value**: default_class
- **Since**: 6.3.0

defaultContentEntryClassName

- **Module**: cmweb-server-adapter
- **Description**: Name of the text class which is set as default for newly created content entries.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: default_class
- **Since**: 6.3.0

defaultIncommingMailClassName

- **Module**: cmas-core-server
- **Description**: Name of the default text class for incoming emails.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: default_class
- **Since**: 6.3.0

defaultNumberOfCustomFieldsColumns

- **Module**: cmweb-server-adapter
- **Description**: Determines the default number of columns for data fields.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 3
- **Since**: 6.2.0

defaultOutgoingMailClassName

- **Module**: cmas-core-server
- **Description**: Name of the default text class for outgoing emails.

- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: default_class
- **Since**: 6.3.0

defaultUnitContentClassName

- **Module**: cmas-core-server
- **Description**: Name of the default class for content created by customers via REST API.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: default_unit_class
- **Since**: 6.16.1

delete.ticket.enabled

- **Module**: cmas-app-admin-tool
- **Description**: Controls if tickets can be deleted in the ticket administration of the Admin Tool. The default value is “true”.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: true
- **Since**: 6.9.4.0
- **Removed in**: 6.16.0.0

diffTrackingEnabled

- **Module**: cmweb-server-adapter
- **Description**: Defines if parallel editing of a case by different users should be possible.
  “false”: Previous way of handling changes when editing a case. If the case has been changed in the meantime, the current user will not be able to submit his changes without being forced to reload the page before submitting.
  “true”: New changes handling mode. If the case has been changed, this will not block the submission of other changes anymore. If the part of the case that was changed was exactly the part that is changed by the submitting user, then an information message will be displayed, but the case change will be stored anyway.
- **Type**: boolean

- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true (default)
- **Since**: 6.10.1
- **Removed in**: 6.11.0

diffTrackingEnabledForUnitAndResource

- **Module**: cmweb-server-adapter
- **Description**: Enables the prevention of concurrent modifications on customers / resources.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 3
- **Since**: 6.11.0.0

diff.tracking.disabled

- **Module**: cmas-restapi-core
- **Description**: Fallback property for disabling difference tracking for CM/Track, which is history-based so it can be heavy.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.10.5.6

disable.admin.task.auto.commit

- **Module**: cmas-core-index-common
- **Description**: Toggle for the automatic execution of administrative index update tasks. If set to “false”, possibly expensive index tasks created as a result of configuration changes will be automatically executed right after creation. It is recommended to set this value to “true”, so that the execution of administrative index tasks needs to be manually triggered in the Web Admin Suite.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no

- **Example value**: false
- **Since**: 6.6.1

domain.map.for.client.config.<DOMAIN_NAME>

- **Module**: cmas-restapi-core
- **Description**: Enables mapping multiple instances of CM/Track to a specific domain. DOMAIN_ NAME is the name of the client configuration (if your CM/Track configuration is called “trackV2customized”, the name of the property is `domain.map.for.client.config.trackV2customized`). You can provide several URLs separated by a comma.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: https://www.consol.de, https://www.consol.com
- **Since**: 6.10.7.0

domain.map.for.oidc.config.<CONFIGURATION_NAME>

- **Module**: cmas-core-security
- **Description**: Enables mapping multiple instances of CM/Track to a specific configuration. Replace “CONFIGURATION_NAME” with the name of the configuration (if your CM/Track configuration is called “trackV2customized”, the name of the property is `domain.map.for.oidc.config.trackV2customized`). You can provide several URLs separated by a comma.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: myserver.consol.de,myserver.consol.com
- **Since**: 6.14.0.3

dwh.administration.refresh.interval.seconds

- **Module**: cmas-app-admin-tool
- **Description**: Internal DWH property, not to be changed manually. This property is used to set the interval for refreshing the list of actions in the Admin Tool, section *Data Warehouse -> Administration -> Actions*.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes

- **Example value**: 10
- **Since**: 6.11.0.1
- **Removed in**: 6.16.0.0

dwh.mode

- **Module**: cmas-dwh-server
- **Description**: Currently active mode for the DWH data transfer. Possible values are OFF, ADMIN, LIVE.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: OFF
- **Since**: 6.0.1

engineer.authentication.method

- **Module**: cmas-core-security
- **Description**: Sets the authentication method for users. Possible values are: “DATABASE”, “LDAP”, “LDAP,DATABASE”, “DATABASE,LDAP”
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: DATABASE
- **Since**: 6.14.0.0

engineer.description.cache.enabled

- **Module**: cmas-core-server
- **Description**: Defines whether user descriptions are cached. The default value is “true”, please do not change it unless advised by ConSol.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.11.0

engineer.description.mode

- **Module**: cmas-core-server
- **Description**: Defines whether the user names in the case history are taken from the database or dynamically rendered using templates. The default value “DYNAMIC” is a bit more costly from the performance perspective, while “PROTOCOL” is faster but returns historical names which might be outdated. Use “PROTOCOL” if you have lots of history entries from many different users.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: PROTOCOL
- **Since**: 6.11.0

engineer.description.template.name

- **Module**: cmas-core-server
- **Description**: Defines the name of the template which is used to render user names for display in the Web Client. The template has to be saved on the *Templates* page of the Web Admin Suite.
- **Type**: String
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: engineer description template name (default value)
- **Since**: 6.11.0

eviction.event.queue.size

- **Module**: cmas-core-cache
- **Description**: The size of the queue holding cache events. The default value is 200000. It is recommended to increase the value slightly (up to 400000) on systems with high traffic or load. [Deprecated - not used for infinispan]
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: 200000 (default value)
- **Since**: 6.4.0

eviction.lifeSpan

- **Module**: cmas-core-cache
- **Description**: Sets the interval (in milliseconds) for cache entry validity. When this time is elapsed, the entry is removed from the cache.
- **Type**: integer
- **Restart required**: yes
- **System**: no
- **Optional**: yes
- **Example value**: 86400000 (default value)
- **Since**: 6.11.1.0

eviction.max.nodes

- **Module**: cmas-core-cache
- **Description**: Sets the maximum size of internal caches. The default value is 100000. Increasing it will lead to higher memory consumption and is not recommended unless explicitly advised by ConSol.
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: 100000 (default value)
- **Since**: 6.4.0

eviction.strategy

- **Module**: cmas-core-cache
- **Description**: Determines the cache eviction strategy. Please see the Infinispan documentation for details.
- **Type**: string
- **Restart required**: yes
- **System**: no
- **Optional**: yes
- **Example value**: LRU, LIRS (default value)
- **Since**: 6.11.1.0

eviction.wakeup.interval

- **Module**: cmas-core-cache
- **Description**: Sets the interval (in milliseconds) between two cache queue event processing cycles. The default value is 3000. It is recommended to decrease it (minimum is 1500) on systems with high traffic or load.

- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: 3000 (default value)
- **Since**: 6.4.0

expert.mode

- **Module**: cmas-core-shared
- **Description**: Activates the expert mode in the Admin Tool to unlock extended features. For example, the system property `initialized` is only available in the expert mode.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.0
- **Removed in**: 6.16.0.0

external.line.access.prefix

- **Module**: cmas-core-server
- **Description**: General prefix to dial before an area code. Set for each customer group separately.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 411
- **Since**: 6.9.3.0

favoritesSizeLimit

- **Module**: cmweb-server-adapter
- **Description**: Determines the maximum number of items in the favorites.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no

- **Example value**: 10
- **Since**: 6.0

fetchLock.interval

- **Module**: cmas-workflow-jbpm
- **Description**:
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5000
- **Removed in**: 6.8.0

fetchSize.strategy

- **Module**: cmas-core-server
- **Description**: Strategy for selecting the fetch size on JDBC result sets.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: FetchSizePageBasedStrategy, FetchSizeThresholdStrategy, FetchSizeFixedStrategy
- **Since**: 6.8.4.1

fetchSize.strategy.FetchSizeFixedStrategy.value

- **Module**: cmas-core-server
- **Description**: Sets the fetch size value if the selected strategy to set the fetch size is `FetchSizeFixedStrategy`. Please do not change this value unless advised by ConSol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 150
- **Since**: 6.8.4.1

fetchSize.strategy.FetchSizePageBasedStrategy.limit

- **Module**: cmas-core-server
- **Description**: Sets the maximum fetch size value if the selected strategy to set the fetch size is `FetchSizePageBasedStrategy`. Please do not change this value unless advised by ConSol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 10000
- **Since**: 6.8.4.1

fetchSize.strategy.FetchSizeThresholdStrategy.value

- **Module**: cmas-core-server
- **Description**: Sets the fetch size threshold border values if the selected strategy to set the fetch size is `FetchSizeThresholdStrategy`. Please do not change this value unless advised by ConSol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 150,300,600,1000
- **Since**: 6.8.4.1

filesystem.polling.threads.number

- **Module**: cmas-nimh
- **Description**: Number of threads started for polling emails from the database queue. The default value is 1.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 10
- **Since**: 6.9.4

filesystem.polling.threads.shutdown.timeout.seconds

- **Module**: cmas-nimh
- **Description**: Waiting time after the shutdown signal. When the timeout is reached, the thread will be terminated.
- **Type**: integer

- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.9.4

filesystem.polling.threads.watchdog.interval.seconds

- **Module**: cmas-nimh
- **Description**: Determines the interval in seconds to execute the watchdog which checks the activity of the threads of the file system poller.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 30 (default value)
- **Since**: 6.9.4

filesystem.task.enabled

- **Module**: cmas-nimh
- **Description**: Allows to disable the service thread related to a given poller.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.9.4

filesystem.task.interval.seconds

- **Module**: cmas-nimh
- **Description**: Default interval (in seconds) for polling the file system.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.9.4

filesystem.task.polling.folder

- **Module**: cmas-nimh
- **Description**: Polling folder location which is scanned for emails in the format of `eml` files. The default value is the `mail` subdirectory of the ConSol CM data directory.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: c://cmas//mail
- **Since**: 6.9.4

filesystem.task.timeout.seconds

- **Module**: cmas-nimh
- **Description**: After this time of inactivity the service thread is considered damaged and automatically restarted. The default value is 120 seconds.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60
- **Since**: 6.9.4

filesystem.task.transaction.timeout.seconds

- **Module**: cmas-nimh
- **Description**: Default transaction timeout for transactions fetching emails from the file system. Should be correlated with number of messages fetched at once.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.9.4

forward.mails.to.representatives

- **Module**: cmweb-server-adapter
- **Description**: Determines if emails which are manually sent from the Web Client are also sent to representing engineers. The default value of the property is “false”, meaning that this kind of emails are not forwarded to the representing engineer. Set the property to “true” if you want to restore the previous behavior, i.e., all emails which are sent to the represented

engineer are automatically forwarded to the representing engineer. Please take into account that this might not be desired if the same person is an engineer and a customer in the CM system.

- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: false (default value)
- **Since**: 6.11.1.7

> ⓘ This property only configures the handling of manually sent emails. The handling of automatically sent emails depends on the used Java method.

globalSearchResultSizeLimit

- **Module**: cmweb-server-adapter
- **Description**: Determines the maximum number of results in the quick search.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 10
- **Since**: 6.0

heartbeat

- **Module**: cmas-core-server
- **Description**: Timestamp that indicates if an instance of the application is connected to the database schema.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1234567899
- **Since**: 6.10.5.3

helpFilePath

- **Module**: cmweb-server-adapter
- **Description**: URL for the online help. If not empty, the *Help* link is displayed in the Web Client.
- **Type**: string
- **Restart required**: no

- **System**: yes
- **Optional**: yes
- **Example value**: http://www.consol.de
- **Since**: 6.2.1
- **Removed in**: 6.14.0.0

hibernate.dialect

- **Module**: cmas-setup-hibernate
- **Description**: The database dialect used by hibernate. Set during the initial setup depending on the database system.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: org.hibernate.dialect.MySQL5InnoDBDialect
- **Since**: 6.0

hideTicketSubject

- **Module**: cmweb-server-adapter
- **Description**: If set to “true”, the case subject is hidden in the Web Client.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.2.1

ignore-queues

- **Module**: cmas-dwh-server
- **Description**: A comma-separated list of queue names which are not not transferred to the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: QueueName1,QueueName2,QueueName3
- **Since**: 6.6.19
- **Removed in**: 6.8.1

index.attachment

- **Module**: cmas-core-index-common
- **Description**: Determines whether the content of attachments is indexed.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.4.3

index.history

- **Module**: cmas-core-index-common
- **Description**: Determines whether the case and contact history is indexed.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.1.0
- **Removed in**: 6.11.0

index.status

- **Module**: cmas-core-index-common
- **Description**: Index synchronization status, possible values are GREEN, YELLOW or RED. This value is set internally by the index system, please do not change it manually unless advised by ConSol.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: GREEN
- **Since**: 6.6.1

index.task.worker.threads

- **Module**: cmas-core-index-common
- **Description**: Sets the number of threads that will be used to execute index tasks. The default value is 1, we recommend to increase it slightly on systems with large indexer load up to a value of 4.

- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1 (default value)
- **Since**: 6.6.14, 6.7.3. Since 6.8.0 and exclusively in 6.6.21 also normal (live) index updates are affected by this property.

index.version.current

- **Module**: cmas-core-index-common
- **Description**: Holds information about the current index version. This value is maintained internally, please do not change it unless advised by ConSol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1 (default value)
- **Since**: 6.7.0

index.version.newest

- **Module**: cmas-core-index-common
- **Description**: Holds information about which index version is considered newest. This value is maintained internally, please do not change it unless advised by ConSol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1 (default value)
- **Since**: 6.7.0

indexed.assets.per.thread.in.memory

- **Module**: cmas-core-index-common
- **Description**: Determines how many assets should be loaded into memory at once, per thread, during indexing.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no

- **Example value**: 200 (default value)
- **Since**: 6.8.0

indexed.engineers.per.thread.in.memory

- **Module**: cmas-core-index-common
- **Description**: Determines how many users should be loaded into memory at once, per thread, during indexing. Increasing it slightly will lead to better indexer performance at the price of higher memory consumption.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 300 (default value)
- **Since**: 6.6.14, 6.7.3

indexed.resources.per.thread.in.memory

- **Module**: cmas-core-index-common
- **Description**: Determines how many resources should be loaded into memory at once, per thread, during indexing. Increasing it slightly will lead to better indexer performance at the price of higher memory consumption.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 200 (default value)
- **Since**: 6.10.0.0

indexed.tickets.per.thread.in.memory

- **Module**: cmas-core-index-common
- **Description**: Determines how many cases should be loaded into memory at once, per thread, during indexing. Increasing it slightly will lead to better indexer performance at the price of higher memory consumption.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 100 (default value)
- **Since**: 6.6.14, 6.7.3

indexed.units.per.thread.in.memory

- **Module**: cmas-core-index-common
- **Description**: Determines how many contacts should be loaded into memory at once, per thread, during indexing. Increasing it slightly will lead to better indexer performance at the price of higher memory consumption.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 200 (default value)
- **Since**: 6.6.14, 6.7.3

initialized

- **Module**: cmas-setup-manager
- **Description**: Flag indicating if ConSol CM is initialized, i.e. the initial setup has been performed. If this value is missing or not “true”, the setup will be performed. Starting with ConSol CM version 6.11, this property is only available in expert mode.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.0

> ⚠ Be careful with using this property. When you set the value to “false”, the ConSol CM server will perform the system setup at the next start, i.e. all data of the existing system is lost, including system properties.

internal.line.access.prefix

- **Module**: cmas-core-server
- **Description**: Prefix that the company’s telephony system requires for outside lines. Set for each customer group separately.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 199
- **Since**: 6.9.3.0

is.cmrf.alive

- **Module**: cmas-dwh-server
- **Description**: Defines a threshold timeframe (in seconds) for the check if CMRF is working properly. If no response from CMRF is received in the configured timeframe, the DWH operation status is updated to “error”.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1200
- **Since**: 6.7.0
- **Removed in**: 6.14.0.0

java.naming.factory.initial

- **Module**: cmas-dwh-server
- **Description**: Factory class for the DWH context factory.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: org.jnp.interfaces.NamingContextFactory
- **Since**: 6.0.1
- **Removed in**: 6.11.0.0

java.naming.factory.url.pkgs

- **Module**: cmas-dwh-server
- **Description**:
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: org.jboss.naming:org.jnp.interfaces
- **Since**: 6.0.1
- **Removed in**: 6.11.0.0

java.naming.provider.url

- **Module**: cmas-dwh-server
- **Description**: URL of naming provider.

- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: localhost
- **Since**: 6.0.1
- **Removed in**: 6.11.0.0

jmx.full.exposure

- **Module**: cmas-core-server
- **Description**: Controls the availability of methods which can be used to retrieve or delete data from ConSol CM via JMX. The property is set to “false” by default, meaning that these methods cannot be used via JMX. If you set the property to “true”, these methods can be used via JMX to retrieve or delete data.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false (default value)
- **Since**: 6.11.2.5

⚠ It is strictly recommended to have this property set to “false” on production systems.

jobExecutor.adminMail

- **Module**: cmas-workflow-engine
- **Description**: Email address which will get notified about job execution problems (when retry counter is exceeded).
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: admin@consol.de
- **Since**: 6.8.0

jobExecutor.idleInterval

- **Module**: cmas-workflow-jbpm
- **Description**:
- **Type**: integer
- **Restart required**: no

- **System**: yes
- **Optional**: no
- **Example value**: 45000
- **Removed in**: 6.8.0
- **Replaced by**: jobExecutor.idleInterval.seconds

jobExecutor.idleInterval.seconds

- **Module**: cmas-workflow-engine
- **Description**: Determines how often (in seconds) the job executor thread will look for new jobs to execute.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 45 (default value up to CM version 6.10.5.2. The default value for CM versions 6.10.5.3 and up is 5)
- **Since**: 6.8.0

jobExecutor.jobExecuteRetryNumber

- **Module**: cmas-workflow-jbpm
- **Description**:
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5
- **Removed in**: 6.8.0
- **Replaced by**: jobExecutor.jobMaxRetries

jobExecutor.jobMaxRetries

- **Module**: cmas-workflow-engine
- **Description**: Controls the number of retry attempts the job executor will do before declaring a job as failed.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes

- **Example value**: 5 (default value)
- **Since**: 6.8.0

jobExecutor.jobMaxRetriesReachedSubject

- **Module**: cmas-workflow-engine
- **Description**: The subject used in the notification email which administrators receive about failed job executions.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: Job maximum retries reached. Job was removed!!! (default value)
- **Since**: 6.8.0

jobExecutor.lockingLimit

- **Module**: cmas-workflow-engine
- **Description**: Maximum number of jobs which can locked at once (marked for execution) by the job executor thread.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 5 (default value since CM version 6.10.5.3)
- **Since**: 6.8.0

jobExecutor.lockTimeout.seconds

- **Module**: cmas-workflow-engine
- **Description**: Determines how long the job can be locked (marked for execution) by the job executor.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 360 (default value)
- **Since**: 6.8.0

jobExecutor.mailFrom

- **Module**: cmas-workflow-engine
- **Description**: Defines the email address which will be set as From header for notifications to the administrator in case of job execution errors.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: jobexecutor@consol.de
- **Since**: 6.8.0

jobExecutor.maxInactivityInterval.minutes

- **Module**: cmas-workflow-engine
- **Description**: Maximum amount of time (in minutes) of allowed job executor inactivity (e.g. when it is blocked by long timer execution). After this time, executor threads are restarted automatically.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 30 (default value)
- **Since**: 6.9.2.0

jobExecutor.threads

- **Module**: cmas-workflow-engine
- **Description**: Sets the number of job execution threads.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 1 (default value)
- **Since**: 6.8.0

jobExecutor.timerRetryInterval

- **Module**: cmas-workflow-jbpm
- **Description**:
- **Type**: integer
- **Restart required**: no

- **System**: yes
- **Optional**: no
- **Example value**: 10000
- **Removed in**: 6.8.0
- **Replaced by**: jobExecutor.timerRetryInterval.seconds

jobExecutor.timerRetryInterval.seconds

- **Module**: cmas-workflow-engine
- **Description**: Determines how long (in seconds) the job executor thread will wait before a retry after an error occurs during job execution.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 10 (default up to CM version 6.10.5.2. The default value for CM versions 6.10.5.3 and up is 30)
- **Since**: 6.8.0

jobExecutor.txTimeout.seconds

- **Module**: cmas-workflow-engine
- **Description**: Transaction timeout used for job execution.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.8.0

kerberos.v5.enabled

- **Module**: cmas-core-security
- **Description**: Indicates whether user authentication using SSO via Kerberos is enabled.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false (default value if Kerberos was not enabled during system setup)
- **Since**: 6.2.0
- **Removed in**: 6.16.0.0

kerberos.v5.username.regex

- **Module**: cmas-core-security
- **Description**: Regular expression used for mapping Kerberos principals to CM user login names.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: (.*)@.*
- **Since**: 6.2.0
- **Removed in**: 6.16.0.0

last.config.change

- **Module**: cmas-core-server
- **Description**: Random UUID created during the last configuration change. This is a value maintained internally, please do not change it unless advised by ConSol.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 2573c7b7-2bf5-47ff-b5a2-bad31951a266
- **Since**: 6.1.0, 6.2.1

last.config.change.templates

- **Module**: cmas-core-server
- **Description**: Random UUID created during the last change in templates. This is a value maintained internally, please do not change it unless advised by ConSol.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 2573c7c7-2af5-4eff-b9c2-bad31951a266
- **Since**: 6.10.5.0

last.ping.timestamp

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property shows the status of CMRF and is filled automatically by the CMRF. It contains the date of the last response on the ping from the ConSol CM server to the CMRF.

- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 32323214
- **Since**: 6.11.0.1

last.success.live.timestamp

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 13232762
- **Since**: 6.11.2.0

ldap.authentication

- **Module**: cmas-core-security
- **Description**: LDAP authentication method used when using LDAP authentication. Possible values are “anonymous” and “simple”.
- **Type**: string
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: simple (default value)
- **Since**: 6.0
- **Removed in**: 6.14.0.0

ldap.basedn

- **Module**: cmas-core-security
- **Description**: Base DN used for looking up LDAP user accounts when using LDAP authentication.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: ou=accounts,dc=consol,dc=de

- **Since**: 6.0
- **Removed in**: 6.14.0.0

ldap.certificate.basedn

- **Module**: cmas-core-server
- **Description**: Base DN for the certificate location in the LDAP tree. Since version 6.14.0.0 this property is mandatory. In earlier CM versions, if not set, the value of the property `cmas-core-security, ldap.basedn` is used.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: ou=accounts,dc=consol,dc=de
- **Since**: 6.8.4

ldap.certificate.content.attribute

- **Module**: cmas-core-server
- **Description**: Name of the LDAP attribute where the certificate data is stored in the LDAP tree.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: usercertificate (default value)
- **Since**: 6.8.4

ldap.certificate.password

- **Module**: cmas-core-server
- **Description**: Password of the LDAP certificate manager. Since version 6.14.0.0 this property is mandatory. In earlier CM versions, if not set, the value of the property `cmas-core-security, ldap.password` is used.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.8.4

ldap.certificate.providerurl

- **Module**: cmas-core-server
- **Description**: URL of the LDAP certificate provider. Since version 6.14.0.0 this property is mandatory. In earlier CM versions, if not set, the value of the property `cmas-core-security, ldap.providerurl` is used.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: ldap://myserver.consol.de:389
- **Since**: 6.8.4

ldap.certificate.searchattr

- **Module**: cmas-core-server
- **Description**: LDAP attribute name used to search for certificates in the LDAP tree.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: mail (default value)
- **Since**: 6.8.4

ldap.certificate.userdn

- **Module**: cmas-core-server
- **Description**: DN of the LDAP certificate manager. Since version 6.14.0.0 this property is mandatory. In earlier CM versions, if not set, the value of the property `cmas-core-security, ldap.userdn` is used.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.8.4

ldap.contact.<NAME>.basedn

- **Module**: cmas-core-security
- **Description**: Base path to search for the contact DN by LDAP ID.
- **Type**: string
- **Restart required**: no

- **System**: no
- **Optional**: yes
- **Example value**: ou=accounts,dc=consol,dc=de
- **Since**: 6.9.3.0

ldap.contact.<NAME>.password

- **Module**: cmas-core-security
- **Description**: Password to look up the contact DN by LDAP ID. If not set, the anonymous account is used.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Since**: 6.9.3.0

ldap.contact.<NAME>.providerurl

- **Module**: cmas-core-security
- **Description**: Address of the LDAP server (ldap[s]://host:port) used for contact authentication.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: ldap://ldap.consol.de:389
- **Since**: 6.9.3.0

ldap.contact.<NAME>.searchattr

- **Module**: cmas-core-security
- **Description**: Attribute to search for the contact DN by LDAP ID (e.g. uid).
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: uid
- **Since**: 6.9.3.0

ldap.contact.<NAME>.userdn

- **Module**: cmas-core-security
- **Description**: User DN to look up contact DN by LDAP ID. If not set, the anonymous account is used.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Since**: 6.9.3.0

ldap.engineer.<NAME>.basedn

- **Module**: cmas-core-security
- **Description**: Base DN used for looking up LDAP user accounts when using LDAP authentication.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: ou=accounts,dc=consol,dc=de
- **Since**: 6.14.0.0

ldap.engineer.<NAME>.password

- **Module**: cmas-core-security
- **Description**: Password for connecting to LDAP to look up users when using LDAP authentication. Only needed if lookup cannot be performed anonymously.
- **Type**: password
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.14.0.0

ldap.engineer.<NAME>.providerurl

- **Module**: cmas-core-security
- **Description**: URL to LDAP provider, when using LDAP authentication.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no

