Team For Capella

Installation guide

Version v7.0.0, 2024-07-31

Chapter 1. Scope

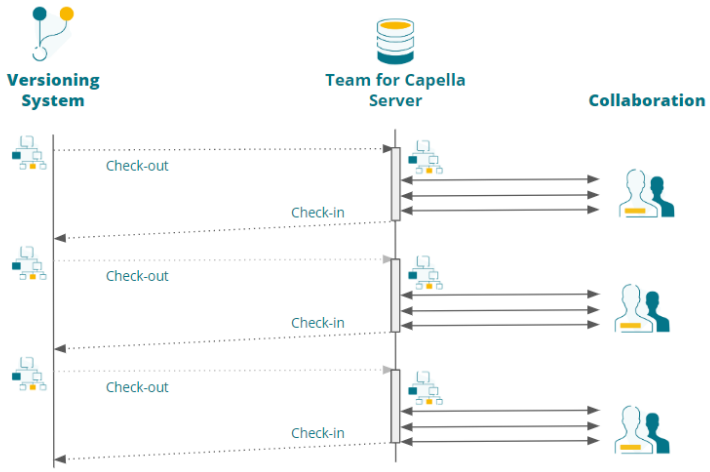
Identification

Description of the product

Team for Capella is a collaborative solution to have several contributors working on the same model, with the granularity as fine as a model element and diagrams.

It decouples the versioning issue which is ensured by a Source Control Management (SCM) tool from the concurrent accesses issue.

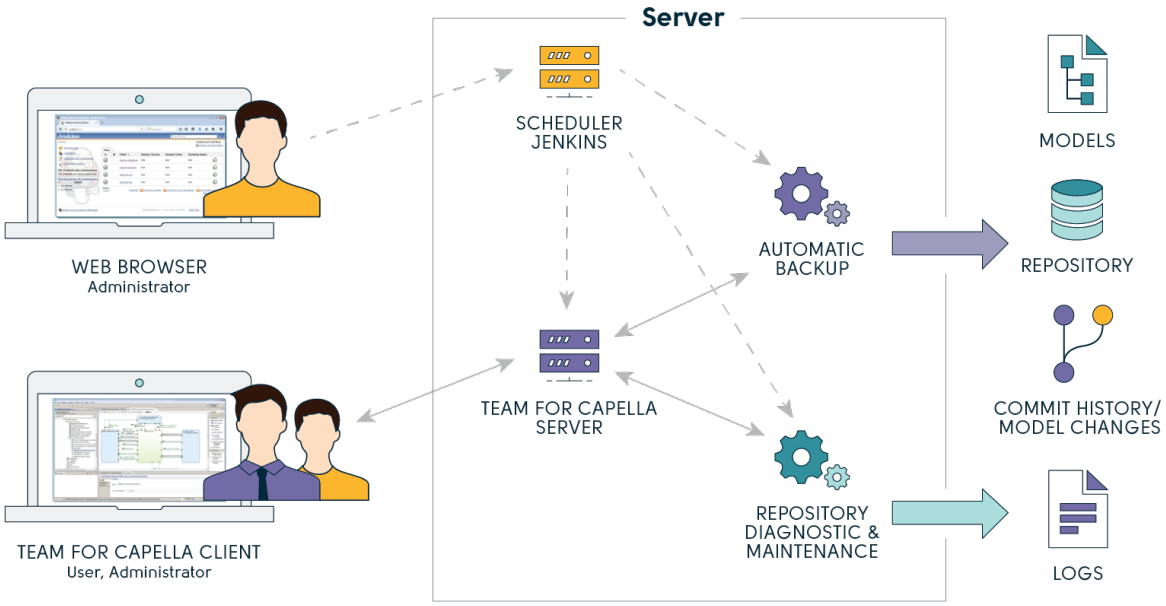
It introduces a shared repository which is populated from the SCM tool and which enables several users to work on the same model.



Users can simultaneously edit the same Capella model without conflicts. Only modified elements are locked, not the whole model, and other users can see in live modifications made by connected teammates. It is no longer necessary to split the model into fragments or to merge concurrent modifications.

It is recommended to version exported models with a SCM tool, for instance Git. Please refer to the Capella embedded documentation at the section Capella Guide > User Manual > Version Control with GIT for more details.

Here is an overview of the Team for Capella architecture:



Team for Capella is composed of three parts:

- A server, to manage the model repositories, and associated features (such as locks, etc.);
- An administration module, to schedule automatic backups (models, changes and database) and trigger diagnostic and maintenance tasks;
- An add-on, packaged as an update site, to bring the multi-user functionalities on top of the standard Capella rich client.

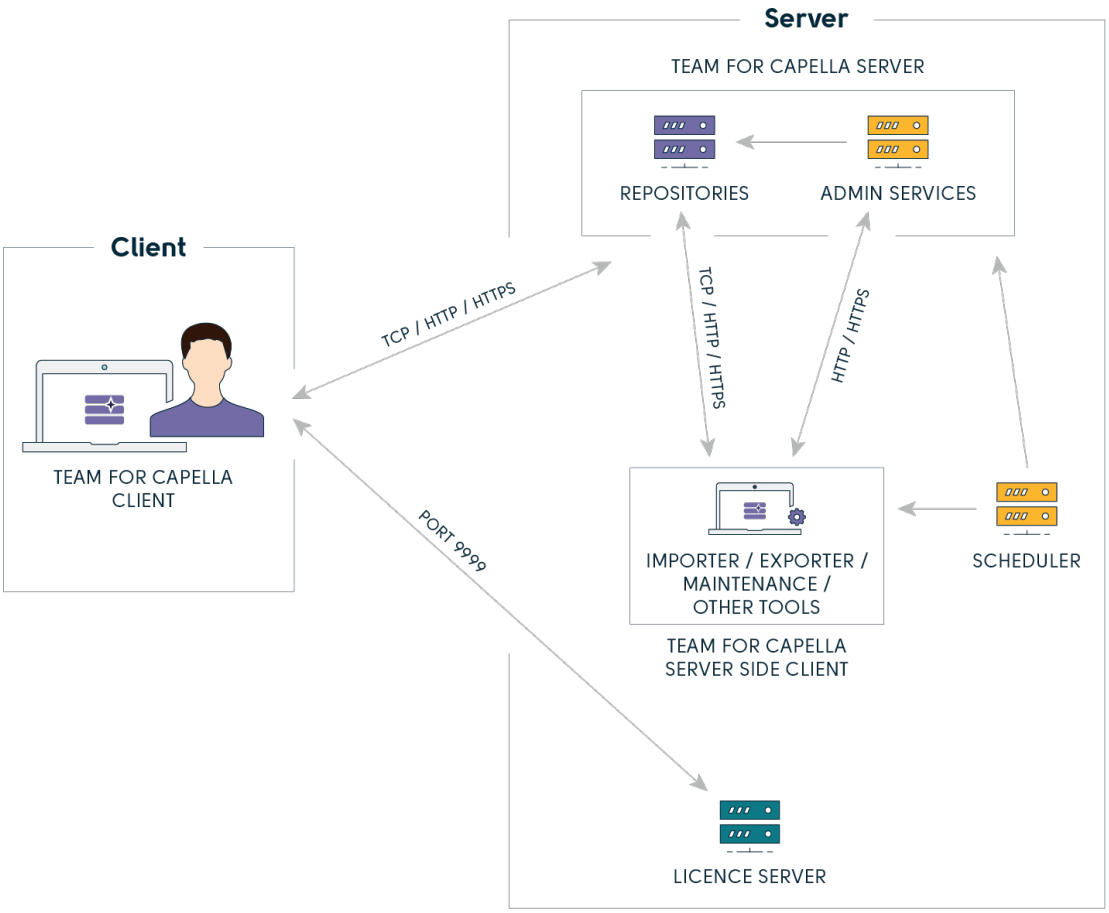
Please refer to the Team for Capella documentation at the section Introduction to Team for Capella > Rationale and Concepts for more information.

Team for Capella is available in 64 bits version for Windows and for Linux.

The Team For Capella server is composed of the CDO repositories server and an HTTP Jetty server. By default, the Jetty admin server is automatically started with the CDO server on the port 8080.

The REST admin server is used:

- to manage repositories with the REST Admin API
- by applications (importer, maintenance application, console application) to execute code on server



Please refer to the Team for Capella documentation at the section System Administrator Guide > Server Configuration for more information.

Deployment modes

Team for Capella can be deployed on different modes:

- **Cloud:** With the Cloud deployment mode, Team for Capella server is installed and administrated by Obeo, and Team for Capella clients are accessible through a remote desktop technology.
- **On-Premise**: With the On-Premise deployment mode, Team for Capella is installed and administrated on the client's infrastructure. The multiple ways to install/deploy Team for Capella are described in section Installation Architecture.

Applicability

This guideline is applicable to following versions of Capella and Team for Capella:

Value name	Value
AppChk	[INSTALLDIR]\capella\capella.exe;
Product Name	Capella x64 / Team for Capella x64
Product Version	7.0.0

This can be followed and adapted to the Linux version of Capella / Team for Capella. Path are

equivalents, the Linux bundle provides .sh scripts which are equivalent to the Windows .bat scripts described in this guide.

Document overview

This document is intended for persons in charge of installing Team for Capella.

It describes the nominal installation, configuration and uninstallation of Team for Capella.

Please refer to the Team for Capella documentation for more details and advanced configuration.

The **Team for Capella Guide** is available from the documentation embedded in the client, but also online, see Referenced Documentation.

To read the embedded documentation, launch a Team for Capella client (after installation), click on the *Help* menu in the top menu bar. Then click on the *Help Contents* button. A web page will appear with all documentation and especially the **Capella Guide** and **Team for Capella Guide** sections.

Chapter 2. Referenced documentation

Table 1. Reference Documents

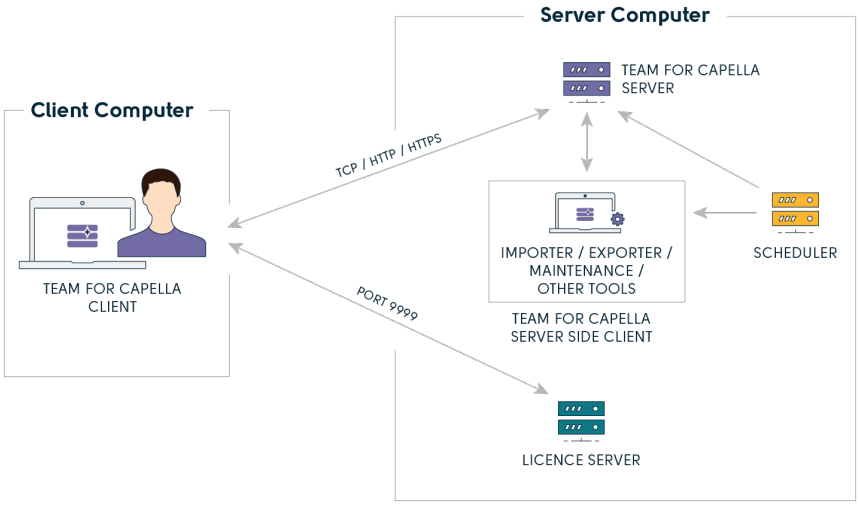
Title	Version
Team for Capella Guide: • Embedded documentation within Team for Capella • online documentation • PDF extracted from online documentation	7.0.0
REST API documentation of the Administration Server	7.0.0
Capella Online Installation Guide	7.0.0
Capella Release Notes	7.0.0
Team for Capella Release Notes	7.0.0

Chapter 3. Team for Capella Server installation

Installation Architecture

There are two main ways to install Team for Capella:

- **Local clients**:
  - server computer runs the Team for Capella Server and Scheduler,
  - n client computers run the Team for Capella Clients,

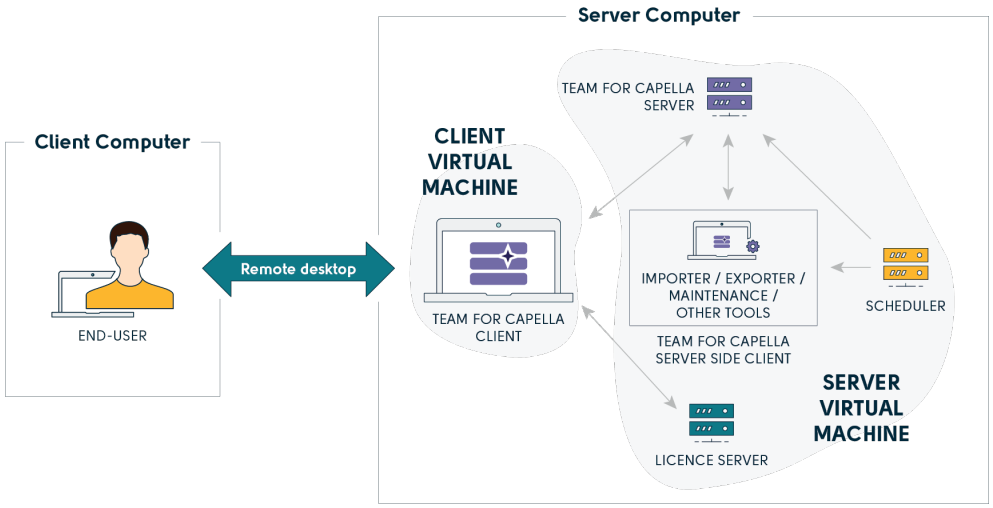


- **Remote clients**:

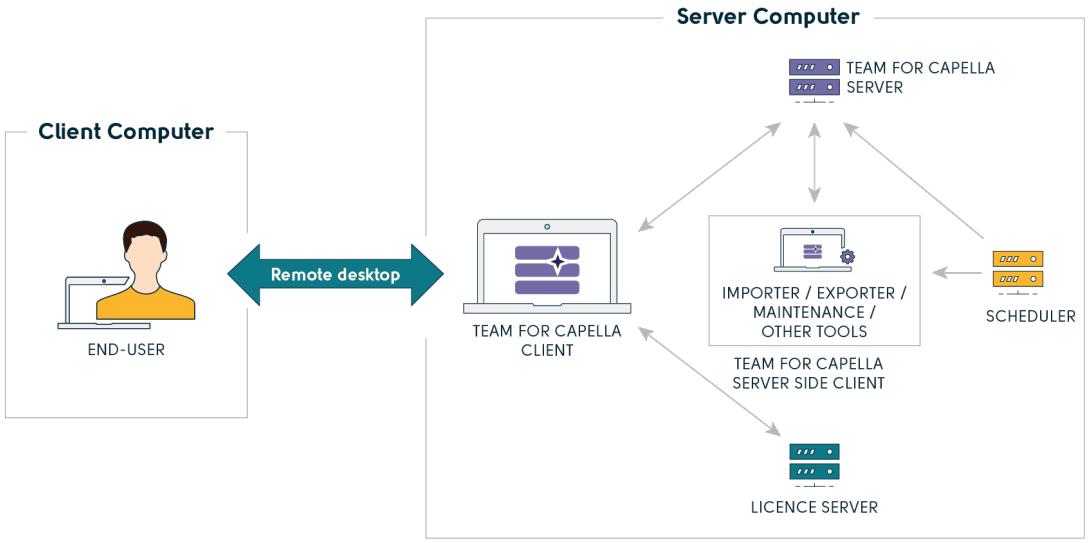
  The same computer runs the Team for Capella Server and Scheduler and n Team for Capella Clients (users connect to this computer using RDP). This allows to make the exchanges between the clients and the server independent of the network's bandwidth.

  In this configuration, the client and server can either be installed on the server computer or separated in two virtual machines. The latter is the recommended installation:

  - Client and server separated in two virtual machines:



- Client and server on the same server machine:



Recommended system requirements

Server computer recommended system requirements

If you installed a tool to act as a scheduler, it is mandatory to run it on the same computer as the Team for Capella Server.

For successful installation of Team for Capella Server, your computer must meet the following requirements:

- Local client installation (server side only):
  - 2 GHz processor,
  - RAM: 4 GB for the Team for Capella server + 3GB for the importer client
  - 15 GB of available hard disk space
- Remote client installation:

    - Multicore processor (2GHz)
        - 2 cores for Team for Capella Server, scheduler and license server
        - 1 core per running Team for Capella Client,
    - RAM:
        - 4 GB for the Team for Capella server + 3GB for the importer client
        - 3 GB per Team for Capella Client,
    - 15 GB of available hard disk space + 2 GB per Team for Capella Client,
    - 2 hard drives are recommended:
        - The first containing system files and software installation files (a SSD hard drive is mandatory if more than 8 users),
        - The second containing the Team for Capella Server file.
- System requirements:
    - Microsoft Windows 10/11 64 bits
        - no known issues with Windows 7/8
    - Microsoft Windows Server 2019/2022 64 bits
        - no known issues Windows Server 2008/2012/2016
    - Compatibility with Open JDK 17, see Capella Online Installation Guide
        - Capella and Team for Capella are configured to use the JRE provided by Capella (Eclipse Adoptium Open JDK 17.0.11, JDK)
    - Jenkins LTS 2.440.3 installed as a service on the server computer.
- Team for Capella database must be stored on a local hard drive,
- Security policies:
    - Virus scanner:
        - Team for Capella Server database files must not be scanned (*.db).
        - In addition, it should not be activated either on Capella models files: `*.aird, \*.capella, *.airdfragment, \*.capellafragment` and `\*.afm` ,
        - The license server hosts a collection of licenses stored in several encrypted `.ols` files. Those licenses pools must not be scanned.
        - Periodic analyses should not be launched when users are working (launch them at night),
    - Firewall:
        - At least 2 ports must be opened: the Team for Capella Server port (by default 2036) and the license server port (by default 9999),
        - In addition, the Scheduler port (by default 8036), the license server monitoring port (8086) and the Admin server port (8080)
- The computer should be fully dedicated to Team for Capella.

Client computer recommended system requirements (only for local client installation)

For successful installation of Team for Capella Client, your computer must meet the following requirements:

- 2 GHz processor
- 3 GB for Team for Capella client
- Microsoft Windows 10/11 64 bits
  - no known issues with Windows 7/8
- Compatibility with Java Runtime Environment 17 (Eclipse Adoptium Open JDK 17.0.11 is provided by Capella)
- Security policies:
  - Virus scanner:
    - It should not be activated on Capella models files: `*.aird, \*.capella, *.airdfragment, \*.capellafragment` and `\*.afm`.

Deployment recommendations

Network

Latency: Client and Team Server

It is recommended to provide a network with the lowest possible latency between the client and the server: in the order of 1 to 10 ms for a round-trip.

Latency: Team server and DB server

It is strongly recommended that the Team server and the DB server are located on the same physical server as latency between the Team server and DB server will impact greatly the overall performances of the solution.As such the best performing deployment is achieved by using the H2 database in embedded mode with its `.db` database file located on the same disk than the Team server.

If there is a requirement on the database that prevents from using H2, make sure that the latency is as low as possible.

Network stability

VPN are not recommended (it is a latency factor) as well as other network elements that could drop connections which are more or less inactive. As such wireless connection are also not recommended as any loss of connectivity might lead to instability in the product and loss of data. However, if a network element of this kind is mandatory, an SSH tunnel could be used as a workaround to avoid client/server disconnections.

Server isolation

It is strongly discouraged to deploy the server on a public WAN. Team for Capella should be the only way to edit the information stored in the database.

Scalability and size of models

Scalability and performances are highly dependent on the design of the domain metamodel, the implementation of this metamodel and the Viewpoint Specification Models. The following figures are given with an Ecore model and the EcoreTools tooling which applies the Sirius best practices.

The minimum physical memory dedicated to the Team server is **4 GB** for a deployment where the expected model size is in the order of **300 000 model elements**. The heap memory available for the server should be increased to support bigger models: **8GB** should support **600 000 model elements**.

The memory usage of the clients will increase when the model which is shared among the clients grows as such these resources might need to be increased for larger models with **8GB** being expected for models with **600000 elements** (the exact value might vary depending on the amount of information each model element holds).

The latency of end user operations requiring a full analysis of the model increase as the model grow, this includes: opening and closing a project, deleting model elements and representations, launching a transformation or a code generation. Opening a project (and hence collecting the model from the network) might take **up to 1 min** for a model with **500K elements**.

Models having **1 000 000 model elements** are the considered the upper limit for a Collaborative Server deployment.

A given server is expected to be used by **10 to 20 users** simultaneously depending on their level of activity.

Disclaimer

Notwithstanding what was stated previously, Team for Capella product is not warranted to run without any error or interruption. Obeo does not make any warranty regarding the statements that are under the chapter Deployment recommendations, this chapter is provided for information purposes.

You acknowledge and accept the risks involved in using these products which could include without limitation, downtime, loss of connectivity or data, system crashes, bad performances or performance degradation.

Installation procedure

The date/time and the time zone of the server must be correct to make the scheduler work as expected.

To use existing models in a new version of Team for Capella, copies of these models have to be kept (in files format) before removing the old version.Once the

> new version is installed, the migration procedure will be performed on these models.

Team for Capella Server installation procedure

Installation

Preparation steps:

1. Extract the archive `TeamForCapella-7.0.0-win32.win32.x86_64.zip` in a given directory.It contains a `TeamForCapella` directory with 5 sub-folders and 1 file:
   - `lic-server`: contains a floating license server which allows several users to share the same product licenses.Each license can be used by only one user simultaneously.
   - `server`: the Team for Capella server
   - `tools`: contains some scripts, properties files and pre-configured jobs to configure a Jenkins as scheduler
   - `updateSite`: the Team For Capella update site for the client
   - `license.html`
2. Download the Capella 7.0.0 bundle from eclipse.org repository.

Client Installation

1. Unzip Capella bundle.
2. Move the capella and samples folders into the `TeamForCapella` directory.
3. Resulting structure of `TeamForCapella`
   - `capella`
   - `lic-server`
   - `samples`
   - `server`
   - `tools`
   - `updateSite`
4. Navigate to the tools folder and execute `installTeamForCapellaInCapella.bat` The installation script will install the Team for Capella features in Capella, update the splash screen and update some properties in `capella.ini` and `config.ini`.

It is configured by default to retrieve the Team For Capella update site in the folder: `TeamForCapella\updateSite`

The *repository* property can be updated in the script to reference it from another location.

> This Capella client (capella folder) should be used only for the Scheduler jobs: it must not be moved or renamed as its .exe files are referenced from the pre-configured jobs (Scheduler) and scripts (tools folder).

> It can also be zipped and provided to user in case of local client installation, Team for Capella Client installation procedure.
>
> In remote clients installation, you need to **copy the full** capella **folder** and **rename it into** *capella_client*. Then this client can be started on Windows Server and accessed with Remote Desktop. If you want to install additional functionality, it will have to be done on *capella_client*. and will not impact the *capella* folder.

