TEAMCENTER

Simulation Process and Data Management — Deployment and Administration

Teamcenter 2412

SIEMENS

About Siemens Digital Industries Software

Siemens Digital Industries Software is a global leader in the growing field of product lifecycle management (PLM), manufacturing operations management (MOM), and electronic design automation (EDA) software, hardware, and services. Siemens works with more than 100,000 customers, leading the digitalization of their planning and manufacturing processes. At Siemens Digital Industries Software, we blur the boundaries between industry domains by integrating the virtual and physical, hardware and software, design and manufacturing worlds. With the rapid pace of innovation, digitalization is no longer tomorrow’s idea. We take what the future promises tomorrow and make it real for our customers today. Where today meets tomorrow. Our culture encourages creativity, welcomes fresh thinking and focuses on growth, so our people, our business, and our customers can achieve their full potential.

Support Center: **support.sw.siemens.com**

Send Feedback on Documentation: **support.sw.siemens.com/doc_feedback_form**

Contents

Configuring and customizing Simulation Process and Data Management for Active Workspace

CAE action handlers

Troubleshooting

1. Setting up Simulation Process and Data Management

Design is an iterative process and often designs need to be modified due to manufacturing constraints or conflicting requirements. It is possible to reduce costs and minimize the time spent on verification and testing by arriving at a good design in the initial stages of product development and by continuously validating the design throughout the product lifecycle. Simulation Process and Data Management is a solution used for validating or improving a design in the early stages of the product lifecycle.

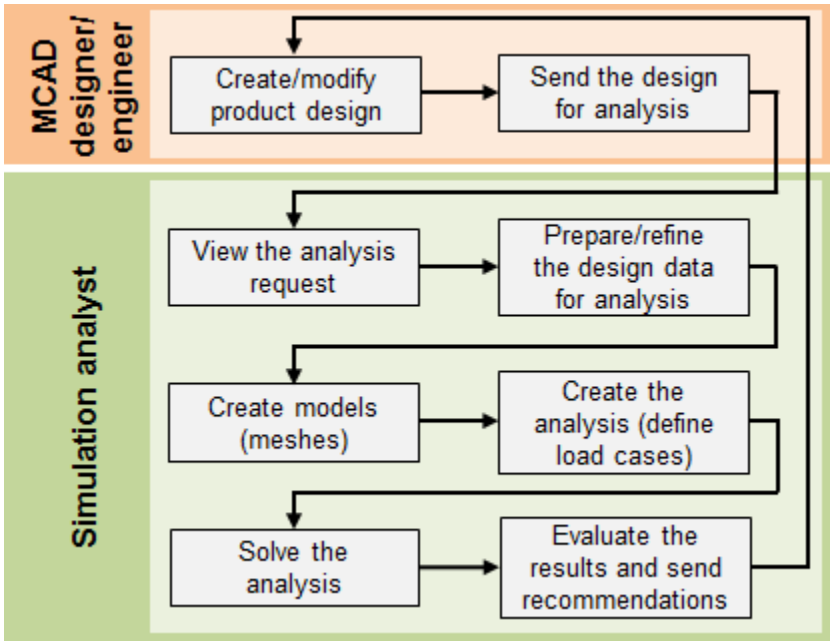
Where do I go from here?

 Simulation analyst	See *Simulation Process and Data Management on Rich Client — Usage* or *Simulation Process and Data Management on Active Workspace — Usage*.
 Simulation administrator	
Install Simulation Process and Data Management	Install Simulation Process and Data Management **using Teamcenter Environment Manager** or **Deployment Center**.
Configuring Simulation Process and Data Management tasks	Setting up Simulation Process and Data Management involves setting preferences to control Teamcenter's behavior and appearance, configuring data map rules, configuring simulation tools, configuring simulation dashboards, specifying comparison options for model and product structures, and other tasks. See *About configuring Simulation Process and Data Management*.
 Users with DBA privileges	
Edit style sheets to expose custom revision types in the Simulation tab for Active Workspace	Edit style sheets to expose custom revision types in the context of the product, geometry, model, analysis, or the result revision. Also, configure traversal paths to define how to traverse the data structure and specify which relationships are of interest and what should be done when these relationships are found. See *Expose custom revision types in the context of the product revision*.
Customize the tool launch page in Active Workspace	The simulation administrator configures simulation tools and the simulation analyst uses these preconfigured simulation tools to launch preprocessors, solvers, or postprocessors. Each tool might require different inputs based on the requirements of the tool. After configuring simulation tools, an administrative user with DBA privileges can create multiple customized launch pages for different launch tools and associate specific tools to the customized pages. See *About customizing the tool launch page*.

2. Introduction to Simulation Process and Data Management

What is Simulation Process and Data Management?

The workflow for managing the simulation process is as follows:



Understanding Simulation Process and Data Management using an example

1. *View the analysis request*

   Let us assume that you (as a simulation analyst) are an expert in simulated automobile crash testing. The analysis request is to perform a simulated side-impact test for a design change made to the driver-side door.

2. *Prepare or refine the design data for analysis*

   Your first task is to import the product geometry and simplify it to make it *relevant* for the analysis. The driver-side door has a speaker; switches for the door lock; power window; and electric mirror; an intrusion beam; steel frames; door panels; and other parts. The speaker and switches are not relevant for the simulated side-impact analysis. You import the complete product geometry and use a simplification tool to remove the speaker and switches. Then, you create geometry revisions for the *intrusion beam*, *steel frame*, and *door panel*, and save the simplified geometry for each component.

3. *Create models*

   The model you want to analyze includes the mesh definition, connections, and material and physical properties. In this step, you create model revisions—in the context of the geometry revisions—for the *intrusion beam*, *steel frame*, and *door panel*. Then, you select the appropriate meshing tool to generate meshes for each component.

4. *Create the analysis and solve the analysis*

   The analysis you want to perform includes load cases, solver parameters, and boundary conditions. In this step, you create analysis revisions—in the context of the model revisions—for the *intrusion beam*, *steel frame*, and *door panel*. Then, you select the appropriate solver tool to specify load cases and solver parameters.

   In this example, you create different analyses to vary the load cases, for example, a **3000** pound SUV-like barrier hits the driver side door at **30** mph. You also create another analysis model to change the barrier weight to **3200** pounds and the speed to **35** mph.

5. *Evaluate the results and send recommendations*

   The intrusion beam acts as an energy-absorbing material during a side impact. The results show that the intrusion beam cannot withstand forces of **3200** pounds at **35** mph. After evaluating results from various scenarios, you sign off the workflow by suggesting a design change for the intrusion beam.

Objects you work with

- *Geometry revisions*

  The CAD model is used to define the geometry of a part or an assembly. During the final stages of the design process, the geometry contains numerous details, such as sharp edges, bolt holes, or fillets, which are not required for the analysis. The geometry used for simulation analysis is often different from the product geometry. It may be a simplified or an abstracted version or an approximation when the product geometry is not available. **CAE 3D Geometry** revisions are workspace objects for storing the simplified geometry. Geometry revisions are created in the context of the item revisions in the product structure.

- *Model revisions*

  A mesh represents a geometric object as a set of finite elements. Finite element analysis (FEA) is a computerized method for simulating how a part reacts to conditions such as force, heat, vibration, and other physical effects in the real world. **CAE 3D Model** revisions are workspace objects for storing the mesh definition, connections, and material and physical properties. Model revisions are created in the context of geometry revisions.

- *Analysis revisions*

An *analysis revision* represents the specific simulation you want to perform. **CAE 3D Analysis** revisions are workspace objects for including load cases, solver parameters, and boundary conditions. Analysis revisions are created in the context of model revisions.

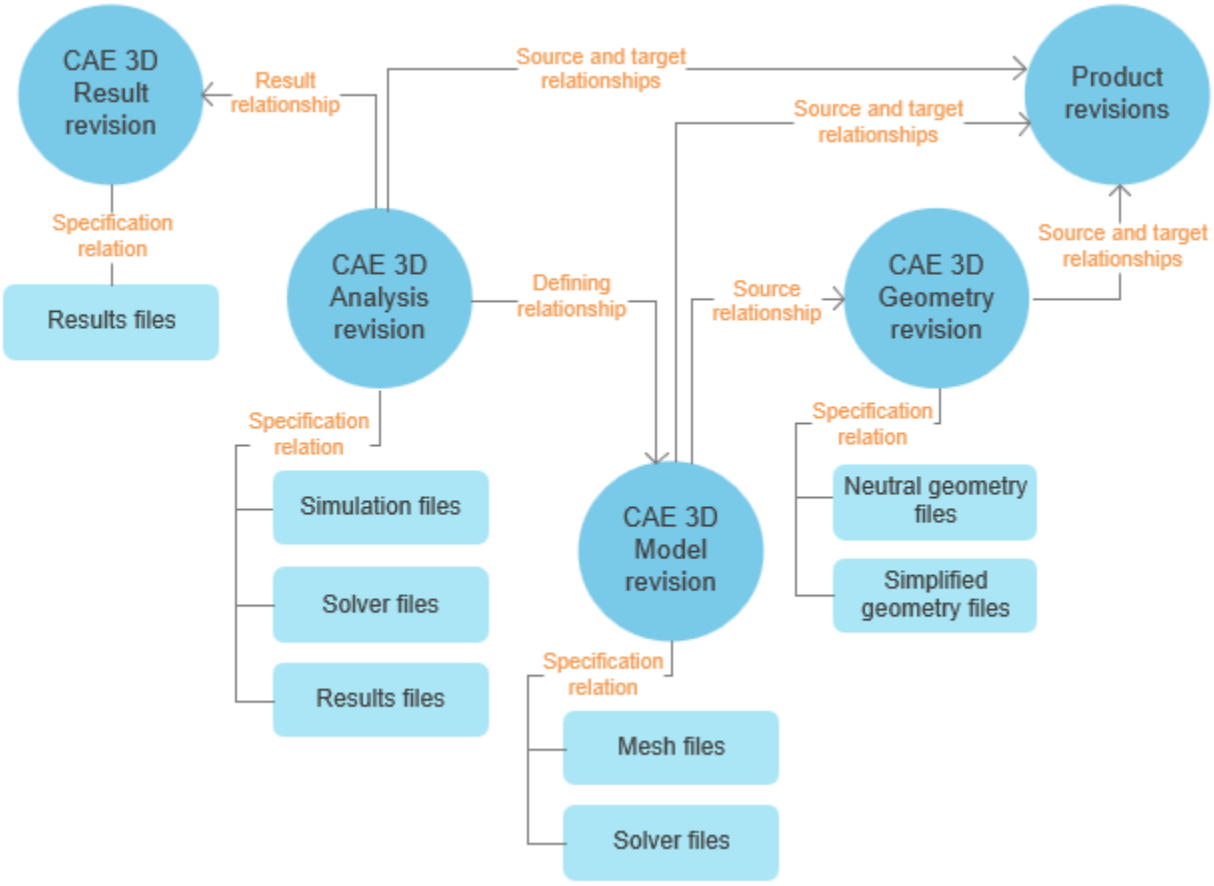
- *Result revisions*

  A *result revision* represents the results of a simulation analysis. **CAE 3D Result** revisions are workspace objects for storing the results of a solve. They may contain JT files or other visualization files that are the results of a solve. Result revisions are associated with analysis revisions.

You can create the following CAE revisions in the context of other revisions along with relationships:

Context revision	CAE revisions you can create for the context revision	Default relationships
Product structure	**CAE 3D Geometry**	**TC_CAE_SOURCE** and **TC_CAE_TARGET**
	CAE 3D Model	
	CAE 3D Analysis	
CAE 3D Geometry	**CAE 3D Model**	**TC_CAE_SOURCE**
	CAE 3D Analysis	
CAE 3D Model	**CAE 3D Analysis**	**TC_CAE_DEFINING**
CAE 3D Analysis	**CAE 3D Analysis**	**TC_CAE_INCLUDE**
	CAE 3D Result	**TC_CAE_RESULTS**

The following is an example of the default simulation data model.



Note:

CAE Item is the base type from which all other default objects such as **CAE Geometry**, **CAE Model**, and **CAE Analysis** are created as its subtypes. You must not create custom subtypes of **CAE Item** directly. However, you can create subtypes of **CAE Geometry**, **CAE Model**, **CAE Analysis**, and such objects.

3. Installing Simulation Process and Data Management

Install Simulation Process and Data Management features using TEM

The following procedures assume that you are installing Simulation Process and Data Management applications on an existing Teamcenter set up and that you are familiar with Teamcenter Environment Manager (TEM).

For more information about installing Teamcenter with Active Workspace, see *Teamcenter Installation on Windows Using TEM* or *Teamcenter Installation on Linux Using TEM*.

Enterprise tier

Run TEM on the Enterprise tier and select the following features in the **Features** panel:

Feature	Description
Extensions→CAE Simulation Management→Simulation Process Management	Installs the Simulation Process and Data Management templates for the corporate server.
Extensions→CAE Simulation Management→Extended Simulation Process Management	Installs enhancements to improve the default data model with additional attributes for the corporate server.
Extensions→CAE Simulation Management→Product Configurator for Simulation Process Management	Provides support for Product Configurator within CAE Manager in Teamcenter. If this template is installed, simulation analysts can use the **Derive Structures** command in CAE Manager and use Product Configurator variants, that is, families, features, and configurator rules to derive structures from an existing structure. If this template is not installed, the **Derive Structures** command is available only for the legacy variant configuration method by using classic variants. Note: If you want to use Product Configurator within CAE Manager in Teamcenter, select the **Product Configurator for Simulation Process Management** template while

Feature	Description
	upgrading from a version prior to 13.1 to the latest.
Base Install→Active Workspace→Server Extensions→CAE Simulation Management→Simulation Process Management	Installs the Simulation Process and Data Management templates for Active Workspace.
Base Install→Active Workspace→Server Extensions→CAE Simulation Management→Extended Simulation Process Management	Installs enhancements to improve the default data model with additional attributes on Active Workspace enterprise tier.
Base Install→Active Workspace→Server Extensions→CAE Simulation Management→Simulation Process Management with Git integration	Provides support for integration of GIT based tools. The feature allows engineers to manage gold copy models in Teamcenter by performing create, update, and compare of collections from GIT tag branches.
Base Install→Active Workspace→Server Extensions→CAE Simulation Management→Simulation Process Management with Parameter Management	Provides support for creating or updating enhanced MBSE parameters inside CAE MDAO and CAE 1D business objects. These parameters can be exchanged with relevant simulation tools through out-of-the-box tool integrations.
Base Install→Active Workspace→Server Extensions→Subscription	Provides support for managing subscriptions and notifications. Note: The **Subscription** tile is enabled on the **Analyst** workspace even if this feature is not installed. This feature must be installed for subscriptions and notifications to be available.

Client tier

Run TEM on the Client tier and select the following features in the **Features** panel:

Feature	Description
Base Install→Active Workspace→Client→CAE Simulation Management→Simulation Process Management	Installs the Simulation Process and Data Management templates on Active Workspace.
Base Install→Active Workspace→Client→CAE Simulation Management→Extended Simulation Process Management	Installs enhancements to improve the default data model with additional attributes on Active Workspace.
Base Install→Active Workspace→Client→CAE Simulation Management→Simulation Process Management with Git Integration	Provides support for integration of GIT based tools. The feature allows engineers to manage gold copy models in Teamcenter by performing create, update, and compare of collections from GIT tag branches.
Base Install→Active Workspace→Client→CAE Simulation Management→Simulation Process Management with Parameter Management	Provides support for creating or updating enhanced MBSE parameters inside CAE MDAO and CAE 1D business objects. These parameters can be exchanged with relevant simulation tools through out-of-the-box tool integrations.
Base Install→Active Workspace→Client→Subscription	Provides support for managing subscriptions and notifications in the client.

Install Simulation Process and Data Management using Deployment Center

Add the Simulation Process and Data Management application to your existing Teamcenter environment.

Procedure

1. Log on to Deployment Center and select the environment to which you want to add Simulation Process and Data Management.

2. Go to the **Applications** task. Click **Add or Remove Selected Applications**.

3. In the **Available Applications** panel, use the web browser search to find the following applications.

Application	Description
Extended Simulation Process Management	Installs enhancements to improve the default data model with additional attributes for the corporate server.
Extended Simulation Process Management for Active Workspace	Installs enhancements to improve the default data model with additional attributes for Active Workspace.

Application	Description
Simulation Process Management	Installs the Simulation Process and Data Management templates for the corporate server.
Simulation Process Management for Active Workspace	Installs the Simulation Process and Data Management templates for Active Workspace.
Product Configurator for Simulation Process Management	Provides support for Product Configurator within CAE Manager in Teamcenter. If this template is installed, simulation analysts can use the **Derive Structures** command in CAE Manager and use Product Configurator variants, that is, families, features, and configurator rules to derive structures from an existing structure. If this template is not installed, the **Derive Structures** command is available only for the legacy variant configuration method by using classic variants.
Subscription	Provides support for managing subscriptions and notifications. Note: The **Subscription** tile is enabled on the **Analyst** workspace even if this feature is not installed. This feature must be installed for subscriptions and notifications to be available. This application is selected by default when you select Teamcenter in the **Software** task.
Simulation Process Management with Parameter Management	Provides support for creating or updating enhanced MBSE parameters inside CAE MDAO and CAE 1D business objects. These parameters can be exchanged with relevant simulation tools through out-of-the-box tool integrations.
Simulation Process Management with Git integration	Provides support for integration of GIT based tools. The feature allows engineers to manage gold copy models in Teamcenter by performing create, update, and compare of collections from GIT tag branches.
Simulation Process Management with Measurable Attribute for Active Workspace	Allows simulation engineers find relevant historical simulations during the creation of new verification requests by providing

Application	Description
	suggestions. It helps streamline the process of finding similar data for their current tasks.

4. Select the applications, and then click **Update Selected Applications**.

   Deployment Center automatically selects any additional dependent applications.

5. In the **Selected Components** list, note any remaining components whose configuration status is not **100%**. Select each incomplete component, enter required parameters, and save component settings until all components in the environment show a configuration status of **100%**.

   When all components are fully configured, the **Deploy** task is enabled.

6. Go to the **Deploy** task. Click **Generate Install Scripts** to generate deployment scripts you will use to update affected machines.

   When script generation is complete, note any special instructions in the **Deploy Instructions** panel.

7. Locate deployment scripts, copy each script to its target machine, and then run each script on its target machine.

   For more information about running deployment scripts, see *Deploy task* in *Deployment Center — Usage*.

Update CAE templates for data model changes

Templates are the files that hold the data model changes. You must install the **Simulation Process and Data Management** template or update the CAE templates.

1. Copy the following XML files from the patch kit.

   - **feature_CAE.xml**
   - **feature_foundation.xml**

2. Start TEM, select the **Configuration Manager** option, and click **Next**.

3. Select the **Perform maintenance on an existing configuration** option, and click **Next**.

4. Select the configuration for which you have installed **Simulation Process and Data Management**, and click **Next**.

5. Select the **Update Database (Full Model - System downtime required)** option, and click **Next**.

TEM displays all the templates you have currently installed.

6. Click the **Browse** button and change to the *patch_kit**platform***install** directory to which you copied the XML files.

7. Confirm your selection to update the database.

8. To confirm whether the cache is cleared, run the following commands from the Teamcenter command prompt:

   - **generate_metadata_cache -u=***user_id* **-p=***password* **-g=***dba* **generate all**
   - **generate_client_meta_cache -u=***user_id* **-p=***password* **-g=***dba* **generate all**

Tasks to perform after installing Teamcenter patches

After installing Teamcenter patches on the Linux or Windows server, perform the following tasks:

Run the cae_configure_dm_propertyset utility if you makes changes to the data map file or the configuration file

Run the **cae_configure_dm_propertyset** utility each time you make changes to the **NodeXMLConfig.xml** configuration file or the data map file. This utility reads the **NodeXMLConfig.xml** file and creates a custom property set that is used for data mapping. To view the command line help for this utility, type **cae_configure_dm_propertyset -h** on the Teamcenter command prompt.

Example:

cae_configure_dm_propertyset -u=*user_id* **-p=***password* **-g=***dba*
-log=*name_of_log_file_containing_utility_results*

Migrate extended properties

The following is an example of how to migrate extended properties from a custom form attached to a **CAE 3D Model** revision to the same revision. Users may use a custom form to include extended properties such as material properties, inertia, center of gravity (CoG), and mesh type and attach them to a model revision. You can extend these properties to the model revision by migrating these properties and making the duplicate properties obsolete from the custom form in the database.

1. Install the **Extended Simulation Process Management** template by **using TEM** or **using Deployment Center**.

   This template installs enhancements to improve the default data model with additional attributes.

2. Create a mapping file (text file) to map the properties you want to extend to the **CAE 3D Model** revision.

In this example, **cae0CoGx** is an extended property in a custom form, and the system migrates it to the **sim0CoGx** property in the database. Similarly, you can map other extended properties.

Example of using extended properties in a mapping file with a single section:

```
# This is a valid file with one section
# This is the basic scenario when attributes from single source BO/Form
needs to be
# migrated to single target BO
#
# target_object_type, relation_type, source_form_type,
source_form_attribute
# and target_attribute are valid strings if they contain following
characters ONLY
# - a to z (Both capital and small case)
# - numbers
# - underscore ['_'] character
# - space [‘ ’] character

# @target_object_type:relation_type:source_form_type
@CAEModelRevision:IMAN_master_form:CAEModelRevisionMaster
#source_attribute,target_attribute
solver_name,solver_name
analysis_types,analysis_types
cae0CoGx,sim0CoGx
cae0CoGy,sim0CoGy
cae0Ip3,sim0Ip3
cae0Ixy,sim0Ixy
```

Example of using extended properties in a mapping file with multiple sections:

```
# This is a valid example file with more than one sections
# This is a scenario when attributes from more than one source BO/Form
# needs to be migrated to one or more target BO.

# target_object_type:relation_type:source_form_type
@CAEModelRevision:IMAN_specification:CAE0EngProperties
# source_form_attribute,target_attribute
cae0CoGx,sim0CoGx
cae0Disciplines,sim0Disciplines
cae0Ip1,sim0Ip1
cae0Ip2,sim0Ip2
cae0Ixx,sim0Ixx
cae0Iyz,sim0Iyz
cae0Izz,sim0Izz
# there must be at least one blank line to separate two different sections.

#@target_object:relation_name:source_object
```

```
@CAEModelRevision:IMAN_master_form:CAEModelRevisionMaster
#source_attribute,target_attribute
solver_name,solver_name
analysis_types,analysis_types
```

> Caution:
>
> If you use the same target object in multiple sections in the attribute mapping file, it may cause a failure in the migration of some objects. A log file is created and you can view it to see both successful and unsuccessful instances of the migration of extended properties.

3. Run the **cae_migrate_extended_cae_properties** utility by including the mapping file you created. To view the command line help for this utility, type **cae_migrate_extended_cae_properties -h** on the Teamcenter command prompt.

   When you run this utility, it copies the properties from the custom form to the **CAE 3D Model** revision.

4. Using BMIDE, mark the duplicate custom properties in the database as obsolete.

5. Edit the **datamapping.xml** file.

   You can add the extended properties to the **datamapping.xml** file so that these properties are available when a simulation analyst:

   - Creates model revisions using structure maps and data map rules; and
   - Derives one or more model structures from an existing model structure.

   a. Locate the **datamapping.xml** file and open it.

      The **datamapping.xml** sample file is located in the *TC_DATA* directory. However, the file used at runtime is managed in a **CAEStructureMap** dataset attached to the item revision indicated by the **CAE_datamap_files_location** preference.

      The following is an example of the file before adding custom properties:

```
<!-- ************************************************************
*********     CAE 3D Model Revision attributes          *************

The input CAE 3D Model revision attributes that need to populate
the output's CAE 3D Model revision atrributes go here.
************************************************************-->
<smn:ATTR_NODES>
<!-- Input's Description transferred to output's attribute of same name-->
     <smn:ATTR_NODE INTERNAL_NAME="object_desc">
          <xsl:attribute name="VALUE">
```

```
                <xsl:value-of select="$currentIRnodeline/smn:ATTR_NODES/
                smn:ATTR_NODE[@INTERNAL_NAME=&apos;object_desc&apos;]/
@VALUE" />
          </xsl:attribute>
     </smn:ATTR_NODE>
<!-- Input's Name transferred to output's attribute of same name-->
     <smn:ATTR_NODE INTERNAL_NAME="object_name">
          <xsl:attribute name="VALUE">
               <xsl:value-of select="concat('',$currentIRnodeline/
smn:ATTR_NODES/
                smn:ATTR_NODE[@INTERNAL_NAME=&apos;object_name&apos;]/
@VALUE)" />
          </xsl:attribute>
     </smn:ATTR_NODE>
</smn:NODE_LINE>
```

b. Add the custom properties.