- **Example value**: ldap://myserver.consol.de:389
- **Since**: 6.14.0.0

ldap.engineer.<NAME>.searchattr

- **Module**: cmas-core-security
- **Description**: Search attribute for looking up the LDAP entries associated with a CM login.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: uid
- **Since**: 6.14.0.0

ldap.engineer.<NAME>.userdn

- **Module**: cmas-core-security
- **Description**: LDAP user for connecting to LDAP to look up users when using LDAP authentication. Only needed if the lookup cannot be performed anonymously.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.14.0.0

ldap.initialcontextfactory

- **Module**: cmas-core-security
- **Description**: Class name for the initial context factory of the LDAP implementation when using LDAP authentication. Usually this is `com.sun.jndi.ldap.LdapCtxFactory`.
- **Type**: string
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: com.sun.jndi.ldap.LdapCtxFactory
- **Since**: 6.0

ldap.password

- **Module**: cmas-core-security
- **Description**: Password for connecting to LDAP to look up users when using LDAP authentication. Only needed if lookup cannot be performed anonymously.
- **Type**: password

- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.1.2
- **Removed in**: 6.14.0.0

ldap.providerurl

- **Module**: cmas-core-security
- **Description**: URL to LDAP provider, when using LDAP authentication.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: ldap://myserver.consol.de:389
- **Since**: 6.0
- **Removed in**: 6.14.0.0

ldap.searchattr

- **Module**: cmas-core-security
- **Description**: Search attribute for looking up the LDAP entries associated with a CM login.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: uid
- **Since**: 6.0
- **Removed in**: 6.14.0.0

ldap.userdn

- **Module**: cmas-core-security
- **Description**: LDAP user for connecting to LDAP to look up users when using LDAP authentication. Only needed if the lookup cannot be performed anonymously.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.1.2
- **Removed in**: 6.14.0.0

live.calculation.enabled

- **Module**: cmas-dwh-server
- **Description**: Determines if the CMRF should perform an update for the date since the last successful processing of a live package or the last update, whatever happened later. Please do not change this property unless advised by ConSol.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: false
- **Since**: 6.11.2.0

live.error

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: false
- **Since**: 6.11.1.0

live.start

- **Module**: cmas-dwh-server
- **Description**: When the DWH synchronization mode is set to LIVE in the Web Admin Suite, this property is created and set to the current date.
  If LIVE mode is not enabled and there is no data in `cmas_dwh_ser_sync_object`, the property `live.start` is deleted.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes (automatically added in DWH “LIVE” mode)
- **Example value**: 15028802377645
- **Since**: 6.7.0

local.country.prefix

- **Module**: cmas-core-server
- **Description**: Prefix of the local country code. Set for each customer group separately.
- **Type**: integer

- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 48
- **Since**: 6.9.3.0

mail.attachments.validation.handling

- **Module**: cmas-nimh-extension
- **Description**: Defines the behaviour of the system for incoming emails which contain an attachment of a type not listed in the property `attachment.allowed.types`. Possible values are:
  - **pass_rename**: Disarms the attachments by renaming the file extension and lets the email pass.
  - **pass_ignore**: Leaves the attachments untouched and lets the email pass.
  - **reject_email**: Rejects the email with a notification to the sender.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: pass_rename (default value)
- **Since**: 6.13.0.9

mail.attachments.validation.info.sender

- **Module**: cmas-nimh-extension
- **Description**: Sets the From header for error notification emails which are sent when the validation of the attachment type fails. If not set, the email address set in mail.from is used. Only used if mail.attachments.validation.handling is set to “reject_email”.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: example@your_domain.com
- **Since**: 6.9.4

mail.attachments.validation.info.subject

- **Module**: cmas-nimh-extension
- **Description**: Sets the subject for error notification emails which are sent when the validation of the attachment type fails.
- **Type**: string
- **Restart required**: no

- **System**: yes
- **Optional**: no
- **Example value**: Mail was not processed because its attachments were rejected!
- **Since**: 6.9.4

mail.db.archive

- **Module**: cmas-nimh-extension
- **Description**: If property is set to “true”, incoming emails are automatically archived in the database.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.9.4

mail.encryption

- **Module**: cmas-core-server
- **Description**: If the property is set to “true”, the encryption checkbox in the email editor of the Web Client is checked by default. The default value of the property is “false”.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.8.4.0

mail.error.from.address

- **Module**: cmas-nimh-extension
- **Description**: Sets the From address for error notification emails which are sent when the email processing fails.
- **Type**: email
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Since**: 6.9.4

mail.error.to.address

- **Module**: cmas-nimh-extension
- **Description**: Sets the To address for error notification emails which are sent when the email processing fails. As a default the value of the property `admin.email` is used.
- **Type**: email
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Since**: 6.9.4

mail.from

- **Module**: cmweb-server-adapter
- **Description**: Central sender address to be used for email communication. The value set here appears as sender of emails sent from ConSol CM.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: consolcm@example.com
- **Since**: 6.1.2

mail.notification.engineerChange

- **Module**: cmas-core-server
- **Description**: Determines whether notification emails should be sent when the assignee of a case is changed.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.1.0

mail.notification.sender

- **Module**: cmas-core-server
- **Description**: Defines the From address (sender) for system notification emails which are sent when the assignee of a case is changed. If not set, the value in the `from` tag in the notification template is used instead. If this is unset too, the value of the `mail.from` property is used.
- **Type**: string

- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: cm6notification@cm6installation
- **Since**: 6.6.3

mail.on.error

- **Module**: cmas-nimh-extension
- **Description**: If set to “true”, an error notification email is sent to the configured address if an email could not be processed.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default)
- **Since**: 6.9.4

mail.outgoing.encoding

- **Module**: cmas-core-server
- **Description**: Enforces a certain encoding for outgoing emails. By default, the encoding is set to “auto” which means that ConSol CM determines the necessary encoding automatically on content inspection. An explicit setting can be needed to mitigate problems with Office365 and their bugs with 7bit encoding.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: base64
- **Since**: 6.16.2.0

mail.redelivery.retry.count

- **Module**: cmas-core-server
- **Description**: Number of redelivery attempts of an outgoing email.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 3

- **Since**: 6.1.0
- **Removed in**: 6.15.0.0

mail.reply.to

- **Module**: cmweb-server-adapter
- **Description**: Central reply-to address to be used for email communication. The value set here controls the reply-to header of emails sent from ConSol CM.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: consolcm@example.com
- **Since**: 6.0.1

mail.sender.address

- **Module**: cmas-workflow-jbpm
- **Description**: From address for emails from the workflow engine.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Removed in**: 6.8.0
- **Replaced by**: jobExecutor.mailFrom

mail.smtp.email

- **Module**: cmas-core-server
- **Description**: SMTP URL for outgoing emails
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: smtp://mail.mydomain.com:25
- **Since**: 6.0

mail.smtp.email.password

- **Module**: cmas-core-server
- **Description**: SMTP password for outgoing emails. If this property is set, it takes precedence over the value from the SMTP URL.
- **Type**: password
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: password
- **Since**: 6.11.2.2

mail.smtp.email.user

- **Module**: cmas-core-server
- **Description**: SMTP user name for outgoing emails. If this property is set, it takes precedence over the value from the SMTP URL.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: myuser
- **Since**: 6.11.2.2

mail.smtp.envelopesender

- **Module**: cmas-core-server
- **Description**: Email address used as a sender in SMTP envelope. If not set, the From address of the email is used.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: mysender@mydomain.com
- **Since**: 6.5.7

mail.smtp.tls.enabled

- **Module**: cmas-core-server
- **Description**: Activates SMTP via SSL/TLS (SMTPS) for sending emails from the Web Client and scripts. The default value is “false”. If it is set to “true”, SMTPS is activated for sending emails.
- **Type**: boolean

- **Restart required**: yes
- **System**: no
- **Optional**: yes
- **Example value**: true
- **Since**: 6.11.1.6

mail.ticketname.pattern

- **Module**: cmas-nimh-extension
- **Description**: Regular expression pattern used to identify the case name in the subject of incoming mails.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: .?\(Ticket\s#(\S+?)\).*
- **Since**: 6.4.0
- **Removed in**: 6.12.0.5

mailbox.<ID>.connection.authentication

- **Module**: cmas-nimh
- **Description**: Determines how the IMAPS connection is authenticated. The allowed values are “oauth2” (for OAUTH2 authentication) and “password” (for BASIC authentication).
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: password
- **Since**: 6.14.0.10

mailbox.<ID>.connection.oauth2.client.authority

- **Module**: cmas-nimh
- **Description**: Contains the Microsoft authentication endpoint and the Azure tenant ID. Only needed if oauth2 authentication is selected for the mailbox.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes

- **Example value**: https://login.microsoftonline.com/9048cc23-731a-40d8-b1b5-609a48b69e88
- **Since**: 6.14.0.10

mailbox.<ID>.connection.oauth2.client.id

- **Module**: cmas-nimh
- **Description**: Contains the client ID property of the registration of the ConSol CM application in the Azure portal. Only needed if oauth2 authentication is selected for the mailbox.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 123-345-567
- **Since**: 6.14.0.10

mailbox.<ID>.connection.oauth2.client.secret

- **Module**: cmas-nimh
- **Description**: Contains the secret generated for the registration of the ConSol CM application in the Azure portal. Only needed if oauth2 authentication is selected for the mailbox.
- **Type**: password
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 4Z28QV9IyblJi
- **Since**: 6.14.0.10

mailbox.<ID>.connection.oauth2.scope

- **Module**: cmas-nimh
- **Description**: Contains the oauth permission scope. The constant for Microsoft Office365 is “https://outlook.office365.com/.default”. Only needed if oauth2 authentication is selected for the mailbox.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: https://outlook.office365.com/.default
- **Since**: 6.14.0.10

mailbox.<ID>.name

- **Module**: cmas-nimh
- **Description**: Identifier (name) of the mailbox.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: myEmailAccount
- **Since**: 6.11.2.0

mailbox.default.connection.host

- **Module**: cmas-nimh
- **Description**: Host (server name) of a given mailbox which the poller reads emails from.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 10.10.1.157
- **Since**: 6.9.4

mailbox.default.connection.password

- **Module**: cmas-nimh
- **Description**: Password for given mailbox which the poller reads emails from.
- **Type**: password
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: consol
- **Since**: 6.9.4

mailbox.default.connection.port

- **Module**: cmas-nimh
- **Description**: Port of a given mailbox which the poller reads emails from.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes

- **Example value**: 143
- **Since**: 6.9.4

mailbox.default.connection.protocol

- **Module**: cmas-nimh
- **Description**: The protocol used by the email poller, e.g. IMAP or POP3.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: imap
- **Since**: 6.9.4

mailbox.default.connection.username

- **Module**: cmas-nimh
- **Description**: User name for a given mailbox which the poller reads emails from.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: username
- **Since**: 6.9.4

mailbox.default.session.mail.debug

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. Allows for more detailed JavaMail session debugging.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true
- **Since**: 6.9.4

mailbox.default.session.mail.mime.address.strict

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. If this property is set to “false”, the parsing of the email headers is less strict. Therefore, more emails with non-RFC-compliant headers will be imported into the system, instead of being stored in the failed incoming emails in the Web Admin Suite.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.9.4

mailbox.default.session.mail.<PROTOCOL>.connectiontimeout

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. Determines the connection timeout in milliseconds.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 3000
- **Since**: 6.9.4 (IMAP and POP3) / 6.11.2.0 (IMAPS and POP3S)

mailbox.default.session.mail.<PROTOCOL>.fetchsize

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. Determines the size of partial fetch in bytes for the indicated protocol.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 1048576 (default value)
- **Since**: 6.9.4.0 (IMAP and POP3) / 6.11.2.0 (IMAPS and POP3S)

mailbox.default.session.mail.<PROTOCOL>.partialfetch

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. Determines whether the protocol’s partial fetch capability should be used.

- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.9.4.0 (IMAP and POP3) / 6.11.2.0 (IMAPS and POP3S)

mailbox.default.session.mail.<PROTOCOL>.timeout

- **Module**: cmas-nimh
- **Description**: JavaMail property, see JavaMail API documentation. Determines the I/O timeout in milliseconds.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 120 for IMAP(S) and 3000 for POP3(S)
- **Since**: 6.9.4 (IMAP and POP3), 6.11.2.0 (IMAPS and POP3S)

mailbox.default.task.delete.read.messages

- **Module**: cmas-nimh
- **Description**: Defines whether messages should be removed from the mailbox after processing. For the IMAP protocol messages are marked as SEEN by default. For the POP3 protocol, this property is ignored and the emails are always deleted from the email server after they were processed successfully.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.9.4, changed in 6.12.1.0

mailbox.default.task.enabled

- **Module**: cmas-nimh
- **Description**: With this property, the service thread related to the given poller can be disabled.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes

- **Example value**: true (default value)
- **Since**: 6.9.4

mailbox.default.task.interval.seconds

- **Module**: cmas-nimh
- **Description**: Default interval (in seconds) for polling mailboxes.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.9.4

mailbox.default.task.max.message.size

- **Module**: cmas-nimh
- **Description**: Maximum size of email messages (i.e., email plus attachment). Emails exceeding the size limit will not be automatically processed by NIMH but will be stored in the database (table `cmas_nimh_archived_mail`) and will therefore appear in the failed incoming emails in the Web Admin Suite. From there they can be resent, downloaded to the file system, or deleted. For those operations the message size is not relevant.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 10485760 (default value, 10 MB)
- **Since**: 6.9.4

mailbox.default.task.max.messages.per.run

- **Module**: cmas-nimh
- **Description**: Number of messages fetched at once from the mailbox. Must be correlated with the transaction timeout. The default value is 20.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60
- **Since**: 6.9.4

mailbox.default.task.timeout.seconds

- **Module**: cmas-nimh
- **Description**: After this time of inactivity the service thread is considered damaged and automatically restarted. The default value is 120 seconds.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60
- **Since**: 6.9.4

mailbox.default.task.transaction.timeout.seconds

- **Module**: cmas-nimh
- **Description**: Default transaction timeout (in seconds) for email fetching transactions. Should be correlated with number of messages fetched at once.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.9.4

mailbox.polling.threads.mail.log.enabled

- **Module**: cmas-nimh
- **Description**: Enables email logging in the table `cmas_mail_log`. If set to “true”, each processed email is logged in the table to avoid duplicate processing. This is especially important in cluster environments. If set to “false” the mechanism to detect duplicates is disabled.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.9.4.1

mailbox.polling.threads.number

- **Module**: cmas-nimh
- **Description**: Number of threads for accessing mailboxes.
- **Type**: integer

- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 1 (default value)
- **Since**: 6.9.4

mailbox.polling.unstorable.backup.folder

- **Module**: cmas-nimh
- **Description**: Folder on the file system which is used to store emails whose processing failed and which could not be stored in the database either. The value of the property should be the path to the data directory followed by `/mail/unstorable`.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: c:/cmas/mail/unstorable
- **Since**: 6.11.2.7

mailSender.executionInterval.seconds

- **Module**: cmas-core-server
- **Description**: Determines the period in seconds before the system checks if there are new emails to be sent.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 10
- **Since**: 6.12.0.0

mailSender.lockingLimit

- **Module**: cmas-core-server
- **Description**: Determines the number of emails fetched from the database for sending in a single transaction.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 1
- **Since**: 6.12.0.0

mailSender.lockTimeout.seconds

- **Module**: cmas-core-server
- **Description**: Determines the period in seconds before a locked email is released for another lock.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 199
- **Since**: 6.12.0.0

mailSender.maxAttempts

- **Module**: cmas-core-server
- **Description**: Determines the maximum number of attempts, including the first attempt, before an email is removed.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 3
- **Since**: 6.12.0.0

mailSender.nodeId

- **Module**: cmas-core-server
- **Description**: Determines the ID of the node on which the thread for sending emails is started (only in clustered environments). If the value is “all” or no value is provided, the threads are started on every node.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: all
- **Since**: 6.12.0.0

mailSender.retryInterval.seconds

- **Module**: cmas-core-server
- **Description**: Determines the period in seconds before another attempt to send an email is made, if the previous attempt failed.
- **Type**: integer

- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60
- **Since**: 6.12.0.0

mailSender.threads.number

- **Module**: cmas-core-server
- **Description**: Determines the number of threads for sending emails.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 1
- **Since**: 6.12.0.0

mailSender.txTimeout.seconds

- **Module**: cmas-core-server
- **Description**: Determines the transaction timeout for sending emails in seconds.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60
- **Since**: 6.12.0.0

mailTemplateAboveQuotedText

- **Module**: cmweb-server-adapter
- **Description**: Determines the behavior of the email template in the email editor when another email is quoted, i.e. forwarded or replied to. Often used to place the signature correctly.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.2.4

max.licences.perUser

- **Module**: cmas-core-server
- **Description**: Sets the maximum number of licenses which a single user can use (e.g., logging in from different browsers). By default this value is not restricted.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 10
- **Since**: 6.8.4.5

maxSizePerPagemapInMegaBytes

- **Module**: cmweb-server-adapter
- **Description**: The parameter defines the size (in MB) of the file which is created by the Wicket framework per user session. i.e. for each engineer which is currently logged in. The file is used to save pages during the running session. When the defined size limit has been reached and new entries are added, the oldest entries are removed. An engineer who works with an “old” page is redirected to the start page in the Web Client, when the page is removed from the file. This parameter can be increased if engineers work with a great number of open tabs and complain about being redirected to the start page. In large systems, you could use e.g. a value of 45 or 50. Since this is the size of the file which is saved on disk, the maximum value depends on the available disk space, however, a value which is too large is not recommended either.
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: 15
- **Since**: 6.3.5

metabase.secret.key

- **Module**: cmas-analytics
- **Description**: Determines the string used to generate the private key used to validate Metabase JWT messages. Needs to match the setting *String used by the JWT signing key* in the Metabase configuration.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 50e2230c98344f183c996e6a3954c1a1e43ed100cc36f4ea093800aa2d8a1075
- **Since**: 6.13.0.0

metabase.url

- **Module**: cmas-analytics
- **Description**: Determines the URL where the Metabase application runs. Needs to match the setting *Site URL* in the Metabase configuration.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: http://localhost:3000 (default value)
- **Since**: 6.13.0.0

ml.evict.unused.after.hours

- **Module**: cmas-core-server
- **Description**: The number of hours after which a machine learning model is evicted from cache if it is not used. The ConSol CM server checks every hour for machine learning models to evict.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 24 (default value)
- **Since**: 6.13.0.0

ml.learn.worker.threads

- **Module**: cmas-core-server
- **Description**: The number of threads that a machine learning algorithm can use while training its machine learning model.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1 (default value)
- **Since**: 6.13.0.0

monitoring.engineer.login

- **Module**: cmas-core-server
- **Description**: Login for the special user (engineer) used for monitoring.
- **Type**: string
- **Restart required**: no

- **System**: yes
- **Optional**: yes
- **Example value**: nagios
- **Since**: 6.9.3.0

monitoring.unit.login

- **Module**: cmas-core-server
- **Description**: Login for the special user (contact) used for monitoring.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: nagios
- **Since**: 6.9.3.0

nimh.enabled

- **Module**: cmas-core-server
- **Description**: Enables the NIMH service which controls email polling and processing in ConSol CM. Must be suffixed with NODEID in cluster environments e.g. `nimh.enabled.NODEID` = “true”.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false
- **Since**: 6.9.4.0

notification.error.description

- **Module**: cmas-dwh-server
- **Description**: Defines the text for error emails from the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: Error occurred
- **Since**: 6.0.1

notification.error.from

- **Module**: cmas-dwh-server
- **Description**: Defines the From address for error emails from the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: myuser@consol.de
- **Since**: 6.0.1

notification.error.subject

- **Module**: cmas-dwh-server
- **Description**: Defines the email subject for error emails from the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: Error occurred
- **Since**: 6.0.1

notification.error.to

- **Module**: cmas-dwh-server
- **Description**: Defines the To address for error emails from the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: myuser@consol.de
- **Since**: 6.0.1

notification.finished_successfully.description

- **Module**: cmas-dwh-server
- **Description**: Defines the text for emails from the DWH when a transfer finishes successfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes

- **Example value**: Transfer finished successfully
- **Since**: 6.0.1

notification.finished_successfully.from

- **Module**: cmas-dwh-server
- **Description**: Defines the From address for emails from the DWH when a transfer finishes successfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: myuser@consol.de
- **Since**: 6.0.1

notification.finished_successfully.subject

- **Module**: cmas-dwh-server
- **Description**: Defines the subject for emails from the DWH when a transfer finishes successfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: Transfer finished successfully
- **Since**: 6.0.1

notification.finished_successfully.to

- **Module**: cmas-dwh-server
- **Description**: Defines the To address for emails from the DWH when a transfer finishes successfully.
- **Type**: string
- **Restart required**: yes
- **System**: yes
- **Optional**: yes
- **Example value**: myuser@consol.de
- **Since**: 6.0.1

notification.finished_unsuccessfully.description

- **Module**: cmas-dwh-server
- **Description**: Defines the text for emails from the DWH when a transfer finishes unsuccessfully.

- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: Transfer finished unsuccessfully
- **Since**: 6.0.1
- **Removed in**: 6.15.0.0

notification.finished_unsuccessfully.from

- **Module**: cmas-dwh-server
- **Description**: Defines the From address for emails from the DWH when a transfer finishes unsuccessfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: myuser@consol.de
- **Since**: 6.0.1
- **Removed in**: 6.15.0.0

notification.finished_unsuccessfully.subject

- **Module**: cmas-dwh-server
- **Description**: Defines the subject for emails from the DWH when a transfer finishes unsuccessfully.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: Transfer finished unsuccessfully
- **Since**: 6.0.1
- **Removed in**: 6.15.0.0

notification.finished_unsuccessfully.to

- **Module**: cmas-dwh-server
- **Description**: Defines the To address for emails from the DWH when a transfer finishes unsuccessfully.
- **Type**: string
- **Restart required**: no

- **System**: yes
- **Optional**: no
- **Example value**: myuser@consol.de
- **Since**: 6.0.1
- **Removed in**: 6.15.0.0

notification.host

- **Module**: cmas-dwh-server
- **Description**: Email (SMTP) server hostname for sending DWH emails.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: myserver.consol.de
- **Since**: 6.0.1

notification.password

- **Module**: cmas-dwh-server
- **Description**: SMTP password for sending DWH emails (optional).
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.0.1

notification.port

- **Module**: cmas-dwh-server
- **Description**: SMTP port for sending DWH emails.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 25
- **Since**: 6.0.1

notification.protocol

- **Module**: cmas-dwh-server
- **Description**: The protocol used for sending emails from the DWH. Possible values are smtp or smtps.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: smtp

notification.tls.enabled

- **Module**: cmas-dwh-server
- **Description**: Enables SMTP via SSL/TLS (SMTPS) for sending notification emails from the DWH. The default value is “false”. If it is set to “true”, SMTPS is enabled for sending notifications from the DWH.
- **Type**: string
- **Restart required**: yes
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.11.1.6

notification.username

- **Module**: cmas-dwh-server
- **Description**: (SMTP) user name for sending DWH emails.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: myuser
- **Since**: 6.0.1

notifications.enabled

- **Module**: cmas-web-notifications
- **Description**: Determines whether the notifications feature is enabled (“true”) or disabled (“false”). The default value is “true”. If you use the notifications feature on a ConSol CM system behind a proxy, some special proxy settings are needed. Please see section *Operating ConSol CM Behind a Proxy (e.g., Apache HTTPD)* of the *ConSol CM Setup Manual* for further information.

- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: true
- **Since**: 6.12.0.0

number.of.tasks

- **Module**: cmas-core-server
- **Description**: Number of threads to use by the Task Execution Framework (TEF).
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 1
- **Since**: 6.9.4.0

oidc.track.authority.default

- **Module**: cmas-core-security
- **Description**: Indicates the URL of the authenticating authority, e.g. ADFS. Replace “default” by the name of the configuration in the property name to specify this setting for a specific configuration.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: https://localhost/adfs
- **Since**: 6.14.0.3

oidc.track.clientId.default

- **Module**: cmas-core-security
- **Description**: Indicates the client ID (application ID) of the application, as registered in ADFS or Azure AD. Replace “default” by the name of the configuration in the property name to specify this setting for a specific configuration.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.14.0.3

oidc.track.clientSecret.default

- **Module**: cmas-core-security
- **Description**: Indicates the secret of the client, generated using ADFS or Azure AD. Replace "default" by the name of the configuration in the property name to specify this setting for a specific configuration.
- **Type**: password
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.14.0.3

oidc.track.enabled.default

- **Module**: cmas-core-security
- **Description**: Indicates whether user authentication using SSO via OIDC is enabled. Replace "default" by the name of the configuration in the property name to specify this setting for a specific configuration.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false (default value)
- **Since**: 6.14.0.3

oidc.track.redirectUri.default

- **Module**: cmas-core-security
- **Description**: Indicates the redirect URI where authentication responses can be received. This is either the OIDC endpoint on the CM server running CM/Track or on the load balancer. Replace "default" by the name of the configuration in the property name to specify this setting for a specific configuration.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: http://localhost/track/oidc/
- **Since**: 6.14.0.3

oidc.track.usernameClaim.default

- **Module**: cmas-core-security
- **Description**: Indicates the name of the claim in the ID token which is used to map the user to a customer in ConSol CM. The value depends on the ADFS settings; the default values are "upn" and "unique_name". Replace "default" by the name of the configuration in the property name to specify this setting for a specific configuration.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: upn
- **Since**: 6.14.0.3

oidc.track.usernameRegexp.default

- **Module**: cmas-core-security
- **Description**: Defines the regular expression used for mapping the user name claim values to ConSol CM user names. Replace "default" by the name of the configuration in the property name to specify this setting for a specific configuration.
  - **"upn" as claim**: `(.*)@.*` will transform the claim value "user1@sso.yourdomain.com" to "user1" and look up "user1" in the ConSol CM database.
  - **"unique_name" as claim**: `.*\\(.*)` will transform the claim value "SSO\user1" to "user1" and look up "user1" in the ConSol CM database.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Since**: 6.14.0.3

oidc.web.authority.default

- **Module**: cmas-core-security
- **Description**: Indicates the URL of the authenticating authority, e.g. ADFS.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: https://localhost/adfs
- **Since**: 6.16.0.0

oidc.web.clientId.default

- **Module**: cmas-core-security
- **Description**: Indicates the client ID (application ID) of the application, as registered in ADFS or Azure AD.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.16.0.0

oidc.web.clientSecret.default

- **Module**: cmas-core-security
- **Description**: Indicates the secret of the client, generated using ADFS or Azure AD.
- **Type**: password
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.16.0.0

oidc.web.enabled.default

- **Module**: cmas-core-security
- **Description**: Indicates whether user authentication using SSO via OIDC is enabled.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false (default value)
- **Since**: 6.16.0.0

oidc.web.redirectUri.default

- **Module**: cmas-core-security
- **Description**: Indicates the redirect URI where authentication responses can be received. This is either the OIDC endpoint on the CM server or on the load balancer.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no