Extensions installation

If meta-model extensions or add-ons are needed, use **one** of the following ways to install them:

- Use the Help > Install New Software... wizard if the extension are provided as update sites
- Otherwise if they are provided as dropins
  - Either unzip/copy their binary files in the folder `TeamForCapella\capella\dropins`
  - Or:
    - Unzip/copy them in any folder (it can be a shared folder between this server installation and client installations)
    - Modify the configuration file `TeamForCapella\capella\capella.ini` by adding the following parameter, **after -vmargs** :

```
-Dorg.eclipse.equinox.p2.reconciler.dropins.directory=<ExtensionFolder>
```

> Exactly the same extensions have to be installed on **and on all clients** and on the server (`capella` and `capella_client`). Installation of a new version of an addon may require a migration, see Capella Addon migration.

Jenkins scheduler installation

Team for Capella provides applications to manage the CDO repositories with the shared projects.

These applications can be triggered from Capella, but Team for Capella also provides Jenkins jobs in order to manage them with a web interface.

See how to install Jenkins and our specific jobs in the documentation System Administrator Guide > Jenkins Installation.

Administration features installation

For system administrators, it is useful to install administration features such as the "Durable Lock Management" and "User Management" views and the "User Profiles" feature to manage users access rights.

Two ways to install these features in the Team for Capella Client:

- (recommended) Execute the script `TeamForCapella\tools\installAdminFeatures`

- install them manually from the Team for Capella client
  - "Help > Install New Software"
  - select the T4C update site and check the features under "Team for Capella - Administration".

For more details about administration views and User Profiles, please refer to the documentation in System Administrator Guide > Server Administration > Administration Views and System Administrator Guide > Access Control (User Profiles).

License server installation

Server installation

1. The license server is provided in the `TeamForCapella-7.0.0-win32.win32.x86_64.zip` archive. After the preparation steps (see Team for Capella Server installation procedure), it is located in the folder `TeamForCapella\lic-server`
2. Unzip the `OLS.zip` archive in `TeamForCapella\lic-server\OLS`, the `OLS` folder should contain 4 `.ols` files.
3. You can choose to:
   - either launch the license server from the scheduler's job `License Server - Run` (disabled by default)
   - or directly launch the tool using the command `lic-server -keys ./OLS -verbose`

Additional parameters and documentation can be found in the `DOCUMENTATION.txt` file located in `TeamForCapella\lic-server\`.

The server configuration and more especially the `.ols` files should not be modified, moved or even accessed while the server is operating.Stop the license server before doing such operation.

Client configuration

In order to connect Team for Capella instances with the license server, a connection key must be used to retrieve its address and port.

You should have received this key with the license server bundle and your .ols files.If not please contact the support and provide the IPaddress and port to use to connect to the machine that will host the server in your local network.

The `capella\capella.ini` file has to be modified with the following line to define the configuration of the server (after `-vmargs`):

```
-DOBEO_LICENSE_SERVER_CONFIGURATION=<connection key>
```

This must be done on all clients.

A license token is retrieved at the first connection attempt done by Team for Capella. It is then

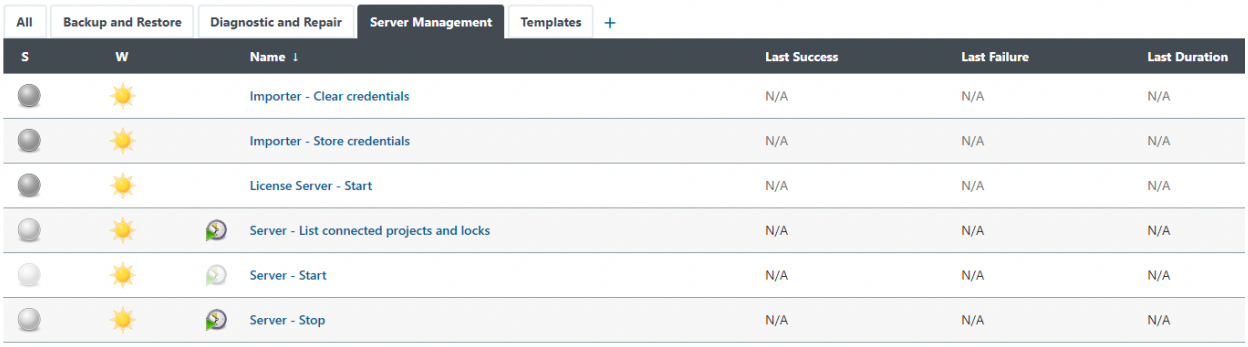
revoked when the last connected Capella project is closed or when Team for Capella closes. This license is verified from times to times with the server while you are using Team for Capella.

Clients are programmatically throttled and will not send requests to the server more frequently than once every minute. This throttling means some of the client actions might have a delay before being distributed to the server, for instance if one user stops using a given feature, 2 minutes of delay at most can be necessary for another user to be able to get the token.

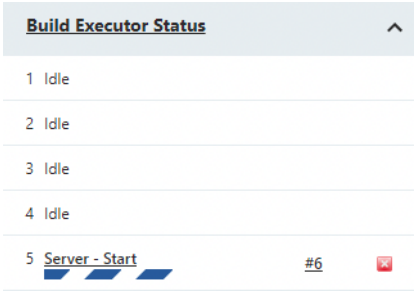
The client/server communication is request based, no connection is kept alive for longer than just a request.

Installation verification

1. Connect to the scheduler admin page using the default URL http://localhost:8036,
2. Check that the job Server – Start has been automatically launched.



On the scheduler main page, the job Server - Run should appear the Build Executor Status section:



3. Once the server is started, it can be used to check the Team for Capella Client installations (see Team for Capella Client installation procedure),
4. Launch the `Server - List connected projects` and check its Console output,
5. Launch the `Database  backup` job from the Backup and Restore tab,
6. Stop the server using the `Server - Stop` job and the `Server - Run` job should be stopped.

Team for Capella server configuration

The Team for Capella Server is provided with a basic configuration. You can find all the documentation to configure it further in System Administrator Guide > Server Configuration. By default, the server authentication is based on a users file.

As a first step of server configuration, it is recommended to check the different authentication methods provided by Team for Capella, such as

- LDAP (subchapter System Administrator Guide > Server Configuration > Activate LDAP Authentication)
- and OpenIdConnect (subchapter System Administrator Guide > Server Configuration > Activate OpenID Authentication).

Also, instead of the default TCP connection, you can further increase security by having the connection to the server using WSS (see in Specific Team for Capella configuration, Tools adaptations, and Client adaptations).

To manage your repository with its shared project, it is recommended to check the documentation of the Jenkins scheduler in Project Administrator Guide > Jenkins Configuration.

How to Reuse previous server configuration when configuring v7.0.x server

When installing a new version of Team for Capella, some parts of the server/client configurations of the previous version can be reused.

Procedure

- Manually import your data or launch the backups jobs.
- Launch the `Server  Stop` job, terminate the `License Server - Run` job or end tasks `server.exe` and `lic_server.exe` in task manager (processes sheet)
- If you were using a Team for Capella version before 7.0.x, stop `TeamForCapellaScheduler` (service sheet) and remove it (see Uninstallation procedure).
- Keep the entire Team For Capella installation which contains all the files you may reuse.
- Choose among the following configuration you want to reuse

Server CDO

- Copy part of the `C:\xxx\T4C\server\configuration\cdo-server.xml` file which contains
  - the acceptor tag can be reused
  - the repository configurations can be reused except for *userManager*, *securityManager* tags.
- if the server was configured with the authenticated configuration, the content of `C:\xxx\T4C\server\configuration\users.properties` can be kept as is.
- If the server was configured with the user profile configuration, the user profile model can be reused. Follow the steps described in System Administrator Guide > Access Control > User Profiles > Import/Export User Profiles Model.
- You may also consider all properties or parameters you may have set in the `server.ini` file.
- If you are doing a new installation with the same version, you can reuse the database. Copy the `C:\xxx\T4C\server\eclipse\db` folder. Note that the database cannot be reused if the *Audit mode* activation changes between both installations.

Scheduler

- In previous versions (before 5.1.0, included), the scheduler was packaged inside Team for Capella, in the `C:\xxx\T4C\scheduler\jenkins_home` folder
  - `config.xml` contains the global Jenkins configuration which can be reused
  - `jobs\` contains the jobs definition with the build history. `jobs\<xxxjob>\config.xml` can be reused modulo some parameter changes that may be needed (for example for importer executable call).
- Since 5.2.0, Team for Capella does not pack a third party tool to manage the repository content (such as Jenkins). You will need to access its installation folder or *Home* directory.
- Customization made on those files/folders might be reported on your new custom installation made for this version.
- Check the modification done in each job with the **Job configuration history** available in each job.
- It is possible to create a v6.1 folder in the Scheduler interface, move the existing jobs in this folder, and check the Jenkins scheduler installation section to update Jenkins to the recommended version and install the new jobs. Then compare your v6.1 jobs with v7.0 jobs.

Tools

- You may consider all system properties or parameters you may have set in the **importer.bat** file (**or importer.ini for older versions**). That information has to be transferred and adapted
  - From `C:\xxx\T4C14\capella\eclipse*importer.bat*` or `C:\xxx\T4C52\tools*importer.bat*`
  - To `C:\xxx\T4C\tools*importer.bat.*`
- Do the same for other scripts you have modified In `C:\xxx\T4C52\tools\\`: **`maintenance.bat`** ,**`command.bat`**.

Client (in remote clients installation)

- You may consider all properties or parameters you may have set in the `eclipse.ini` file. That information have to be transferred to `C:\xxx\T4C\capella_client\capella.ini.`
- If the IP address of the server did not change, you can keep the `OBEO_LICENSE_SERVER_CONFIGURATION` property system (just copy `-DOBEO_LICENSE_SERVER_CONFIGURATION=116168566473385529985114840⋯`.)

License server

- The license server version is now provided in the Team for Capella archive.
- The `*.ols` files may have been thought up to be reusable for your new Team For Capella version. In this case, you can keep your `\*.ols` files.

User Accounts Management

Several modes of access control can be used for each repository on the server:

- Identification (default mode): each user defined in the file `user.properties` is authorized to read and/or modify all models present on the repository.
- User Profiles: discriminating user rights are defined in a User Profiles model stored in the database.
- LDAP Authentication: this mode allows to authenticate with a LDAP server. It can be also used with authenticated or with user profiles.
- Not Authenticated Access: anyone can read and/or modify all models on the repository.

For the complete access control configuration documentation, refer to the documentation System Administrator Guide in chapters:

- System Administrator Guide > Access Control
- System Administrator Guide > Server Configuration
- System Administrator Guide > Server Configuration > Not Authenticated Configuration
- System Administrator Guide > Server Configuration > Authenticated Configuration
- System Administrator Guide > Server Configuration > Activate LDAP Authentication
- System Administrator Guide > Server Configuration > Activate OpenID Connect Authentication
- System Administrator Guide > Server Configuration > Activate WebSocket Connection
- System Administrator Guide > Server Configuration > Activate SSL Connection
- System Administrator Guide > Server Configuration > User Profile Configuration

In the current version, Team for Capella is configured with the “Identification” access control mode, ie. passwords are not encrypted. Refer to System Administrator Guide > Access Control if you need Authentication or Authorization mechanisms.

Uninstallation procedure

To uninstall Team for Capella:

1. Do a backup of projects stored on the Team for Capella Server,
2. Stop the server,
3. Stop the scheduler,
4. If a Windows Service was created, remove it:
   a. Using a Windows Command Line (as administrator), go to the scheduler directory,
   b. Execute the following command: `sc delete TeamForCapellaScheduler` (alternative command: modify the `winservice.bat` to comment the installation steps and uncomment the uninstallation steps).
5. If you installed a third party tool to manage your repository, such as Jenkins, uninstall it.
6. Remove the installation directory.