Example of the same file after adding custom properties.

```
<!-- ************************************************************
*********     CAE 3D Model Revision attributes          *************

The input CAE 3D Model revision attributes that need to populate
the output's CAE 3D Model revision atrributes go here.
************************************************************-->
<smn:ATTR_NODES>
<!-- Input's Description transferred to output's attribute of same name-->
     <smn:ATTR_NODE INTERNAL_NAME="object_desc">
          <xsl:attribute name="VALUE">
          <xsl:value-of select="$currentIRnodeline/smn:ATTR_NODES/
          smn:ATTR_NODE[@INTERNAL_NAME=&apos;object_desc&apos;]/@VALUE" />
          </xsl:attribute>
     </smn:ATTR_NODE>
<!-- Input's Name transferred to output's attribute of same name-->
     <smn:ATTR_NODE INTERNAL_NAME="object_name">
          <xsl:attribute name="VALUE">
          <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
          smn:ATTR_NODE[@INTERNAL_NAME=&apos;object_name&apos;]/@VALUE)" />
          </xsl:attribute>
     </smn:ATTR_NODE>
     <smn:ATTR_NODE INTERNAL_NAME="sim0RepresentationType">
          <xsl:attribute name="VALUE">
          <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
          smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0RepresentationType&apos;]/
          @VALUE)" />
          </xsl:attribute>
     </smn:ATTR_NODE>
     <smn:ATTR_NODE INTERNAL_NAME="sim0MasterFormat">
```

```
          <xsl:attribute name="VALUE">
          <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
          smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0MasterFormat&apos;]/
@VALUE)" />
          </xsl:attribute>
          </smn:ATTR_NODE>
     <smn:ATTR_NODE INTERNAL_NAME="sim0Materials">
          <xsl:attribute name="VALUE">
          <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
          smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0Materials&apos;]/
@VALUE)" />
          </xsl:attribute>
     </smn:ATTR_NODE>
     <smn:ATTR_NODE INTERNAL_NAME="sim0Thicknesses">
          <xsl:attribute name="VALUE">
          <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
          smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0Thicknesses&apos;]/
@VALUE)" />
          </xsl:attribute>
     </smn:ATTR_NODE>
     <smn:ATTR_NODE INTERNAL_NAME="sim0MeshFlags">
          <xsl:attribute name="VALUE">
          <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
          smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0MeshFlags&apos;]/
@VALUE)" />
          </xsl:attribute>
     </smn:ATTR_NODE>
     <smn:ATTR_NODE INTERNAL_NAME="sim0PIDFormula">
          <xsl:attribute name="VALUE">
          <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
          smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0PIDFormula&apos;]/
@VALUE)" />
          </xsl:attribute>
     </smn:ATTR_NODE>
     <smn:ATTR_NODE INTERNAL_NAME="sim0PIDList">
          <xsl:attribute name="VALUE">
          <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
          smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0PIDList&apos;]/@VALUE)" />
          </xsl:attribute>
     </smn:ATTR_NODE>
     <smn:ATTR_NODE INTERNAL_NAME="sim0MIDs">
          <xsl:attribute name="VALUE">
          <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
          smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0MIDs&apos;]/@VALUE)" />
          </xsl:attribute>
     </smn:ATTR_NODE>
     <smn:ATTR_NODE INTERNAL_NAME="sim0MeshDensity">
          <xsl:attribute name="VALUE">
          <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
```

```
          smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0MeshDensity&apos;]/
@VALUE)" />
          </xsl:attribute>
     </smn:ATTR_NODE>
     <smn:ATTR_NODE INTERNAL_NAME="sim0MeshQuality">
          <xsl:attribute name="VALUE">
          <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
          smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0MeshQuality&apos;]/
@VALUE)" />
          </xsl:attribute>
     </smn:ATTR_NODE>
     <smn:ATTR_NODE INTERNAL_NAME="sim0MeshQualityCheck">
          <xsl:attribute name="VALUE">
          <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
          smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0MeshQualityCheck&apos;]/
          @VALUE)" />
          </xsl:attribute>
     </smn:ATTR_NODE>
     <smn:ATTR_NODE INTERNAL_NAME="sim0Mass">
          <xsl:attribute name="VALUE">
          <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
          smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0Mass&apos;]/@VALUE)" />
          </xsl:attribute>
     </smn:ATTR_NODE>
     <smn:ATTR_NODE INTERNAL_NAME="sim0CoGx">
          <xsl:attribute name="VALUE">
          <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
          smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0CoGx&apos;]/@VALUE)" />
          </xsl:attribute>
     </smn:ATTR_NODE>
     <smn:ATTR_NODE INTERNAL_NAME="sim0CoGy">
          <xsl:attribute name="VALUE">
          <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
          smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0CoGy&apos;]/@VALUE)" />
          </xsl:attribute>
     </smn:ATTR_NODE>
     <smn:ATTR_NODE INTERNAL_NAME="sim0CoGz">
          <xsl:attribute name="VALUE">
          <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
          smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0CoGz&apos;]/@VALUE)" />
          </xsl:attribute>
     </smn:ATTR_NODE>
     <smn:ATTR_NODE INTERNAL_NAME="sim0Ixx">
          <xsl:attribute name="VALUE">
          <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
          smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0Ixx&apos;]/@VALUE)" />
          </xsl:attribute>
     </smn:ATTR_NODE>
     <smn:ATTR_NODE INTERNAL_NAME="sim0Ixy">
```

```
     <xsl:attribute name="VALUE">
     <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
     smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0Ixy&apos;]/@VALUE)" />
     </xsl:attribute>
</smn:ATTR_NODE>
<smn:ATTR_NODE INTERNAL_NAME="sim0Ixz">
     <xsl:attribute name="VALUE">
     <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
     smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0Ixz&apos;]/@VALUE)" />
     </xsl:attribute>
</smn:ATTR_NODE>
<smn:ATTR_NODE INTERNAL_NAME="sim0Iyy">
     <xsl:attribute name="VALUE">
     <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
     smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0Iyy&apos;]/@VALUE)" />
     </xsl:attribute>
</smn:ATTR_NODE>
<smn:ATTR_NODE INTERNAL_NAME="sim0Iyz">
     <xsl:attribute name="VALUE">
     <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
     smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0Iyz&apos;]/@VALUE)" />
     </xsl:attribute>
</smn:ATTR_NODE>
<smn:ATTR_NODE INTERNAL_NAME="sim0Izz">
     <xsl:attribute name="VALUE">
     <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
     smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0Izz&apos;]/@VALUE)" />
     </xsl:attribute>
</smn:ATTR_NODE>
<smn:ATTR_NODE INTERNAL_NAME="sim0Ip1">
     <xsl:attribute name="VALUE">
     <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
     smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0Ip1&apos;]/@VALUE)" />
     </xsl:attribute>
</smn:ATTR_NODE>
<smn:ATTR_NODE INTERNAL_NAME="sim0Ip2">
     <xsl:attribute name="VALUE">
     <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
     smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0Ip2&apos;]/@VALUE)" />
     </xsl:attribute>
</smn:ATTR_NODE>
<smn:ATTR_NODE INTERNAL_NAME="sim0Ip3">
     <xsl:attribute name="VALUE">
     <xsl:value-of select="concat('',$currentIRnodeline/smn:ATTR_NODES/
     smn:ATTR_NODE[@INTERNAL_NAME=&apos;sim0Ip3&apos;]/@VALUE)" />
     </xsl:attribute>
</smn:ATTR_NODE>
```

```
</smn:ATTR_NODES>
</smn:NODE_LINE>
```

c. Save the **datamapping.xml** file and upload it to the **CAEStructureMap** dataset attached to the item revision indicated by the **CAE_datamap_files_location** preference.

6. Edit the **NodeXMLConfig.xml** file.

You can add the extended properties to the **NodeXMLConfig.xml** file so that these properties are available when a simulation analyst:

- Creates model revisions using structure maps and data map rules; and
- Derives one or more model structures from an existing model structure.

a. Locate the **NodeXMLConfig.xml** file and open it.

The **NodeXMLConfig.xml** sample file is located in the *TC_DATA* directory. However, the file used at runtime is managed in a **CAEStructureMap** dataset attached to the item revision indicated by the **CAE_datamap_files_location** preference.

The following is an example of the file before adding custom properties:

```
<smn:NODE_LINE CLASS="CAEModelRevision" TYPE="CAEModelRevision">
     <smn:ATTR_NODES>
          <smn:ATTR_NODE INTERNAL_NAME="object_name"
          NAME="Name" TYPE="Attribute"/>
          <smn:ATTR_NODE INTERNAL_NAME="object_desc"
          NAME="Description" TYPE="Attribute"/>
          <smn:ATTR_NODE INTERNAL_NAME="item_id"
          NAME="ID" TYPE="Runtime"/>
          <smn:ATTR_NODE INTERNAL_NAME="structure_revisions"
          NAME="BOM View Revisions" TYPE="Reference"/>
          <smn:ATTR_NODE INTERNAL_NAME="ps_children"
          TYPE="Runtime"/>
          <smn:ATTR_NODE INTERNAL_NAME="project_ids"
          NAME="Project IDs" TYPE="Runtime"/>
     </smn:ATTR_NODES>
</smn:NODE_LINE>
```

b. Add the custom properties.

Example of the same file after adding custom properties.

```
<smn:NODE_LINE CLASS="CAEModelRevision" TYPE="CAEModelRevision">
     <smn:ATTR_NODES>
          <smn:ATTR_NODE INTERNAL_NAME="object_name"
```

```
NAME="Name" TYPE="Attribute"/>
<smn:ATTR_NODE INTERNAL_NAME="object_desc"
NAME="Description" TYPE="Attribute"/>
<smn:ATTR_NODE INTERNAL_NAME="item_id"
NAME="ID" TYPE="Runtime"/>
<smn:ATTR_NODE INTERNAL_NAME="structure_revisions"
NAME="BOM View Revisions" TYPE="Reference"/>
<smn:ATTR_NODE INTERNAL_NAME="ps_children" TYPE="Runtime"/>
<smn:ATTR_NODE INTERNAL_NAME="project_ids"
NAME="Project IDs" TYPE="Runtime"/>
<smn:ATTR_NODE INTERNAL_NAME="sim0RepresentationType"
NAME="Representation Type" TYPE="Attribute"/>
<smn:ATTR_NODE INTERNAL_NAME="sim0MasterFormat"
NAME="Master Format" TYPE="Attribute"/>
<smn:ATTR_NODE INTERNAL_NAME="sim0Materials"
NAME="Materials" TYPE="Attribute"/>
<smn:ATTR_NODE INTERNAL_NAME="sim0Thicknesses"
NAME="Thicknesses" TYPE="Attribute"/>
<smn:ATTR_NODE INTERNAL_NAME="sim0MeshFlags"
NAME="Mesh Flags" TYPE="Attribute"/>
<smn:ATTR_NODE INTERNAL_NAME="sim0PIDFormula"
NAME="PID Formula" TYPE="Attribute"/>
<smn:ATTR_NODE INTERNAL_NAME="sim0PIDList"
NAME="PID List" TYPE="Attribute"/>
<smn:ATTR_NODE INTERNAL_NAME="sim0MIDs"
NAME="MIDs" TYPE="Attribute"/>
<smn:ATTR_NODE INTERNAL_NAME="sim0MeshDensity"
NAME="Mesh Density" TYPE="Attribute"/>
<smn:ATTR_NODE INTERNAL_NAME="sim0MeshQuality"
NAME="Mesh Quality" TYPE="Attribute"/>
<smn:ATTR_NODE INTERNAL_NAME="sim0MeshQualityCheck"
NAME="Mesh Quality Check" TYPE="Attribute"/>
<smn:ATTR_NODE INTERNAL_NAME="sim0Mass"
NAME="Mass" TYPE="Attribute"/>
<smn:ATTR_NODE INTERNAL_NAME="sim0CoGx"
NAME="CoGx" TYPE="Attribute"/>
<smn:ATTR_NODE INTERNAL_NAME="sim0CoGy"
NAME="CoGy" TYPE="Attribute"/>
<smn:ATTR_NODE INTERNAL_NAME="sim0CoGz"
NAME="CoGz" TYPE="Attribute"/>
<smn:ATTR_NODE INTERNAL_NAME="sim0Ixx"
NAME="Ixx" TYPE="Attribute"/>
<smn:ATTR_NODE INTERNAL_NAME="sim0Ixy"
NAME="Ixy" TYPE="Attribute"/>
<smn:ATTR_NODE INTERNAL_NAME="sim0Ixz"
NAME="Ixz" TYPE="Attribute"/>
<smn:ATTR_NODE INTERNAL_NAME="sim0Iyy"
NAME="Iyy" TYPE="Attribute"/>
<smn:ATTR_NODE INTERNAL_NAME="sim0Iyz"
```

```
            NAME="Iyz" TYPE="Attribute"/>
            <smn:ATTR_NODE INTERNAL_NAME="sim0Izz"
            NAME="Izz" TYPE="Attribute"/>
            <smn:ATTR_NODE INTERNAL_NAME="sim0Ip1"
            NAME="Ip1" TYPE="Attribute"/>
            <smn:ATTR_NODE INTERNAL_NAME="sim0Ip2"
            NAME="Ip2" TYPE="Attribute"/>
            <smn:ATTR_NODE INTERNAL_NAME="sim0Ip3"
            NAME="Ip3" TYPE="Attribute"/>
      </smn:ATTR_NODES>
</smn:NODE_LINE>
```

c. Save the **NodeXMLConfig.xml** file and upload it to the **CAEStructureMap** dataset attached to the item revision indicated by the **CAE_datamap_files_location** preference.

Migrate MDAO object table properties to Teamcenter Parameter Management System

Multidisciplinary Analysis and Optimization (MDAO) is a software-driven methodology that lets you optimize systems through simulation and across engineering disciplines.

For more information, see *MDAO in Active Workspace* in *Model-Based Systems Engineering*.

Execute the **swp_migrate_parameters** utility to convert the **Input Variable** and **Output Response** properties created during Teamcenter 14.2 or prior to 14.2 to Teamcenter Parameter Management System.

To view the command line help for this utility, type **swp_migrate_parameters -h** on the Teamcenter command prompt.

Install Teamcenter Tool Launcher Client for Active Workspace

- Why install TTLC?
- Install TTLC using quick setup shell script (Linux)
- Install TTLC using quick setup bat file (Windows)
- Uninstall TTLC

Why install TTLC?

In Active Workspace, the simulation analyst creates geometry, model, and analysis revisions and launches a preconfigured simulation tool. The tool uploads the simplified geometry, meshing data, or analysis data to the respective application, runs the application, and then uploads the resulting data to the corresponding revision created by the analyst. To run simulation tools or open external links, you must install Teamcenter Tool Launcher Client (TTLC) on the client machine.

The simulation analyst can launch simulation tools on Active Workspace using TTLC only if the simulation administrator has configured simulation tools using the **Local Launch** option. TTLC is not required for launch methods such as **Remote Launch** and **Server Launch**.

TTLC supports single sign-on (SSO) access to SSO-enabled Teamcenter applications. This eliminates the need for users to log on to each Teamcenter application separately. For more information on installing Security Services, see Security Services Configuration.

> Note:
>
> When an SSO setup is done based on the **Kerberos** protocol for Teamcenter, then TCCS must be installed on the client machine where TTLC is installed. On this machine, ensure that the **FMS_HOME** environment variable is set to the valid path. For more information, see Configure TCCS for Kerberos.

TTLC connects to FSC by using the values defined in the **Fms_BootStrap_Urls** preference.

For information about TTLC and Data Share Manager combinations that are not supported with OpenJDK, search for *TTLC and DSM* in the knowledge base articles on Support Center.

Install TTLC using quick setup shell script (Linux)

1. Set **JAVA_HOME**.

2. Copy the **additional_applications/TTLC_installer.zip** file from the installation kit to a local machine in a directory where you want to install Tool Launcher Client and unzip its contents. For example, you can unzip its contents to the **workdir/TTLC_installer** directory.

3. From a command line, change to the directory where you have copied the Tool Launcher Client installer.

4. To set up the Tool Launcher Client environment, run the following command:

   ttlc_quick_setup.sh

5. To access sample usage help provided by this script, run the following command.

   ttlc_quick_setup.sh -h

6. To verify the TTLC version number, run the following command after installing TTLC:

   ttlc_quick_setup.sh -version

7. After installing TTLC, you must set the default application for launching simulation tools and verify the port number and server session time out properties.

Install TTLC using quick setup bat file (Windows)

1. Set **JAVA_HOME**.

2. Copy the **additional_applications\TTLC_installer.zip** file from the installation kit and unzip its contents to a local machine in a folder where you want to install Tool Launcher Client. For example, you can unzip its contents to the **D:\workdir\TTLC\11.4.0.1\TTLC_installer** directory.

3. Open a command prompt with *administrative privileges* and browse to the folder where you want to install Tool Launcher Client, for example, **D:\workdir\TTLC\11.4.0.1\TTLC_installer**.

4. To set up the Tool Launcher Client environment, run the following command:

   ttlc_quick_setup.bat

5. To access sample usage help provided by this script, type **ttlc_quick_setup.bat -h** in the command prompt.

6. To verify the TTLC version number, run the following command after installing TTLC:

   ttlc_quick_setup.bat -version

7. After installing TTLC, you must set the default application for launching simulation tools and verify the port number and server session time out properties.

Uninstall TTLC

To uninstall TTLC, delete the directory where you have copied the installation files.

After installing TTLC

Set the default application for launching simulation tools after installing TTLC

After you install TTLC, you can set the default application for launching simulation tools as **Teamcenter Tool Launch Client**. To do so:

1. Launch a simulation tool.

   The system generates the **.tcsimxml** file and downloads it on your browser.

2. Browse to the location of **.tcsimxml** file, right-click, and select **Properties**.

3. (Linux) Select the **Open With** tab, choose the **Teamcenter Tool Launch Client** application, and click **Set as default**.

   OR

   (Windows) Select the **General** tab, click the **Change** button beside the **Opens with** option, click **More Apps**, choose **Look for another app on this PC**, select the **tcsimtoollauncher.bat** file from the location where you have installed TTLC, and click **OK**.

Verify the port number and server session time out properties after installing TTLC

1. Open the **TTLC_config.properties** file from the location where you have installed the Tool Launcher Client.

2. Verify the port number for the **TTLC_port** property value. The default value is **9876**.

   The TTLC client socket connects to the TTLC server socket on this port. For more information, see the **TTLC_config.properties** file.

   This is the TTLC server socket number and not the Teamcenter server port number.

   To find out if this port number is being used, type **netstat -ano | findstr 9876** (Windows example) from a command prompt. If you want to change the port number, edit this property value. Be sure to change it before you start any simulation tools.

3. Verify the **TTLC_server_session_timeout** property value. The default idle time is **60** minutes.

   The TTLC server socket log outs from the Teamcenter server and stops the TTLC server socket after the specified idle time.

   For more information, see the **TTLC_config.properties** file.

4. To find the Active Workspace URL and Web server port number, after you launch a simulation tool, open the **.tcsimxml** file. Look for the **HostPath** property:

   Example:

   <HostPath>http://*machine_name***:3000/tc/</HostPath>**

   In the above example, the Active Workspace URL is followed by **3000**, which is the default Web server port number.

Create mass client deployment scripts for TTLC

The process for creating mass deployment scripts for Teamcenter Tool Launcher Client (TTLC) is as follows:

1. **Create mass client deployment scripts for TTLC**
2. **Create the mini kit for the TTLC installer**
3. **Install TTLC using the deploy script and the mini kit**

Why install TTLC?

In Active Workspace, the simulation analyst creates geometry, model, and analysis revisions and launches a preconfigured simulation tool. The tool uploads the simplified geometry, meshing data, or

analysis data to the respective application, runs the application, and then uploads the resulting data to the corresponding revision created by the analyst. To run simulation tools or open external links, you must install Teamcenter Tool Launcher Client (TTLC) on the client machine.

The simulation analyst can launch simulation tools on Active Workspace using TTLC only if the simulation administrator has configured simulation tools using the **Local Launch** option. TTLC is not required for launch methods such as **Remote Launch** and **Server Launch**.

TTLC supports single sign-on (SSO) access to SSO-enabled Teamcenter applications. This eliminates the need for users to log on to each Teamcenter application separately. For more information on installing Security Services, see Security Services Configuration.

TTLC connects to FSC by using the values defined in the **Fms_BootStrap_Urls** preference.

For information about TTLC and Data Share Manager combinations that are not supported with OpenJDK, search for *TTLC and DSM* in the knowledge base articles on Support Center.

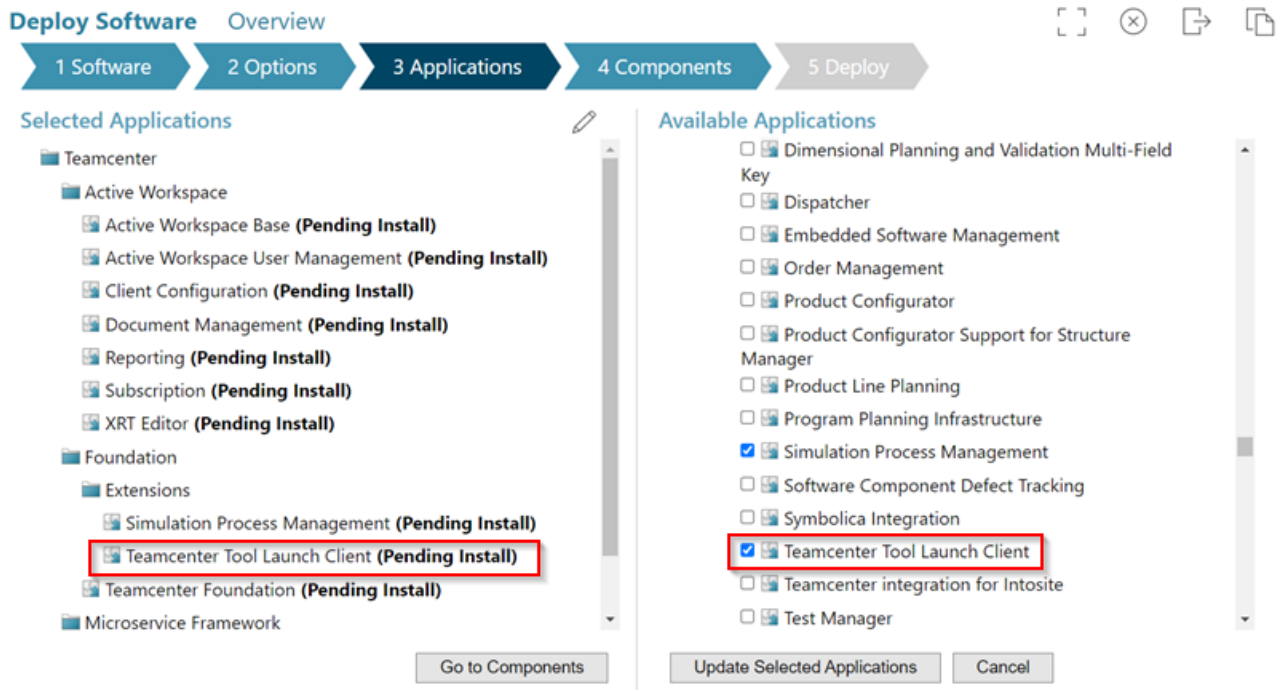
Create mass client deployment scripts for TTLC

The following procedures assume that you are installing Simulation Process and Data Management applications on an existing Teamcenter set up with Active Workspace and that you are familiar with Deployment Center.

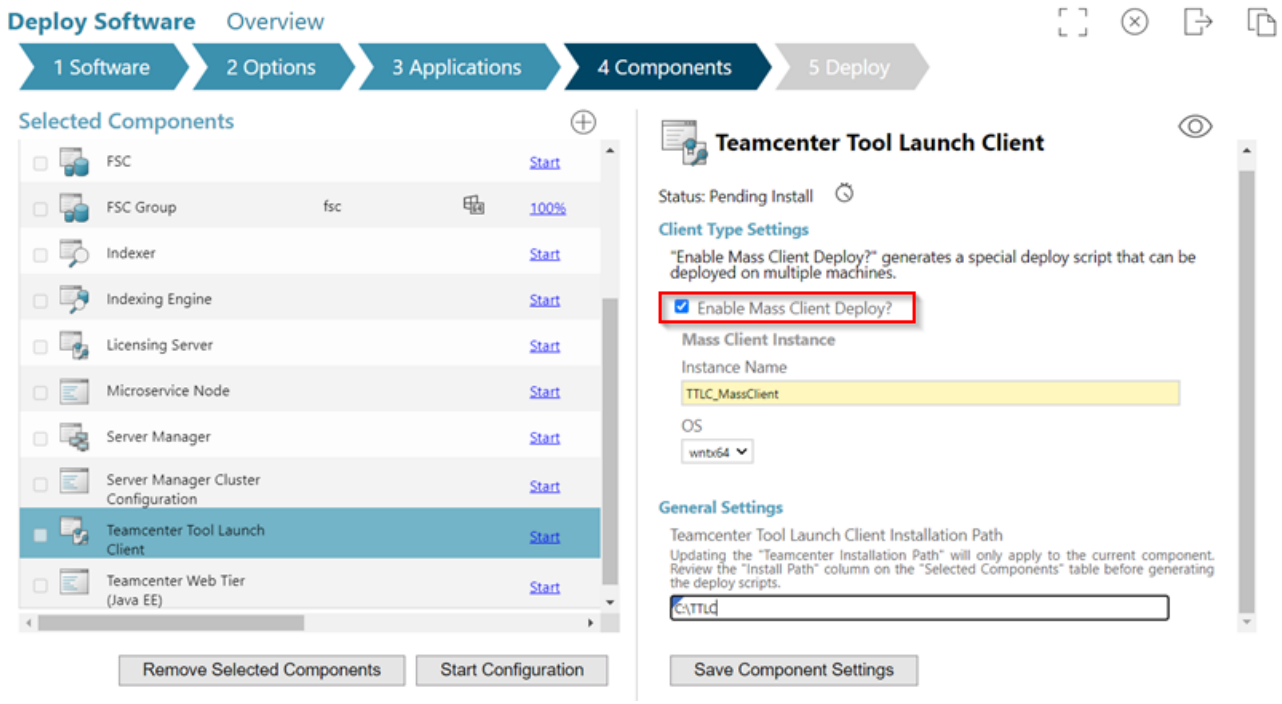
For information about using Deployment Center, see *Deployment Center — Usage*.