- **Example value**: http://localhost/cm-client/oidc/
- **Since**: 6.16.0.0

oidc.web.usernameClaim.default

- **Module**: cmas-core-security
- **Description**: Indicates the name of the claim in the ID token which is used to map the user to an engineer in ConSol CM. The value depends on the ADFS settings; the default values are “upn” and “unique_name”.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: upn
- **Since**: 6.16.0.0

oidc.web.usernameRegexp.default

- **Module**: cmas-core-security
- **Description**: Defines the regular expression used for mapping the user name claim values to ConSol CM user names. Replace “default” by the name of the configuration in the property name to specify this setting for a specific configuration.
    - **“upn” as claim**: `(.*)@.*` will transform the claim value “user1@sso.yourdomain.com” to “user1” and look up “user1” in the ConSol CM database.
    - **“unique_name” as claim**: `.*\\(.*)` will transform the claim value “SSO\user1” to “user1” and look up “user1” in the ConSol CM database.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Since**: 6.16.0.0

outdated.lock.age

- **Module**: cmas-workflow-jbpm
- **Description**:
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 60000

- **Removed in**: 6.8.0
- **Replaced by**: cmas-workflow-engine, jobExecutor.lockTimeout.seconds

pagemapLockDurationInSeconds

- **Module**: cmweb-server-adapter
- **Description**: Number of seconds to pass before a pagemap is considered to be locked for too long.
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: yes
- **Example value**: 60
- **Since**: 6.7.3

password.reset.mail.from

- **Module**: cmas-core-security
- **Description**: Defines the from address (sender) of emails for resetting the password. If not set, the value of the `admin.email` property is used instead.
- **Type**: String
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: mypwreset@consol.de
- **Since**: 6.11.0.1

policy.password.age

- **Module**: cmas-core-security
- **Description**: Defines the maximum validity period (in days) of passwords before the user needs to change the password. Example “183” (6 months), default value: “5500” (= 15 years, i.e., no password change enforced).
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 5500 (15 years, default value)
- **Since**: 6.10.1.0

policy.password.pattern

- **Module**: cmas-core-security
- **Description**: Defines a regular expression pattern which new passwords have to match. The default value is `^(?=.*[0-9])(?=.*[A-Z])(?=.*[a-z]).{7,}$` (at least 7 characters, of which one capital letter, one lower case letter and one number).
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: ^(?=.*[0-9])(?=.*[A-Z])(?=.*[a-z]).{7,}$ (default value)
- **Since**: 6.10.1.0

policy.rotation.ratio

- **Module**: cmas-core-security
- **Description**: Defines how often passwords may repeat. For example, setting the value to X means that the new password cannot be present among the user's X previous passwords.

> ⓘ Since 6.11.2.8, this property is also taken into account for CM/Track. Nevertheless, any value greater than 1 is interpreted as 1, so that it is only possible to compare the new password to the current one.

> ⓘ You can set the property to 0 to disable password rotation checks in CM/Track.

- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 5 (default value)
- **Since**: 6.10.1.0, changed in 6.11.2.8

policy.track.username.case.sensitive

- **Module**: cmas-core-security
- **Description**: Defines whether customer user names in CM/Track case-sensitive.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.11.0.0

policy.username.case.sensitive

- **Module**: cmas-core-security
- **Description**: Defines whether user names are case-sensitive.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.10.1.0

postActivityExecutionScriptName

- **Module**: cmweb-server-adapter
- **Description**: Defines the name for the script which should be executed after every workflow activity. If no script should be executed, leave the value empty.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: postActivityExecutionHandler
- **Since**: 6.2.0

queue.polling.threads.number

- **Module**: cmas-nimh
- **Description**: Number of threads started for polling the email queue in the database.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 1 (default value)
- **Since**: 6.9.4

queue.polling.threads.shutdown.timeout.seconds

- **Module**: cmas-nimh
- **Description**: Waiting time after the shutdown signal for the email poller. When the timeout is reached, the thread will be terminated.
- **Type**: integer
- **Restart required**: no
- **System**: no

- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.9.4

queue.polling.threads.watchdog.interval.seconds

- **Module**: cmas-nimh
- **Description**: Determines the interval in seconds to execute the watchdog which checks the activity of the threads of the email queue poller.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 30 (default value)
- **Since**: 6.9.4

queue.task.error.pause.seconds

- **Module**: cmas-nimh
- **Description**: Maximum number of seconds, the queue poller waits after infrastructure (e.g. database) error.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 180 (default value)
- **Since**: 6.9.4

queue.task.interval.seconds

- **Module**: cmas-nimh
- **Description**: Determines the interval in seconds for the main email queue polling thread.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 15 (default value)
- **Since**: 6.9.4

queue.task.max.retries

- **Module**: cmas-nimh
- **Description**: Maximum number of email processing retries after an exception. When reached, the email is moved to the email archive. This email can be rescheduled again using the NIMH API or the Web Admin Suite.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 10
- **Since**: 6.9.4

queue.task.timeout.seconds

- **Module**: cmas-nimh
- **Description**: After this time of inactivity (in seconds), the service thread is considered damaged and automatically restarted.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 600 (default value)
- **Since**: 6.9.4

queue.task.transaction.timeout.seconds

- **Module**: cmas-nimh
- **Description**: Transaction timeout (in seconds) for email processing in the pipe.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.9.4

queuesExcludedFromGS

- **Module**: cmweb-server-adapter
- **Description**: Comma-separated list of queue names which are excluded from the quick search.
- **Type**: string
- **Restart required**: no

- **System**: yes
- **Optional**: yes
- **Example value**: Helpdesk,FAQ
- **Since**: 6.0

recent.items.cleanup.cluster.node.id

- **Module**: cmas-core-server
- **Description**: Value of a `-Dcmas.clusternode.id` designating the node which will clean up recent items.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 1 (assuming the cluster node started with `-Dcmas.clusternode.id=1` parameter)
- **Since**: 6.11.0.1

recent.items.cleanup.interval.minutes

- **Module**: cmas-core-server
- **Description**: Determines the time interval (in minutes) in which recent items should be checked for removal.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 60
- **Since**: 6.11.0.1

recent.items.max.per.engineer

- **Module**: cmas-core-server
- **Description**: Maximum number of preserved recent items per engineer while cleaning up (older recent items will be deleted).
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 500
- **Since**: 6.11.0.1

recent.items.persistence.enabled

- **Module**: cmas-core-server
- **Description**: Enables persistence of recent items. If the property is set to “false”, it prevents storing new recent items. If a recently visited widget (`recentlyVisitedWidget`) is used on a dashboard, this property needs to be set to “true”. Otherwise, you can set it to “false” to save system resources.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.11.1.0

recoverable.exceptions

- **Module**: cmas-dwh-server
- **Description**: Comma-separated list of exception definitions: CLASS[+][:REGEX]. The exceptions included in the list do not stop CM from sending to the CMRF process, but force it to try again. If optional '+' after CLASS is present, classes which extend CLASS are matched.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: java.sql.SQLRecoverableException,java.lang.RuntimeException+:.*T.1\,2T.*
- **Since**: 6.8.4.6

refreshTimeInCaseOfConcurrentRememberMeRequests

- **Module**: cmweb-server-adapter
- **Description**: Sets the refresh time (in seconds) after which the page is reloaded in case of concurrent remember me requests. This feature prevents one user from occupying many licenses.
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: yes
- **Example value**: 5
- **Since**: 6.8.2

rememberMeLifetimeInMinutes

- **Module**: cmweb-server-adapter
- **Description**: Lifetime for *remember me* in minutes.

- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: 1440
- **Since**: 6.0

request.log.threshold.milliseconds

- **Module**: cmas-restapi-core
- **Description**: Time threshold in milliseconds that needs to be exceeded to generate an entry in the file `operationtimes-rest.log`.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1000 (default value)
- **Since**: 6.13.0.2

request.log.threshold.milliseconds

- **Module**: cmweb-server-adapter
- **Description**: Defines the execution time threshold of web requests in milliseconds that needs to be exceeded to generate an entry in the file `operationtimes.log`.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 1000 (default value)
- **Since**: 6.15.0.0

request.log.username

- **Module**: cmas-restapi-core
- **Description**: Name of the user whose requests are logged in the file `operationtimes-rest.log`. Only one name is supported. If the property is empty (default value), the requests of all users are logged.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes

- **Example value**: admin
- **Since**: 6.13.0.2

request.scope.transaction

- **Module**: cmweb-server-adapter
- **Description**: It allows to disable request scope transactions. By default one transaction is used per request. Setting this property to “false” will cause one transaction per service method invocation.
- **Type**: boolean
- **Restart required**: yes
- **System**: yes
- **Optional**: yes
- **Example value**: true
- **Since**: 6.8.1

resetCode.expirationPeriod

- **Module**: cmas-core-security
- **Description**: Defines the expiration period for the link when resetting the password for the Web Client or CM/Track (in milliseconds).
- **Type**: Integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 86400000 (default value, 24 hours)
- **Since**: 6.10.1

resource.replace.batchSize

- **Module**: cmas-core-server
- **Description**: Defines the number of objects to be processed in a single resource replacement batch.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5
- **Since**: 6.10.0.0

resource.replace.timeout

- **Module**: cmas-core-server
- **Description**: Determines the transaction timeout (in seconds) of a resource replacement action step.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 120
- **Since**: 6.10.0.0

scene

- **Module**: cmas-setup-scene
- **Description**: Name of the scene file which was imported during the setup (can be empty).
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: vfszip:/P:/dist/target/jboss/server/cmas/deploy/cm-dist-6.5.1-SNAPSHOT.ear/APP-INF/lib/dist-scene-6.5.1-SNAPSHOT.jar/META-INF/cmas/scenes/helpdesk-sales_scene.jar/
- **Since**: 6.0

script.evict.unused.after.hours

- **Module**: cmas-core-server
- **Description**: Determines the number of hours for which unused scripts remain in the cache. After this time, the compiled class of the script is removed. The ConSol CM server checks for scripts to evict every hour.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 24 (default)
- **Since**: 6.11.1.14

script.logging.threshold.seconds

- **Module**: cmas-core-server
- **Description**: Determines the time (in seconds) after which a warning is written to the log file if it is exceeded during script execution.

- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 10 (default)
- **Since**: 6.10.1.0

script.validation.interval.seconds

- **Module**: cmas-app-admin-tool
- **Description**: Sets the interval in seconds between two code checks in the script editor of the Admin Tool and Process Designer.
- **Type**: Integer
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 1 (default value)
- **Since**: 6.11.0.1
- **Removed in**: 6.16.0.0

searchPageSize

- **Module**: cmweb-server-adapter
- **Description**: The global default page size for search results in the Web Client. Only used for new users who have not yet set preferences.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 20
- **Since**: 6.0

searchPageSizeOptions

- **Module**: cmweb-server-adapter
- **Description**: Paging options for the page size of search results in the Web Client.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: no

- **Example value**: 10|20|30|40|50|75|100
- **Since**: 6.0

security.fields.customer.exposure.check.enabled

- **Module**: cmas-restapi-core
- **Description**: If set, only case fields with *Availability via REST* set to “Full access” or “Read-only access” are shown in CM/Track.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.10.5.4

security.restrict.unit.access.to.own.data

- **Module**: cmas-restapi-core
- **Description**: Defines if an additional check is performed when a user logs in as a customer using the REST API, e.g. CM/Track. If set to “true”, only the company of the user or other contacts of the user’s company are returned when the user requests customer data. If set to “false”, no additional security check is performed (not recommended).
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true (default value)
- **Since**: 6.9.2.14

serial.mods.tracking.enabled

- **Module**: cmas-core-server
- **Description**: Low level technical flag deciding whether serial diff tracking for entities is enabled. If enabled, there will be no `StackOverflow` error in case a dependency between two entities (for example user and case) causes an infinite loop first and then as a result, the stack overflow.