Chapter 4. Team for Capella Client installation

Requirements

Recommended system requirements

For successful installation of Team for Capella Client, your computer must meet the following requirements:

- 2 GHz processor
- 3 GB for (for big models, 4GB RAM)
- Microsoft Windows 10/11 64 bits
  - no known issues with Windows 7/8
- Compatibility with Java Runtime Environment 17 (Eclipse Adoptium Open JDK 17.0.11 is provided by Capella)
- Virus scanner should not be activated on Capella models files: `*.aird, *.capella, *.airdfragment, *.capellafragment and *.afm`.

Installation Procedure

Team for Capella Client installation procedure

Default client installation

You may take your Team For Capella client as it has been installed during the server installation.In that case, you need to make an archive of the capella folder and distribute it to users.Remember to include the samples folder if you want to ship the example model.

For Windows and Linux, it possible to follow the server installation procedure without the server nor Jenkins deployment and keep only the capella folder in the end.

Before archiving the Capella client that will be deployed on others computers, don't forget to remove the workspace used for the installation from the recent workspaces list (`Window > Preferences > General > Startup and Shutdown > Workspaces`).

It is also possible to manually install Team for Capella but note that the installation with the `tools/installTeamForCapellaInCapella.bat` is recommended.

Manual client client installation

To manually install Team for Capella client in a Capella bundle (Windows/Linux/macOS), please follow the next steps:

1. Extract the archive `TeamForCapella-7.0.0-win32.win32.x86_64.zip` and keep only the `updateSite` and `tools` folders.
2. Download and unzip Capella 7.0.0 bundle (https://www.eclipse.org/capella/download.html)
3. Launch Capella (`capella.exe` for Windows, capella for Linux, Capella for macOS)
4. Click Menu *Help/Install New Software...* , add the archive file in the `updateSite` folder and select features in the *Team for Capella* category:
   - Team for Capella
   - User Interface to request a License
   - Other features from *Team for Capella* and *Team for Capella – Administration* categories are optional and not installed by the installation script.
5. Then you need to do the additional tasks performed by the installation script:
   - change the splashscreen:
     - replace the value of `osgi.splashPath` by `platform\:/base/plugins/com.thalesgroup.mde.melody.collab.ui` in `capella\configuration\config.ini` (`Capella.app/Contents/Eclipse/configuration/config.ini` for macOS)
   - copy the `license.html` file from `tools` to `capella` folder
   - avoid performance impact of CDO internal tracing:
     - add `-Dorg.eclipse.net4j.util.om.trace.disable=true` at the end of the `-vmargs` section in `capella\capella.ini` (`Capella.app/Contents/Eclipse/capella.ini` for macOS)
   - avoid third parties components logging noise
     - add `-Dlogback.configurationFile=configuration/logback.xml` at the end of the `-vmargs` section in `capella\capella.ini` (`Capella.app/Contents/Eclipse/capella.ini` for macOS)
     - copy the `logback.xml` file from `tools/resources/client_rootfiles` to `capella/configuration` folder (`Capella.app/Contents/Eclipse/configuration/` for macOS)
   - add possibility to override default repository connection information:
     - add `-pluginCustomization pluginCustomization.ini` before the `-vmargs` section in `capella\capella.ini` (`Capella.app/Contents/Eclipse/capella.ini for macOS`)
     - copy the `pluginCustomization.ini` file from `tools/resources/client_rootfiles` to `capella` folder (`Capella.app/Contents/Eclipse/` for macOS)
   - add the `license.html` in `capella` folder, copied from `tools\license`.
6. Add the license server configuration key as described in Client configuration. If you have installed Team for Capella in a Capella bundle already completed with some add-ons, remember to check if they provide a “CDO Feature” for Team for Capella compatibility.

Steps 4 and 5 are done by the installation script.

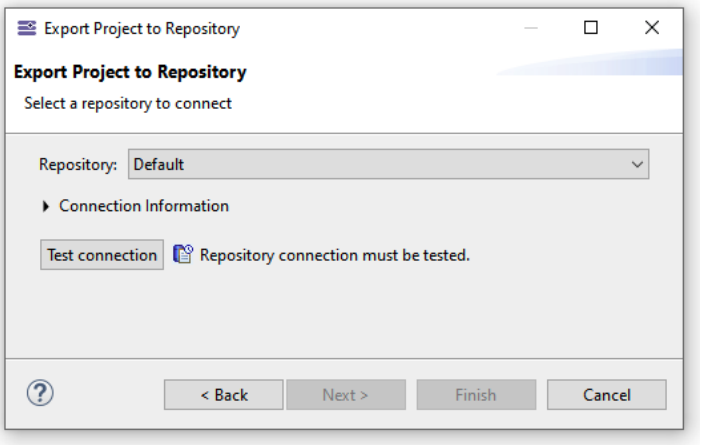
Team for Capella Client configuration

For the complete client configuration documentation, refer to the chapter User Guide > Client Configuration > Preferences

1. Launch Team for Capella (`capella/capella.exe`),
2. Optional: clean the user's Secure Storage (it contains the save Login/Password, if "Remember Me" option was used): Go to menu **Window** > **Preferences** > **General** > **Security** > **Secure Storage**,
   a. Open the `Contents` tab,
   b. Select `[Default Secure Storage]`,
   c. Click on `Delete`,
   d. Upon request, restart the Team for Capella client.
3. Go to the menu **Window** > **Preferences** > **Sirius** > **Team collaboration**
   a. Set the server location with the hostname or the IP address of the Team for Capella Server (localhost if the server is setup on the same machine),
   b. Click on `Apply`

Verification installation procedure

1. **Create a new project** > **Right click in the Capella Explorer** > **New** > **Capella Project**
   a. Call it Test for example > Finish
2. Export the project to the remote repository
   a. Right click on the project > Export > Team for Capella > Export model to remote repository>Next,
   b. (Optional) Expand *Connection Information* if the deployed repository with default parameters
   c. Click on Test Connection :

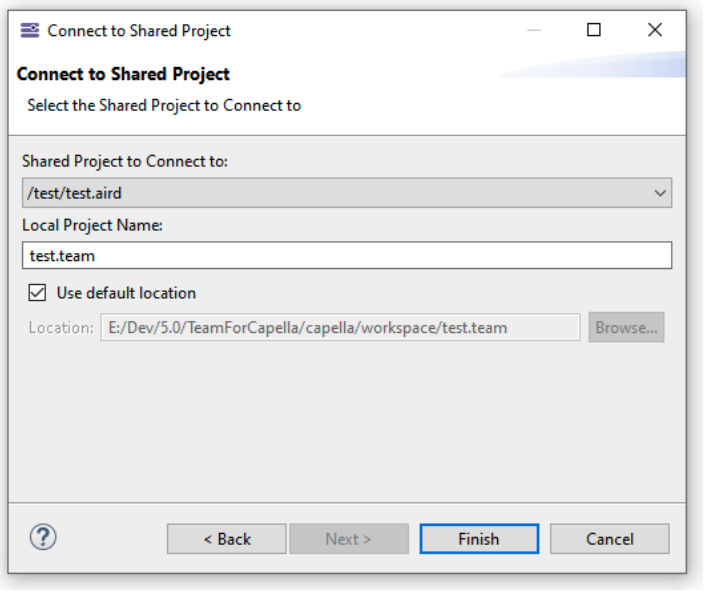


   d. Provide the user name and password (by default: user1/user1, user2/user2, user3/user3, admin/admin).
   e. Click on Finish.
3. Connect to the remote project previously exported,

a. Right click in the Capella Explorer > New > Capella Connected Project

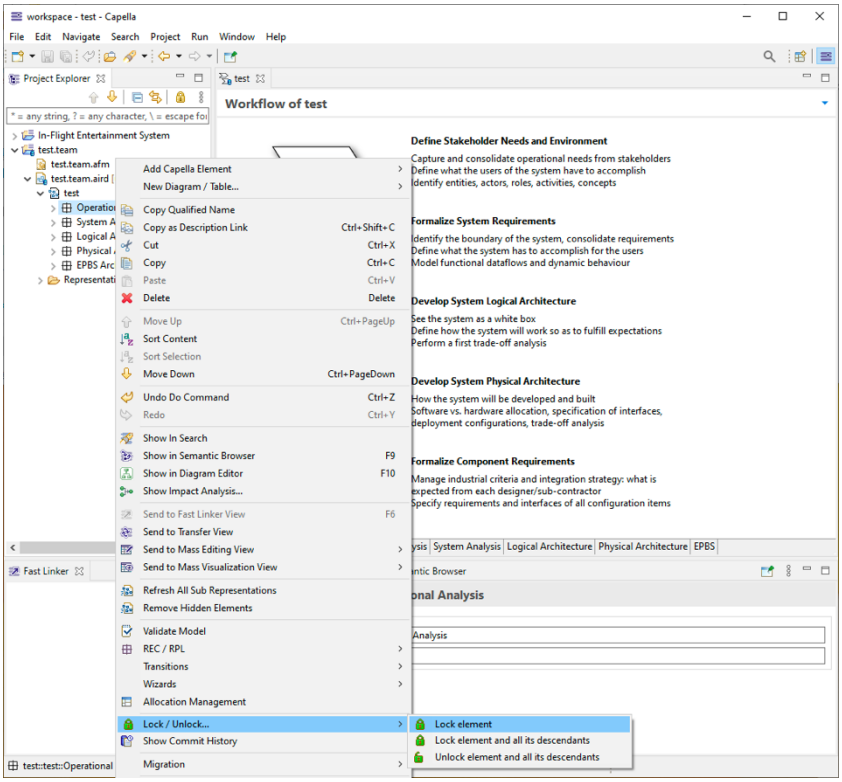
b. Test Connection,

c. Select sthe Shared project from the list:



d. Click on Finish,

4. You should now be able to work on the project on the remote repository.



Chapter 5. Migration process

Version compatibility

Table 2. Version Compatibility (sorted by delivery dates)

Team for Capella	(based on) Capella	(third party software) Sirius
1.0.3 (32bits & 64bits)	1.0.3	Sirius 3.1.6
1.1.4 (32bits & 64bits)	1.1.4	Sirius 4.1.9
1.2.2 (64bits)	1.2.2	Sirius 5.1.4
1.3.2 (64bits)	1.3.2	Sirius 6.1.4
1.4.0 (64bits)	1.4.0	Sirius 6.3.0
1.4.1 (64bits)	1.4.1	Sirius 6.3.1
1.4.2 (64bits)	1.4.2	Sirius 6.3.3
5.0.0 (64bits)	5.0.0	Sirius 6.4.0
5.1.0 (64bits)	5.1.0	Sirius 6.5.0
5.2.0 (64bits)	5.2.0	Sirius 6.6.0
6.0.0 (64bits)	6.0.0	Sirius 7.0.1
6.1.0 (64bits)	6.1.0	Sirius 7.1.0
7.0.0 (64bits)	7.0.0	Sirius 7.4.1

Model migration from previous version to v7.0

“**The CDO Framework cannot handle multiple version of the same metamodel on the same repository. Capella projects of different Capella versions needs to be exported on separate repositories**”.

To use a previous version model in Team for Capella v7.0, a migration must be done to be compliant with the new metamodel and file extensions.

This model migration is provided by Capella and must be done between each minor version (from v1.4.x to v5.x, for example).