1. Log on to Deployment Center.

2. Select **Teamcenter Tool Launch Client** in the **Applications** task.



3. Select the **Enable Mass Client Deploy** option in the **Components** task and select other required options or specify information as appropriate.



4. Run the mass client deploy script for the generated software mini kit.

For more information, see *How to deploy using a software mini kit* in *Deployment Center — Usage* on Support Center.

5. Install TTLC using the deploy script and mini kit.

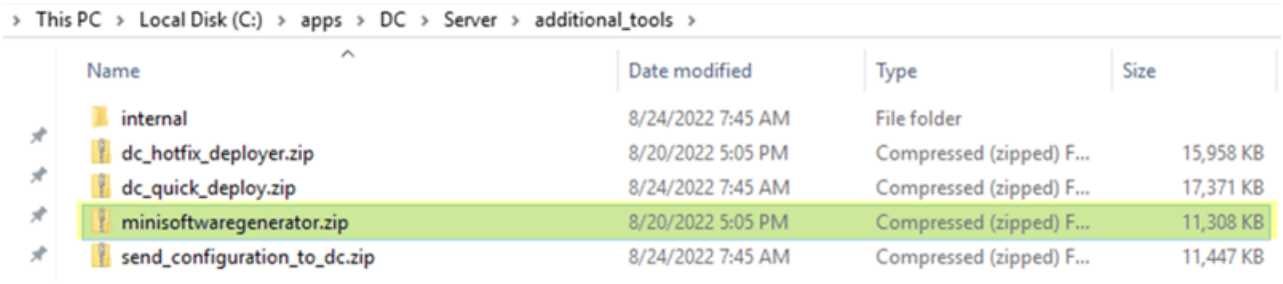
Create the mini kit for the TTLC installer

You can reduce the size of your mass client deployment files by generating a mini software kit. Running the **generateMiniSoftware** utility extracts only the necessary software from the full software kits and builds a smaller, more portable software kit for the specific client deployment.

For more information, see Generating mini kits for mass client installations in *Deployment Center — Usage* on Support Center.

1. Run the **generateMiniSoftware.bat** utility after the mass client deploy script for the TTLC installer is generated from **step 4**.

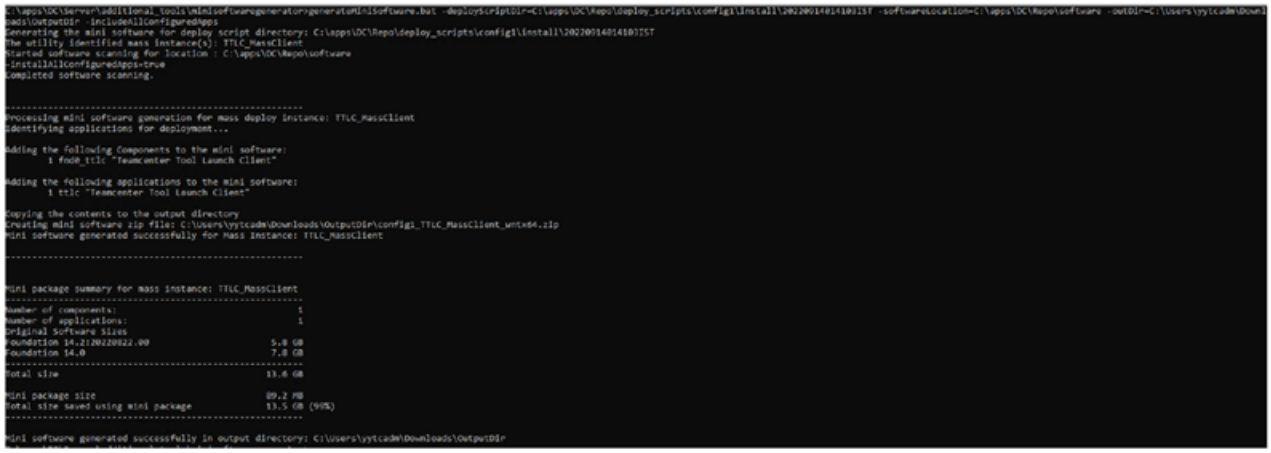
   This utility is located in a zip file **minisoftwaregenerator.zip** in the **additional_tools** directory.

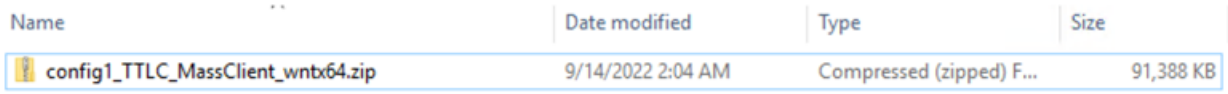


2. To generate a kit with TTLC installed, run the following command:

   generateMiniSoftware.bat
   -deployScriptDir=*C:\apps\DC\Repo\deploy_scripts\config1\install\20220914014103IST*
   -softwareLocation=*C:\apps\DC\Repo\software* **-outDir=***C:\Downloads\OutputDir*
   -includeAllConfiguredApps

   The output of the utility looks as follows:





Install TTLC using the deploy script and the mini kit

For more information, see Run the deployment scripts in *Deployment Center — Usage* on Support Center.

1. Use the mini kit created in **Create the mini kit for the TTLC installer** to install TTLC. The command for the deploy script is as follows:

   deploy.bat -dcusername=*User_Name* **-dcpassword=***Password*
   -softwareLocation=*C:\Downloads\OutputDir\config1_TTLC_MassClient_wntx64*

   The software location is now the path of the mini kit instead of the original software location.

2. Verify whether TTLC is installed properly.

3. **After installing TTLC**, you must set the default application for launching simulation tools and verify the port number and server session time out properties.

4. Configuring Simulation Process and Data Management

About configuring Simulation Process and Data Management

Role	Task	Description
Simulation administrator	Sets preferences to control Teamcenter's behavior and appearance	Use Teamcenter preferences to control various aspects of Teamcenter's behavior and appearance and to suit the requirements at your site. For information about retrieving a list of preferences, see *Where can I get a list of preferences?* in *Teamcenter Preferences*.
Simulation administrator	**Configures data map rules**	Define data map rules to automate the creation of model structures. Data map execution is driven by the contents of two XML files: a data map file (**datamapping.xml**) and a configuration file (**NodeXMLConfig.xml**). The rules describing the mappings are contained in the data map file.
Lead simulation analyst or a designated simulation analyst	Creates structure map rules	Define structure map rules to automate the creation of model structures. Structure map rules are used to create model structures for specific types of analyses.
A user with DBA privileges; OR a group administrator; OR a simulation analyst	**Configures rules to derive structures at the site, group, or user level**	Define derivative rules and variant configuration rules. Analysts can use these predefined rules to quickly derive one or more simulation variants from an existing one.
Simulation administrator	**Maps material revisions from the product structure to the model structure**	Create rules to allow the model structure to be exported with the material IDs and the associated material details.
Simulation administrator	**Configures the workflow process for running structure maps**	Configure a workflow process to allow simulation analysts to create model structures by using structure map rules.
Simulation administrator	**Specifies comparison options for comparing a CAE model structure against a product structure**	Define product structure attributes, simulation structure attributes, and root node attributes for comparing structures.

Role	Task	Description
Simulation administrator	**Configures attribute values for structure comparison**	Define attribute values for comparing structures. These comparison options are used by simulation analysts to investigate differences in attributes such as thickness, associated files or load cases, and file content.
Simulation administrator or a user with DBA privileges	**Configures simulation tools**	Configure rules to allow simulation analysts to launch simulation tools that can include preprocessors, solvers, and postprocessors.
An administrative user with DBA privileges	**Specify HPC connection and profile information**	Identifies business needs for a simulation launch tool to use the high performance computing (HPC) configuration. Based on the needs, an HPC configuration is created that includes connection and profile information.
Simulation administrator	**Configures simulation tools to extract the key performance indicator (KPI) values automatically**	Set up sample configurations by running the **tcsim_quick_setup.pl** script. The sample simulation tool configurations includes the **Extract KPI from Result** tool.
Simulation administrator	**Sets up Dispatcher to launch simulation tools from different machines**	Set up Dispatcher on different machines or modules that have simulation tools. Simulation analysts can launch simulation tools on the required modules.
Simulation administrator	**Enables SimProcess and NxNastran services on Dispatcher**	Enable the **SimProcess** and **NxNastran** services on Dispatcher to launch a simulation process or NX Nastran remotely.
A user with DBA privileges; OR a simulation administrator; OR a group administrator; OR a simulation analyst	**Configures analysis or CAE packages at the site, group, or user level**	Configure CAE packages to help simulation analysts standardize the creation a set of CAE item revisions for a specific type of analysis.
Simulation administrator	**Enables pedigree operations**	Enable pedigree operations and specify valid relation types for model relations and analysis relations. Pedigree information captures the exact configuration of various structures, including the revision rule, effectivity, and variant rules, and persists that information in the database.
Simulation administrator	**Configures file upload rules to support analysis on local desktops**	Define file upload rules for simulation analysts to upload or download the analysis files to or from Teamcenter.
Simulation administrator	**Configures the simulation dashboard**	Configure analysis or model dashboards.

Role	Task	Description
Simulation administrator	Runs scripts to quickly set up sample configurations	Run scripts to quickly set up sample configurations on a virtual machine or a sandbox environment. This provides an easy setup for presales or customers who want to explore the Simulation Process and Data Management capabilities.
Simulation administrator	Installs Teamcenter Tool Launcher Client for Active Workspace	Install Teamcenter Tool Launcher Client (TTLC) on each client machine to run simulation tools or open external links in Active Workspace.
A user with DBA privileges	Edits style sheets to expose custom revision types in the Simulation tab for Active Workspace	Edit style sheets to expose custom revision types in the context of the product, geometry, model, analysis, or the result revision. Also, configure traversal paths to define how to traverse the data structure and specify which relationships are of interest and what should be done when these relationships are found.
An administrative user with DBA privileges	Customize the tool launch page in Active Workspace	The simulation administrator configures simulation tools and the simulation analyst uses these preconfigured simulation tools to launch preprocessors, solvers, or postprocessors. Each tool might require different inputs based on the requirements of the tool. After configuring simulation tools, an administrative user with DBA privileges can create multiple customized launch pages for different launch tools and associate specific tools to the customized pages.

Define schemas and expose object properties in the GUI

Define schema definitions

You can configure and manage the CAE schema and attributes in Business Modeler IDE.

The following schema definitions and attributes are defined in Teamcenter to manage CAE data.

Form storage classes:

CAEMasterStore	CAERevMasterStore
CAEAnalysisMasterStore	CAEAnalysisRevMasterStore
CAEGeometryMasterStore	CAEGeometryRevMasterStore
CAEModelMasterStore	CAEModelRevMasterStore
CAEResultMasterStore	CAEResultRevMasterStore
StructureMapMasterStore	StructureMapRevMasterStore
	CAESolverManagerStore

Item classes and item revision classes:

CAEItem	CAEItemRevision
CAE Analysis	CAE Analysis Revision
CAE Geometry	CAE Geometry Revision
CAE Model	CAE Model Revision
CAE Result	CAE Result Revision
CAE Structure Map	CAE Structure Map Revision

CAEAnalysisMasterStore, **CAEAnalysisRevMasterStore**, **CAE Analysis** item class, and **CAE Analysis Revision** item revision class are enhanced and customized for CAE Manager.

Expose object properties in the GUI

The Business Modeler IDE (Integrated Development Environment) is a tool for configuring and extending the data model of your Teamcenter installation. In Business Modeler IDE, you can use the **Operation Descriptor** tab on business objects to make properties visible and required on creation dialog boxes in the rich client. These properties are exposed when these objects are created in the rich client.

Simulation Process and Data Management provides a specialized wizard, the **New CAE Item** wizard, for creating items and item revisions that are subtypes of the CAE item. This wizard exposes properties of these item subtypes that are declared in the **Operation Descriptor** tab in Business Modeler IDE. The wizard provides steps for establishing certain references to other item revisions by leveraging CAE relationship types. When you invoke this wizard from CAE Manager, the wizard presents logical defaults for these references based on the item revisions you select or load in CAE Manager.

> Note:
>
> The administrator can alter subtypes of the CAE item using the **Operation Descriptor** tab and the analyst can subsequently launch the **New CAE Item** wizard to enter item properties and subtype

properties. However, if the administrator creates subtypes of the CAE item directly, the wizard does not support creation of the subtypes that are created directly.

Map product items to model items using a data map

Map product items to model items

In CAE Manager, analysts can generate a CAE structure from an existing product structure. The data map determines the default mapping of product items to CAE model items before any structure map is applied.

Using the sample data map file, a **CAE 3D Model** item is created for each generic item in the product structure. In practice, your company will have defined one or more custom item types to handle product structure data as part of its business model. For each custom product item type in your company's business model, you (as administrator) need to create a mapping rule that determines the corresponding CAE item type in the CAE structure. Generally, you need to do this only once, unless your company's business model changes.

You can create domains to define different rules, for example, your company might have different CAE disciplines for safety, durability, and so on. The company wants data mapping to be different for each discipline, but there is only one data map XML file for the whole site. In such situations, you can create and define domains.

Data map execution is driven by the contents of two XML files—a data map file (**datamapping.xml**) and a configuration file (**NodeXMLConfig.xml**). The rules describing the mappings are contained in the data map file. These rules determine:

- The CAE items and item revisions that are created when you generate a CAE structure from a product structure.
- The relations between product item revisions and CAE item revisions.
- The naming of the resulting CAE items and item revisions.
- The mapping of all attached datasets of a particular type to input item revisions having a specific relationship to be referenced or copied to the output structure with the same relationship or a different relationship type.

To manage data map definition files:

- Run the **cae_manage_datamap_definition** utility to update the data map and the **NodeXML** configuration file in the data map set. To view the command line help for this utility, type **cae_manage_datamap_definition -h** on the Teamcenter command prompt.
- The **CAE_datamap_files_location** preference is used to manage the location of the data map definition files.

The preference indicating the item revision containing this information is a site preference. Only a user with DBA privileges can set it.

- Any type of item revision can be used to manage these files.

  The item revision is owned by the user who is responsible for defining this information in the customer's environment. Often, this is a user with the simulation administrator role.

- A **CAEStructureMap** dataset must be attached to the item revision using a **Specifications** relationship.

  This dataset:

  - Must contain at least one XML reference, which is the **datamapping.xml** file.
  - May contain a second XML reference, which is the **NodeXMLConfig.xml** file.

The simulation administrator can edit the sample **NodeXMLConfig.xml** file to control the items, item revisions, forms, or BOM line attributes that are processed. The administrator can edit the required attributes in the XML file in such a way that Teamcenter processes only the required attributes. This significantly improves the performance and memory usage of data map and structure map execution operations.

Tasks to perform before customizing the data map

Data map namespaces and schemas

The data map file is an XML file that resides in the *TC_DATA* directory. It uses elements from three different namespaces:

- `xmlns:smr` - Declares the **SMERule** schema definition. The elements in this schema define the data mapping rules and the XSL templates to apply. The **SMERule** schema definition is named **tcsim_sm_rule.xsd**, and is located in the *TC_DATA* directory. The schema definition file is fully documented.

- `xmlns:smn` - Declares the **SMENodes** schema definition. The elements in this schema are used to read and write information about nodes in a structure, such as item class, type, relations, name, and description. The **SMENodes** schema definition is named **tcsim_sm_node.xsd**, and is located in the *TC_DATA* directory. The schema definition file is fully documented.

- `xmlns:xsl` is the XSL Transformations (XSLT 1.0) language as defined by the World Wide Web Consortium (W3C). Elements in this namespace are used to define conditional statements and the XSL templates to apply to nodes that meet the defined conditions.

```
<smr:RULES
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
```

```
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:smn="http://www.ugs.com/Schemas/SMENode"
  xmlns:smr="http://www.ugs.com/Schemas/SMERule"
  xsi:schemaLocation="http://www.ugs.com/Schemas/SMERule
tcsim_sm_rule.xsd">
</smr:RULES>
```

Before customizing the data map

- To prepare for customizing the data map, have the following information ready:
  - The name of each custom item type and item revision type your company uses to manage CAD data.
  - The number of different product item types you need to support.
  - The default naming scheme to apply to CAE items created by the data map.
  - Relations and attributes that you want to map before executing data map rules.

    A relationship is required to copy or reference a dataset. You can define this relationship as a simple property on the source object, that is, the **CAE 3D Model** item revision.

    You must turn on **Operation Descriptor**s for the attributes and relationships of any objects that you want to map before executing data map rules against those objects.

    The Business Modeler IDE (Integrated Development Environment) is a tool for configuring and extending the data model of your Teamcenter installation. In Business Modeler IDE, you can use the **Operation Descriptor** tab on business objects to make properties visible and required on creation dialog boxes in the rich client. These properties are exposed when these objects are created in the rich client.
  - A compound property is a property that is displayed on one object (the display object) although it is defined and resides on a different object (the source object). The display object and source object are related by one or more Teamcenter relations and reference properties. If you have a custom form attached to an item revision with a custom relationship, and the output is a **CAE 3D Model** item revision, then you must add the custom relation as a compound property to the output.

    Example: If *xyz* custom form is attached to the **CAE 3D Model** item revision with *abc* custom relationship, you must use the **Operation Descriptor** tab on business objects in Business Modeler IDE to add the *abc* custom relationship as a compound property to the **CAE 3D Model** item revision.
- Before data mapping objects in Business Modeler IDE, apply schema changes, install templates, and update the database using TEM.

Changed business objects and properties are considered as schema changes. You can use Business Modeler IDE to extend Teamcenter with your own business objects, properties, and business behavior. To do so, store extensions using a template and then deploy this template.

- Identity all attributes of any objects that you want to map in the **NodeXMLConfig.xml** configuration file for *each* domain. The **FOCUS="IN"** attribute identifies objects types that are the source objects and **FOCUS="OUT"** attribute identifies object types that are the target objects. You must set the **STATUS="ACTIVE"** attributes for all objects for custom queries to pull in the attributes.

- Run the **cae_configure_dm_propertyset** utility each time you make changes to the **NodeXMLConfig.xml** configuration file. To view the command line help for this utility, type **cae_configure_dm_propertyset -h** on the Teamcenter command prompt.

  This utility reads the **NodeXMLConfig.xml** file and creates custom property sets that are used for data mapping.

  Example:

  cae_configure_dm_propertyset -u=*user_id* **-p=***password* **-g=***dba*

> Note:
>
> After editing the **datamapping.xml** file, you must save the file with UTF-8 encoding if the attributes have special characters in it.

Sample data map file

Teamcenter provides a sample data map file that you can use as a template for developing your own data maps. The sample data map file, **datamapping.xml**, is located in the *TC_DATA* directory.

This data map file reflects the *commercial-off-the-shelf (COTS)* functionality of CAE Manager in Teamcenter. You can use this file as a template when defining a data map that supports your business model.

The sample data map file applies a rule to a selected product structure that traverses the nodes in the product structure and performs the following actions:

- If the current node is an item of the type *Item*, the data map creates a corresponding **CAE 3D Model** item in the CAE structure, with a **CAE Target** relation to the product item.

- If the current node contains an item revision of the type *ItemRevision*, the data map creates a corresponding **CAEModelRevision** item revision.

- If the current node represents a product structure occurrence (**PSOccurrence**), the data map creates a corresponding occurrence in the CAE structure.

- In each case, the data map extracts the name and description of the node from the product structure and assigns it to the corresponding node in the CAE structure.

The following is an example. For more information, see the **datamapping.xml** file in the *TC_DATA* directory.

```
<smr:RULES
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:smn="http://www.ugs.com/Schemas/SMENode"
  xmlns:smr="http://www.ugs.com/Schemas/SMERule"
  xsi:schemaLocation="http://www.ugs.com/Schemas/SMERule tcsim_sm_rule.xsd">
  <smr:RULE
    TYPE="MAPPING"
    DOMAIN="CAE"
    NAME="Item Mapping"
    DESCRIPTION="">
    <xsl:if
      test="smn:NODES/smn:NODE/
            smn:NODE_LINE[@CLASS=&apos;Item&apos; and @TYPE=&apos;Item&apos;]">
      <xsl:call-template
        name="Item">
        <xsl:with-param
          name="node"
          select="."/>
        <xsl:with-param
          name="ruleName"
          select="&apos;Item Mapping&apos;"/>
      </xsl:call-template>
    </xsl:if>
  </smr:RULE>
```

Configuring subtypes of input and output item types

The default **datamapping.xml** and **datamapping_NXCAE_CustomInputType.xml** provides guidance to replace **Item** and **CAEModel** with the subtype names. However, there are a lot of instances to be replaced and it can be error prone. The files should be such that a novice user should do changes at a few places at the beginning of the file instead of doing a *find* and *replace* in the entire file.

- A parametrized **datamaping_parametrized.xml** file is provided as an example in addition to the default **datamapping.xml** file. It allows users makes changes in fewer places to make it work with subtypes.

  For more information, refer the *TC_DATA*/**datamapping_parametrized.xml** file.

- The **datamapping_NXCAE_CustomInputType.xml** file is changed so that the user needs to make minimal changes to make it work with subtypes.

  For more information, refer the *TC_DATA*/**datamapping_NXCAE_CustomInputType.xml** file.

Use BOM line attributes to define data mapping rules

Analysts can create a simulation structure by applying a set of rules to a product structure. Before the analysts can create the simulation structure, the administrator or lead analyst must define certain data mapping rules.

- Example 1: Your site has a predictable number of levels in the product structure representing a similar product organization. In such cases, your site administrator can define data mapping rules to leverage properties from specific levels above the current object in the structure. for example, *two* levels above the current object.

- Example 2: Your site has predictable types of objects in the product structure representing product organization. In such cases, your site administrator can define data mapping rules to leverage properties from a specific type of object that occurs above the current object in the structure. An example of this is, the *closest ancestor* of the **ProductDesign** type, which is defined as a custom item type at your site.

> Note:
>
> Teamcenter captures data mapping rules in an XML file. The administrator or lead analyst can use any suitable text editor to create or edit the data mapping file.

The following are the default **BOMLine** attributes available in the *TC_DATA***datamapping.xml** file:

- **Find No.**: Find the number of the line in the structure.

- **Relative Transformation Matrix**: Structure Manager fills the absolute transformation matrix fields when it rolls up the relative transformation matrices of the structure.

- **ID In Context (Top Level)**: Shows only the identifier assigned to the line in the context of the loaded top-level line. Any absolute occurrence identifiers defined at a level lower than the currently selected top-level line are not visible.

- **ID In Context (All Levels)**: Shows the identifiers assigned to the line in all contexts.

To view more attributes, select an item revision in the **Product** or **Model** view in CAE Manager, and click the **Generate Node XML** icon. You can add any of these attributes by customizing the *TC_DATA***datamapping.xml** file.

Common customization points in the data mapping file

The amount and type of data map customization needed is highly dependent on your business model. However, the following are the most common customization points, which you can perform by using the sample **datamapping.xml** file as a template and modifying it as needed:

- Supporting custom item types and item revision types. This is the most common customization needed. The COTS behavior is to create a **CAE 3D Model** item and item revision for each generic item type in the product structure. In practice, most companies define custom item types to manage product structure data.

- Supporting multiple product structure item types. For each product structure item type defined in your business model, you need to create a corresponding rule in the data map. Depending on your business model and workflows, each rule may call the same template, or you can define unique templates for each rule.

- Defining rule-based naming for CAE items. You can define different naming conventions for the resulting **CAE 3D Model** items and item revisions, depending on the input product structure type.

  For example, suppose you always mesh Pre/Post CAD data in Pre/Post, but you mesh IGES CAD data in FEMAP. You can prepend **CAE 3D Model** item names created from Pre/Post data with the string **FEM-**, whereas **CAE 3D Model** items created from IGES data use the string **MOD-**.

> Note:
>
> The combination of standard XSL with the provided rule and node schemas is powerful and flexible, and depending on your knowledge of XSL, PLM XML, and your business model, much more extensive customization is possible.

Create domains to define different data mapping rules

You can create domains to define different rules, for example, your organization might have different CAE disciplines for safety, durability, and so on. The organization wants data mapping to be different for each discipline, but there is only one data map XML file for the whole site. In such situations, you can create and define domains. They help provide the flexibility to include different data mapping methods for each discipline while executing structure maps.

After you define a domain, analysts can use the domain when they create a **CAE Structure Map** item using the **New CAE Item** wizard.

Domains in the data map file

The following domains are available by default:

- **CAE**

  This is the default domain.

  Simulation analysts can select this domain to generate a model structure from an existing product structure or a product collector for both the rich client and Active Workspace.

  In addition, simulation analysts can generate model structures from an existing CAE geometry structure on Active Workspace.

- **CFD**

  This domain option is similar to the default **CAE** option. It additionally maps the JT files from the source structure to the output structure.

  Simulation analysts can select this domain to generate a model structure with mapped JT files from an existing product structure or a product collector for both the rich client and Active Workspace.

  In addition, simulation analysts can generate model structures with mapped JT files from an existing CAE geometry structure on Active Workspace.

- **CAEGA**

  Simulation analysts can select this domain to generate a geometry structure from an existing product structure or a product collector that contains CAD assemblies. This is supported only on Active Workspace.