> ⚠ Please do not enable this property unless advised by ConSol. It is a low level technical flag with intricate consequences for the system behavior.

- **Type**: boolean
- **Restart required**: no
- **System**: no

- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.10.7.0, 6.11.0.5

server.instance.task.crash.period.seconds

- **Module**: cmas-core-server
- **Description**: Determines the period (in seconds) for detecting live and crashed ConSol CM server instances.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 60 (default value)
- **Since**: 6.11.1.6

server.instance.task.period.seconds

- **Module**: cmas-core-server
- **Description**: Determines the interval (in seconds) between repeated task executions. The task will be executed repeatedly with the given number of seconds between each execution.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 5 (default value)
- **Since**: 6.11.1.6

server.session.archive.reaper.interval

- **Module**: cmas-core-server
- **Description**: Determines the interval (in seconds) when the reaper for archived server sessions is executed (refers to the database table `cmas_user_session`).
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 60
- **Since**: 6.7.1

server.session.archive.timeout

- **Module**: cmas-core-server
- **Description**: Server sessions archive validity timeout (in days). After this time session info is removed from the database (refers to the database table `cmas_user_session`).
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 3 or 7
- **Since**: 6.7.1

server.session.heartbeat.enabled.WEB_CLIENT

- **Module**: cmas-core-server
- **Description**: Determines if the session heartbeat is enabled for the Web Client. If this property is set to “true”, a heartbeat is sent to the ConSol CM server every 20 seconds and with every page reload. If the server does not receive a heartbeat during the configured interval, the session is considered orphaned and ended according to the configured mode.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.11.2.9

server.session.heartbeat.mode

- **Module**: cmas-core-server
- **Description**: Determines when orphaned sessions are ended if the session heartbeat is enabled. There are two possible values:
    - **LAZY**: Orphaned sessions are ended when the same user authenticates again or when there are no more licenses available (default value).
    - **EAGER**: Orphaned sessions are ended immediately.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: LAZY (default value)
- **Since**: 6.11.2.9

server.session.heartbeat.timeout.seconds

- **Module**: cmas-core-server
- **Description**: Determines the period after which a session is considered orphaned if the session heartbeat is enabled and no heartbeat is received. The default value is 300, meaning that orphaned sessions are ended after 5 minutes.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 300 (default value)
- **Since**: 6.11.2.9

server.session.reaper.interval

- **Module**: cmas-core-server
- **Description**: Determines the interval (in seconds) when the reaper for inactive (ended) server sessions is executed, refers to the database table `cmas_user_session`.
- **Type**: integer
- **Restart required**: only Session Service
- **System**: yes
- **Optional**: no
- **Example value**: 10800 (3 hours)
- **Since**: 6.6.1, 6.7.1

server.session.timeout

- **Module**: cmas-core-server
- **Description**: Server session timeout (in seconds) for connected clients (database table `cmas_user_session`). Each client can overwrite this timeout with custom value using its ID (ADMIN_TOOL, WEB_CLIENT, WORKFLOW_EDITOR, TRACK, ETL, REST, WAS) appended to property name, e.g., `server.session.timeout.WAS`.
  Please see also the page customization attributes *updateTimeServerSessionActivityEnabled* and *updateTimeServerSessionActivity*, both of type *cmApplicationCustomization*.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1800
- **Since**: 6.6.1, 6.7.1

serverPoolingInterval

- **Module**: cmweb-server-adapter
- **Description**: Defines the time in seconds for the pooling server to invalidate caches on the web layer.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5
- **Since**: 6.1.0

show.error.stacktrace

- **Module**: cmweb-server-adapter
- **Description**: If this property is set to “true”, the full stracktrace can be shown in the red error message which is displayed in the Web Client if a user action caused an exception.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.12.1.0

skip-ticket

- **Module**: cmas-dwh-server
- **Description**: Cases are not transferred during transfer/update.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.6.19
- **Removed in**: 6.8.1

skip-ticket-history

- **Module**: cmas-dwh-server
- **Description**: The case history is not transferred during transfer/update.
- **Type**: boolean
- **Restart required**: no

- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.6.19
- **Removed in**: 6.8.1

skip-unit

- **Module**: cmas-dwh-server
- **Description**: Contacts are not transferred during transfer/update.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.6.19
- **Removed in**: 6.8.1

skip-unit-history

- **Module**: cmas-dwh-server
- **Description**: The contact history is not transferred during transfer/update.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.6.19
- **Removed in**: 6.8.1

skip.wfl.transfer.cleanup

- **Module**: cmas-core-server
- **Description**: Enables skipping the cleanup process after the deployment of a new workflow version. The cleanup checks whether elements which were deleted in the workflow model need to be preserved and - if not - removes them physically. Changing the default (“false”) to “true” might help if problems with deleted elements during the deployment of a new workflow version occur.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes

- **Example value**: false (default value)
- **Since**: 6.9.4.1

skip.wfl.transfer.translations.cleanup

- **Module**: cmas-core-server
- **Description**: Enables skipping the cleanup of localized properties of removed workflow elements.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.10.5.5

split.history

- **Module**: cmas-dwh-server
- **Description**: Changes the SQL that fetches the case history during DWH transfer so that the history is not fetched for all cases at once but only for one case per SQL.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: false
- **Since**: 6.8.0
- **Removed in**: 6.8.4.0

start.groovy.task.enabled

- **Module**: cmas-app-admin-tool
- **Description**: Determines if it is possible to execute tasks in the Admin Tool, section *Services -> Task Execution*. Set this system property to “true” to enable task execution.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true
- **Since**: 6.9.4.0
- **Removed in**: 6.16.0.0

statistics.calendar

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.client.group

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.contact.role

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.content.entry

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.content.entry.class

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.content.entry.history

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.cube

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.13.0.2

statistics.customer.definition

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.engineer

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.enum.group

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.field.definition

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.group.definition

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.locale

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.localized.property

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.mla

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.project

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.queue

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.resource

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.resource.group

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.resource.history

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.resource.relation.definition

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.resource.type

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.ticket

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.ticket.function

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.ticket.history

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.time.booking

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.timestamp

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.unit

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.unit.history

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.unit.relation.definition

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

statistics.workflow

- **Module**: cmas-dwh-server
- **Description**: Internal DWH property, not to be changed manually. This property is filled automatically by the CMRF to store statistical data from the DWH. This data is displayed in the Web Admin Suite, on the *Monitor* page.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.1

strict.utf.bmp.enabled

- **Module**: cmas-core-server
- **Description**: ConSol CM installations which use a MySQL database with a 2-byte encoding / collation setting cannot handle incoming emails which contain 4-byte UTF8 characters.
  If you set this property to “true”, all 4-byte UTF8 characters are filtered out from email subjects before any database interaction. Since 6.11.2.7, it also filters out 4-byte UTF8 characters from other header fields, and since 6.11.2.8 UTF8 characters are filtered out from the names of email attachments.
  By default, the property value is “true” for MySQL databases, and “false” for any other database where the underlying problem should not occur. Do not change it for a MySQL database, unless you are sure that the database supports 4-byte characters.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: false
- **Since**: 6.10.6.0, changed in 6.11.2.7 and 6.11.2.8

supportEmail

- **Module**: cmweb-server-adapter
- **Description**:
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Since**: 6.0
- **Removed in**: 6.11.0.1

synchronize.archive.timeout.minutes

- **Module**: cmas-core-index-common
- **Description**: Determines the time period in minutes during which index updates are archived. The default value of this property is “60”. If the master node is switched, the new master can recover missing data in case it had not been fully synchronized with the previous master indexing server at the moment of switch.

  ⓘ The property is ignored for non-clustered environments.

- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 60
- **Since**: 6.12.0.0

synchronize.master.address

- **Module**: cmas-core-index-common
- **Description**: Value of `-Dcmas.http.host.port` specifying how to connect to the indexing master server. Since 6.12.0.0 it is possible to switch the node acting as master node by entering the address of the new master node. The index updates are archived for the time configured in the property `cmas-core-index-common`, `synchronize.archive.timeout.minutes`, so the new master can recover missing data in case it had not been fully synchronized with the previous master indexing server at the moment of switch.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 127.0.0.1:80
- **Since**: 6.6.0

synchronize.master.security.token

- **Module**: cmas-core-index-common
- **Description**: The password for accessing the index snapshot via URL, e.g., for index synchronization or for backups. This value is negotiated between nodes internally, please do not change it manually unless advised by ConSol.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes

- **Example value**: token
- **Since**: 6.6.0

synchronize.master.security.user

- **Module**: cmas-core-index-common
- **Description**: The user name for accessing the index snapshot via URL, e.g., for index synchronization or for backups. This value is negotiated between nodes internally, please do not change it manually unless advised by ConSol.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: user
- **Since**: 6.6.0

synchronize.master.timeout.minutes

- **Module**: cmas-core-index-common
- **Description**: Determines how much time (in minutes) the master server may continually fail until a new master gets elected. 0 means that master server will never change (failover mechanism is disabled).
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5 (default value)
- **Since**: 6.6.0

synchronize.megabits.per.second

- **Module**: cmas-core-index-common
- **Description**: Determines how much bandwidth (in megabits) the master server may consume when transferring index changes to all slave servers. Do not use all available bandwidth to transfer index changes between hosts, as doing so will most probably partition the cluster due to some subsystems being unable to communicate.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 85 (default value)
- **Since**: 6.6.0

synchronize.sleep.millis

- **Module**: cmas-core-index-common
- **Description**: Determines how often each slave server polls the master server for index changes.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1000 (default value)
- **Since**: 6.6.0

system.flavour

- **Module**: cmas-core-shared
- **Description**: Allows to define the system type. Possible values are: “dev”, “test”, “stage” and “prod”. On prod systems, it is not possible to perform a scene import with the option to delete all existing data. In the Web Admin Suite, the system type is indicated in the header, which is colored according to the system type: green for dev systems, lilac for test systems, yellow for stage systems and red for prod systems.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: prod
- **Since**: 6.14.0.3

system.name

- **Module**: cmas-core-shared
- **Description**: Allows to give the system a name. In the Web Admin Suite, the name is displayed in the header.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: Test system
- **Since**: 6.14.0.3

task.execution.interval.seconds

- **Module**: cmas-core-server
- **Description**: Time in seconds between the end of a completed task and the start of the next task in the Task Execution Framework.
- **Type**: Integer
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 5
- **Since**: 6.9.4.0

task.execution.notification.mail.from

- **Module**: cmas-core-server
- **Description**: Defines the email address which will be set as From header for notifications to the administrator in case of task execution errors.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: admin@mydomain.de
- **Since**: 6.14.0.8

task.execution.node.id

- **Module**: cmas-core-server
- **Description**: Only relevant in clustered environments. The ID of the node where scripts of the TEF (Task Execution Framework) will be executed. This applies to both scripts called from the workflow and scripts called manually using the Web Admin Suite, which can be started from any node.
- **Type**: Integer
- **Restart required**: yes
- **System**: no
- **Optional**: yes
- **Example value**: 2
- **Since**: 6.11.0.1

task.panel.refresh.interval.seconds

- **Module**: cmas-app-admin-tool
- **Description**: Sets the time in seconds after which the task list in the Admin Tool is refreshed.
- **Type**: Integer

- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 10
- **Since**: 6.10.5.3 (not added automatically during update from versions prior to 6.10.5.3!)
- **Removed in**: 6.16.0.0

themeOverlay

- **Module**: cmweb-server-adapter
- **Description**: Defines the user interface theme (CSS file) for the Web Client layout. Set this property to the name of your custom skin if you are using one.
- **Type**: string
- **Restart required**: yes
- **System**: yes
- **Optional**: yes
- **Example value**: consolINT
- **Since**: 6.0, functionality changed in 6.11.2.0

ticket.delete.timeout

- **Module**: cmas-core-server
- **Description**: Transaction timeout (in seconds) for deleting cases in a bulk.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 60
- **Since**: 6.1.3

ticket.from.incoming.message.accepted.links

- **Module**: cmas-core-server
- **Description**: List of domains to which links are clickable in the case history. The whitelist applies to links in incoming emails, links in comments added via REST API, and links entered in comments and rich text fields in CM/Track. Regular expressions can be used to specify the allowed URLs. It is possible to add several URLs by using a whitespace as delimiter. The URL must start with one of the allowed protocols (http, https, ftp, ftps, file, mailto). All other links are removed, i.e., the link is displayed in the case history as text but it cannot be clicked. If the property is left empty, all links are removed. The regular expression `.+` can be used to allow all domains.
- **Type**: string
- **Restart required**: no