The process to migrate a model to v7.0 from a shared repository follows the following steps:

1. If Team for Capella users have created local diagrams (i.e.: diagrams stored in the local `.aird` file), they have to move all diagrams they want to keep to a remote `.aird` or `.airdfragment` (“**Move Diagrams**” action),

2. Import locally the model to migrate (“**Import... / Team for Capella / Import model from remote repository**”) using previous version (v5.x) of Team for Capella. Make sure that the associated team server is running before importing from remote repository. Another way is to use the last valid import of the Scheduler.

→ Make a baseline of the imported model,

3. Migrate the previously imported model (“**Migrate Project toward current version**”) in a Team for Capella v7.0. (The migration can also be done in Capella v7.0).To do so, please refer to *Migration* dedicated chapter of Capella online installation guide,

   → Make a baseline of the migrated model,

4. Export the migrated model to the Team for Capella server v7.0 (“**Export...** / **Team for Capella** / **Capella Project to Remote**”) using Team for Capella v7.0. Make sure that associated team server is running before exporting to remote repository.

After performing these steps, the model in the shared repository is in right version.Team for Capella has to be upgraded on client’s computers.Then users can connect and work on the model.They do not need to do any migration.

Capella Addon migration

Some Capella addons extend the Capella metamodel with new metaclasses. Installing a new version of an addon may introduce a different version of the metamodel extension that alters existing metaclasses. In that case, a migration will be needed, Capella projects using these addons will need to be exported to a new repository. Therefore, the same procedure as the previous part is needed.

Model migration from an older version

The model migration is necessary between each version considering only minor and major version change.

- For the first model migration, you need to reproduce only steps **1**, **2** and **3** described above.
- For the following intermediary model migrations, you need to reproduce only step **3**.
- For the last model migration, you need to reproduce only steps **3** and **4**

For example, you start from v1.4.2 model

- do steps **1**, **2** and **3** with *previous version = 1.3.2* and new version = 1.4.2,
- do step **3** with *previous version = 1.4.2* and *new version = 5.2.0*
- do step **3** with *previous version = 5.2.0* and *new version = 6.1.0*
- do steps **3** and **4** with *new version = 7.0.0.*

Changes for Server Installation from v6.1 to v7.0.0

Changes in Server Configuration

Changes in `admin-server.properties`

- The property `admin.server.api.project.get.impl.config.XMLImportFilePath` in v6.1 becomes `admin.server.api.project.get.impl.config.importFilePath` in v7.0.0

- The property `openIDConnect.webServerURL` in v6.1 becomes `auth.openIDConnect.webServerURL` in v7.0.0

Changes in `cdoServer.xml`

- The websocket configuration `<acceptor type="ws" listenAddr="YourAcceptorName" />` in v6.1 becomes `<acceptor type="ws" name="YourAcceptorName" />` in v7.0.0

Changes in Jenkins Jobs Installation/Configuration

The recommended Jenkins version is now 2.440.3 LTS for the Team for Capella 7.0.0 and the required plugins list has changed. You can read more about the jenkins installation in section System Administrator Guide > Jenkins Installation of the user manual.

The configuration of the Jenkins jobs installation is no longer directly in the installation script `install-TeamForCapellaAppsOnJenkins.bat`/`install-TeamForCapellaAppsOnJenkins.sh`. In v7.0.0, the configuration is now in a separate file `install-TeamForCapellaAppsOnJenkins.properties`. You can read more about the Jenkins jobs installation in section System Administrator Guide > Jenkins Installation > Install Jenkins plugins and jobs required for Team for Capella of the user manual.

Changes in Jenkins Jobs

The following jobs have been renamed:

- `License Server - Start` in v6.1 becomes `License Server - Run` in v7.0.0
- `Server - Start` in v6.1 becomes `Server - Run` in v7.0.0
- `Projects - Import` in v6.1 becomes `Projects - Import - repoCapella` in v7.0.0

The following jobs have been added:

- `Repository - Commit history`, more information in the section Project Administrator Guide > Jenkins Configuration > Team for Capella Scheduler > Backup and Restore > Repository — Commit history of the user manual
- `Repository - Import projects from history`, more information in the section Project Administrator Guide > Jenkins Configuration > Team for Capella Scheduler > Backup and Restore > Repository — Import projects from history of the user manual
- `Repository - List projects`, more information in the section Project Administrator Guide > Jenkins Configuration > Team for Capella Scheduler > Backup and Restore > Repository — List project of the user manual

A new parameter was added to select the repository for the following jobs:

- All `Repository` Jobs
- All `Tools` Jobs
- `Projects - Export`

Other Changes

In the importer application:

- The parameter `-XMLImportFilePath` in v6.1 becomes `-importFilePath` in v7.0.0
- The parameter `-archivefolder <folder>` in v6.1 is replaced by two parameters `-archiveProject true -outputFolder <folder>` in v7.0.0

Chapter 6. Performance consideration feedback

This chapter will provide you some useful information about the performance you should expect and about performance on production context (with big sized model and many users)

Reference time

You may wonder, if the performances you have are normal, that is, if the time elapsed, when doing usual action in Team for Capella, is the right one compared to what should be expected.

Ensuring that the elapsed time doing your actions is the expected one, will help you to ensure that there are no issues that could come from other cause like network lag, lack of memory, antivirus process etc..

Case Studies

Some studies have been done on reference Capella models:

- The open source *IFE* model packaged with Capella
- *Combined IFEs*, a modified version of *IFE* which is artificially swollen to have an important sized model.

The next figure presents some characteristics on those models and the expressions used to compute them.

Table 3. Reference models

Characteristics	**Project**		
	IFE	**Combined IFEs**	
File (size in KB)			
Project	19 179	203 310	
.capella	1 475	13 464	
.aird	17 705	189 846	
Model (number of elements)			**AQL expressions to use in Interpreter view**
Capella elements	5 378	49 389	aql:self.eAllContents()→size() *select the first child of the .aird file in the Project Explorer*

Characteristics	Project		
Components	110	1 230	aql:self.eAllContents(cs::Component)→size() *select the first child of the .aird file in the Project Explorer*
Functions	224	2 652	aql:self.eAllContents(fa::AbstractFunction)→size() *select the first child of the .aird file in the Project Explorer*
Diagrams	111	880	aql:self.eResource().getContents()→filter(diagram::DDiagram)→size() *select a diagram in the Project Explorer*
Tables	1	8	aql:self.eResource().getContents()→filter(table::DTable)→size() *select a diagram in the Project Explorer*
Elements displayed in all representations	6 384	70 850	aql:self.eResource().getContents()→filter(viewpoint::DRepresentation).eAllContents(viewpoint::DRepresentationElement)→size() *select a Diagram in the Project Explorer (only for local projects)*

Refer to How to get characteristics of your model to see how to get the characteristics of your own model in order to compare it with those reference models.

Measures

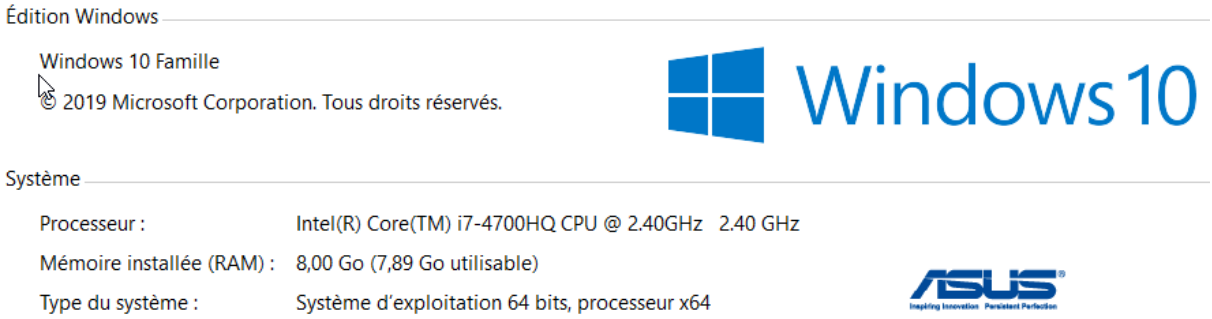
Measures contained in this document give some idea and order of magnitude of the performances expected for comparable environments.

They have been taken on two different deployments:

- *Local*: single computer with both client and server - IFE and Combined IFEs
- *OVH*: two virtual machines, one for the client and one for the server as recommenced in Deployment recommendations.

Computer used for the tests

Local Windows computer



OVH Windows computer

Reference elapsed time

The memory allowed to the Team for Capella client can be configured with the *capella.ini* file beside *capella.exe* executable.

Table 4. Measured time on test models (5.0.0)

Elapsed time (s)	IFE	Combined IFEs
	Xmx 3GB	Xmx 3GB
Export To Server	28,7	114
Open Session (first time)	3,7	9
Test Close Session	1	3
Open Session	2	6,5
Open the diagram (1)	4	6
Create a LFBD diagram on Root Logical Function. (2)	3	6
Save the session after steps from (3)	2	7

Analysis and conclusion

The majority of the scenarios could be done with the minimum recommended memory of 3Go.

Nevertheless, with model of big size (*Combined IFEs*), we need more RAM to work with constant performance for some actions like *Refresh all representations* or *Validate* on the whole model.

If the *Team for Capella* Client reaches, at execution, the maximum allowed memory, the performance may drastically fall particularly for actions that consume a lot of memory such as:

- Import a Capella project locally from the server
- Do the V*alidation* of the whole project
- Start *Refresh All Representation* action
- Save if many objects have been changed or deleted
- Some particular Capella functionality such as *functional transition*

Dedicated tests on *Combined IFEs*

The aim of this test session was to ensure that Team for Capella is able to have to keep the right performances after having used for a while.

- *Combined IFEs* is used as the Capella Project. It is a large sized project.
- 20 users are connected to the Capella project on the server
- 8 users are really working on the model. During the test, they will do standard actions such as
  - Open many diagrams
  - Create, Refresh, modify diagrams
  - Make functional transition for some of them
  - Regularly create semantic element through diagram or not
  - Use tool in diagram palette
  - Use the *Activity Explorer,* F8 and F9 to navigate
  - Validate some parts of the model

But also, other actions that demands significant amount of RAM

- The tests have been done with 21 virtual machines hosted on OVH cloud (20 clients and 1 server):
  - Client Xmx=3Go
  - Server Xmx=4Go

Server

Throughout the test, the server never reaches its maximum allowed memory of 4Go RAM and not even 3Go. It keeps constant performance.

Client

The client behaves constantly with good performance throughout the tests.