For more information, see the **Mapping Rule to map the 4GD structure using a Subset Item** section in the **datamapping.xml** file.

```
<!-- Mapping Rule to map the 4GD structure using a Subset Item-->
<smr:RULE TYPE="MAPPING" DOMAIN="CAE" NAME="Subset Item Mapping" DESCRIPTION="">

<smr:RULE TYPE="MAPPING" DOMAIN="CAE" NAME="CAEGeometry Mapping" DESCRIPTION="">

<smr:RULE TYPE="MAPPING" DOMAIN="CAE" NAME="Collector Mapping" DESCRIPTION="">

<smr:RULE TYPE="MAPPING" DOMAIN="CFD" NAME="Item Mapping with JT Files" DESCRIPTION="">

<smr:RULE TYPE="MAPPING" DOMAIN="CFD" NAME="CAEGeometry Mapping For CFD" DESCRIPTION="">

<smr:RULE TYPE="MAPPING" DOMAIN="CFD" NAME="Collector Mapping For CFD" DESCRIPTION="">

<smr:RULE TYPE="MAPPING" DOMAIN="CAEGA" NAME="Item Mapping For CAEGA" DESCRIPTION="">

<smr:RULE TYPE="MAPPING" DOMAIN="CAEGA" NAME="Collector Mapping For CAEGA" DESCRIPTION="">
```

Create domains

You can create domains to define different data mapping rules.

Prerequisites

For domains to be effective, the domain value in the **New CAE Item** wizard must match the domain value defined in the **datamapping.xml** file.

Procedure

1. Define all domains in Business Modeler IDE.

After you define domains, they are available to analysts in the **Enter Additional Item Information** panel of the **New CAE Item** wizard.

The default domain is **CAE** and it is defined in the **datamapping.xml** file.

2. Add the domains to the **datamapping.xml** file.

Create a custom data map from a template

Use **datamapping.xml** as a template while creating a custom data map.

1. In the *TC_DATA* directory, locate the **datamapping.xml** file.
2. Create a copy of the data map file and give it a unique name.
3. Open the copy in a dedicated XML editor or a plain text editor.
4. Modify the file as needed to support your users and your business model.
5. To manage the data mapping definition file, in Teamcenter, create an item revision.
   a. Click **File→New→Item**.
   b. Click **Assign** to automatically generate the ID and revision identifiers.
   c. Type a name for the item and click **Finish**.
6. Create a **CAEStructureMap** dataset in the item revision you created using a **Specifications** relationship and import the customized data map file.
   a. Choose the item revision within the item master for the item revision you created.
   b. Click **File→New→Dataset**.
   c. Click **More** and choose **CAEStructureMap**.
   d. To import the customized data map file into the dataset as an XML reference, click **Import**.
   e. From the **Relation** menu, select **Specifications**.
7. Modify the **CAE_datamap_files_location** preference to point to the created item revision.

> Note:
>
> Only a user with dba privileges can perform this step.
>
> You can perform this any time after the item revision is created.

8. To test your custom data map, locate an existing product structure in My Teamcenter, right-click, and choose **Send To→CAE Manager**.

9. In the **Product** view, select the BOM line (product structure), and from the view buttons, click the **CAE 3D Model from Data Map** icon.

10. Select a domain to execute the data map.

Clone or reference variant information

Simulation analysts can use predefined data map rules and structure map rules to create a model structure from the product structure. Additionally, they can use predefined derivative rules to derive one or more structures from an existing model structure. Derivative rules use both the data map and structure map rules to derive structures.

The source structure contains variant information such as variant conditions, variant options, saved variant rules (SVRs), option defaults, and rule checks. By default, the system *clones* the variant options and SVRs to the target structure and maps the rest of the variant information to the target structure.

Some companies may have business practices that prevent the cloning of variant information. In such cases, you (as a simulation administrator) can edit the **varMappingValue** attribute in the **datamapping.xml** file to *reference* the variant options and SVRs instead of cloning the information.

1. Open the **datamapping.xml** file.

2. To reference variant information while creating a model structure from a product structure:

   a. Search for **Variants** in the **Item Revision** template.

      Example:

```
<!--************************************************************
      *********       Item Revision template            ************
      ************************************************************-->
        <xsl:template name="ItemRevision">
        <xsl:param name="node" />
        <xsl:variable name="currentIRnodeline" select="$node/smn:NODES/
smn:NODE/
          smn:NODE_LINE[@CLASS=&apos;ItemRevision&apos;
          and @TYPE=&apos;ItemRevision&apos;]">
        </xsl:variable>
```

```
        <smn:NODE_LINE CLASS="CAEModelRevision" TYPE="CAEModelRevision"
          STATE="MAPPING">
            <!--
**********************************************************
           *********     Item Revision attributes          *************

           The input item revision attributes that need to populate
           the output's item revision atrributes go here.
           ************************************************************-->
            <smn:ATTR_NODES>
                <!-- Input's Description transferred to output's attribute
of
                  same name-->
                <smn:ATTR_NODE INTERNAL_NAME="object_desc">
                    <xsl:attribute name="VALUE">
                        <xsl:value-of select="$currentIRnodeline/
smn:ATTR_NODES/

smn:ATTR_NODE[@INTERNAL_NAME=&apos;object_desc&apos;]/
                          @VALUE" />
                    </xsl:attribute>
                </smn:ATTR_NODE>
                <!-- Input's Name transferred to output's attribute of
                same name-->
                <smn:ATTR_NODE INTERNAL_NAME="object_name">
                    <xsl:attribute name="VALUE">
                        <xsl:value-of
select="concat('CAE-',$currentIRnodeline/
                          smn:ATTR_NODES/

smn:ATTR_NODE[@INTERNAL_NAME=&apos;object_name&apos;]/
                          @VALUE)" />
                    </xsl:attribute>
                </smn:ATTR_NODE>
            </smn:ATTR_NODES>
        </smn:NODE_LINE>

        <xsl:call-template name="Variants">
            <xsl:with-param name="node" select="$node" />
            <xsl:with-param name="varMappingValue"
select="&apos;CLONE&apos;" />
        </xsl:call-template>

    </xsl:template>
```

b. To reference SVRs, variant conditions, variant options, option defaults, and rule checks, edit as follows:

   Default **varMappingValue**:

```
<xsl:with-param name="varMappingValue"
select="&apos;CLONE&apos;" />
```

Change to:

```
<xsl:with-param name="varMappingValue"
select="&apos;REFER&apos;" />
```

> Tip:
>
> If you do not want to use variant information at your site, remove the **Variants** template.

3. To reference variant information while deriving structures:

   a. Search for **Variants** in the **CAE 3D Model Revision** template.

   Example:

```
<!--************************************************************
  *********       CAEModel Revision template            ************
     ************************************************************-->
    <xsl:template name="CAEModelRevision">
        <xsl:param name="node" />
        <xsl:variable name="currentIRnodeline" select="$node/smn:NODES/
smn:NODE/
          smn:NODE_LINE[@CLASS=&apos;CAEModelRevision&apos;
          and @TYPE=&apos;CAEModelRevision&apos;]">
        </xsl:variable>
        <smn:NODE_LINE CLASS="CAEModelRevision" TYPE="CAEModelRevision"
          STATE="MAPPING">
            <!--
***********************************************************
           *********     CAEModel Revision attributes          *************

           The input CAEModel revision attributes that need to populate
           the output's CAEModel revision atrributes go here.
           ************************************************************-->
            <smn:ATTR_NODES>
                <!-- Input's Description transferred to output's attribute
of
                  same name-->
                <smn:ATTR_NODE INTERNAL_NAME="object_desc">
                    <xsl:attribute name="VALUE">
                        <xsl:value-of select="$currentIRnodeline/
smn:ATTR_NODES/

smn:ATTR_NODE[@INTERNAL_NAME=&apos;object_desc&apos;]/
```

```
                        @VALUE" />
                    </xsl:attribute>
                </smn:ATTR_NODE>
                <!-- Input's Name transferred to output's attribute of
                same name-->
                <smn:ATTR_NODE INTERNAL_NAME="object_name">
                    <xsl:attribute name="VALUE">
                        <xsl:value-of select="concat('',$currentIRnodeline/
                          smn:ATTR_NODES/

smn:ATTR_NODE[@INTERNAL_NAME=&apos;object_name&apos;]/
                          @VALUE)" />
                    </xsl:attribute>
                </smn:ATTR_NODE>
            </smn:ATTR_NODES>
        </smn:NODE_LINE>

        <xsl:call-template name="Variants">
            <xsl:with-param name="node" select="$node" />
            <xsl:with-param name="varMappingValue"
select="&apos;CLONE&apos;" />
        </xsl:call-template>

    </xsl:template>
```

b. To reference SVRs, variant conditions, variant options, option defaults, and rule checks, edit as follows:

Default **varMappingValue**:

```
<xsl:with-param name="varMappingValue"
select="&apos;CLONE&apos;" />
```

Change to:

```
<xsl:with-param name="varMappingValue"
select="&apos;REFER&apos;" />
```

Configure data map and structure map rules to improve system performance

In CAE Manager, analysts can execute the **Generate Node XML** command after selecting a BOM line in the **Product** or **Model** view. Teamcenter processes all items, item revisions, forms, and BOM line attributes for each BOM line in the input structure. If the structures are very large, it impacts the performance and memory usage of the system.

Simulation Process and Data Management provides a configuration XML file that the simulation administrator can edit to control which items, item revisions, forms, or BOM line attributes are processed. The administrator can edit the required attributes in the XML file in such a way that

Teamcenter processes only those needed attributes. This significantly improves the performance and memory usage of data map and structure map execution operations.

The **NodeXMLConfig.xml** sample file is located in the *TC_DATA* directory. However, the file used at runtime is managed in a **CAEStructureMap** dataset attached to the item revision indicated by the **CAE_datamap_files_location** preference.

The attributes of the **NodeXMLConfig.xml** sample file are as follows:

> Note:
>
> All attributes are mandatory except the **DOMAIN** attribute.

- **SM_CONFIG_ITEM ITEM_TYPE=”Item”**: The item type for which you need to configure attributes.

  If you do not specify the **ITEM_TYPE** attribute, then all attributes are used and the configuration is *not* applicable.

- **DOMAIN="CAE"**: The domain to which the configuration is applied.

  If no domain is specified, Teamcenter applies the configuration to all the domains.

- **FOCUS="IN"**: You can set it to:

  - **IN** to apply the configuration to the input structure.
  - **OUT** to apply the configuration to the output structure.
  - **“”** (no value): to apply the configuration to both the input and output structures.

- **STATUS="INACTIVE"**: You can set it to:

  - **INACTIVE** for Teamcenter to generate the node XML file for all the attributes. This means that all data map and structure map rules defined for any attributes are executed.
  - **ACTIVE** for Teamcenter to generate the node XML file for only the defined attributes. This means that data map and structure map rules for only configured attributes are executed.

    > Note:
    >
    > You must configure all attributes that you want to use as **ACTIVE**. If you do not configure them as active, analysts cannot use any data mapping rules.
    >
    > If product structures are very large, they impact the performance and memory usage of the system. Therefore, you must set only those attributes that are required for processing as **ACTIVE**.

- Define list of mapped attributes for output objects.

  While using the **CAE Attribute Compare** dialog box, analysts can compare some or all mapped attributes in a model structure to current values in the product structure if you set the **MAPPED** attribute to **true**.

The following are some examples. For more information, see the **NodeXMLConfig.xml** sample file in the *TC_DATA* directory.

```
<SM_CONFIG_ITEM ITEM_TYPE="Item" DOMAIN="CAE" FOCUS="IN" STATUS="ACTIVE">
    <smn:NODES xmlns:smn="http://www.ugs.com/Schemas/SMENode">
      <smn:NODE>
        <smn:NODE_LINE CLASS="Item" TYPE="Item">
          <smn:ATTR_NODES>
            <smn:ATTR_NODE INTERNAL_NAME="object_name" NAME="Name"/>
            <smn:ATTR_NODE INTERNAL_NAME="object_desc" NAME="Description"/>
            <smn:ATTR_NODE INTERNAL_NAME="item_id" NAME="ID"/>
            <smn:ATTR_NODE INTERNAL_NAME="object_type" NAME="Type"/>
            <smn:ATTR_NODE INTERNAL_NAME="project_ids" NAME="Project IDs"/>
          </smn:ATTR_NODES>
        </smn:NODE_LINE>
      </smn:NODE>
    </smn:NODES>
  </SM_CONFIG_ITEM>
</SM_NODEXML_CONFIGURATION>

<SM_CONFIG_ITEM ITEM_TYPE="Item" DOMAIN="CAE" FOCUS="IN" STATUS="ACTIVE">
    <smn:NODES xmlns:smn="http://www.ugs.com/Schemas/SMENode">
      <smn:NODE>
        <smn:NODE_LINE CLASS="Item" TYPE="Item">
          <smn:ATTR_NODES>
            <smn:ATTR_NODE INTERNAL_NAME="object_name" NAME="Name"/>
            <smn:ATTR_NODE INTERNAL_NAME="object_desc" NAME="Description"/>
            <smn:ATTR_NODE INTERNAL_NAME="item_id" NAME="ID"/>
            <smn:ATTR_NODE INTERNAL_NAME="object_type" NAME="Type"/>
            <smn:ATTR_NODE INTERNAL_NAME="project_ids" NAME="Project IDs"/>
          </smn:ATTR_NODES>
        </smn:NODE_LINE>
      </smn:NODE>
    </smn:NODES>
  </SM_CONFIG_ITEM>
</SM_NODEXML_CONFIGURATION>
```

Define data mapping rules to map datasets or map project information

You can create a **CAE 3D Model** structure by applying a set of data mapping rules to a product structure.

The procedure for defining data mapping rules to map datasets and map project information is similar. The simulation administrator first edits the **datamapping.xml** file to add data mapping rules. The analyst then loads the product structure and creates a model structure by choosing the **CAE 3D Model from DataMap** option.

1. In CAE Manager, click **Edit→Options→CAE→CAE Structure Map**.

2. Click the **Browse** button beside the **Data Map Files Location** box to open the **Search** dialog box.

3. Enter the item revision that is configured for the data map definition location and click the **Search** button.

4. Download the data map definition file (**datamapping.xml** file) from the dataset attached to the item revision.

5. Add additional rules to map datasets or map project information by editing the **datamapping.xml** file.

   Example 1 — Attach all datasets of **DirectModel** type with **Rendering** relationship to **ItemRevision** to be mapped as copied to the output model with the same relation.

   For this example, assume that the analyst loads a product item revision with a **DirectModel** dataset type with **Rendering** relationship, **DirectModel** dataset type with **Specifications** relationship, and **CAEMesh** dataset type with **Specifications** relationship in the **Product** view. When the analyst applies the following data mapping rule (example 1), the system creates a corresponding model structure in the **Model** view and only **DirectModel** dataset types with **Rendering** relationship are copied across to the model structure. Similarly, for each product item type in your business model, you can create data mapping rules and create corresponding CAE item types based on your site requirements.

```
<!-- All attach datasets of type DirectModel to
input item revisions -->
        <xsl:for-each select="$node/smn:NODES/smn:NODE/
   smn:NODE_LINE[@CLASS=&apos;Dataset&apos; and
   @TYPE=&apos;DirectModel&apos;]">

            <!-- Only relationship Rendering to input itemrevision
to be copied to
     output CAE 3D Model revisions with same relationship -->
            <xsl:if test="smn:NODE_LINKS/smn:NODE_LINK/
@RELATIONSHIP_TYPE=
     &apos;IMAN_Rendering&apos;">
              <xsl:call-template name="Dataset">
                <xsl:with-param name="operationtype">COPY</xsl:with-
param>
                <xsl:with-param name="outrelationshiptype"
select="smn:NODE_LINKS/
       smn:NODE_LINK/@RELATIONSHIP_TYPE"/>
```

```
                <xsl:with-param name="parenttype">CAEModelRevision</
xsl:with-param>
                <xsl:with-param name="datasettype" select="@TYPE"/>
              </xsl:call-template>
            </xsl:if>
```

Example 2 — Map project information and assign the default project ID.

For this example, assume that the analyst loads a product item revision in the **Product** view and assigns **P1** and **P2** projects from the **Project** application. When the analyst applies the following data mapping rule (example 2), the system automatically assigns **P1** and **P2** project objects to the item revision.

> Note:
>
> If a configuration file is used for data mapping, you must configure project IDs (**project_ids**) for appropriate types.

```
<!--Example: To map the project information and
assign default project id -->
            <smn:ATTR_NODE INTERNAL_NAME="project_ids">
                <xsl:attribute name="VALUE">
                    <xsl:value-of select="$currentnodeline/
smn:ATTR_NODES/
           smn:ATTR_NODE[@INTERNAL_NAME=&apos;project_ids&apos;]/
@VALUE"/>
                </xsl:attribute>
            <!--Example of default project id -->
                <xsl:attribute name="DEFAULT_VALUE"></xsl:attribute>
            </smn:ATTR_NODE>
```

6. Save the **datamapping.xml** file and upload it to the dataset attached to the item revision.

Best practices to optimize performance while using data mapping and reuse rules

Use preferences to optimize performance

- **CAE_persist_StructureMapLog**: Controls whether a log is attached as a dataset to the root item revision of the created structure in a data map or a structure map.

  The default value for this user preference is **0**, that is, no log is attached as a dataset to the root item revision.

  To improve performance, do not generate logs.

- **CAE_structuremap_log_content_scope**: Persists the summary level information in the dataset captured by the structure map log. The structure map log, identified by the **CAE_structuremap_log_dataset_name** preference, is captured in a dataset and attached to the root item revision of the output structure.

  The default value for this user preference is **Summary** to persist the summary level information in the dataset.

  You can improve performance by persisting only the summary level information and this minimizes the amount of time spent writing to a file.

- **CAE_datamap_object_creation_limit**: Sets the limit for objects that can be created in memory before committing the objects to the database.

  The default value for this site preference is **5000**.

  The goal is to increase this preference value to effectively reduce the number of round trips to the minimum, for example, setting a limit of 5000 objects would approximately be a 625 BOMLine structure (see below). If the target BOM structure is 1000 BOM lines, then increasing the limit to 10,000 reduces the number of round trips to 1.

  The preference was designed to restrict the structure map engine from consuming excessive memory. You must limit the balance because consuming large amounts of memory can force the hardware to swap memory more frequently, slowing down parallel processes.

  An item/revision consists of several objects, for example, typical item aggregation consists of the following (8–12 objects): item, master form, item revision, item revision master form, target relationship, source relationship, BOMView, BOMView Revision, specification relationship and the dataset. Additional custom forms or datasets can affect the number proportionally.

  Since the preference is also impacted by the number of attributes being mapped, you must reduce the number of objects. If you map a large number of attributes, it can make the objects unusually large in aggregation.

- **CAE_enable_AncestorMapping**: Enables ancestor mapping by adding the node XML interpretations for all BOMline ancestors of BOMlines to structure maps in CAE Manager.

  By default, ancestor mapping is not enabled. The default value for this site preference is **0**.

  This preference allows data mapping to pull attributes from parent objects into the hierarchy of the structure. However, that requires additional processing and can reduce performance. For best performance, you should disable ancestor mapping.

- **CAE_enable_Mark_up_to_date**: Enable or disable the **Mark-Up-To-Date** option for attachment changes to **CAE 3D Model** and **CAE 3D Analysis** item revisions in CAE Manager to improve the performance of data mapping rules against large structures.

In a complex product development environment, different analysts perform different tasks in the overall analysis. In such scenarios, it becomes critical to know when analysis data, possibly with multiple dependencies, is out of date. The analyst can then act on it and ensure that the analysis is built with the correct set of data to deliver accurate results. In CAE Manager, analysts can check for later revisions of item revisions attached to **CAE 3D Model** and **CAE 3D Analysis** item revisions. They can also check for changes to any attachments of item revisions.

The default value for this user preference is **OFF**.

You can improve performance by retaining this setting, that is, keep this preference set to **OFF**. This reduces the number of secondary objects that are created. However, note that analysts must subsequently mark the BOMlines as up-to-date.

Guidelines for setting attributes using the configuration file

You can edit the **NodeXMLConfig.xml** configuration XML file to control which items, item revisions, forms, or BOM line attributes are processed. You can edit the required attributes in the XML file in such a way that Teamcenter processes only those needed attributes. This significantly improves the performance and memory usage of data map and structure map execution operations.

While editing the **NodeXMLConfig.xml** file:

- Ensure that each item type you want to use for data mapping or structure mapping for both output and input is present and the state is set to **STATUS="ACTIVE"**. This means that data map and structure map rules are executed only for the configured attributes.
- Identify the attributes and NODELines you want to use for data mapping or to be compared against in structure map rules and set only those attributes that are required for processing as **ACTIVE**.

Guidelines for specifying data map rules

- You should identify the data map rules for the map item types or attributes exposed in the **NodeXMLConfig** file.
- You must map the datasets you identify as reference and not as copy. Datasets that are mapped as copy require extra processing and impact the performance of the data mapping operation.
- Mapping project IDs:
  - Minimize the number of projects you want to map. If all the projects associated with the product structure are not relevant to the **CAE 3D Model** structure, exclude them from the maps. This can be done using detailed data map rules.
  - Ensure that project IDs are valid for the user to access. Invalid project IDs can reduce the performance of data mapping.

  - Map the project IDs at the item revision level. Assigning project IDs at the item level requires the system to assign the project ID to the item as well as the item revision.

- For optimal performance, use the **CAE Target Occurrence** relation for components the analyst wants to evaluate at the attribute level. This is typically the lower subassembly levels.

  Set the context once at the root or at distinct subassembly levels. Avoid multiple overlapping contexts as they degrade performance significantly and cause unintended data mapping results with the introduction of multiple occurrences.

Guidelines for setting reuse rules

- The **CAE_Apply_Reuse_Rules_to_SubAssemblies** preference determines whether to apply reuse rules to subassemblies or just to component level product structures within a structure.

  By default, this site preference is set to **TRUE**. When set to **TRUE** all reuse rules are applied to every BOMLine in a BOM structure, thus reducing the performance.

  To improve performance, you must set this preference to **FALSE** to restrict executing reuse rules to the leaf nodes of a product BOM structure.

- Implement multiple subtypes for how **CAE 3D Model** objects are reused. Consider creating subtypes of the **CAE 3D Model** based on group ownership or analysis type. Reuse criteria is applied to all objects of the type returned by the rule scope definition. Therefore, restricting that list to the objects of a particular analysis or group can improve performance.

- The order of reuse rules is important. Rules are executed one at a time against the entire list of objects related to the target object. The first rule is applied to each object in that list before the second rule is applied. To improve performance, order the rules based on the frequency of execution.

Define data map and structure map preferences

Set preferences to control data maps and structure maps

Analysts can use CAE Manager to generate a CAE structure from an existing product structure or BOM. Typically, the CAE structure is similar to but not the same as the product structure. They can use structure maps to automate the creation of CAE structures.

The administrator can use the following preferences:

- **CAE_create_GeneratedByRelationship** to control whether a structure map relationship is maintained between the root item revision of the created structure and the defining structure map.

- **CAE_persist_StructureMapLog** to control whether a log is attached as a dataset to the root item revision of the created structure in a data map or a structure map.

- **CAE_enable_AncestorMapping** to enable ancestor mapping. This adds the node XML interpretations of all the BOMline ancestors of BOMlines to structure maps in the CAE Manager application.
- **CAE_ancestorMapping_level** to specify the maximum levels of ancestors to include in an XML node during ancestor mapping in the CAE Manager application.
- **CAE_structureMap_log_dataset_name** to specify the name of the dataset to be created while executing the data map and structure map to maintain the data map and structure map log file.
- **CAE_datamap_files_location** to specify the unique ID (UID) of the item revision used to manage the data map definition files. The administrator must specify this preference for analysts to use data map or structure map functions.

> Note:
>
> Only a user with DBA privileges can define structure map preferences.

Define data map location and other data map and structure map options

1. In CAE Manager, choose **Edit→Options→CAE→StructureMap**.
2. In the **Options** dialog box, hover over a field to find the preference name associated with it.

   For information about retrieving a list of preferences, see *Where can I get a list of preferences?* in *Teamcenter Preferences*.
3. (Optional) To specify the data map file location.

   a. In **Data Map Files Location** field, click **Browse**.

   b. In the **Search** dialog box, choose **Item Revision**.

   c. From the **Type** menu, choose any type of revision, and click **Search**.

   d. From the **Folders** list, select an item revision.

   e. In the **Selected Item Revision** list, click **Add** and then click **OK**.
4. Ensure that the **Create relationship between output structure and CAE StructureMap** check box is selected (default setting) to control whether a structure map relationship is maintained between the root item revision of the created structure and the defining structure map.
5. Specify structure map log options.

a. (Optional) Select the **Persist CAE StructureMap log** log check box to control whether a log is attached as a dataset to the root item revision of the created structure in a data map or a structure map.

The **CAE StructureMap Log Dataset Name** dataset is used to store the data map or structure map log and this log is persisted across the site.

b. To specify detailed logs for structure maps, select **Detailed**. The default option is **Summary**.

6. (Optional) Select the **Enable ancestor data mapping** check box to enable ancestor mapping.

You can select the appropriate value to specify the maximum levels of ancestors to include in an XML node during ancestor mapping.

This adds the node XML interpretations of all the **BOMLine** ancestors of **BOMLine**s to structure maps in the CAE Manager.

7. Specify the appropriate value in the **CAE StructureMap maximum objects in memory** box. The default value is **5000**.

8. To apply reuse rules to subassemblies, select **True**. This is the default option.

9. To disable mark up-to-date, select **OFF**. This is the default option.

10. To select a folder for the output objects, select the **Newstuff**, **None**, or **Other Folder** option.

Newstuff is the default option.

If you select **Other Folder**, click **Browse** to select another folder.

11. To enable automatic validation of structure map rules, select **True**. This is the default option.

12. To copy output objects as items or item revisions in the selected output folder (created in step 10), select **Item** or **Item Revision**.