- **System**: no
- **Optional**: no
- **Example value**: https://.*\.consol\.de/.* (allows links to “https://<any>.consol.de/”)
- **Since**: 6.11.1.7

⚠ Please note that whitelisting domains might make ConSol CM vulnerable to cross-site scripting and other attacks. Choose the domains you whitelist carefully!

ticketListRefreshIntervalInSeconds

- **Module**: cmweb-server-adapter
- **Description**: Refresh interval for the case list (in seconds).
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 180
- **Since**: 6.0

ticketListSizeLimit

- **Module**: cmweb-server-adapter
- **Description**: Maximum number of cases in the case list.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 100
- **Since**: 6.0

tickets.delete.size

- **Module**: cmas-core-server
- **Description**: Defines the number of cases deleted in a single transaction.
- **Type**: integer
- **Restart required**: only Session Service
- **System**: yes
- **Optional**: no
- **Example value**: 10 (default value)
- **Since**: 6.8.1

time.buffer

- **Module**: cmas-dwh-server
- **Description**: Number of minutes to extend the date of the start of the live mode.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 5
- **Since**: 6.8.1.11

tinymce.enabled

- **Module**: cmweb-server-adapter
- **Description**: Determines whether Froala (“false”) or TinyMCE (“true”) is used as an editor for the Web Client.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.12.1.0
- **Removed in**: 6.14.0.0

transaction.timeout.minutes

- **Module**: cmas-core-server
- **Description**: Sets the transaction timeout (in minutes) for the task execution service, i.e., one run of a task must finish before this timeout is reached. The changes are visible only for new tasks, the execution of which started after the configuration change.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 10*60 (10 hours - default value)
- **Since**: 6.10

ttl.days

- **Module**: cmas-web-notifications
- **Description**: Determines the time (in days) after which unpinned notifications are removed.
- **Type**: integer

- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 15
- **Since**: 6.12.0.0

tx.read.only.mode.enabled

- **Module**: cmweb-server-adapter
- **Description**: Enables read-only transactions for faster page loading. This transactional behavior was introduced in 6.11.0, and this property acts as a safety guard to restore the old behaviors. Do not change this value unless you are facing transaction problems and advised by ConSol.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: true (default value)
- **Since**: 6.11

unit.description.mode

- **Module**: cmas-core-server
- **Description**: Defines whether contact descriptions in the case history are taken from the database or dynamically rendered using templates. The value, “DYNAMIC”, is a bit more costly from the performance perspective, while “PROTOCOL” is faster but returns historical names which might be outdated. Use “PROTOCOL” if you have lots of history entries from many different contacts.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: PROTOCOL (default)
- **Since**: 6.11.0

unit.replace.batchSize

- **Module**: cmas-core-server
- **Description**: Defines the number of objects to be processed in a single customer replacement batch.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no

- **Example value**: 5
- **Since**: 6.8.2

unit.replace.timeout

- **Module**: cmas-core-server
- **Description**: Determines the transaction timeout (in seconds) of a customer replacement action step.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 120
- **Since**: 6.8.2

unit.transfer.order

- **Module**: cmas-dwh-server
- **Description**: Defines in which order customer field groups should be transferred to the DWH.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: company;customer
- **Since**: 6.6.19
- **Removed in**: 6.8.1

unitIndexSearchResultSizeLimit

- **Module**: cmweb-server-adapter
- **Description**: Defines the maximum number of hints for all contact autocomplete or suggestion components (e.g. when adding contact to a case). Components customizations (maxHints, maxSuggestions) may change that value.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 5
- **Since**: 6.0

unused.content.remover.cluster.node.id

- **Module**: cmas-core-server
- **Description**: Value of a `cmas.clusternode.id` designating which node will remove unused case attachments and contact content entries.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: 1 (assuming cluster node started with the parameter `-Dcmas.clusternode.id=1`)
- **Since**: 6.9.0.0

unused.content.remover.enabled

- **Module**: cmas-core-server
- **Description**: Specifies whether a removal of unused case attachments and contact content entries should take place.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.9.0.0

unused.content.remover.polling.minutes

- **Module**: cmas-core-server
- **Description**: Determines the time interval (in minutes) for checking unused case attachments and contact content entries for removal.
- **Type**: integer
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 15
- **Since**: 6.9.0.0

unused.content.remover.ttl.minutes

- **Module**: cmas-core-server
- **Description**: Minimum interval (in minutes) while removed attachments and comments are preserved before finally deleting them from the database.
- **Type**: integer

- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: 1440
- **Since**: 6.9.0.0

update.6.11.0.0.sleep

- **Module**: cmas-setup-hibernate
- **Description**: Helper property for the update preparation scripts introduced in context of CM database refactoring in version 6.11. This is an optional setting allowing a delay (in milliseconds) after each loop iteration of the preparation scripts. Setting the delay should lower the database load, for example during working hours. This property may be removed after the update preparation tasks finish.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: 0
- **Since**: 6.11.0.0, for use in 6.10.5.x

update.6.11.0.0.timezone

- **Module**: cmas-setup-hibernate
- **Description**: Helper property for the case history migration (the new way of counting history groups). Since 6.11.0.0 the groups are constant (2h time span), but before 6.11.0.0 groups were not constant and depended on the customer's time zone. Migration scripts use an old algorithm to calculate groups and therefore need information about the time zone. The property should be set to the timezone which is most commonly used by the customers. If the property is not set, the default server time zone is used (`TimeZone.getDefault()`). The property should be set before updating to 6.11.0.0 and will be removed automatically after migration. The list of accepted timezones can be found for example here: http://joda-time.sourceforge.net/timezones.html.
- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: Europe/Berlin
- **Since**: 6.11.0.0, for use before updating to this version

urlLogoutPath

- **Module**: cmweb-server-adapter
- **Description**: URL which is opened when the user logs out. There are three configuration possibilities:
  - Empty value: The user is redirected to the login page.
  - <CM URL>/cm-client/logout: The user is redirected to the logout page, which includes a link to log in again.
  - Any URL: The user is redirected to the given URL outside ConSol CM.
- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: https://mycmserver/cm-client/logout
- **Since**: 6.3.1, behavior changed in 6.11.2.0

url.track

- **Module**: cmas-core-server
- **Description**: Determines the URL of CM/Track to be used by the class `LinkService` to generate links to cases in CM/Track. The following example shows how the use the convenience method to generate a link to a case (the `ticket` object is passed as a parameter).

```
url = linkTo.track(ticket);
```

- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: http://localhost:8080/track
- **Since**: 6.13.0.0

url.webclient

- **Module**: cmas-core-server
- **Description**: Determines the URL of the Web Client to be used by the class `LinkService` to generate links to cases, contacts or resources in the Web Client. The following example shows how the use the convenience method to generate a link to a contact (the `unit` object is passed as a parameter).

```
url = linkTo.webclient(unit);
```

- **Type**: string

- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: http://localhost:8080/cm-client
- **Since**: 6.13.0.0

url.webhooks

- **Module**: cmas-core-server
- **Description**: Determines the central URL for webhooks exposed by this system. It is used by the class `LinkService` to generate Webhook submit URLs, and by `intgTokenService` to generate token-secured caller URLs.

```
url = linkTo.webhook;
```

- **Type**: string
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: http://localhost:8080/intg
- **Since**: 6.15.1.0

voCacheEnabled

- **Module**: cmweb-server-adapter
- **Description**: This property enables additional caching for the Web Client, voCaching, of complete objects, thus improving performance.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: yes
- **Example value**: true
- **Since**: 6.11.1.0

Notes:

- Since ConSol CM version 6.11.1.1, the default value is “true” for non-clustered environments. The value is set to “true” automatically during the setup or update of ConSol CM 6.11.1.1.
- When voCaching is enabled and lazy loading is used for folding case history entries, once the user unfolded an entry, he cannot fold it again by reloading the page or opening the case from the workspace.
- When using the dynamic mode for displaying user and contact names in the case history (as configured in the system properties cmas-core-server, engineer.description.mode and cmas-core-server, unit.description.mode), the new version of the user and/or contact name is only displayed after the case has been changed.
- This system property is ignored for clustered environments (environments with cmas-core-shared, cluster.mode set to “true”). In clustered environments, voCaching is always disabled to avoid problems that changes made to objects on one node are not visible on the other nodes.

warmup.executor.enabled

- **Module**: cmas-core-server
- **Description**: Specifies whether the server should asynchronously warm up during startup (e.g., fill some of the internal caches) for better performance.
- **Type**: boolean
- **Restart required**: no
- **System**: yes
- **Optional**: no
- **Example value**: true
- **Since**: 6.9.4.2

webSessionTimeoutInMinutes

- **Module**: cmweb-server-adapter
- **Description**: Session timeout in minutes.
- **Type**: integer
- **Restart required**: yes
- **System**: yes
- **Optional**: no
- **Example value**: 180
- **Removed in**: 6.7.1
- **Replaced by**: cmas-core-server, server.session.timeout

wfl.sticky.transfer.disabled

- **Module**: cmas-core-server
- **Description**: Enables using preserved original names of workflow elements.
- **Type**: boolean
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: false
- **Since**: 6.10.1.0

wicketAjaxRequestHeaderFilterEnabled

- **Module**: cmweb-server-adapter
- **Description**: This enables the filter for Wicket AJAX requests, coming from stale pages with Wicket 1.4 scripting (CM < 6.8.0), after an update to CM > 6.8.0.
- **Type**: boolean
- **Restart required**: yes
- **System**: yes
- **Optional**: yes
- **Example value**: false
- **Since**: 6.8.1

workflowDefinitionsLimit

- **Module**: cmas-was-web
- **Description**: Sets the maximum number of versions of each workflow to be loaded. The number is calculated excluding deployed versions, which are always loaded.
- **Type**: integer
- **Restart required**: no
- **System**: no
- **Optional**: no
- **Example value**: 2
- **Since**: 6.15.2.0

workflow.deploy.cache.eviction.disabled

- **Module**: cmas-core-server
- **Description**: Determines if the infinispan cache should be cleared after deployment (“true”) or not (“false”). If the property is set to “true”, all mappings are removed from the cache.
- **Type**: boolean
- **Restart required**: no

- **System**: no
- **Optional**: yes
- **Example value**: false (default value)
- **Since**: 6.11.1.4

X-Frame-Options

- **Module**: cmweb-server-http-headers
- **Description**: Example property to illustrate the configuration of HTTP headers to be returned with the page in the Web Client. In this case the delivered HTTP header contains the field *X-Frame-Options* with the value “SAMEORIGIN”.

  Each property in the module `cmweb-server-http-headers` represents one header field. The name of the property is the field of the response header, and the value of the property is the field value sent in the header.

  > ⚠ Please be aware that additional HTTP response headers must be correctly defined with the exact spelling as officially specified! Please note also that the correct interpretation and application of these headers is fully in the realm and responsibility of the client browser which requested the page!

- **Type**: string
- **Restart required**: no
- **System**: no
- **Optional**: yes
- **Example value**: SAMEORIGIN
- **Since**: 6.10.8

J.3 Email addresses

This chapter discusses the following:

J.3.1 Introduction to emails in ConSol CM

ConSol CM interacts with one or several email servers to send and receive emails for internal and external communication. The email connections and addresses can be configured separately for different purposes. This allows sending notifications directly to the responsible people instead of to one central account.

The email connections are determined on the following pages of the Web Admin Suite.

- **Email configuration**: Allows to define the email server to be used for incoming and outgoing emails from ConSol CM.
- **DWH administration**: Allows to define the email server to be used for outgoing emails from the CMRF.

The email accounts and addresses are determined on the following pages of the Web Admin Suite.

- **Email configuration**: Allows to define the email accounts for sending and receiving emails from ConSol CM.
- **DWH administration**: Allows to define the email account for sending emails from the CMRF, and the email addresses which should receive notifications from the CMRF.
- **System properties**: Allows to define the email addresses for special purposes used by ConSol CM, e.g. the system-wide reply-to address or addresses which should receive specific notifications.

> ⓘ You can also see and modify the values of all system properties holding email addresses in the *Addresses* tab of the *Email configuration* page.

J.3.2 Important email addresses

- **Administrator email address**: During the installation of ConSol CM, the administrator email address must be provided. It is used as a fallback if no specific email address is configured for a certain purpose later on. It is saved in the system property cmas-core-security, admin.email.
- **Reply-to address**: Set the property cmweb-server-adapter, mail.reply.to to define a reply-to address which is used in all emails sent from ConSol CM.