Nevertheless, the client may reach 3Go of memory if high memory consumption actions are performed such as:

- Import a Capella project locally from the server
- Do the *Validation* of the whole project
- Start *Refresh All Representation* action

Once the maximum allowed memory is reached:

- The user keeps having good performance for standard actions
- But the performance may fall drastically for high memory consuming actions. For that cases, the performance issue can be solved
  - using more allowed memory for the client (Xmx=4 or more)
  - simply restarting the client

Note that the potential performance issue encountered by one user has no impact on another user

conclusion

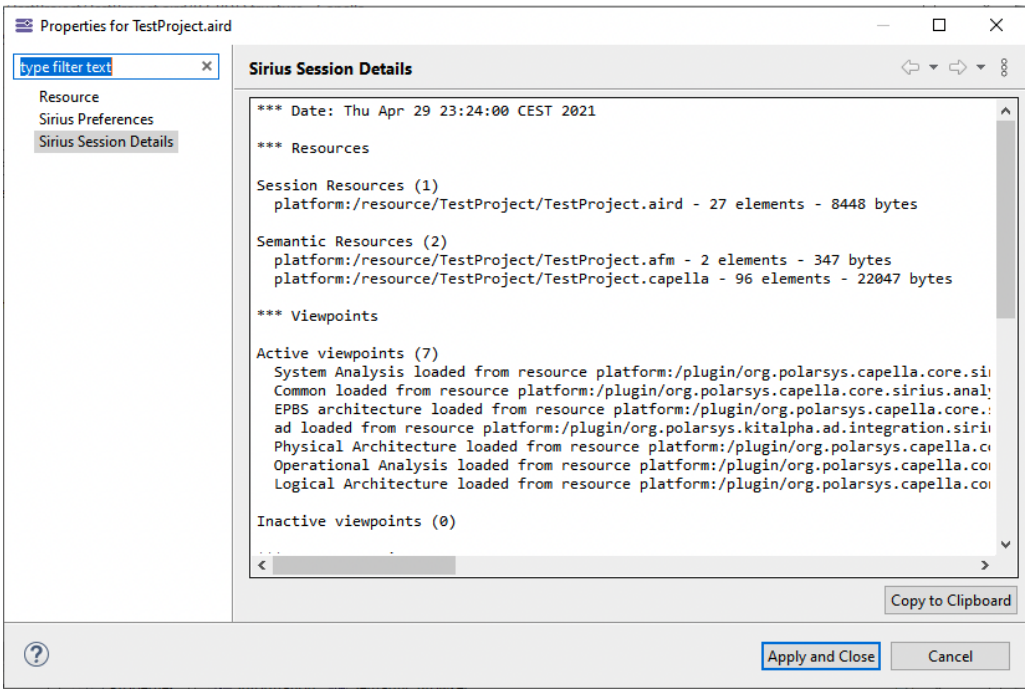
The model used is what we consider as a medium-big sized model.

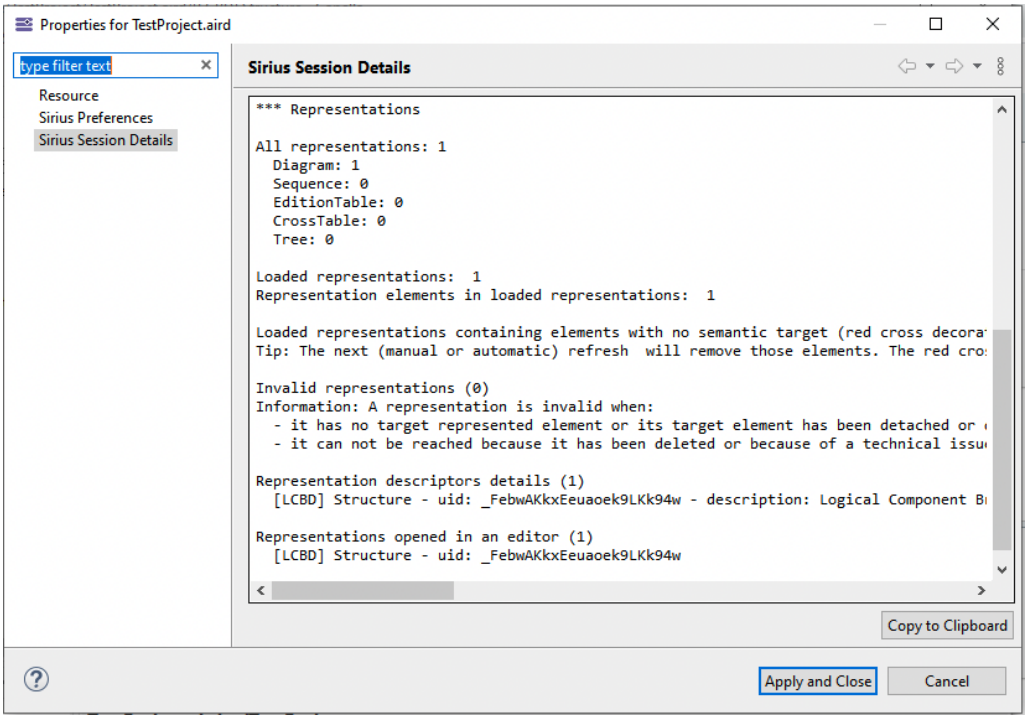
20 users are working on it.

- The server 4Go is well fitted for the use
- The client 3Go RAM is sufficient for most cases.It can be increased to have best performance when the user uses high memory consuming action

How to get characteristics of your model

The simplest way to get characteristics of your model is to use the *Sirius Session Details* tab available in the *Properties* contextual menu of the .aird file of your project:

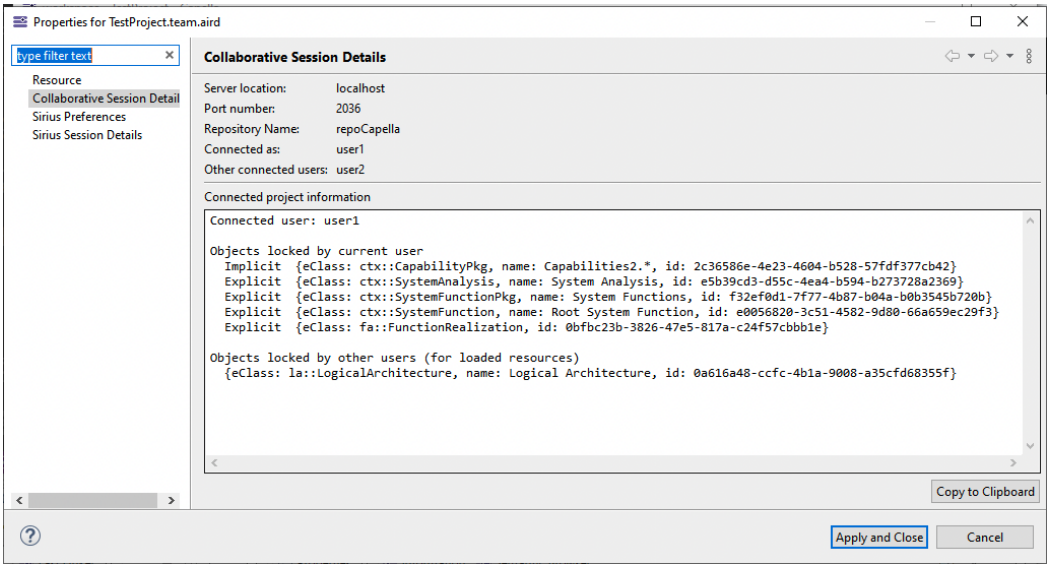




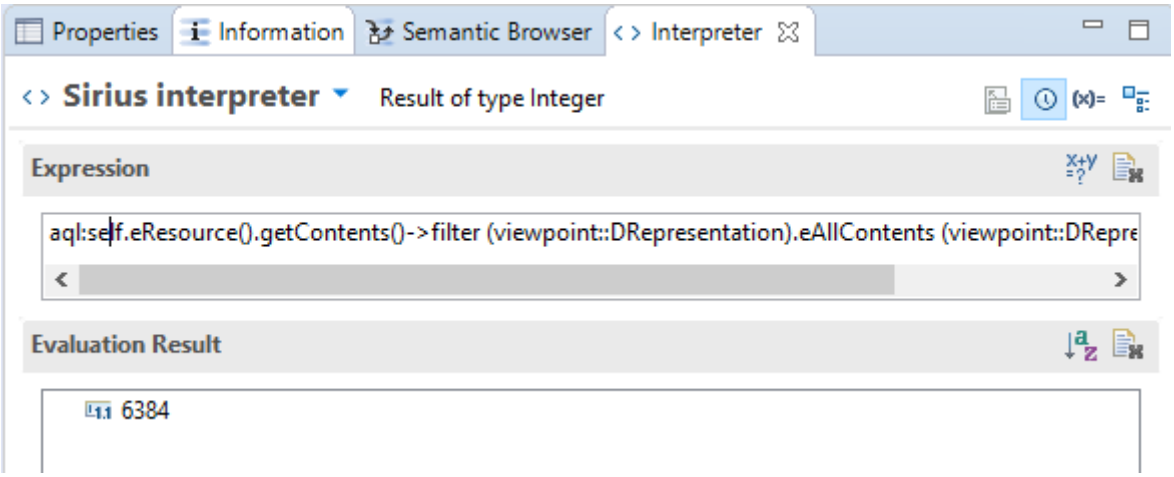
It will give you the following information about your project:

- Number, name, path and number of contained elements of each resources
- Selected Viewpoints
- Number of representations
- Some details about each representation (name type, id, status)
- Number of loaded representations
- Number of representations opened in editor

On shared projects, the *Collaborative Session Details* tab gives information about connected users and locks:

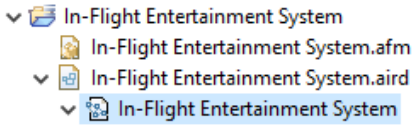


It is also possible to additional model characteristics thanks to the *Interpreter* view:



To get the characteristics of the *.capella* semantic resource :

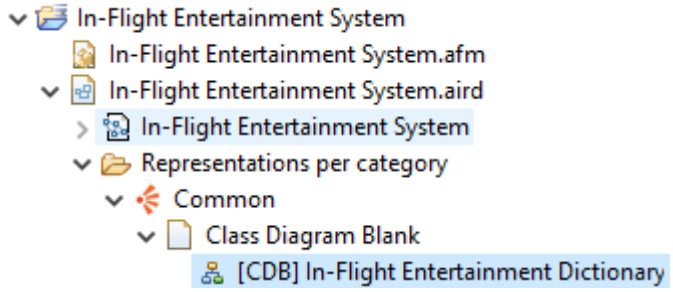
- select any semantic element



- write the AQL query in the *Interpreter* view to get the result in the lower part of the view.

To get the characteristics of the representations in the *.aird*

- select any representation



- write the AQL query in the *Interpreter* view to get the result in the lower part of the view

Chapter 7. Additions

7.1. Jenkins offline installation and configuration procedure

Summary

This procedure provides a complete guide for installing Jenkins offline on a Windows machine as part of the T4C installation process. It is intended for deployments where the target server hosting the Scheduler has no internet access. For context on the execution of this procedure, please refer to the following documents:

- Team for Capella Download page
- Team for Capella Installation Guide (pdf)
  - Jenkins scheduler installation
- Team for Capella System Administrator Guide (pdf)
  - Jenkins Installation

This procedure consists of four steps:

- The first step describes the prerequisites necessary for an offline installation of Jenkins.
- The second step involves downloading, preparing, and packaging the necessary assets for installation from a host that is connected to the internet.
- The third step involves deploying the scheduler server (Jenkins) on the offline host using the assets packaged in the previous step.
- The last step involves preparing for needed plugin installation through an automatic procedure.

Prerequisites

To complete this procedure, you will need two hosts:

- **Target_Host**: This is the host on which Jenkins will be installed. It must be **offline** (i.e., not connected to the internet) and must have already been prepared using the T4C procedure, with **Capella already deployed** (see Client Installation section). Admin rights are required on this host to install Jenkins. The following variable must be known on this host:
  - **TEAMFORCAPELLA_APP_HOME**: This variable contains the absolute path of the T4C and Capella installation folder (which contains the folders capella, lic-server, samples, server, tools, updateSite, etc.). Example: `C:\T4C\`
  - **WORKING_PATH**: This variable contains the absolute path of a temporary folder used to copy and extract assets.

  This host need to have a tools to create a `zip` archive (Example: `7-zip`)

Windows integrated (un)archiver tools is not recommended for managing Capella related archives. Contrary to recent versions of the File Explorer, the tool is not able to manage long file paths yet.

If administrator rights are not available on the **Target_Host**, an alternative is available in section Non administrator alternative.