If this option is set to:

- **Item**, the system pastes the output *items* to the configured output folder.
- **Item Revision**, the system pastes the output *item revisions* to the configured output folder.

Define data map rules to generate model structures containing CAE geometry

Define data map rules to generate model structures

You can define data mapping rules to help analysts automate the creation of CAE geometry. You define data mapping rules in the **datamapping.xml** file. You also configure the **NodeXMLConfig.xml** file to specify the required items, item revisions, or BOM line attributes required for processing the model output.

Simulation analysts can create a CAE model structure from an input product structure, using data mapping rules. The default rules are used for creating *primary objects*. These rules determine the creation and naming pattern of model items and item revisions from the product structure, the mapping of product items to model items, and the relations between the product item revisions and model item revisions.

In addition, you can optionally enable data mapping rules for *secondary objects*. These allow simulation analysts to create a CAE model structure containing primary objects from the default rules as well as secondary objects from the additional rules. The secondary objects contain CAE geometry with the appropriate relationships between the input product structure and the secondary object or between the primary output object and the secondary output object.

> Note:
>
> The rules for primary objects are available by default in the **datamapping.xml** file. However, the **xsl: call-template** attributes to create secondary objects are commented, and you must uncomment these lines to enable the creation of secondary objects.

Before customizing the data map for secondary objects, you must know the attributes and relationships you want to map. The Business Modeler IDE is a tool for configuring and extending the data model of your Teamcenter installation. You must turn on **Operation Descriptor**s for the attributes and relationships of any objects that you want to map before executing data map rules against these objects.

> Tip:
>
> By default, all the attributes and their relationships required for secondary objects in the **datamapping.xml** file are defined in the **CreateInput** operation on the **Operation Descriptor** tab for business objects. You need to define new attributes and their relationships in Business Modeler IDE only if you plan to customize the data mapping file by adding new attributes.
>
> Similarly, all the required items, item revisions, or BOM line attributes required for processing the model output for secondary objects are defined in the **NodeXMLConfig.xml** file. You need to edit this file only if you plan to add any new attributes.

Edit the data map file to include CAE 3D Geometry items as secondary objects

You have to edit the data map file to include **CAE 3D Geometry** items as secondary objects.

1. Open the *TC_DATA***datamapping.xml** file in an XML editor.

2. Search for the **SecondaryGeometry** template and uncomment the tags before and after the **xsl:call-template** attributes.

```
<!-- Example 1: Call template to create CAE 3D Geometry as a secondary
output
  object and a node with name geometry1_node_name -->
    <!--Begin SECONDARY_OBJECT_CREATION -->
    <!-- <xsl:call-template name="SecondaryGeometry">
      <xsl:with-param name="node" select="$node"/>
        </xsl:call-template> -->
    <!--End SECONDARY_OBJECT_CREATION -->
```

When an analyst creates a CAE model structure from the input product structure, the system checks each leaf node in the input structure and creates corresponding CAE geometry output items for each item in the input structure. Additionally, it copies input item names and appends the **CAE** string to the output item names for each corresponding item. Similarly, each input item description is copied to the corresponding output item and appended with the **CAE** string.

CAE geometry item attributes example:

```
<!--****************************************
********************     CAE 3D Geometry attributes
*************************
The input item attributes that need to populate the output's atrributes
go here.
*************************************************************************-
->
<smn:ATTR_NODES>
    <!-- Input's Name transferred to output's attribute of same name-->
    <smn:ATTR_NODE INTERNAL_NAME="object_name">
      <xsl:attribute name="VALUE">
        <xsl:value-of select="concat('CAE-',$currentItemnodeline/
smn:ATTR_NODES/
        smn:ATTR_NODE[@INTERNAL_NAME=&apos;object_name&apos;]/@VALUE)"/>
          </xsl:attribute>
    </smn:ATTR_NODE>
    <!-- Input's Description transferred to output's attribute of same
name-->
    <smn:ATTR_NODE INTERNAL_NAME="object_desc">
        <xsl:attribute name="VALUE">
            <xsl:value-of select="concat('CAE-',$currentItemnodeline/
              smn:ATTR_NODES/smn:ATTR_NODE[@INTERNAL_NAME=
                &apos;object_desc&apos;]/@VALUE)"/>
            <!--  Example 1 - To specify ancestor node level -->
            <!--  <xsl:for-each select="$node/smn:NODES/smn:PNODES/
smn:PNODE">
```

```
                    <xsl:if test="@LEVEL = &apos;2&apos;">
                    <xsl:value-of select="concat('CAE-',currentItemnodeline/
                      smn:ATTR_NODES/smn:ATTR_NODE[@INTERNAL_NAME=&apos;
                      object_desc&apos;]/@VALUE)"/>
                    </xsl:if>
                    </xsl:for-each> -->
                    </xsl:attribute>
        </smn:ATTR_NODE>
    </smn:ATTR_NODES>
```

After setting the geometry item attributes, the system calls the geometry revision template to copy the geometry item revision attributes and their description. Each input item revision name and input item revision description is copied to the output structure.

Geometry revision template example:

```
<!--Conditional logic to call the CAE 3D Geometry Revision template-->
    <xsl:if test="$node/smn:NODES/smn:NODE/smn:NODE_LINE[@CLASS=&apos;
      ItemRevision&apos; and @TYPE=&apos;ItemRevision&apos;]">
        <xsl:call-template name="SecondaryGeometryRevision">
            <xsl:with-param name="node" select="$node"/>
                </xsl:call-template>
            </xsl:if>
```

CAE geometry item revision attributes example:

```
<xsl:template name="SecondaryGeometryRevision">
    <xsl:param name="node"/>
        <xsl:variable name="currentIRnodeline" select="$node/smn:NODES/
          smn:NODE/smn:NODE_LINE
          [@CLASS=&apos;ItemRevision&apos; and
@TYPE=&apos;ItemRevision&apos;]">
        </xsl:variable>
        <smn:NODE_LINE CLASS="CAEGeometryRevision"
TYPE="CAEGeometryRevision"
          STATE="MAPPING">

            <!--
************************************************************
           *********     CAE 3D Geometry Revision attributes
*************

           The input Item revision attributes that need to populate
           the output's CAE 3D Geometry revision atrributes go here.
           ************************************************************-->
            <smn:ATTR_NODES>
                <!-- Input's Description transferred to output's attribute
of
```

```
                same name-->
              <smn:ATTR_NODE INTERNAL_NAME="object_desc">
                <xsl:attribute name="VALUE">
                <xsl:value-of select="$currentIRnodeline/smn:ATTR_NODES/
                  smn:ATTR_NODE[@INTERNAL_NAME=&apos;object_desc&apos;]/
                  @VALUE"/>
                  </xsl:attribute>
              </smn:ATTR_NODE>
              <!-- Input's Name transferred to output's attribute of
                same name-->
              <smn:ATTR_NODE INTERNAL_NAME="object_name">
                <xsl:attribute name="VALUE">
                <xsl:value-of select="concat('CAE-',$currentIRnodeline/
                  smn:ATTR_NODES/smn:ATTR_NODE[@INTERNAL_NAME=
                    &apos;object_name&apos;]/@VALUE)"/>
                  </xsl:attribute>
              </smn:ATTR_NODE>
          </smn:ATTR_NODES>
      </smn:NODE_LINE>

</xsl:template>
```

In addition, a **CAE Target** relation is created between the output object (**CAE 3D Model**, **CAE 3D Geometry**, or **CAE 3D Analysis** item revision) and the input item revision as specified in the **NODE_RELATIONS** sections.

CAE Target relation creation example:

```
<smn:NODE_RELATIONS>
    <!-- To create TC_CAE_Target relationship with ItemRevision -->
    <!-- Begin SECONDARY_OBJECT_RELATIONSHIP_CREATION -->
    <smn:NODE_RELATION RELATIONSHIP_TYPE="TC_CAE_Target">
    <smn:PRIMARY_NODE  CLASS="ItemRevision" TYPE="CAEGeometryRevision"
      FOCUS="OUT"/>
    <smn:SECONDARY_NODE CLASS="ItemRevision" TYPE="ItemRevision"
      FOCUS="IN"/>
    </smn:NODE_RELATION>
<!-- End SECONDARY_OBJECT_RELATIONSHIP_CREATION -->
</smn:NODE_RELATIONS>
```

Set the data map files location for the updated data mapping rule

You can define a data mapping rule to create only primary objects with relationships and another rule to create both primary and secondary objects with relationships for CAE geometry. To manage multiple rules, create different item revisions for each rule. Use the **Data Map Files Location** box in the **Options** dialog box to specify the appropriate item revision for the rule you want the analyst to run while creating the CAE model from the input product structure.

1. In CAE Manager, click **Edit→Options→CAE**.

2. Click **Search** in the **Data Map Files Location**, and specify you search parameters.

3. Click the **Execute the search** option and select the appropriate item revision containing the data map rule you want the analyst to run.

4. To specify the item revision, click **Apply**.

Specify comparison options for model and product structures

The **Inspector** view in CAE Manager allows analysts to compare a CAE model structure against a product structure.

You can specify comparison options for the **Inspector** view in the **Options** dialog box.

1. Click **Edit→Options→CAE→Inspector**.

2. In the **Options** dialog box, hover over a field to find the preference name associated with it.

   For information about retrieving a list of preferences, see *Where can I get a list of preferences?* in *Teamcenter Preferences*.

3. (Optional) Click the **General** tab. Use the **Inspector Execution Summary Dataset Name** to specify the name of the dataset in which the inspector summary value is saved as a named reference.

4. Define product structure criteria.

   a. Click the **Product Structure** tab.

   b. Select the appropriate criteria and default actions.

      The following table describes the product structure criteria.

Criteria	Description
Product with no Model	Use this to determine whether the product BOM line is linked to any **CAE 3D Model** BOM line in the model structure.
Missing Model Components	Use this to determine whether the number of BOM lines for a **CAE 3D Model** item revision is less than the number of BOM lines of the linked product item revision.

5. Define simulation structure criteria.

a. Click the **Model Structure** tab.

b. Select the appropriate criteria and default actions.

The following table describes the simulation structure criteria.

Criteria	Description
Model With Different Product Revision Target	Use this to determine if the **CAE 3D Model** BOM line is linked to a different revision of the product BOM line.
Model With No Product	Use this to determine if the **CAE 3D Model** BOM line is not linked to any product BOM line in the product structure.
Extra Model Components	Use this to determine if the number of BOM lines for a **CAE 3D Model** item revision is more than the number of BOM lines of the linked product item revision.

6. Define root node criteria.

a. Click the **Root Node** tab.

b. From the **Default Root BOMLine Action** menu, choose:

- **Do Nothing** to take no action.
- **Use Revised Model** to create a new revision of the model structure.
- **Use New Model** to use a new model to create a new item.

c. From the **Default New Model Action Item Type** menu, select an appropriate item type such as **CAE 3D Model**.

Configure attribute values for comparing structures

Prerequisites for configuring CAE BOM comparison

- Set up the **MEMaxOpenViewsSameType** preference.

  You must set up this preference for configuring CAE BOM comparison. This preference defines the maximum number of **Product** or **Model** views that an analyst can open in CAE Manager. The default value is 10.

- Import the new **CAE BOM Comparison** report to Teamcenter.

Note:

Perform this only for the 10.1.2 release and patch upgrades later.

1. Create a stage directory, for example, **D:\workdir**.

2. Create a **TC_10_1_02_CUS_RPT_CAE_BOM_CMP** folder in the stage directory, for example, **D:\workdir\TC_10_1_02_CUS_RPT_CAE_BOM_CMP**.

3. Create a **Resources** subfolder in the **TC_10_1_02_CUS_RPT_CAE_BOM_CMP** folder, for example, **D:\workdir\TC_10_1_02_CUS_RPT_CAE_BOM_CMP\Resources**.

4. Copy the *TC_DATA***\crf\TC_10_1_02_CUS_RPT_CAE_BOM_CMP.xml** file to the **TC_10_1_02_CUS_RPT_CAE_BOM_CMP** folder.

5. Copy the *TC_DATA***\crf\Resources\cae_bom_compare_rpt_excel.xsl** file to the **Resources** subfolder.

6. Copy the *TC_DATA***\crf\Resources\cae_bom_compare_rpt_excel_OOXML.xsl** file to the **Resources** subfolder.

7. Run the following command from the Teamcenter command prompt:

   import_export_reports -u=*user_id* **-p=***password* **-g=***dba* **-import -overwrite -stageDir=***D:\workdir* **-reportFile=***TC_DATA***\crf\CrfReport.xml -reportId=***TC_10_1_02_CUS_RPT_CAE_BOM_CMP*

8. Import new closure rules for the **CAE BOM Comparison** report to Teamcenter.

   Note:

   Perform this only for the 10.1.2 release and patch upgrades later.

   Run the following commands from the Teamcenter command prompt:

   tcxml_import -u=*user_id* **-p=***password* **-g=***dba* **-scope_rules -scope_rules_mode=***overwrite* **-file=***TC_DATA***\caeBomCompareScopeRules.xml**

   install_callback -u=*user_id* **-p=***password* **-g=***dba* **-mode=***install* **-type=***Accountability-CheckCriteria* **-library=***libcae* **-function=***CAE_BOM_Comparison_find_no_criteria_instance* **-name=***CAE_BOM_Cmp_Find_No_Criteria*

Configure attribute values for structure comparison

Analysts can use Simulation Process and Data Management to quickly create one or more simulation variants from an existing simulation variant, make modifications to the structure, and view the differences between simulation variants. Typically, these simulation variants are complex and it is not possible to compare more than two simulation variants at a time and understand their differences.

They can use CAE Manager to view two different simulation variants side-by-side for a comparison, get a visual understanding of their differences as they are color coded, and can further investigate differences in attributes such as thickness, associated files or load cases, and file content.

You must configure partial match properties by selecting an item revision type and the properties for this item revision type that the analyst can use for comparing structures.

1. Choose **Edit→Options→CAE BOM Comparison**.

2. Select **User**, **Group**, or **Site** as appropriate.

3. Select an item revision type, and select the properties that you want analysts to use for comparing structures.

Configure different types of analysis or CAE packages using standard templates

Why configure analysis packages or CAE packages?

Analyst groups in companies typically perform multiple types of analyses based on a product definition or a set of requirements and validate them, for example, strength analysis, durability analysis, fluid analysis, or crash analysis. They also perform analyses multiple times, for example, based on different load cases or different input geometry variations. The processes and tools used and the input and output may vary across analyses.

To create an analysis package for a specific type of analysis, the analysts must know which items, datasets, and relationships to create and then create multiple CAE item revisions one at a time and establish relationships between different item revisions. The simulation administrator or a user with DBA privileges creates site-level package definitions, the group administrator creates group-level package definitions, and individual users create user-level package definitions based on a common template and save them to the database to make this process easier and more efficient.

Analysts can then use the package definitions created by the simulation administrator or the group administrator to create their own CAE item revisions. They can use these packages to define input parts (item revisions), output items, the relationships between the output item revision to their input parts, and the output datasets within one or more output item revisions.

At a site, there can be different categories of analysis packages:

- Site-level package definitions.

  These are standardized and approved package definitions created by the simulation administrator or a user with DBA privileges and used by multiple analysts across the enterprise to create a set of CAE item revisions for a specific analysis.

- Group-level package definitions.

  These are standardized and approved package definitions created by the group administrator and used by multiple analysts across the enterprise to create a set of CAE item revisions for a specific analysis. Access to these are limited to members of that group.

- User-level package definitions.

  These are created by an analyst and used only by this analyst.

Package definitions are stored in a CAE configuration object. The simulation administrator can use the **cae_migrate_configurations** utility to move packages definitions from one database to another. To view the command line help for this utility, type **cae_migrate_configurations -h** on the Teamcenter command prompt.

The simulation administrator can configure CAE package definitions to allow analysts to:

- Create BOM view revision (BVR) relations between various output and input objects in CAE packages.
- Refer or copy existing datasets as output objects.

> Note:
>
> A user with DBA privileges can run the **cae_convert_package_configuration** utility to convert CAE package configurations prior to the 11.5 release to a new CAE configuration object. To view the command line help for this utility, type **cae_convert_package_configuration -h** on the Teamcenter command prompt.

Specify a folder for output objects

1. In CAE Manager, click **Edit→Options→CAE→CAE Packages**.
2. In the **Options** dialog box, hover over a field to find the preference name associated with it.

   For information about retrieving a list of preferences, see *Where can I get a list of preferences?* in *Teamcenter Preferences*.

3. To specify a folder for output objects, select **Directly in location folder** or **In a subfolder under location folder**.

4. To enable the creation of logs after simulation analysts create CAE packages from predefined definitions, select the **Persist CAE Package Log** option.

   By default, logs are disabled.

5. To copy output objects as items or item revisions in the selected output folder (created in the earlier step), select **Item** or **Item Revision**.

   If this is set to:

   - **Item**, the system pastes the output *items* to the configured output folder.
   - **Item Revision**, the system pastes the output *item revisions* to the configured output folder.

6. Click **OK** to save and exit the **Options** dialog box.

Override the group administrator restriction to create configuration objects

Only a group administrator can create configuration objects at the group level. To avoid this restriction, you can select the **Bypass Group Administrator** option.

1. In CAE Manager, click **Edit→Options→CAE→General**.

2. In the **Options** dialog box, hover over a field to find the preference name associated with it.

   For information about retrieving a list of preferences, see *Where can I get a list of preferences?* in *Teamcenter Preferences*.

3. Click the **CAE Configuration** tab.

4. Select the **Bypass Group Administrator** check box.

   By default, it is set to **false**.

Configure CAE packages

1. In CAE Manager, click the **CAE Configuration** toolbar menu, and choose **CAE Package Configuration**.

2. Select the **SITE**, **GROUP**, or **USER** option, and click the **Create CAE Package** icon.

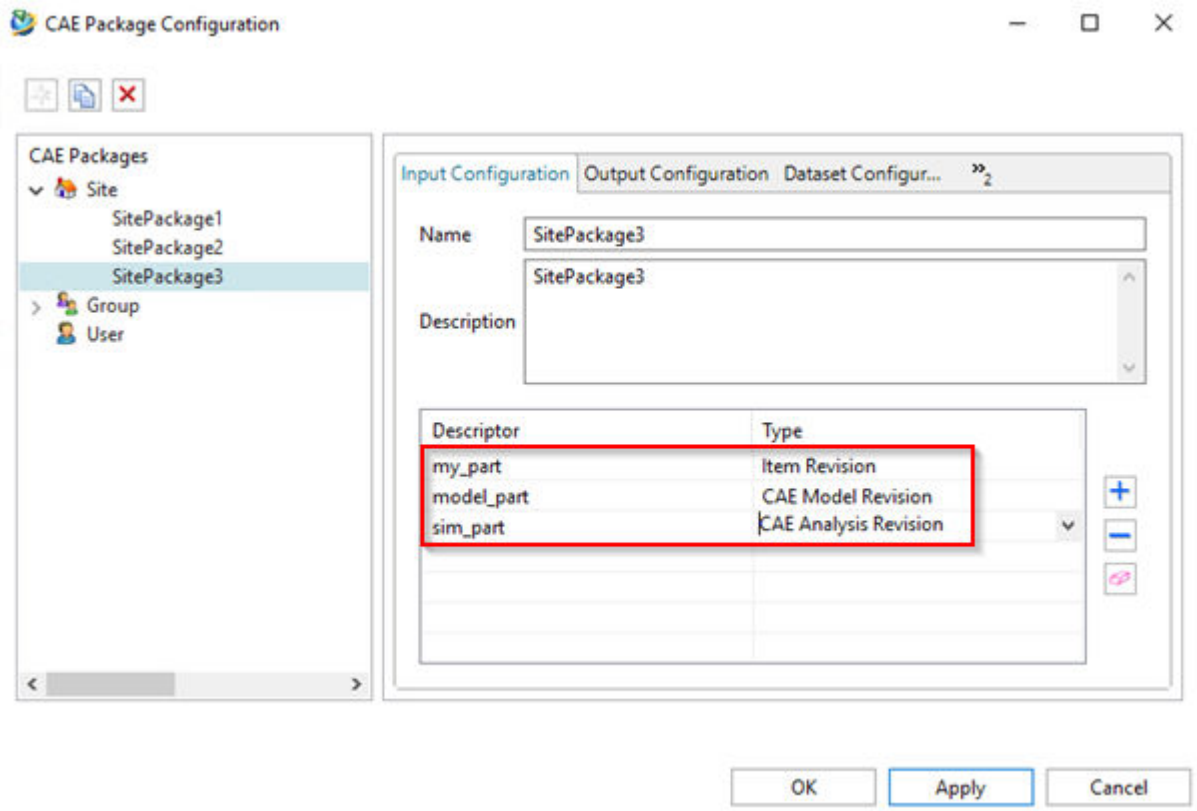
3. (Optional) In the **Create CAE Package** dialog box, to assign an ID, click **Assign** in the **ID** box.

4. (Optional) To assign a revision number, click **Assign** in the **Revision** box.

5. (Mandatory) To specify a derivative rule name, type a unique name in the **Name** box.

6. (Optional) Type a description in the **Description** box.

7. Configure input parameters.



   a. Click the **Input Configuration** tab.

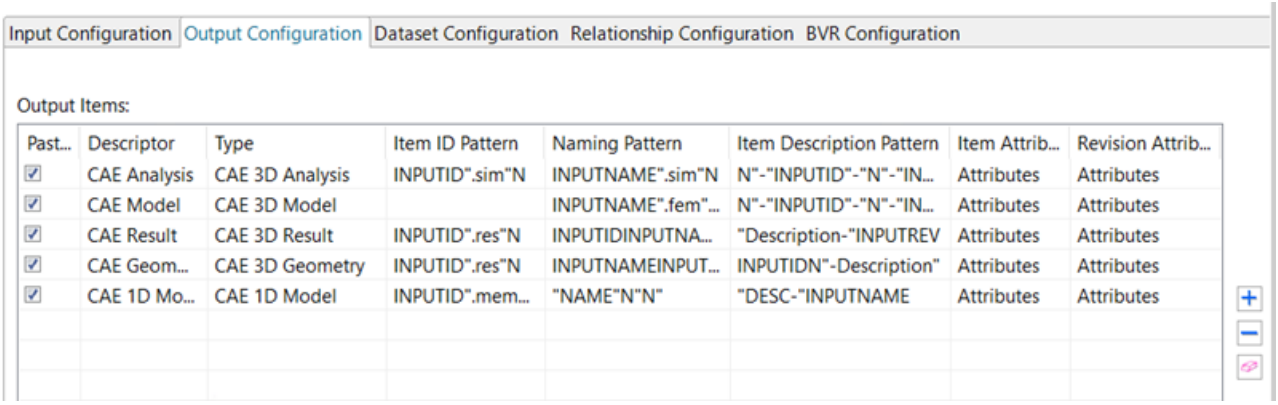
   b. (Optional) Enter a description for the CAE package in the **Description** box.

   c. Click the **Add** button to add parameters.

   d. In the **Descriptor** column, choose the default value or overwrite it with a unique value, and from the **Type** column, select an appropriate item revision.

   e. Click the **Apply** button.

8. Configure output parameters.



You can configure a CAE package containing a standalone output item (with or without datasets) and multiple output items (with or without datasets) where one or more items are connected or not connected to other items.

a. Click the **Output Configuration** tab, and click the **Add** button to add parameters.

b. (Optional) Clear the check boxes in the **Paste in Output Folder** column if you do not want to paste the output items and item revisions to the default output folder.

> Note:
>
> For all migrated packages from previous releases, the check boxes remain selected, that is, the system pastes the output items and item revisions to the default output folder.

c. In the **Descriptor** column, choose the default value or overwrite it with a unique value, and from the **Type** column, select an appropriate option.

d. Specify appropriate values in the **Item ID Pattern**, **Naming Pattern**, and **Item Description Pattern** columns.

- The item ID pattern must have the **N** keyword to generate an output item ID with a unique value and with the next available number for the **N** keyword.
- If the output item is not associated to any input item by any relation, then the configured pattern keywords are replaced with an empty value.
- If the output item ID pattern is not configured but **N** is configured on the other properties, then the **N** keyword is not replaced with any counter value and is kept unchanged.

You can use these keywords individually or as a combination. When you specify a combination, you must not repeat the keywords. You can use the following keywords as a combination by including text in quotes (see examples that follow).

- The **INPUTNAME** pattern keyword is replaced with input item's name.
- The **INPUTID** pattern keyword is replaced with input item's ID.

- The **INPUTREV** pattern keyword is replaced with the input item's revision value.
- The **N** pattern keyword represents the next available number for the output item ID. It is calculated at runtime when it is configured with the item ID pattern for the output item. If the **N** pattern keyword is configured across the other properties for the output item or for the item revision, the value is not recalculated. The value is repeated from the first usage, which is calculated from the output item ID.