J.3.3 List of email addresses

The *Addresses* tab of the *Email configuration* page shows a description of all email addresses which are saved in system properties. It also allows you to see and modify the property values:

- cmas-core-security, admin.email
- cmweb-server-adapter, mail.from
- cmweb-server-adapter, mail.reply.to
- cmas-nimh-extension, mail.error.from.address
- cmas-nimh-extension, mail.error.to.address
- cmas-core-security, password.reset.mail.from
- cmas-core-server, mail.notification.sender
- cmas-core-server, mail.smtp.envelopesender
- cmas-core-server, task.execution.notification.mail.from
- cmas-nimh-extension, mail.attachments.validation.info.sender
- cmas-workflow-engine, jobExecutor.adminMail
- cmas-workflow-engine, jobExecutor.mailFrom
- cmas-dwh-server, notification.error.to
- cmas-dwh-server, notification.error.from
- cmas-dwh-server, notification.finished_successfully.to
- cmas-dwh-server, notification.finished_successfully.from

J.4 List of manuals

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

- **Administrator Manual**
  Detailed manual about the configuration of the system and the implementation of worflows in the Web Admin Suite
  Target group: CM experts who adapt ConSol CM for specific use cases
- **DWH Manual**
  Detailed explanation of the ConSol CM data warehouse (DWH) concept, the database schema and a list of all table structures
  Target group: CM experts who are in charge of reporting with ConSol CM
- **Operations Manual**
  Description of the integration of ConSol CM into IT environments and the operation of the system
  Target group: IT administrators
- **Setup Manual**
  Technical description for installing ConSol CM in different IT environments
  Target group: IT administrators
- **System Requirements**
  List of all requirements that have to be met to install ConSol CM
  Target group: IT administrators
- **Release Notes**
  Technical information about all changes to ConSol CM in a given version, including new features and bug fixes
  Target group: CM experts, IT administrators and key users
- **User Manual**
  Introduction to the ConSol CM Web Client
  Target group: end users

J.5 TecDoc server

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

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

J.6 Legal notice

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

J.7 Gender disclaimer

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

J.8 Copyright

© 2024 ConSol Consulting & Solutions Software GmbH - All rights are reserved.

J.9 Trademarks

- The Apache Commons Codec ™ library is a trademark of the Apache Software Foundation. See Apache Commons Codec web page.
- Apache OpenOffice™ – Apache and the Apache feather logos are trademarks of The Apache Software Foundation. OpenOffice.org and the seagull logo are registered trademarks of The Apache Software Foundation. See Apache OpenOffice Trademarks web page.
- Google Maps™ – Google Maps is a trademark of Google Inc. See Google trademark web page for details.
- HAProxy – HAProxy is copyright of Willy Tarreau. See HAProxy website.
- Microsoft® – Microsoft and Windows are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. See Microsoft trademark web page.
- Microsoft® Active Directory® – Microsoft and Microsoft Active Directory are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. See Microsoft trademark web page.
- Microsoft® Exchange Server – Microsoft and Microsoft Exchange Server are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. See Microsoft trademark web page.
- Microsoft® Office – Microsoft and Microsoft Office are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. See Microsoft trademark web page.
- Microsoft® SQL Server® – Microsoft and Microsoft SQL Server are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. See Microsoft trademark web page.
- Microsoft® Windows® operating system – Microsoft and Windows are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. See Microsoft trademark web page.
- Microsoft® Word® – Microsoft and Microsoft Word are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. See Microsoft trademark web page.
- MongoDB® – Mongo, MongoDB, and the MongoDB leaf logo are registered trademarks of MongoDB, Inc.
- NGiNX – NGiNX is copyright of Igor Sysoev and Nginx, Inc. See NGiNX license page.
- OpenStreetMap - OpenStreetMap® is open data, licensed under the Open Data Commons Open Database License (ODbL) by the OpenStreetMap Foundation (OSMF). See OpenStreetMap Copyright and License page.
- Oracle® – Oracle is a registered trademark of Oracle Corporation and/or its affiliates. See Oracle trademarks web page.
- Oracle® WebLogic – Oracle is a registered trademark of Oracle Corporation and/or its affiliates. See Oracle trademarks web page.
- Pentaho® – Pentaho and the Pentaho logo are registered trademarks of Pentaho Inc. See Pentaho trademark web page.
- Vis.js – Vis.js is copyright of Almende B.V. See Vis.js license page.

- Wicket^{TM} - Apache Wicket and Wicket, Apache, the Apache feather logo, and the Apache Wicket project logo are trademarks of The Apache Software Foundation. See, for example, the hint at the bottom of the Wicket home page

Glossary

A

ACF

ACF is the abbreviation of Activity Control Form. ACFs can be used in workflow activities to force the engineer to fill out certain fields before proceeding.

ACIM

Activity item - entry in the history section of a ticket (e.g., comment, email, attachment, time booking entry).

AD

Microsoft Active Directory - an LDAP-based directory service for Microsoft Windows domain networks.

additional customer

Additional customers are customers (companies or contacts) who are interested in the ticket. They are optional and usually have a role indicating the reason why they were added.

AJP

Apache JServ Protocol, see, for example https://en.wikipedia.org/wiki/Apache_JServ_Protocol

B

BI

Business Intelligence - methods, technologies, and architectures to transform data into useful information for business purposes.

C

CFEL

Custom Field Expression Language - Java classes and methods of the ConSol CM API to access data in ticket fields, customer fields and resource fields.

CIDR

Classless Inter-Domain Routing, notation for IP address ranges

CM/Archive

CM/Archive is a ConSol CM add-on which allows archiving tickets, so that old tickets can be deleted from the ConSol CM database.

CM/Doc

A standard module of ConSol CM which enables the engineer via ConSol CM Web Client to work with Microsoft Word or OpenOffice documents prefilled with ConSol CM ticket or customer parameters.

CM/EBIA

ConSol CM addon which allows to create reports directly in the Web Client. The abbreviation means Embedded BI and Analytics.

CM/Phone

The ConSol CM module which provides CTI for CM.

CM/Resource Pool

CM/Resource Pool is an optional add-on which allows to store different kinds of objects as resources in ConSol CM.

CM/Track

CM/Track is the portal of ConSol CM. Customers can access their tickets through CM/Track.

CMDB

ConSol CM database - the working database of the CM system.

CMRF

ConSol CM Reporting Framework - a JEE application which synchronizes data between the ConSol CM database and the DWH.

company

The company is the upper hierarchical level of a two-level customer model. A company can have several contacts.

contact

The contact is the lower hierarchical level of a two-level customer model. A contact can only belong to one company.

CRM

Customer Relationship Management. Approach to manage a company's customers, e.g., to collect data from different sources and integrate the data to generate information which allows, e.g., to optimize the services for the customers.

CTI

Computer Telephony Integration - a denomination for any technology that facilitates interaction between a telephone and a computer.

customer

The customer represents the external side of a ticket. It designates the person or object that gave the reason for creating a ticket. A customer can either be a company or a contact.

customer action

Part of the Action Framework. An action which is performed for a customer object, i.e., a contact or company object.

customer data model

The customer data model is the definition of the customers. It determines the available data fields and possible relations.

customer field

A field where data for customers (contacts or companies) can be stored. Similar to ticket fields for ticket data. Previously called Data Object Group Field.

customer field group

A group of fields where data for customers (contacts or companies) can be stored. Similar to ticket field group for ticket data. Previously called Data Object Group.

customer group

The customer group determines which customer data model is used for its customers and which actions are available.

customer object

A customer (a contact or a company). Formerly called Data Object. The term Unit is used in the programming context.

D

Dashboard

A type of report which integrates data from different sources providing an overall perspective of a certain topic. Often times graphical representation is used.

DWH

Data Warehouse - A database used for reporting and data analysis. In a standard ConSol CM distribution, a DWH is included and only has to be installed and configured.

E

engineer

Engineers are the users who work on the tickets in the Web Client

ERP system

Enterprise Resource Planning - often used for this type of enterprise management software.

ESB

Enterprise Service Bus - a software architecture used for communication between mutually interacting software applications in a service-oriented architecture (SOA).

ETL

Extract Transform Load - extracts data from one source (a database or other source), transforms it, and loads it into a database, e.g., a data warehouse.

F

FlexCDM

Flexible Customer Data Model - the customer data model introduced in ConSol CM in version 6.9. For each customer group, a specific customer data model can be defined.

G

GUI

Graphical User Interface

H

history

The history contains all changes which were carried out for the ticket, customer, or resource.

HMAC

Hash-based Message Authentication Code, message authentication function using hashes

I

IMAP

Internet Message Access Protocol - Internet standard protocol to access email on a remote email server. Can be used as plain IMAP or as secure IMAP (IMAPs). In the latter case, proper certificates are required.

J

Java EE

Java Enterprise Edition

JMS

Java Message Service - Java EE component used to send messages between JMS clients.

JRE

Java Runtime Environment. Provides a Java Virtual Machine for Clients.

K

KPI

Key Performance Indicator - parameter used for performance measurement for companies, projects, etc.

L

LDAP

LDAP is the abbreviation of Lightweight Directory Access Protocol. It is a protocol used to manage login information for several applications.

LDAPS

LDAP over SSL

M

mailbox

Destination to which email messages are delivered. Mailboxes are managed on an email server. ConSol CM can access one or more mailboxes to retrieve emails.

main customer

The main customer is the customer who gave the reason for creating the ticket. The main customer is mandatory for a ticket.

Mule

An open source Java-based Enterprise Service Bus (ESB).

N

NIMH

New Incoming Mail Handler - module for retrieving incoming emails.

P

participants

Participants are users who have a specific purpose, which depends on your business process. Usually, they have to carry out certain tasks within the process.

PCDS

Page Customization Definition Section

Pentaho

PentahoTM is a business intelligence (BI) suite which is available in open source and as enterprise editions.

permission

Permissions determine which objects the users can see in the Web Client and which actions they are allowed to perform. Permissions are always granted via roles, i.e., they are not assigned to a single user but to a group of users sharing a common role. Usually these users belong to the same team and/or have similar functions in the company.

POP

Post Office Protocol - Internet standard protocol to retrieve emails from a remote server via TCP/IP. Can be used as plain POP or as secure POP (POPs).

In the latter case, proper certificates are required.

portal

CM/Track - provides customer access to ConSol CM.

Q

queue

The queue contains thematically related tickets which should be handled in the same way and follow the same business process (workflow). Permissions and other parameters are also defined based on queues.

R

RDBMS

Relational Database Management System - e.g. Oracle ® , MS SQL Server ® , MySQL.

relation

Relations are connections between different data objects in ConSol CM. This can be a relation between two objects of the same type, e.g., between tickets, customers, and resources, or a relation between objects of different types, e.g., between a ticket and a resource or a customer and a resource.

representation

Representations are temporary replacements. The representing user receives a copy of all emails which are sent from the CM system to the represented user's email address. In addition, his ticket list has an option to display the tickets of the represented user.

resource

Resources are objects managed in CM/Resource Pool.

resource action

Part of the Action Framework. An action performed for a resource object.

resource field

A field where resource data can be stored.

resource field group

A group of fields where data for resources can be stored. Similar to ticket field group for ticket data.

resource type

The resource type is the definition of the resources. It determines the available data fields and possible relations and actions.

REST

Representational State Transfer - conventions for transferring data over HTTP connections.

role

Roles are assigned to users. They define the users' access permissions and views.

S

script

Program written for a specific run-time environment that can interpret and automate the execution of tasks. In ConSol CM, scripts are stored in the Web Admin Suite.

search action

Part of the Action Framework. An action performed for the result set of a search.

SMTP

Simple Message Transfer Protocol - standard protocol for sending emails.

T

TAPI

Telephony Application Programming Interface - a Microsoft Windows API which provides computer/telephony integration and enables PCs running Microsoft Windows to use telephone services.

TEF

Task Execution Framework - a ConSol CM module which can execute tasks asynchronously.

template

Templates contain predefined and preformatted text. They can be used for comments, emails, and documents.

ticket

The ticket is the request of the customer which the engineer works on. It is the object which runs through the business process defined by the workflow.

ticket field

A field where ticket data can be stored. Previously called Custom Field

ticket field group

A group of ticket fields where ticket data can be stored. Previously called Custom Field Group.

ticket list

The ticket list is located to the left of the main working area of the Web Client. It shows tickets which are relevant for the current user.

time booking

Time bookings allow the engineers to register the time they worked on a ticket or project.

U

Unit

Java class which represents a customer object. i.e. a contact is an object of class Unit and a company is also an object of class Unit.

user

Users are the people who work with the ConSol CM system. Users who work on tickets in the Web Client are also called engineers.

V

view

Views limit the tickets which are shown in the ticket list in the ConSol CM Web Client to those tickets matching specific criteria (scopes from one or more workflows). Views are assigned to roles.

W

Web Admin Suite

ConSol CM component, web application to configure and manage the ConSol CM system.

Web Client

The Web Client is the primary access to the system for the users. They use the Web Client to work on tickets.

Wicket

Apache Wicket is an open source, component oriented, serverside, Java web application framework. See https://wicket.apache.org/ for details information.

workflow

The workflow is the implementation of the business process managed in ConSol CM. It contains a series of steps which are carried out by the engineers.