- **Download_Station**: This is the workstation from which you will download all the elements needed to install the Jenkins server. Admin rights are not required on this host. This host must be connected to the internet and have access to the following resources:
  - Jenkins MSI installer download URL: https://get.jenkins.io/windows-stable/
  - Jenkins plugins download URL: https://updates.jenkins.io/download/plugins/

  The following variable must be known on this host:
  - **T4C_BUNDLE**: Local path of the T4C bundle extracted on the host.
  - **DDL_PATH**: Local path of an empty folder used to download assets. This folder must have the name `t4c_jenkins_assets`

  This host need to have a tools to extract a `zip` archive
- On the procedure, some values are used between chevron. This values are described here :
  - **JENKINS_VERSION**: version of Jenkins to use, we recommend to use the version on which Team for Capella has been tested and validate: the LTS version **2.440.3**. If you choose to deploy a more recent version, we strongly recommend to use a release from the LTS (Long Term Support) stable releases stream available at Jenkins.io. In case of a newer LTS, it recommended to take the shortname plugin list. This list might be incomplete for a non tested version: some plugins might have renamed, or have new dependencies. They can me manually downloaded and installed afterwards. The errors while loading plugins will be displayed in Jenkins.

Procedure

Step 1: Check of prerequisites

1. On the **Download_Station** :
   - Check the access of the listed websites.
   - Download or get the Jenkins plugins list provided in the T4C bundle from the file `<T4C_BUNDLE>/tools/resources/scheduler/requiredPlugins_Jenkins_LTS_<*JENKINS_VERSION*>.txt`.
2. On the **Target_Host** :
   - Check that the available disk space is greater than 200 MB.
   - Create the folder **DDL_PATH** if it does not exist.
   - Create the folder **DDL_PATH**\plugins if it does not exist.

Step 2: Download assets

On the **Download_Station** :

1. Download the Jenkins MSI installer file from the url: https://get.jenkins.io/windows-stable/<*JENKINS_VERSION*>/jenkins.msi
2. Copy the downloaded jenkins.msi file on the folder **DDL_PATH**

The jenkins.msi file checksum (SHA-256) can be verified using the provided checksum given from the url: https://get.jenkins.io/windows-stable/<*JENKINS_VERSION*>/jenkins.msi.sha256 and the Windows command :

- using **cmd.exe**: `certutil -hashfile jenkins.msi SHA256`
- using **PowerShell**: `Get-FileHash jenkins.msi`

The signature of the jenkins.msi file can also be verified using the procedure described in the following url: https://www.jenkins.io/download/verify/

3. Download the needed plugins using the file `requiredPlugins_Jenkins_LTS_<*JENKINS_VERSION*>.txt` given in the T4C bundle on path `<T4C_BUNDLE>\tools\resources\scheduler\`.

   These plugins can be manually downloaded using the URLs provided in the file.

The plugins download step can be automated using this PowerShell script:

```
# Set the path to the file containing the URLs
$url_file = "C:\Path\To\requiredPlugins_Jenkins_LTS_.*.txt"

# Set the directory where the downloaded files will be saved
$output_directory = "C:\Path\To\DDL_PATH\plugins"

# Read the contents of the URL file and store each URL in an array
$url_array = Get-Content $url_file | Where-Object { $_ -notmatch
"^\s*#.*" -and $_ -ne "" }

# Iterate through the array and download each file
foreach ($url in $url_array) {
    $output_file = Join-Path $output_directory
([System.IO.Path]::GetFileName($url))
    Invoke-WebRequest $url -OutFile $output_file
}
```

To execute this script, copy it in a file with the ".ps1" file and adapt the file paths. On execute, if you have an error that the script "cannot be loaded because running scripts is disabled on this system", then you need to execute the command "Set-ExecutionPolicy RemoteSigned" in PowerShell.

Check that the number of resulting plugins in the target folder corresponds to

the number of plugins in the list.

The checksum (SHA-256) of each plugin file can be verified using the provided checksum given on the plugin description page for the selected version. (https://updates.jenkins.io/download/plugins/<plugin name>/) and the windows command:

- using **cmd.exe**: `certutil -hashfile <plugin file>.hpi SHA256`.
- using **PowerShell**: `Get-FileHash <plugin file>.hpi`.

4. Package the **DDL_PATH** folder on a zip archive named `t4c_jenkins_assets.zip`.

   The zip archive generated must contain the folder `t4c_jenkins_assets`, and inside this folder, all downloaded assets.

5. Get the checksum of this archive with the following Windows commands:
   - using **cmd.exe**: `certutil -hashfile t4c_jenkins_assets.zip SHA256`.
   - using **PowerShell**: `Get-FileHash t4c_jenkins_assets.zip`.
6. Copy the generated zip file on a media available from the **Target_Host**.

Step 3: Deploy offline Jenkins

On the **Target_Host** :

1. Copy the previously generated zip from the media on the **Target_Host** on the **WORKING_PATH**.
2. Extract the zip archive on the **WORKING_PATH** to have the folder structure :

```
WORKING_PATH
* t4c_jenkins_assets
** jenkins.msi
** plugins
*** <plugin name>.hpi
*** <plugin name>.hpi
*** ...
```

3. Install Jenkins by running the jenkins.msi file following the official documentation: https://www.jenkins.io/doc/book/installing/windows/

There are some recommendations for this Jenkins installation :

- Use the port **8036** for the service.
- Use the **JRE provided** by the T4C deployment: `TEAMFORCAPELLA_APP_HOME\capella\jre\`.
- Activate the **Start Service**.
- Activate the **Firewall Exceptioin**.

On the Post-installation setup wizard :

- Use the port **8036** for the url to access your Jenkins instance.
- During Jenkins installation, choose the option to "Skip Plugin Installations" to indicate that the installation is offline.

Step 4: Prepare plugins installation

On the **Target_Host** :

1. Create a `TEAMFORCAPELLA_APP_HOME\tools\resources\scheduler\plugins` folder
2. Populate this folder by copying all `WORKING_PATH\t4c_jenkins_assets\plugins\*.hpi` files in it.
3. Generate the plugins file list `TEAMFORCAPELLA_APP_HOME\tools\resources\scheduler\requiredPlugins_Jenkins_local.txt` with the local relative path from `TEAMFORCAPELLA_APP_HOME\tools\resources\scheduler` path of each plugin file. The file create must have this look :

```
C:\...\plugins\<plugin name>.hpi
C:\...\plugins\<plugin name>.hpi
C:\...\plugins\<plugin name>.hpi
...
```

This file can be created with this PowerShell command

```
# Set the path to the folder containing the plugins files
$folder_path =
"*TEAMFORCAPELLA_APP_HOME*\tools\resources\scheduler\plugins"

# Set the target file that list the plugins files
$output_file =
"*TEAMFORCAPELLA_APP_HOME*\tools\resources\scheduler\requiredPlugins
_Jenkins_local.txt"

# List files and generate the list on the file
Get-ChildItem $folder_path -File | foreach { "file:" + $_.FullName }
| Set-Content $output_file
```

4. Update the plugin installation script to use this newly generated plugin file list :
   - Edit the file `TEAMFORCAPELLA_APP_HOME/tools/resources/scheduler/install-TeamForCapellaAppsOnJenkins.properties`
   - replace the variable **PLUGINS_LIST_FILE** value with `requiredPlugins_Jenkins_local.txt` instead of `requiredPlugins_Jenkins_LTS_<*JENKINS_VERSION*>.txt`

All needed plugins will be now installed by the `install-TeamForCapellaAppsOnJenkins.bat` script on the System Administrator Guide > Jenkins Installation > Install Jenkins plugins and jobs required

for Team for Capella section.

The execution of the `install-TeamForCapellaAppsOnJenkins.bat` will by default use the JRE embedded in Capella.The JRE to use to run the script can be changed in the file `TEAMFORCAPELLA_APP_HOME/tools/resources/scheduler/install-TeamForCapellaAppsOnJenkins.properties`

```
JAVA_CALL="..\..\..\capella\jre\bin\java"
```

Non administrator alternative

This section explain how to adapt the previous procedure to use a **Target_Host** without administrative rights.

In this alternative, there are some constraints :

- The windows installer (MSI) cannot be used.So some operations must be done manually
- Jenkins cannot be run as a Windows service.This means that Jenkins server must be started manually at each host start.

On this alternative, define a folder to store the Jenkins home: `JENKINS_HOME` on an availlable space to the used users.

Step 2: Download assets

1. Replace the download of the jenkins.msi file by the WAR alternative: https://get.jenkins.io/war-stable/<*JENKINS_VERSION*>/jenkins.war

   The checksum of this file is available from the url https://get.jenkins.io/war-stable/<*JENKINS_VERSION*>/jenkins.war.sha256

Step 3: Deploy offline Jenkins

1. Prepare the Jenkins execution folder :

   Create the folder `TEAMFORCAPELLA_APP_HOME\jenkins`

2. Copy the `jenkins.war file on this folder *TEAMFORCAPELLA_APP_HOME\jenkins`
3. Create the Jenkins temporary folder :
   - Create the folder `TEAMFORCAPELLA_APP_HOME\jenkins\temp`
4. Create a `startScheduler.bat` script to start the Jenkins product with the following content (prepared using System Administrator Guide > Jenkins Installation)

   ```
   REM Set env var needed to run Jenkins
   SET JENKINS_HOME=*TEAMFORCAPELLA_APP_HOME*\jenkins
   ```

```
SET PATH=%PATH%;*TEAMFORCAPELLA_APP_HOME*\capella\jre\bin

REM Start Jenkins webserver
java -jar *TEAMFORCAPELLA_APP_HOME*\jenkins.war
-Djava.io.tmpdir=%JENKINS_HOME%\temp --httpPort=8036
--extractedFilesFolder="%JENKINS_HOME%\temp
```

5. To start the Jenkins server, run the script **startScheduler.bat** from a Windows terminal (cmd.exe or PowerShell)

Continue this step by following the "Post-Installation" recommendations of the standard procedure.

Step 4: Prepare plugins installation

This step is not affected by the use of a non-administrator user.

7.2. Team for Capella - Advanced installation and configuration sample procedure

Summary

This procedure adapts the default Team for Capella installation procedure for an offline deployment with wss:// and https:// for all components.

Installation steps are listed, sorted, adapted or referenced from Team for Capella installation guide, Team for Capella documentation and Jenkins offline installation procedure for Team for Capella.

For context on the execution of this procedure, please refer to the following documents:

- Team for Capella Download page
- Team for Capella Installation Guide (pdf)
  - Jenkins offline installation procedure
- Team for Capella User Manual (pdf)

Prerequisites

Several items will need to be know at installation time:

- IP address or DNS name of the Server VM needs to be transmitted to Obeo
  - required to generate the client connection key and the token pool for the license server
- FQDN of the server
  - needed to generate the https certificates
  - needed on identity provider side to redirect url
- Certificate(s) for the chosen FQDN

- can be self-signed
- needed for https configuration for the T4C Server, License server and Jenkins

To complete this procedure, you will need:

- a computer with internet access to prepare the resources for the Jenkins offline installation
- access to the Server VM with the possibility to upload the needed artifacts
- access to the Client VM with the possibility to transfer the prepared client from the Server VM to the Client VM.

Before the installation, you will need to download/get:

- Capella bundle:
  - prepared bundle Capella + add-ons.
  - or open source Capella bundle.
- Jenkins 2.440.3 msi: https://get.jenkins.io/windows-stable/2.440.3/
- Team for Capella
  - Windows bundle from https://www.obeosoft.com/en/team-for-capella-download
  - Installation guide from Team for Capella Installation Guide (pdf)
  - Documentation from Team for Capella User Manual (pdf)
  - Jenkins offline installation and configuration procedure provided with this document

The target host should have a tool to extract a zip archive such as 7-zip to avoid "long path issues" during extraction of Capella and its plugins.