Example 1:

Item ID pattern configuration for CAE package	After the package is executed with the input item ID as 000387	After the package is executed the second time with the same input item ID	After the package is executed with a different input item ID as 000341
CAE Analysis is configured as **INPUTID".sim"N**	Item ID for **CAE Analysis** is **000387.sim1**	Item ID for **CAE Analysis** is **000387.sim2**	Item ID for **CAE Analysis** is **000341.sim1**
CAE Model is configured as **INPUTID".fem"N**	Item ID for **CAE Model** item ID is **000387.fem1**	Item ID for **CAE Model** is **000387.fem2**	Item ID for **CAE Model** is **000341.fem1**
CAE Result is configured as **INPUTID".res"N**	Item ID for **CAE Result** is **000387.res1**	Item ID for **CAE Result** is **000387.res2**	Item ID for **CAE Result** is **000341.res1**
CAE Geometry is configured as **INPUTID".i"N**	Item ID for **CAE Geometry** is **000387.i1**	Item ID for **CAE Geometry** is **000387.i2**	Item ID for **CAE Geometry** is **000341.i1**

Example 2:

CAE package configuration for CAE Analysis	After the package is executed with input ID as 000387 and input item name as NX_Numbers for CAE Analysis
The item ID pattern is configured as **INPUTID".sim"N**.	The item ID is **000387.sim1**
The name pattern is configured as **INPUTNAME".sim"N**.	The item name is **NX_Numers.sim1**
The item description pattern is configured as **"Test Package for .sim - "N**.	The item description is **Test Package for .sim - 1**

Example 3:

CAE package configuration for CAE Analysis	After the package is executed with input ID as 000387 and input item name as NX_Numbers for CAE Analysis
The item ID is not configured, that is, the item ID pattern is empty.	The item ID is automatically generated by the system.
The item name pattern is configured as **INPUTNAME".sim"N**.	The item name is **NX_Numers.simN**
The item description pattern is configured as **"Test Package for .sim - "N**.	The item name description is **Test Package for .sim - N**

Example 4:

Item ID pattern configuration for CAE Geometry	Existing numbers in the database for the pattern N	Missing numbers in the database for the pattern N	Allocated numbering for the pattern N after the package is executed
INPUTID".i"N	1, 2, 3, 4, 5, 7, 9	6, 8	6, 8, 10, 11, and so on

e. In the **Item Attributes** column, click the icon to the right of the column to open the **Define Attribute** dialog box. Click the **Add** button to add parameters. In the **Attribute Name** column, select an attribute name (for example, **Name**) and enter the value in the **Value** column.

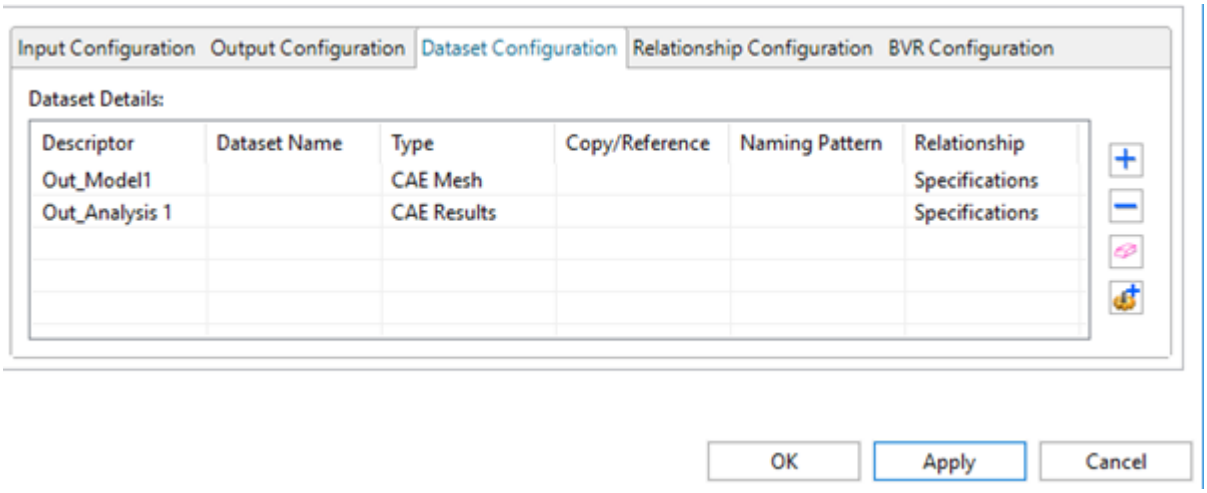
   (Optional) You can use keywords described in **step d**, for example, specify **Attribute Name** as **Project ID** and **Value** as **N"Proj001"N**.

f. In the **Revision Attributes** column, click the icon to the right of the column to open the **Define Attribute** dialog box. Click the **Add** button to add parameters. In the **Attribute Name** column, select an attribute name, for example, **Revision** and enter the value in the **Value** column.

   (Optional) You can use keywords described in **step d**, for example, specify **Attribute Name** as **Solver Name** and **Value** as **INPUTID"Name"N**.

g. Click the **Apply** button.

9. Define dataset parameters.



a. Click the **Dataset Configuration** tab, and click the **Add** button to add parameters.

b. In the **Descriptor** column, choose a default value.

   This list contains all output items defined in the **Output Configuration** pane.

c. To add a dataset, click the **Dataset Name** column. Select a dataset from the **Home** view and click the **Add Dataset** option. Alternatively, click the **Search** icon in the column and search for a dataset.

d. In the **Type** column, select a dataset type, for example, **CAEMesh**.

e. In the **Copy/Reference** column, select **Copy** or **Reference** to allow analysts to copy or reference existing datasets as part of output objects, respectively.

f. In the **Naming Pattern** column, enter a naming pattern.

> Tip:
>
> You can create multiple dataset naming patterns.

   The naming pattern supports one of these or any combination of these:

   - Constant string enclosed in double quotes
   - **ITEMREVID** keyword

     The item revision ID applied here is the item revision ID of the object to which the dataset is going to be attached.

g. From the **Relationship** column, select the relevant option.

h. Click the **Apply** button.

Example:

Descriptor	Dataset Name	Type	Copy/ Reference	Naming pattern	Relationship
model_output	*dataset_-name1*	**CAEMesh**	**Copy**	**"CAE-"-ITEMREVID**	**IMAN_based_on** or **NX Simulations**
sim_output	*dataset_-name2*	**CAE-AnalysisDS**	**Reference**	**"CAE-"-ITEMREVID**	**CAE Target**

10. (Optional) Define relationship parameters.

You can specify Generic Relationship Management (GRM) rules to apply constraints on the relationship between two business objects. A GRM rule applies constraints on the relationship between two business objects. When you create a GRM rule, you select the primary and secondary business objects for the relationship, the relationship they have to one another, and the constraints to be applied.

a. Click the **Relationship Configuration** tab, and click the **Add** button to add parameters.

b. From the **Primary Object**, **Relationship** and **Secondary Object** columns, select appropriate options.

All the descriptors you created are available for selection in the **Primary Object** and **Secondary Object** columns.

The column provides a list of values and displays the available relationships for the primary object.

Example:

Primary object	Relationship	Secondary object
model_output	**CAE Target**	*my_part* (**ItemRevision** type)
sim_output	**CAE Defining**	*model_output*

> Note:
>
> You can have relationships between output and input, output and output, or input and output. However, you cannot have a relationship between input and input or the same descriptor as primary object and secondary object.

c. Click the **Apply** button to apply your data.

11. (Optional) Configure BOM view revisions.

    A BVR is a workspace object that stores the single-level assembly structure of an item revision. When you add a component to an assembly, you create an occurrence of that item or item revision in the assembly, which is stored on the BVR. This occurrence is displayed as a BOM line. A BVR is a single-level structure that contains occurrences of its immediate children. A multilevel structure is constructed from several single-line BVRs. Any modification to the product structure (including changing any of the occurrence attributes or adding a substitute) changes the BVR of the parent assembly.

    a. Click the **BVR Configuration** tab.

    b. In the **Parent Descriptor Table** area, click the **Add** button to add parent descriptors.

    c. From the **Parent Descriptor** and **Relation** columns, select appropriate options.

       The **Parent Descriptor** list contains all input and output descriptors you have defined in the **Input Configuration** and **Output Configuration** panes.

    d. In the **Child Descriptor Table** area, click the **Add** button to add child descriptors.

12. To create a duplicate of an existing package configuration, click **Clone CAE Package** and specify a name for the duplicate.

13. Click **OK** to save and exit the **CAE Package Configuration** dialog box.

Modify style sheets for CAE packages

Analysts can open CAE packages from the **Create CAE Packages** wizard by choosing **File→New→CAE Package**. You can modify style sheets to add a link for analysts to open this wizard from the **Summary** pane of item revisions for frequently used CAE packages.

1. In My Teamcenter, type **BaseItemRevSummary** in the search menu at the top of the navigation pane, select **Dataset Name**, and click **Perform Search**.

   > Tip:
   >
   > The **ItemRevision.SUMMARYRENDERING** preference controls the dataset used for rendering an item revision in the graphical user interface (GUI). You can use another dataset by changing the value of this preference to the name of that dataset.

2. In the **Viewer** pane, add the following command action key and parameter value in the first section titled **tc_xrt_actions**:

   Example:

```
<section titleKey="tc_xrt_actions" commandLayout="vertical">
  <command actionKey = "copyAction" commandId = "com.teamcenter.rac.copy"/>
  <command actionKey = "reviseAction" commandId =
"com.teamcenter.rac.revise"/>
  <command actionKey = "saveAsAction" commandId =
"org.eclipse.ui.file.saveAs"/>
  <command actionKey = "newProcessAction" titleKey = "tc_xrt_newProc"
      commandId = "com.teamcenter.rac.newProcess"/>

<! -- Add a link to open Create CAE Packages wizard -->
   <command actionKey="openCAEPackageAction"
 commandId="com.teamcenter.rac.tcsim.commands.opencaepackage"
 title="Package_Name">
  <parameter value="Package_Name"
   name="com.teamcenter.rac.tcsim.OpenCAEPackage"/>
 </command>
</section>
```

3. Replace *Package_Name* with the name of the package for which you want to modify style sheets.

4. Click **Apply**.

Configure rules to derive structures

Define derivative rules and variant configuration rules

You (as a simulation administrator) can create derivative rules and variant configuration rules to allow analysts to derive one or more structures from an existing structure.

During the virtual validation of a vehicle, simulation analysts may create hundreds of different deck structures or simulation variants to understand the effects of different materials or to optimize the weight of the vehicle. This is done by varying the load cases, materials, thickness, meshes, or geometry and then generating results to validate the vehicle against these different load cases. Many variants are created for this purpose, with each variant representing a load case.

Analysts can use predefined derivative rules and variant configuration rules to quickly derive one or more simulation variants from an existing one. They can then make modifications to the structure by varying load cases, meshes, or materials to evaluate different options and view the complete traceability between the different variants.

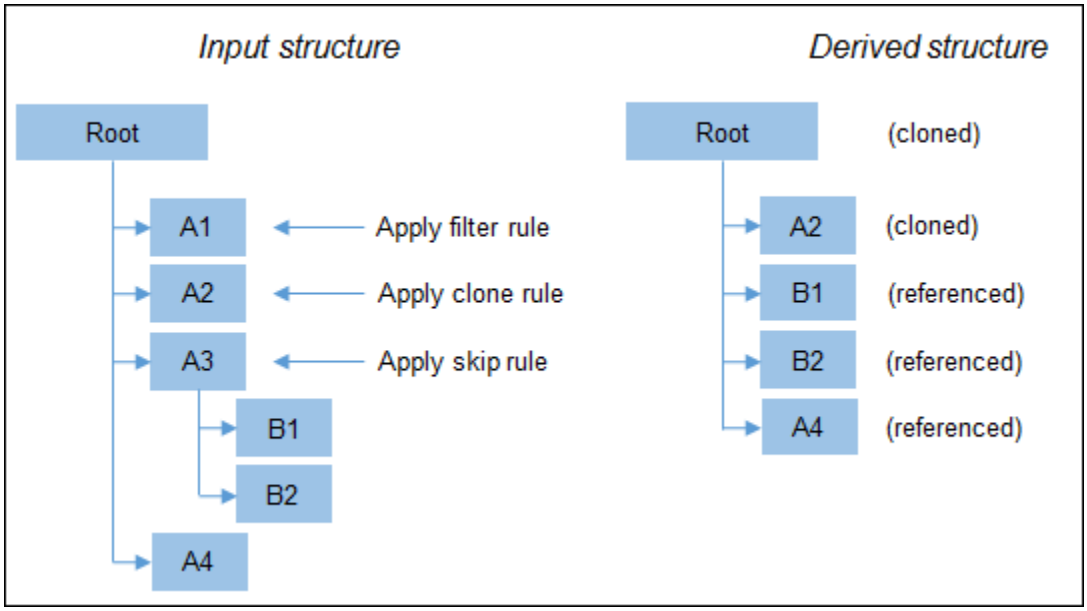
> Note:
>
> Derivative rules and variant configuration rules are defined at the site level by a user with DBA privileges, at the group level by the group administrator, and at the user level by the analyst.

Derivative rules

A derivative rule contains filter, skip, and clone rules to indicate which components of the source CAE structure are used to create a derived CAE structure.

Each rule may contain a conditional clause limiting the rule to a specific item or item revision types. The conditional clauses may contain expressions based on items, item revisions, or form attribute values. You can define any number of conditional clauses for each derivative rule. Each conditional clause is evaluated against each component in the source CAE structure. If components:

- Satisfy the conditional clauses, the system *filters*, *skips*, or *clones* the components in the derived CAE structure.
- Do not satisfy the conditional clauses, the system directly *references* the components, along with their children (if any), in the derived CAE structure.



Variant configuration rules

A variant configuration rule contains variant options and values to indicate which components of the source structure are used to create a derived structure.

Consider the source structure as an unconfigured product structure (150% BOM) with a V8 or V6 engine, an automatic or a manual gearbox, and 2 door or 4 door options. You can define a variant configuration rule to allow analysts to derive *two* specific structures using the same rule: one for the V8 engine, automatic gearbox, and 4 door combination and another for the V6 engine, manual gearbox, and 2 door combination.

You can specify the configuration details for the above example as follows:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<DERIVATIVE_VARIANT_CONFIG>

<CONFIGURATION>
<DESCRIPTION>V8, automatic, and 4 door vehicle</DESCRIPTION>
<VARIANT_OPTION NAME="Engine" VALUE="V8"/>
<VARIANT_OPTION NAME="Gearbox" VALUE="Automatic"/>
<VARIANT_OPTION NAME="Door" VALUE="4"/>
</CONFIGURATION>

<CONFIGURATION>
<DESCRIPTION>V6, manual, and 2 door vehicle</DESCRIPTION>
<VARIANT_OPTION NAME="Engine" VALUE="V6"/>
<VARIANT_OPTION NAME="Gearbox" VALUE="Manual"/>
<VARIANT_OPTION NAME="Door" VALUE="2"/>
</CONFIGURATION>