The target host should have an advanced text editor such as Notepad++ (or an equivalent) to help editing of multi- ine configuration files.

Details and tips to generate keystores can be found at Jetty Operations Guide - Configuring SSL/TLS KeyStores

Procedure

TEAMFORCAPELLA_APP_HOME: This variable contains the absolute path of the T4C and Capella installation folder (which contains the folders capella, lic-server, samples, server, tools, updateSite, etc.). Example: C:\T4C\

Step 1: Team for Capella installation

1. Follow section Installation of the Team for Capella installation guide but use the Capella bundle instead of the open-source bundle.
   - Note at the end of the section Team for Capella Server installation procedure is not necessary if remote clients are not launched on the same VM as the server.

- Section Extensions installation might not be necessary: plugins already packaged as dropins with their T4C specific extensions.

2. Follow section Administration features installation of the Team for Capella installation guide.

Step 2: "Jenkins offline installation" preparation step

Follow Step 1: Check of prerequisites and Step 2: Download assets from "Jenkins offline installation and configuration procedure". Pay attention to the modified plugins list note for 7.0.

Step 3: Specific Team for Capella configuration

Server

Activation of the wss:// instead of plain text tcp communications requires to change several configuration files of the Team for Capella server to enable WS/WSS transport and deploy the Jetty server in https instead of https:

1. `TEAMFORCAPELLA_APP_HOME\server\configuration\fr.obeo.dsl.viewpoint.collab.server.admin\admin-server.properties`
   - disable http
     - line 10: **admin.server.jetty.http.enabled=false**
   - enable https:
     - line 14: **admin.server.jetty.https.enabled=true**
     - uncomment line 16 to 19

```
admin.server.jetty.ssl.host=0.0.0.0
admin.server.jetty.ssl.port=8443
#admin.server.jetty.ssl.keystore.path=${currentDir}/keystore ①
admin.server.jetty.ssl.keystore.path=../certs/keystore ②
admin.server.jetty.ssl.keystore.passphrase=secret
```

     ① keystore file, named keystore, located in the same folder than `admin-server.properties`

     ② keystore file, named keystore, located in `TEAMFORCAPELLA_APP_HOME\certs`

   - enable websocket transport:
     - line 52: admin.server.jetty.net4j.enabled=true
2. `TEAMFORCAPELLA_APP_HOME\server\configuration\cdo-server.xml`
   - comment line 3
   - line 4, add: **<acceptor type="ws"/>**

Keystore management tips:

- System Administrator Guide > Server Configuration > Managing certificate

- Jetty Operations Guide - Configuring SSL/TLS KeyStores
- example for a self-signed certificate:

```
*TEAMFORCAPELLA_APP_HOME*\capella\jre\bin>keytool.exe -genkey
-keyalg "RSA" -dname CN=poc.obeosoft.cloud -alias keystore_alias
-keystore keystore -storepass secret -validity 730 -keysize 4096
-ext "SAN:c=DNS:localhost,IP:127.0.0.1"
```

Tools adaptations

Admin server properties need to be replaced in tools properties:

- From:

```
-httpHost localhost ^
-httpPort 8080 ^
-httpsConnection false ^
```

- To:

```
-httpHost T4C_SERVER_FQDN ^ ①
-httpPort 8443 ^
-httpsConnection false ^
```

① T4C_SERVER_FQDN must be replaced with the chosen FQDN

This needs to be done in:

- TEAMFORCAPELLA_APP_HOME\tools\command.bat
- TEAMFORCAPELLA_APP_HOME\tools\exporter.bat
- TEAMFORCAPELLA_APP_HOME\tools\importer.bat
- TEAMFORCAPELLA_APP_HOME\tools\maintenance.bat

Once jobs are deployed in Jenkins, **Server - Start Repository** will need to be updated: remove -httpPort, -httpsConnection - httpHost arguments from the executed command)

Client adaptations

Some configuration needs to be changed on the client properties:

1. TEAMFORCAPELLA_APP_HOME\capella\pluginCustomization.ini

```
#fr.obeo.dsl.viewpoint.collab/PREF_DEFAULT_REPOSITORY_LOCATION=localhost
fr.obeo.dsl.viewpoint.collab/PREF_DEFAULT_REPOSITORY_LOCATION=T4C_SERVER_FQDN ①
```

```
fr.obeo.dsl.viewpoint.collab/PREF_DEFAULT_REPOSITORY_PORT=8443
fr.obeo.dsl.viewpoint.collab/PREF_DEFAULT_CONNECTION_TYPE=WSS
```

① T4C_SERVER_FQDN must be replaced with the chosen FQDN

2. `TEAMFORCAPELLA_APP_HOME\capella\capella.ini`
   - if self-signed certificates are used
     - trust all certificate (self-signed), complete the **-vmargs** section with:

```
-Dfr.obeo.dsl.viewpoint.collab.https.jetty.ssl.context.trustall=true
```

     - or install certificate in jre trustore:

```
keytool -import -cacerts  -alias <aDifferentAliasForEachCertFile> -file
<pathToTheCerFile> ①
```

① Default password for cacerts is changeit or empty, `-cacerts` is equivalent to `-keystore ../lib/security/cacerts`, it is also possible to pass the store pass on the command line with -storepass.

Step 4: Jenkins installation

The section Jenkins scheduler installation redirects to System Administrator Guide > Jenkins Installation which requires internet access.

This section requires the result from Step 2: "Jenkins offline installation" preparation step

If Jenkins has already been installed, skip the first sub step of section Step 3: Deploy offline Jenkins.

1. Run Step 3: Deploy offline Jenkins from the Jenkins offline installation and configuration procedure.
   - use port **443**
2. Run Step 4: Prepare plugins installation from the Jenkins offline installation and configuration procedure.
   - This step corresponds to the beginning of the Jenkins plugins and jobs required for Team for Capella - Automatic installation section
3. Run the latest steps of the System Administrator Guide > Jenkins Installation > Install Jenkins plugins and jobs required for Team for Capella > Automatic installation section
   - Declare the **TEAMFORCAPELLA_APP_HOME** environment variable
   - Change the number of executors, the locale, the default view and the markup formatter
4. Configure Jenkins with https:

- Edit the jenkins.xml file (installed into %PROGRAMFILES{x86)%/Jenkins/jenkins.xml by default on Windows) and change the following arguments being passed to java when launching jenkins:

```
--httpPort=-1
--httpsPort=443
--httpsKeyStore=*TEAMFORCAPELLA_APP_HOME*\certs\keystore ①
--httpsKeyStorePassword="<cleartext-password-to-keystore>"
```

① Could also reference its own keystore with `--httpsKeyStore="%JENKINS_HOME%\jenkins.example.com.jks"`

Database - Backup needs a fix to create its workspace at the expected location, replace `$WORKSPACE` with `%WORKSPACE%` in its build step.

Jenkins configuration with OpenId Connect is not part of this procedure.

Step 5: Licenses installation

1. Follow sections License server installation and Client configuration of the Team for Capella installation guide.
   - OLS.zip and connection-key.txt are required at this step.
2. In the Scheduler (Jenkins), enable the **License Server - Run** job.
3. Change its configuration to enable **https on port 9443**

   ```
   eclipsec.exe --launcher.ini lic-server.ini -verbose -consoleLog -keys ./OLS
   -httpPort 9443 -useHttps
   ```

4. Complete License server **-vmargs** section to reference the keystore, in . `TEAMFORCAPELLA_APP_HOME\lic-server\lic-server.ini`

   ```
   eclipsec.exe --launcher.ini lic-server.ini -verbose -consoleLog -keys ./OLS
   -httpPort 9443 -useHttps
   ---
   -vmargs
   #Https_monitoring:
   #-Dorg.eclipse.jetty.ssl.keystore.path=keystore
   -Dorg.eclipse.jetty.ssl.keystore.path=../certs/keystore
   #Https_monitoring_optional_parameters:
   -Dorg.eclipse.jetty.ssl.password=secret
   #-Dorg.eclipse.jetty.ssl.keypassword=KEY_MANAGER_PASSWORD
   #-Dorg.eclipse.jetty.ssl.keystore.type=JKS
   #-Dorg.eclipse.jetty.ssl.protocol=TLS
   ```

Step 6: Client installation

1. Zip the capella and samples plugins from `TeamForCapella_InstallationFolder`
2. Upload this bundle on the client VM
3. Extract to the chosen `TeamForCapella_ClientInstallationFolder`
4. Perform the multi-user configuration:
    - On a multi-user client installation, with workspace store in location not readable by other users, configuration and p2 folders must also be re-located.
    - See Strategy 2: Shared installation
    - Several possible options:
        - in case of a custom launch script: change workspace and configuration area
        - or in `TeamForCapella_ClientInstallationFolder\capella\capella.ini`: **-Dosgi.configuration.area=@user.home/xxx/capella/configuration**
        - or in `TeamForCapella_ClientInstallationFolder\capella\configuration.ini\config.ini`: **osgi.configuration.area=@user.home/TeamForCapella_7.0.0/configuration**
            - force the worskpace (equivalent to -data): **osgi.instance.area=@user.home/TeamForCapella_7.0/workspace**
            - also force the default workspace path: in **osgi.instance.area.default=@user.home/TeamForCapella_7.0/workspace**

Step 7: Installation verification

1. Restart Jenkins service
2. Check that the **Server - Run** and the **License Server - Run** jobs are running
3. Launch two Capella clients from the client VM, from two different Citrix sessions
4. User 1: export one project to Team for Capella repository, and connect to it
5. User 2: connect to the share project.
6. User 1 / USer 2: perform some modifications, save, take locks, …
7. Admin: check the license server monitoring
8. Admin: check that the **Backup - Database**, **Project - Import**, **Server - Start repository**, **Server - Stop repository**, **Server - Stop** jobs are working fine
9. Admin:
    - stop the server with **Server - Stop**
    - delete **TEAMFORCAPELLA_APP_HOME**\server\db-auditng
    - start the server with **Server - Run**

Step 8: OpenID Connect

The user registration is by default a user manager which stores the accepted users and their credentials in `TEAMFORCAPELLA_APP_HOME\server\configuration\users.properties`.

To enable the use of **OpenID Connect**, there are two options:

- the authentication only can be delegated to the identity provider but the list of users is still managed per Team for Capella repositories
  - uncomment and configure the OpenID Connect properties in `TEAMFORCAPELLA_APP_HOME\server\configuration\usermanager-config.properties`
  - User registrations by editing the user.properties to add authorized logins or via the User Management view in Capella
- or the authentication and registration are both delegated to the identity provider:
  - in `TEAMFORCAPELLA_APP_HOME\server\configuration\cdo-server.xml`
    - comment the **userManager** element
    - uncomment the **<authenticator type="openidconnect"** element
  - configure the OpenID Connect properties in `TEAMFORCAPELLA_APP_HOME\server\configuration\openid-config.properties`

The authentication servlet must also been enabled in `TEAMFORCAPELLA_APP_HOME\server\configuration\fr.obeo.dsl.viewpoint.collab.server.admin\admin-server.properties`:

- line 45: **admin.server.jetty.auth.openidconnect.enabled=true**

Configuration details are described in System Administrator Guide > Server Configuration > Activate OpenID Connect authentication

Step 9: Installation verification

Rerun step 7. Some technical users might be required regarding the configuration of the tool. They will be defined in `TeamForCapella_InstallationFolder\server\configuration\technicalUsers.properties`.