</DERIVATIVE_VARIANT_CONFIG>
```

Each variant option in the configuration details contains a value limiting the rule to derive a structure containing only those values. The analyst can override these values or add additional values while deriving a structure.

Specify a folder location and naming pattern for derived structures

1. In CAE Manager, click **Edit→Options→CAE→Derive**.

2. In the **Options** dialog box, hover over a field to find the preference name associated with it.

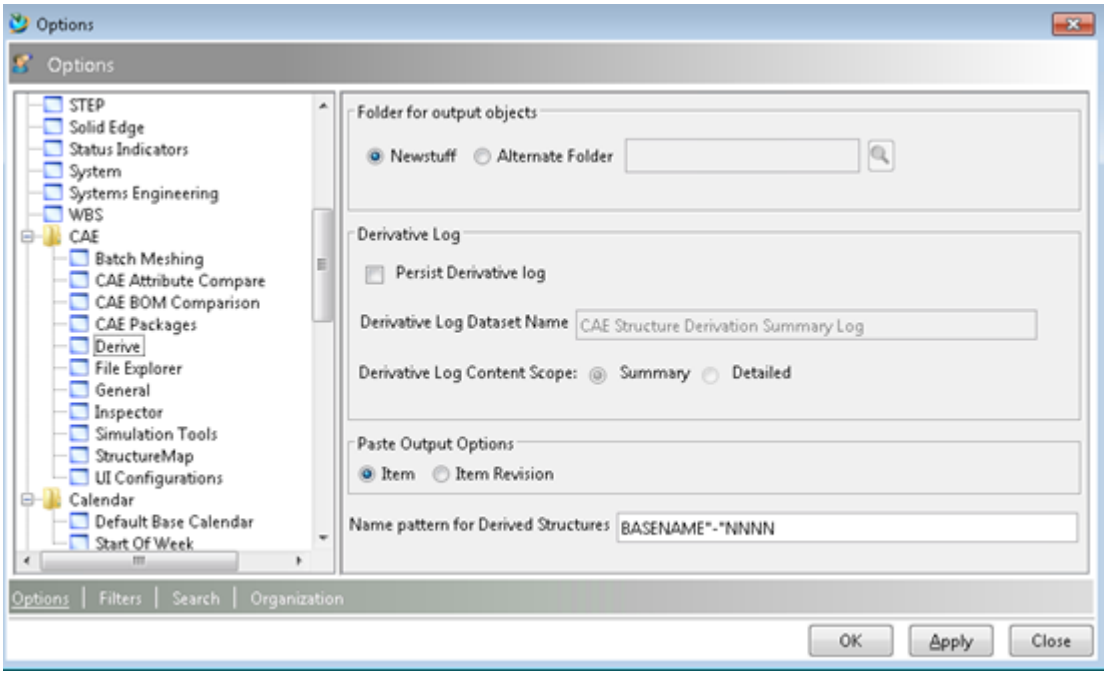
   For information about retrieving a list of preferences, see *Where can I get a list of preferences?* in *Teamcenter Preferences*.

3. (Optional) To specify a different folder for storing the derived structures, choose **Alternate Folder** and click **Search** to search and select an alternate folder.

4. To create a derivative log, select the **Persist Derivative Log** check box.

5. Specify the derivative log content scope by selecting the **Summary** or **Detailed** option.

   The option determines whether the derivation log is a detailed log or a summary log.



6. To copy output objects as items or item revisions in the selected output folder (created in the earlier step), select **Item** or **Item Revision**.

   The **CAE_derivative_paste_output_options** preference determines the configured value. If this preference is set to:

   - **Item**, the system pastes the output *items* to the configured output folder.
   - **Item Revision**, the system pastes the output *item revisions* to the configured output folder.

7. Specify a naming pattern for derived structures.

   You can specify a name pattern as a combination of fixed strings in double quotes and keywords. The valid keywords are **BASENAME**, which corresponds to the base name specified by the simulation analyst in the **Derive CAE Structure** dialog box before the execution, and **N** for digit.

   The pattern you specify must contain *both* keywords only once. You can optionally specify fixed strings and they can appear more than once in double quotes.

   Examples:

   BASENAME"-Deck"NNNN

   NNN"-"BASENAME"-Deck"

   NNNN"-"BASENAME

The default value is **BASENAME"-"NNNN**.

Consider the following while specifying a naming pattern:

- No blank values or blank spaces in the pattern string that are *not* within quotes.
- No leading or trailing white spaces.

Override the group administrator restriction to create configuration objects

Only a group administrator can create configuration objects at the group level. To avoid this restriction, you can select the **Bypass Group Administrator** option.

1. In CAE Manager, click **Edit→Options→CAE→General**.
2. In the **Options** dialog box, hover over a field to find the preference name associated with it.

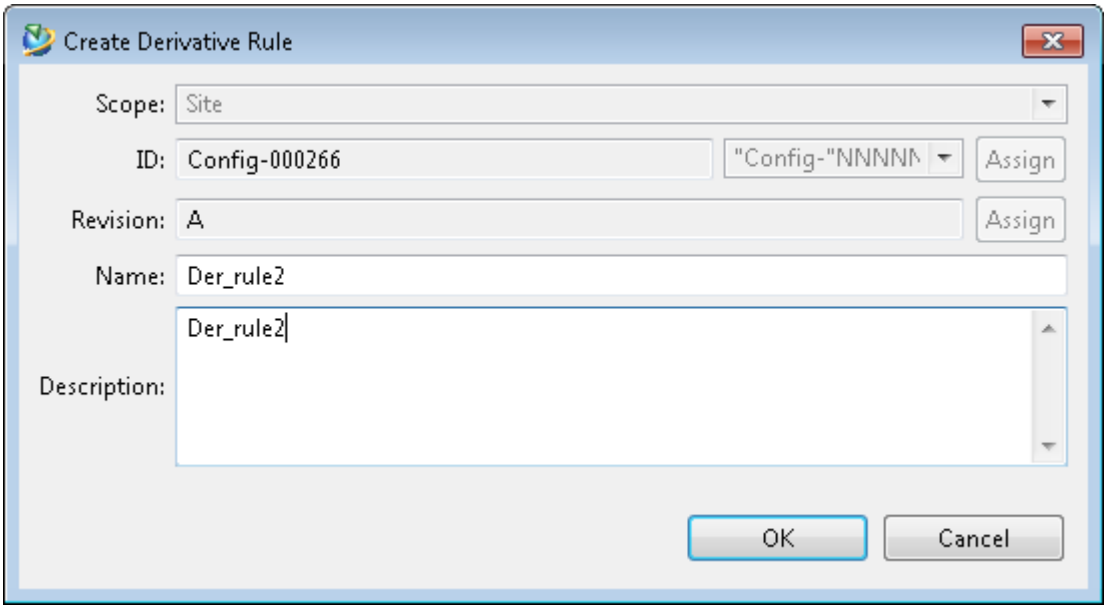
   For information about retrieving a list of preferences, see *Where can I get a list of preferences?* in *Teamcenter Preferences*.
3. Click the **CAE Configuration** tab.
4. Select the **Bypass Group Administrator** check box.

   By default, it is set to **false**.

Configure the derivative rule

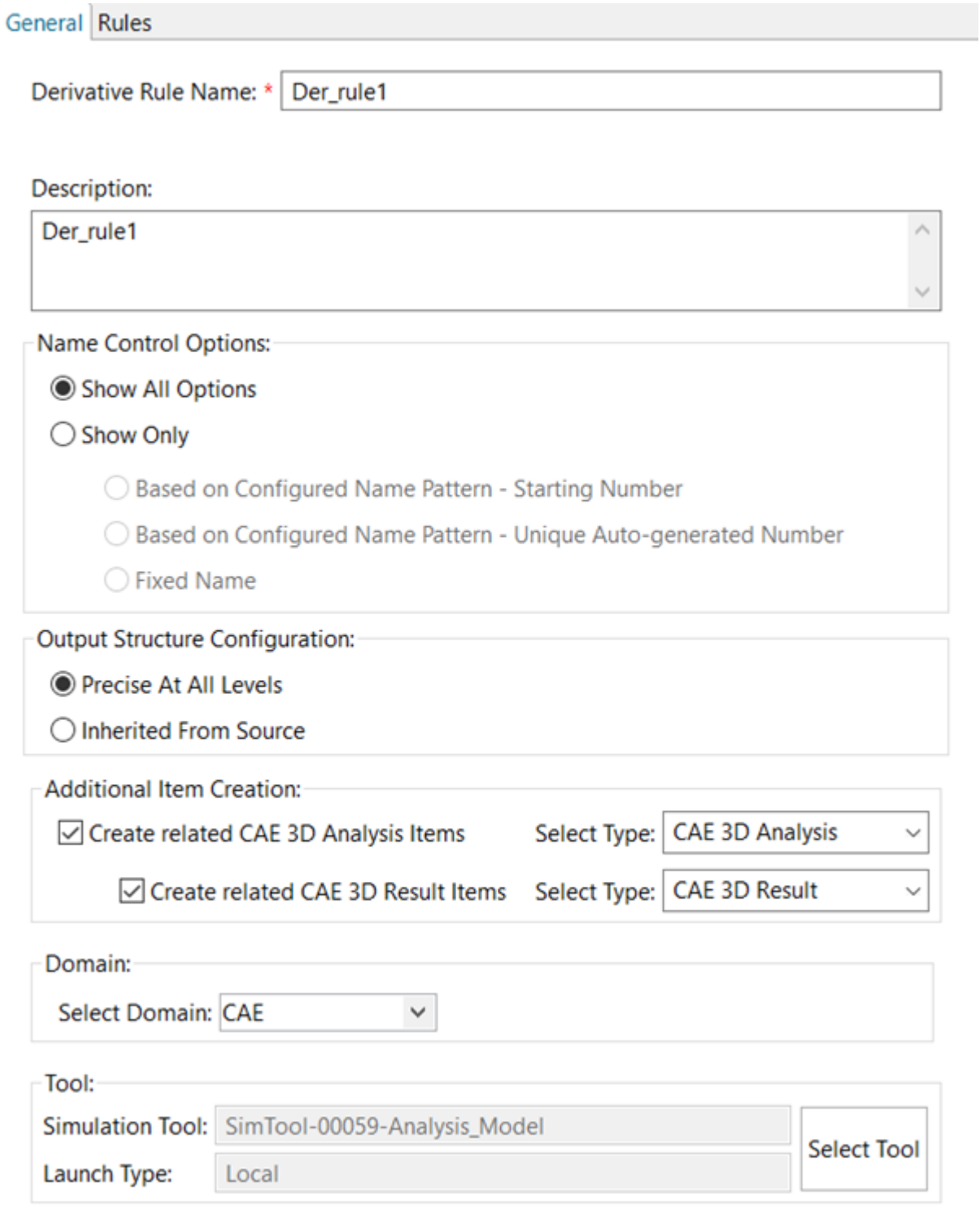
Specify a name and define the output structure configuration

1. In CAE Manager, click the **CAE Configuration** toolbar menu, and choose **Derivative Rules Configuration**.
2. Select the **Site**, **Group**, or **User** scope option.
3. In the **Derivative Rules Configuration** view, click the **Create New Derivative Rule** view bar icon.
4. To assign an ID, click **Assign** in the **ID** box of the **Create Derivative Rule** dialog box.
5. To assign a revision number, click **Assign** in the **Revision** box.
6. To specify a derivative rule name, type a unique name in the **Name** box.



7. (Optional) Type a description in the **Description** box.

8. Select the **General** tab.



9. For **Name Control Options**, you can select **Show All Options** or **Show Only**.

   - **Show All Options** shows all available options.

This is the default option if you have migrated derivative rules from a previous version of Teamcenter and imported them to a new environment by running the **cae_migrate_configurations** utility.

- **Based on Configured Name Pattern - Starting Number** allows analysts to generate derived structures using a starting number or override it while deriving structures.
- **Based on Configured Name Pattern - Unique Auto-generated Number** allows analysts to generate derived structures by using a unique and automatically generated number or override it while deriving structures.

  For example, if an analyst specifies the **BASENAME** as **SV01** and selects the unique autogenerated option, the system generates the next unique number, **0001**, and computes the name based on the specified base name, for example, **SV01-Deck0001**.

- **Based on Configured Name Pattern - Fixed Name** allows analysts to generate derived structures using a fixed name or override it while deriving structures.

10. Specify the output structure configuration by selecting the **Precise At All Levels** or **Inherited From Source** option.

    When an analyst derives a CAE structure from another structure, the system retains the precise or imprecise rules specified in the selected clone rule set definition. If the clone rule set definition is set to:

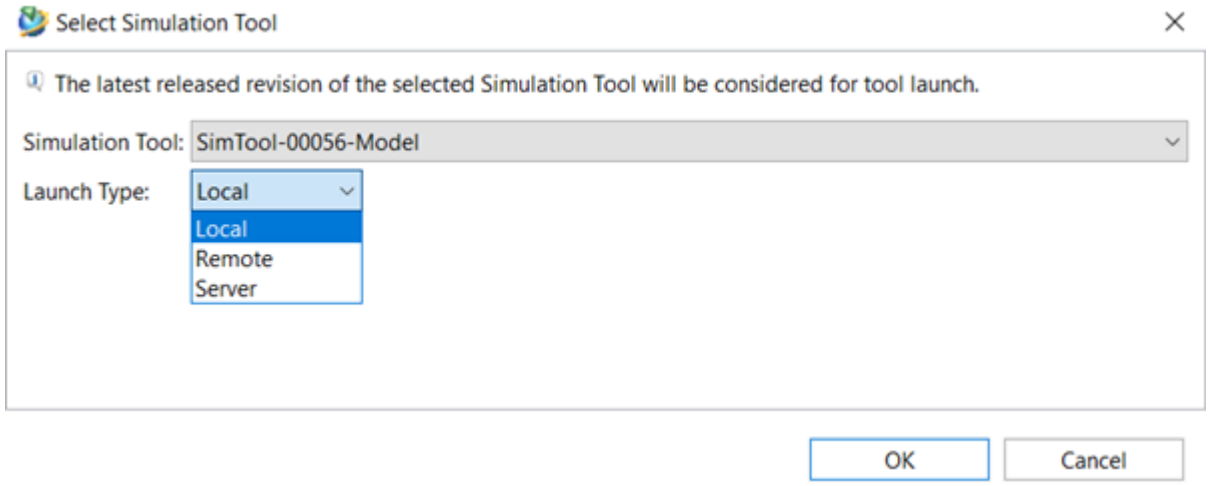
    - **Precise At All Levels**: Every cloned assembly or subassembly in the derived CAE structure is set to precise.
    - **Inherited From Source**: Every cloned assembly or subassembly in the derived CAE structure is set to the same state (precise or imprecise) as its corresponding object in the source structure.

    > Note:
    >
    > When a child component of a cloned subassembly is referenced, the derived subassembly includes the revision of the child component based on the precision setting of the source subassembly and the active revision rule.

11. Enter additional item creation information by selecting the **Create related CAE Analysis Items** and the **Create related CAE Result Items** options.

12. Select a domain from the **Select Domain** list.

13. To allow simulation analysts to launch a simulation tool after deriving a structure, click **Select Tool** and select a tool from the **Select Simulation Tool** dialog and the launch type.



The tools you select should be released and should have **CAE 3D Model** revision as its primary object. This is the same tool list that is displayed when you configure a simulation tool for structure maps. If the launch type for the tool is configured as **Single Launch**, only a single process is launched. However, if it is configured as **Multiple Launch**, multiple processes are launched.

14. To save the derivative rule, click **Save** in the view toolbar.

15. Select the **Rules** tab and create a Filter, Skip, or Clone rule.

Add filter, skip, and clone rules

You can create a derivative rule to indicate which components of the source CAE structure to use to create a derived CAE structure. The derivative rule consists of the following rules:

- Filter rule identifies the components and their children you want to filter in the derived structure.
- Skip rule skips any level in a structure. This rule skips one level of the structure while continuing to process any children of the identified level of the structure.
- Clone rule duplicates a structure.

1. In CAE Manager, click the **CAE Configuration** toolbar menu, and choose the **Derivative Rules Configuration** toolbar icon.

2. Select a derivative rule you have created and click the **Rules** tab.

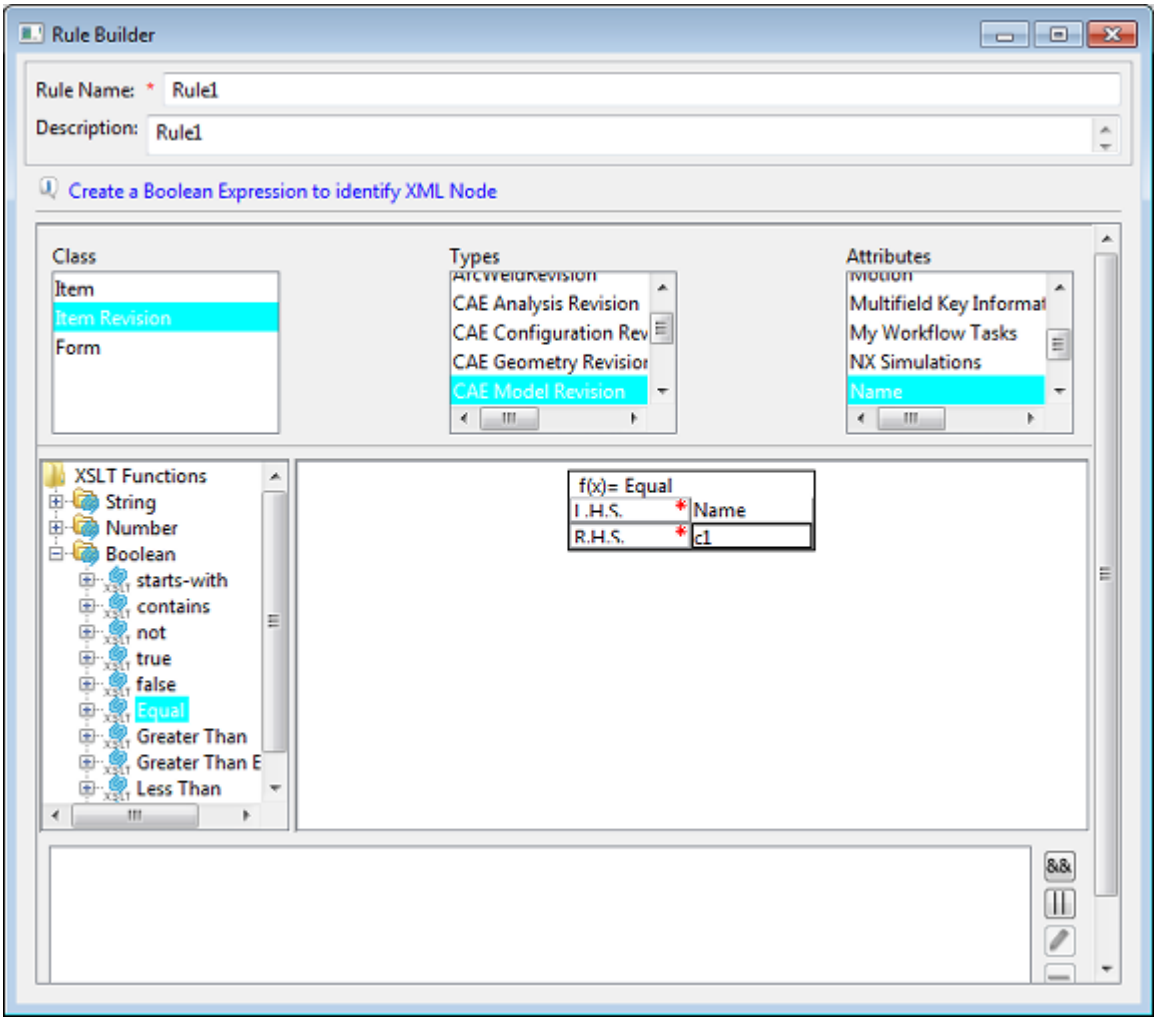
3. From the **Rules** list, select **Filter**, **Skip**, or **Clone** and click **Add**.

4. In the **Rule Builder** dialog box, specify a rule name in the **Rule Name** box.

5. (Optional) Specify a description in the **Description** box.

6. Select appropriate options in the **Class**, **Types**, and **Attributes** lists.

7. Perform the following steps to create a conditional expression:



- Select an attribute.
  - Select the **Item**, **Item Revision**, or **Form** class from the **Class** list, for example, **ItemRevision**.
  - Select the required type from the **Types** list, for example, **CAEModelRevision**.
  - Select the required attribute from the **Attributes** list, for example, **Name**.
- Define a conditional expression.
  - Select the required **XSLT** function in the **XSLT Function** tree.

An **XSLT** function is required to create a conditional expression that has a primary return type such as **String**, **Number**, or **Boolean**. Examples include **Boolean** and **Equal**.

- Drag the selected **XSLT** function in the pane next to the **XSLT Function** tree.
- Provide values for the **XSLT** function in one of the following ways:
  - Type static values in the parameter box.
  - Drag the required attributes in the parameter box.
  - Select an **XSLT** function from the **XSLT Function** tree and drag it in the parameter box.

    For example, drag the **Name** attribute to the **L.H.S** box and type **c1** in the **R.H.S** box.

    Here **c1** represents the name of the item type you want to filter, skip, or clone.
- Click the && button and the || button to add the rule condition to the set of conditions.

You can also use the **Edit** button to edit a rule condition, the **Delete** button to remove a rule condition, or the **Clear** button to clear all rule conditions.

8. To create another rule, repeat steps 3-7.
9. To create the derivative rule, click **Finish**.

Create derivative variant configuration

1. In CAE Manager, click the **CAE Configuration** toolbar menu, and click **Derivative Rules Configuration**.
2. Select the **Variant Configurations** tab.
3. Select the **Site**, **Group**, or **User** scope option.
4. In the **Derivative Rules Configuration** view, click the **Create** view bar icon.
5. To assign an ID, click **Assign** in **ID** box of the **Create Derivative Variant Configuration** dialog box.
6. To assign a revision number, click **Assign** in the **Revision** box.
7. To specify a rule name, type a unique name in the **Name** box.
8. (Optional) Type a description in the **Description** box.

9. Click **OK**.

10. In the **Configuration Details** box, specify a configuration in XML format.

    To view or copy a sample configuration, see the *TC_DATA***CAEDerivativeVariantConfig.xml** file.

    Example:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!--
For a master deck that has 3 variants options configured on the root -
01_Market, 02_BodyStyle , 03_Engine and 04_Roof -
the following Derivative Variant Configuration will create 3 derived
decks as follows:
1. Vehicle configuration for European market with 3 doors,
V6 engine and sunroof
2. Vehicle configuration for Asia Pacific's market with 4 doors,
V6 engine and standard roof
3. Vehicle configuration for American market with 3 doors,
V4 engine and sunroof
-->
<DERIVATIVE_VARIANT_CONFIG>
  <CONFIGURATION>
    <DESCRIPTION>Deck1 representing Europe vehicle configuration</
DESCRIPTION>
      <VARIANT_OPTION NAME="01_Market" VALUE="EU"/>
      <VARIANT_OPTION NAME="02_BodyStyle" VALUE="3DR"/>
      <VARIANT_OPTION NAME="03_Engine" VALUE="V6"/>
      <VARIANT_OPTION NAME="04_Roof" VALUE="Sun"/>
  </CONFIGURATION>
  <CONFIGURATION>
    <DESCRIPTION>Deck2 representing Asia-Pacific vehicle
      configuration</DESCRIPTION>
      <VARIANT_OPTION NAME="01_Market" VALUE="APAC"/>
      <VARIANT_OPTION NAME="02_BodyStyle" VALUE="4DR"/>
      <VARIANT_OPTION NAME="03_Engine" VALUE="V6"/>
      <VARIANT_OPTION NAME="04_Roof" VALUE="Standard"/>
  </CONFIGURATION>
  <CONFIGURATION>
    <DESCRIPTION>Deck3 representing USA vehicle configuration</DESCRIPTION>
      <VARIANT_OPTION NAME="01_Market" VALUE="USA"/>
      <VARIANT_OPTION NAME="02_BodyStyle" VALUE="3DR"/>
      <VARIANT_OPTION NAME="03_Engine" VALUE="V4"/>
      <VARIANT_OPTION NAME="04_Roof" VALUE="Sun"/>
    </CONFIGURATION>
</DERIVATIVE_VARIANT_CONFIG>
```

11. To save this rule, click the **Save** view bar icon.

Configure relationship types for model and analysis item revisions

Configure relationship types

In a complex product development environment, different analysts perform different tasks in the overall analysis. For example, abstractions are delivered by one group, model builds by another group, load cases defined by another group, and solver ready decks are built by another group. In such scenarios, it becomes critical to know whether analysis data, with possibly multiple dependencies, is out of date. The analyst can then act on it and ensure that the analysis is built with the correct set of data to deliver accurate results.

In CAE Manager, analysts can check for later revisions of item revisions attached to **CAE 3D Model** or **CAE 3D Analysis** item revisions. They can also check for changes to any attachments of **CAE 3D Model** or **CAE 3D Analysis** item revisions in the **Attachments** view.

When analysts perform these checks, the simulation administrator can configure Simulation Process and Data Management to consider only those objects attached via relation types that are significant for the company's business model. The simulation administrator can create significant relation types for **CAE 3D Model** and **CAE 3D Analysis** item revisions. These significant relation types are different for **CAE 3D Model** and **CAE 3D Analysis** item revisions. However, a significant relation type for the **CAE 3D Model** item revision, for example, applies to all its subtypes.

> Note:
>
> The significant relation type is a site-wide configuration set by a simulation administrator and it cannot be overridden by an analyst.

The process is as follows:

- At a site, the simulation administrator configures significant relationship types for **CAE 3D Model** and **CAE 3D Analysis** item revisions and its subtypes.
- The analyst, checks for changes to **CAE 3D Model** and **CAE 3D Analysis** item revisions and marks them as up to date.

1. In CAE Manager, click **Edit→Options→CAE→General**.
2. In the **Options** dialog box, hover over a field to find the preference name associated with it.

   For information about retrieving a list of preferences, see *Where can I get a list of preferences?* in *Teamcenter Preferences*.
3. Create signification relationship types for **CAE 3D Model** item revisions.

   a. Click the **Out-of-date Significant Relations** tab.

b. (Optional) Select a relationship type for **CAE 3D Model** from the menu and click the **Add** button to add the relationship type to **Significant Relationship Types for CAE 3D Model** list.

c. (Optional) Click the **Remove** button to remove a relationship type from the **Significant Relationship Types for CAE 3D Model** list.

4. Create signification relationship types for **CAE 3D Analysis** item revisions.

a. Click the **Out-of-date Significant Relations** tab.

b. (Optional) Select a relationship type for **CAE 3D Analysis** from the menu and click the **Add** button to add the relationship type to the **Significant Relationship Types for CAE 3D Analysis** list.

c. (Optional) Click the **Remove** button to remove a relationship type from the **Significant Relationship Types for CAE 3D Analysis** list.

Configure highlight colors

The highlight colors you configure are applied when using highlighting in the CAE Manager secondary views.

1. In CAE Manager, click **Edit→Options→CAE→General**.

2. In the **Options** dialog box, hover over a field to find the preference name associated with it.

   For information about retrieving a list of preferences, see *Where can I get a list of preferences?* in *Teamcenter Preferences*.

3. Click the **Out-of-date Highlights** tab.

4. Choose an appropriate color option for **Action**, **Caution**, **No Action** and **Out of Scope** options.

   - **Action** — Indicates that an action is needed on an object. (The default color is red.)
   - **Caution** — Indicates that caution is needed on an object. (The default color is yellow.)
   - **No Action** — Indicates that no action is needed on an object. (The default color is green.)
   - **Out of Scope** — Indicates that an object is not considered by the operation. (The default color is gray.)

Capture the exact configuration of product and model structures

Enable pedigree operations and specify model and analysis relations

You can enable or disable pedigree-related commands using the **Enable Pedigree Operations** option. This option is stored as the **CAE_enable_pedigree_operations** preference in the database. You can set this preference at the **Site**, **Role**, **Group**, or **User** level. By default, this preference is set to **true** at the **User** level.

You can also specify valid relation types for model and analysis objects and all their subtypes.

1. In CAE Manager, click **Edit→Options→CAE→General**.
2. In the **Options** dialog box, hover over a field to find the preference name associated with it.

   For information about retrieving a list of preferences, see *Where can I get a list of preferences?* in *Teamcenter Preferences*.
3. Click the **Pedigree Operations** tab.
4. (Optional) Clear the **Enable Pedigree Operations** check box to disable pedigree options. This option is selected by default.
5. Specify the valid relation types for **Model Relations**.

   These relationships are applicable to model objects and all their subtypes. The system follows the specified relationships in capturing the pedigree. You can optionally specify custom relations.
6. Specify the valid relation types for **Analysis Relations**.

   These relationships are applicable to analysis objects and all their subtypes. The system follows the specified relationships in capturing the pedigree. You can optionally specify custom relations.

Configure simulation tools to launch preprocessors, solvers, and postprocessors

Verify the compatible versions of simulation integration tools

The integration matrix for compatible versions of tools integrated with Simulation Process and Data Management is available on Support Center.

Verify the version compatibility of simulation integration tools in the **Simulation & Test Integrations** tab in **Teamcenter Integrations Availability Matrix.xlsx** file.

This file is available at **Support Center** > **Teamcenter** > **Downloads** > **Hardware and Software Certifications** > **Teamcenter Certifications and Information** > **Integrations Matrix**.

Why configure simulation tools?

Simulation Process and Data Management provides a framework for configuring and launching simulation tools that can include preprocessors, solvers, postprocessors, and other tools to perform custom actions. It allows you to:

- Define and organize specific tools to gather process inputs.
- Create Teamcenter objects to hold tool output.
- Import tool output into Teamcenter.
- Configure a simulation tool to specify an object (item, item revision, or dataset) in the data model as the object to hold the output data files.
- Define rules to navigate from the primary input object to the output object through a combination of relationships where the originating item revision is either the primary or secondary object of the relationship.
- Define a naming pattern for each of the objects (item, item revision, or dataset) that are created during the tool launch.
- (For server and remote modes) Export the product or model structure as a secondary object by leveraging the pedigree information to configure the respective structure.

> Note:
>
> Only the simulation administrator or a user with DBA privileges can configure simulation processes.

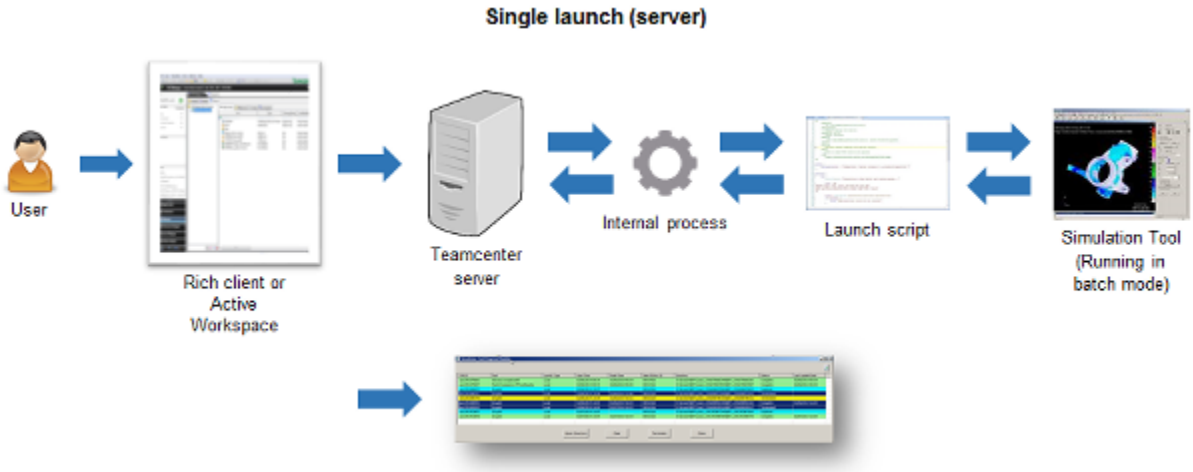
After the simulation administrator configures the simulation tools, analysts can run the tools from the Teamcenter rich client on their desktops and place the tool output in Teamcenter upon completion.

Analysts can launch simulation tools as a local launch, local detached launch, remote launch, or server launch, depending on how you configure launch parameters at your site. Simulation Process and Data Management uses Dispatcher Server components to launch external simulation tools as a local detached launch or a remote launch.

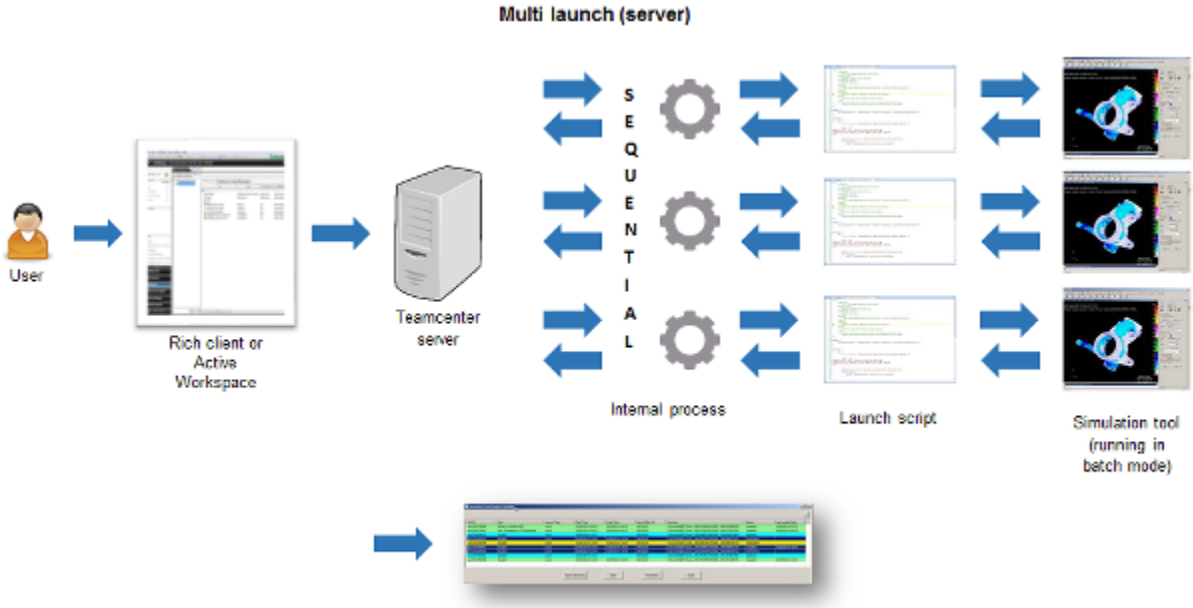
> Note:
>
> The local detached launch option has been deprecated and will be removed in a future version of Teamcenter.

Dispatcher Server components are not required for the local launch and server launch options.

Server launch without Dispatcher



For *single launch*, the user selects an input item revision, selects a simulation tool, and selects **Server Launch**. Teamcenter creates the job ID, exports the data to the **ESP** directory, creates a new internal process, and executes the process.



For *multiple launch*, the user selects multiple item revisions, selects a simulation tool, and selects **Server Launch**. Teamcenter creates the job IDs, exports the data to the **ESP** directories, creates new internal processes, and executes the processes *sequentially*.

Prerequisites for simulation tool launch dialog box

A Web browser is a prerequisite for **Launch Simulation Tool** dialog box.

For information about certified browser versions, see the Hardware and Software Certifications knowledge base article on Support Center.

> Note:
>
> To display the **Launch Simulation Tool** dialog box on the Linux platform, you must set the **MOZILLA_FIVE_HOME** environment variable to point to the **/usr/bin/firebox/** directory, and then prepend this directory to the **LD_LIBRARY_PATH** variable.

Define the dataset name to store simulation tool configurations

A user with DBA privileges must configure the **CAE_simulation_tool_config_dsname** preference, and the simulation administrator must have write access to the indicated dataset. If no such dataset exists, it is created the first time a privileged user (dba or a simulation administrator) saves a simulation tool configuration.

The simulation tool configuration file is managed in a **TCCAESimToolConfig102009** dataset indicated by the **CAE_simulation_tool_config_dsname** preference.

Set the credential token expiry time for tool launch

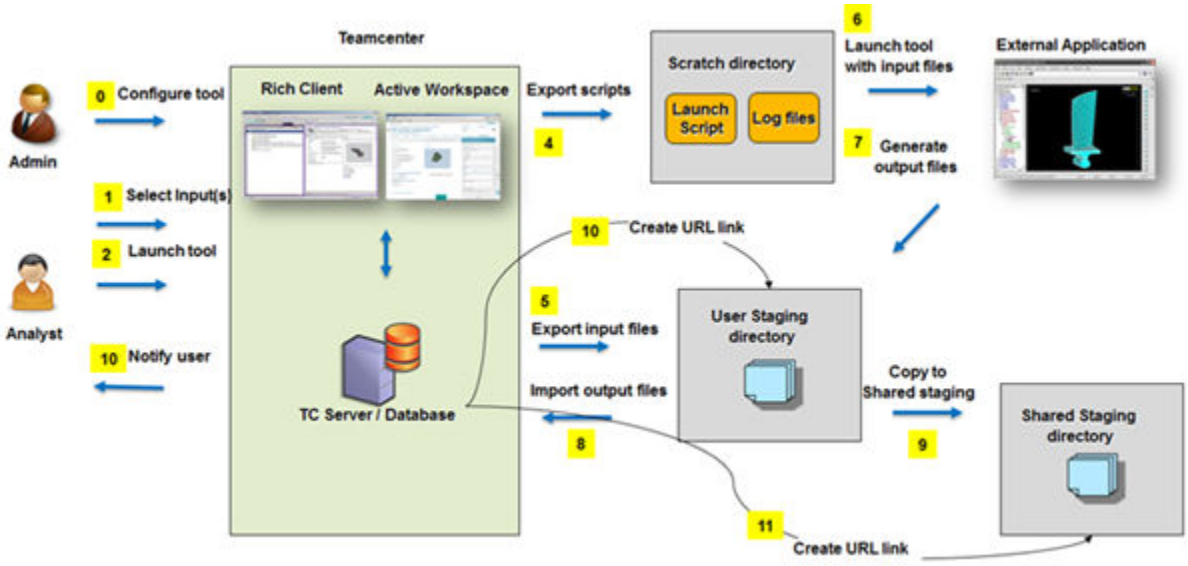
As an administrator, you can set the credential token expiry time for tool launch by editing the **CAE_expire_time_for_credential_token** site preference and specifying an appropriate value.

For information about retrieving a list of preferences, see *Where can I get a list of preferences?* in *Teamcenter Preferences*.

Configure the user and shared staging locations for simulation tools

The simulation tool launch framework uses the default scratch location as the directory to store all the input and output files. However, if the user staging location is configured, the tool launch uses the user staging location as the directory for storing all the input and output files for the **Local Launch and Server Launch options**.

In addition, a shared staging location can be configured at the site, group, and user level. This is done to avoid exporting the same result files across simulation tools by multiple users. These result files can also be easily shared across multiple users.



During tool launch, the system provides the path to the **User Staging Directory** and the **Shared Staging Directory** to the launch script. The system exports or imports the required files from the **User Staging Directory**. The launch script uses the configured **Shared Staging Directory** path to copy the files.

1. In CAE Manager, click **Edit** → **Options** → **CAE**→**Simulation Tools**.

2. In the **Options** dialog box, hover over a field to find the preference name associated with it.

   For information about retrieving a list of preferences, see *Where can I get a list of preferences?* in *Teamcenter Preferences*.

3. To set the root item of a simulation tool configuration, in the **CAE Tool Configuration Root** box, click the **Search** icon, choose **CAE Simulation Tool** from the **Type** menu, and execute the search to find the simulation tool you want to set as the root.

   After you create a simulation tool structure, you can specify the root tool using this option.

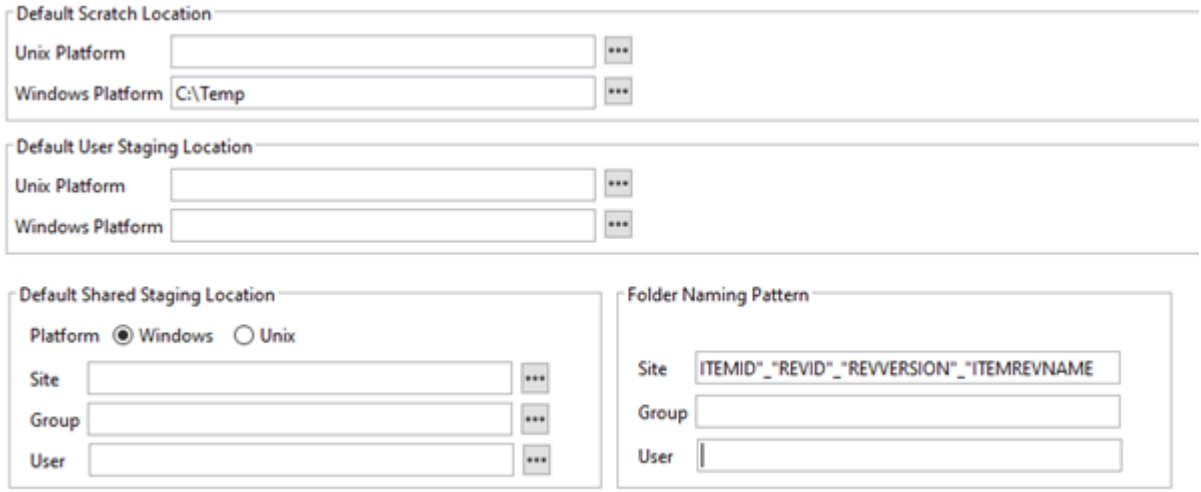
4. In the **Default Scratch Location** box, specify a location for the appropriate operating system. This defines the temporary location to be used by the simulation tool launch process on the Windows or the Linux platform.

   You can click the **Browse** button, and select the scratch location.

   The tool launch framework uses the default scratch location as the location to store all the input and output files. However, if the user staging location is configured (see step 5), the tool launch framework uses the staging location as the location for all input and output files for the **Local Launch** and **Server Launch** options.

5. In the **Default User Staging Location** box, specify a location for the appropriate operating system. This defines the user staging location to be used by the simulation tool launch process on the Windows or the Linux platform.

   You can click the **Browse** button, and select the user staging location. By default, no location is set for these options.



> Note:
>
> This is applicable for server launch only.
>
> Shared paths can be used for the default scratch location and the user staging location.
>
> The OS user who is running the Teamcenter server should have the full control to share access for the shared path.
>
> Windows services do not support mapped drives. You must use a UNC path or a path that is local to the machine for the shared staging location.

6. To specify a shared staging location, in the **Default Shared Staging Location** box, click the **Browse** button and select the staging location. You can specify a location at the site, group, and user level for the appropriate operating system. By default, no location is set for these options.

7. Specify a folder naming pattern for the shared staging location at the site, group, and user level. The naming pattern is a combination of fixed strings in double quotation marks and keywords. The supported keywords are **ITEMID**, **REVID**, **ITEMREVID**, **ITEMREVNAME**, and **REVVERSION**.

   The fixed strings are optional and they can appear more than once in double quotation marks. You cannot name the folder name with a character that is not supported by the target file system.

   Naming pattern examples:

- *** ITEMID"-"ITEMREVNAME**

  The system evaluates this as **00069-Item**.

- *** ITEMID"_"REVID"_"REVVERSION"_"ITEMREVNAME**

  The system evaluates this as **00079_A_1_Item**.

By default it is set to **ITEMID"_"REVID"_"REVVERSION"_"ITEMREVNAME** at the site level.

> Example:
>
> If you specify **C:\Temp** as the user staging location and **ITEMID_REVID** as the folder naming pattern, the input and output files are available in the following directories:
>
> - **Local Launch**: **C:\Temp***ITEMID_REVID*
> - **Server Launch**: **C:\Temp***ITEMID_REVID\ID_of_the_Logged_In_User*
>
> The system creates a subfolder with the ID of the logged on user in the folder created as per the naming pattern.

8. To disable the context-based menu for simulation tool launch, clear the **Enable Context Based Menu for Simulation Tool Launch** check box. This check box is selected by default.

> Note:
>
> The **Enable Context Based Menu for Simulation Tool Launch** is controlled by a user with DBA privileges and the check box determines whether a tool is available in the menus based on whether the selected item revision is valid for that particular tool. If the context-based menu check box is cleared, analysts can specify input types that do not match the selected input item revision.

By default, this check box is selected by default.

Set favorite simulation tools

1. In CAE Manager, click **Edit→Options→CAE→Simulation Tools**.

2. In the **Options** dialog box, hover over a field to find the preference name associated with it.

   For information about retrieving a list of preferences, see *Where can I get a list of preferences?* in *Teamcenter Preferences*.

3. To add simulation tools to the **Simulation Tools** toolbar in CAE Manager, select the configured tools from the following sections:

- **My Favorite Simulation Tools**: Defines the simulation tools configured as favorite tools to appear in the **Simulation Tools** → **Favorite Tools** menu in CAE Manager.

  After the simulation administrator configures the simulation tools, analysts can use the **Simulation Tools** section in the **Options** dialog box to add their simulation tools to the **Simulation Tools**→**Favorite Tools** menu in CAE Manager.

- **Released Simulation Tools in Toolbar**: Defines the simulation tools configured to appear in the global toolbar in CAE Manager.

  After the simulation administrator configures the simulation tools, analysts can use the **Simulation Tools** section in the **Options** dialog box to add their simulation tools to the global toolbar in CAE Manager.

  Note:

  If **Released Simulation Tools in Toolbar** are updated due to a change in the **CAE_my_simulation_tools_in_toolbar** preference or due a session change by the user, the analyst has to switch or reopen the perspective to refresh the toolbar. This is due to a known limitation in the Eclipse platform.

4. Select a revision rule for the simulation tools in the toolbar.

   By default, it set to **Any Status; No Working** based on the **CAE_tool_menu_rev_rule** user preference.

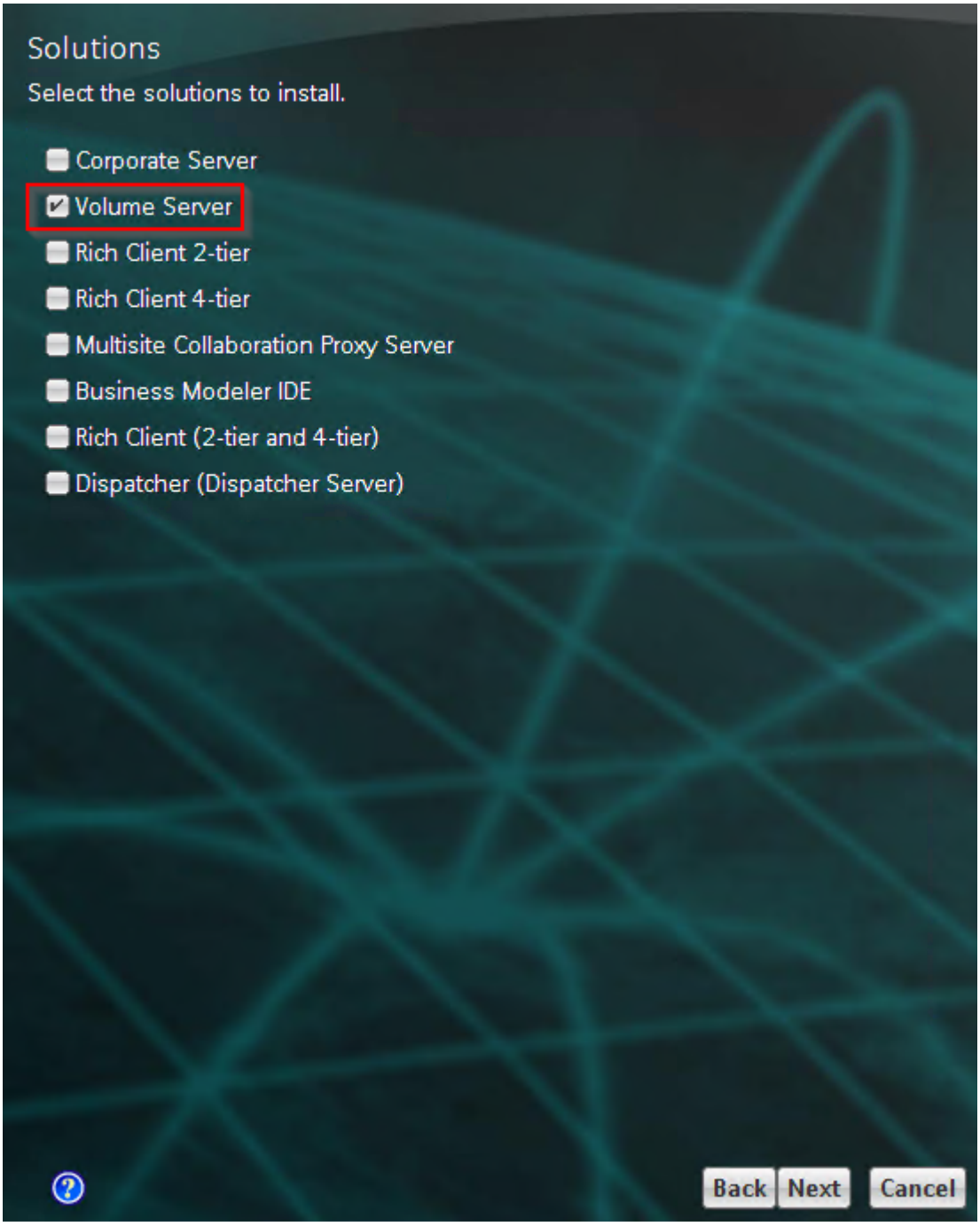
Configure server cache for simulation tools using the PLM XML export method

The Teamcenter administrator can configure the Teamcenter File Management System (FMS) and file system cache service (FSC) cache such that simulation analysts can launch preconfigured simulation tools faster. The simulation analyst uses model structures to perform different types of analyses and for different load cases. Each time a different analysis or a different load case is being performed, the system fetches the required input files from the Teamcenter database. Moreover, in large organizations, there are hundreds of simulation analysts who typically perform these different kinds of analyses using similar kinds of model structures. The Teamcenter administrator can use the server cache and make the required input files available locally to quicken the tool launch process.

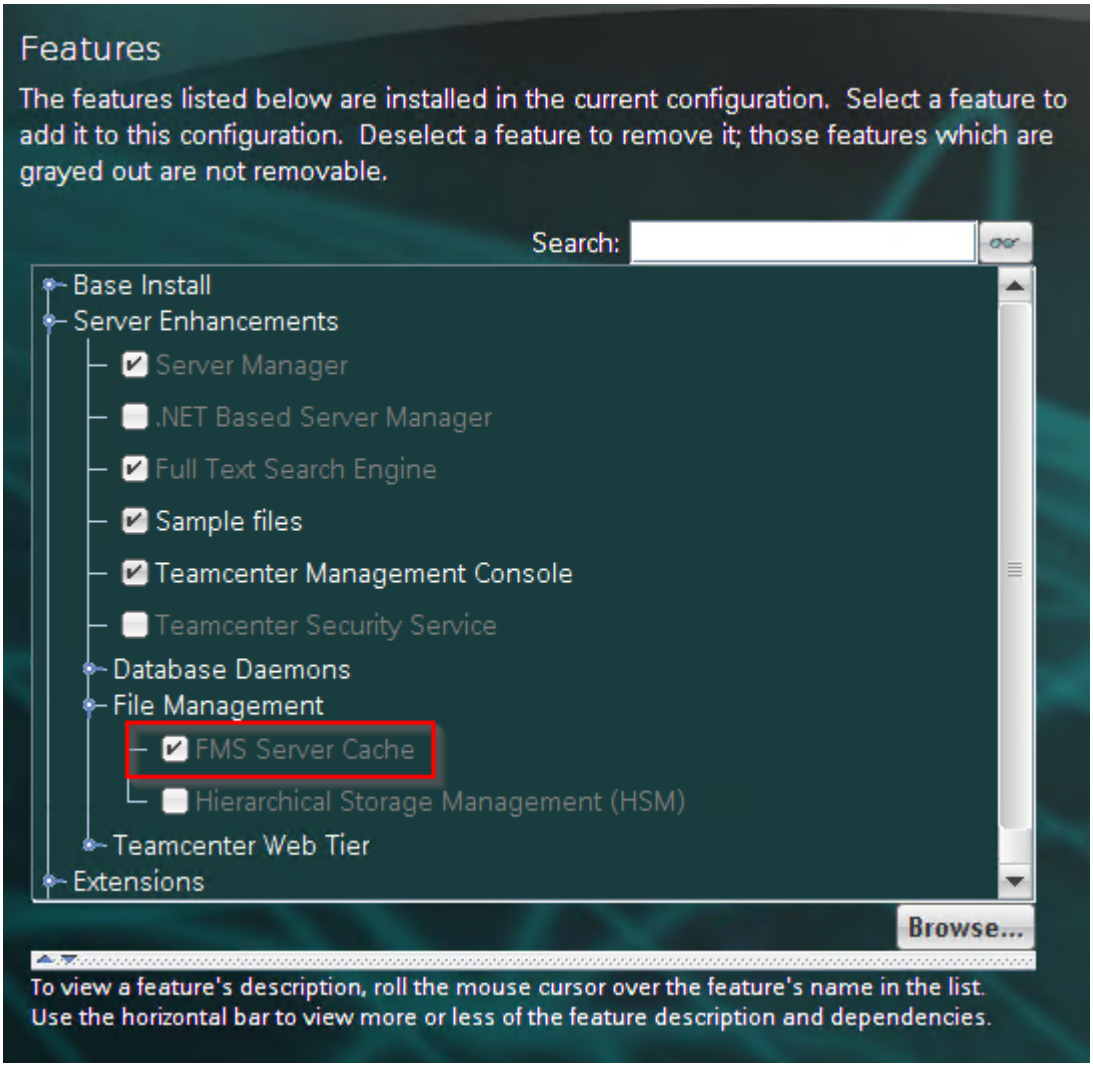
The simulation administrator configures simulation tools including preprocessors, solvers, and postprocessors. While configuring the tools, the simulation administrator can specify primary input rules and additional input rules. Both these rules use the file input method or the PLM XML export method to gather all the required inputs required for the simulation tools. The Teamcenter administrator can configure the server cache for the PLM XML export method. This configuration is supported for all simulation launch methods such as local launch, remote launch, and server launch.

Install the standalone cache server

1. Run TEM and in the **Solutions** panel, select **Volume Server**.

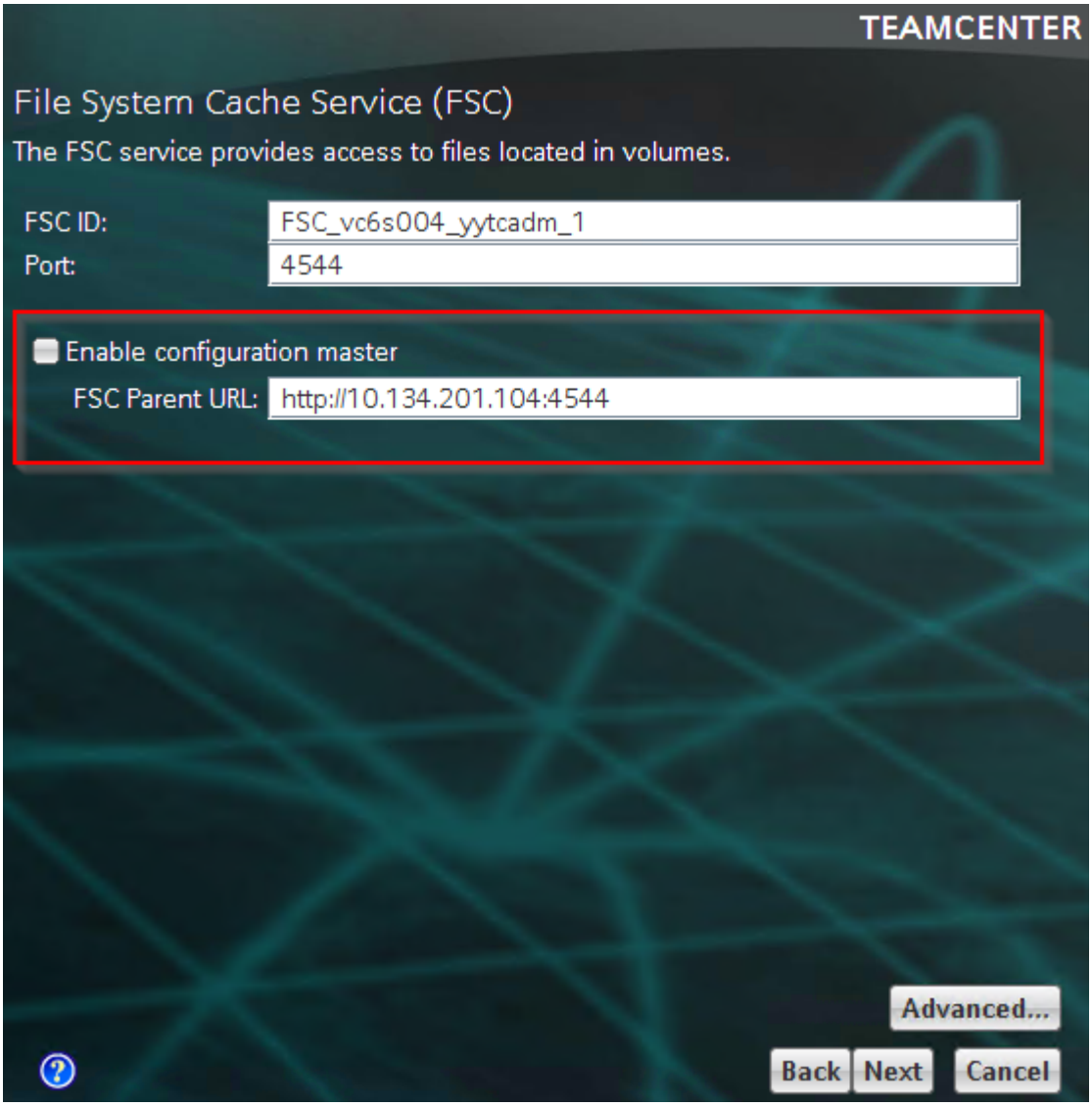


2. In **Server Enhancements → File Management**, select **FMS Server Cache**.



3. In the **File System Cache Service (FSC)** panel, clear the **Enable configuration master** check box.

4. Specify the **FSC Parent URL** value. This specifies the URL to the parent FSC if the current FSC is not a master.



5. Complete the installation.

6. Edit the *TC_ROOT***fsc****FSC**_*Machine_Name_User_Name*.**xml** file and set the **FSC_FilesPerWholeFileCacheDir** property value to **1024**.

> Caution:
>
> File caching does not happen if this is set to zero (**0**) or as a negative value.

Modify the FMS master configuration file on the Teamcenter server

1. Edit the *TC_ROOT***fsc****fmsmaster_FSC**_*Machine_Name_User_Name*.**xml** file.

   Example:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fmsworld SYSTEM "fmsmasterconfig.dtd">

<fmsworld>
  <fmsenterprise id="-1841327028" volumestate="normal">
    <fccdefaults>
      <property name="FCC_CacheLocation" value="$HOME/FCCCache|/tmp/
            $USER/FCCCache" overridable="true" />
      <property name="FCC_HashBlockPages" value="6144" overridable="true" />
      <property name="FCC_LogFile" value="$HOME/fcc.log|/tmp/$USER/fcc.log"
            overridable="true" />
      <property name="FCC_MaxExtentFileSizeMegabytes" value="256"
            overridable="true" />
      <property name="FCC_MaxExtentFiles" value="11" overridable="true" />
      <property name="FCC_MaxReadCacheSize" value="1000M"
overridable="true" />
      <property name="FCC_MaxWriteCacheSize" value="1000M"
overridable="true" />
      <property name="FCC_MaximumNumberOfFilePages" value="28672"
            overridable="true" />
      <property name="FCC_MaximumNumberOfSegments" value="10688"
            overridable="true" />
    </fccdefaults>
    <fscgroup id="mygroup">
      <fsc id="FSC_vc6s004_yytcadm"
            address="http://vc6s004.net.plm.eds.com:4544" ismaster="true">
        <volume id="00f55d7b4440923f944c" enterpriseid="-1841327028"
            root="C:\apps\tc\tc12\TV" priority="0" />
        <transientvolume id="8409518771a8915f7f0d8264fd06e68a"
            enterpriseid="-1841327028" root="C:\\apps\\tc\\tc12\\TRV" />
      </fsc>
      <clientmap subnet="127.0.0.1" mask="0.0.0.0">
        <assignedfsc fscid="FSC_vc6s004_yytcadm" priority="0" />
      </clientmap>
    </fscgroup>
  </fmsenterprise>
</fmsworld>
```

Add the cache server configuration to the FMS master configuration. Be sure to include the IP address of the cache server machine and set the **ismaster** value to **false**.

Change as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fmsworld SYSTEM "fmsmasterconfig.dtd">

<fmsworld>
  <fmsenterprise id="-1841327028" volumestate="normal">
    <fccdefaults>
```

```
      <property name="FCC_CacheLocation" value="$HOME/FCCCache|/tmp/
            $USER/FCCCache" overridable="true" />
      <property name="FCC_HashBlockPages" value="6144" overridable="true" />
      <property name="FCC_LogFile" value="$HOME/fcc.log|/tmp/$USER/fcc.log"
            overridable="true" />
      <property name="FCC_MaxExtentFileSizeMegabytes" value="256"
            overridable="true" />
      <property name="FCC_MaxExtentFiles" value="11" overridable="true" />
      <property name="FCC_MaxReadCacheSize" value="1000M"
overridable="true" />
      <property name="FCC_MaxWriteCacheSize" value="1000M"
overridable="true" />
      <property name="FCC_MaximumNumberOfFilePages" value="28672"
            overridable="true" />
      <property name="FCC_MaximumNumberOfSegments" value="10688"
            overridable="true" />
    </fccdefaults>
    <fscgroup id="mygroup">
      <fsc id="FSC_vc6s004_yytcadm"
            address="http://vc6s004.net.plm.eds.com:4544" ismaster="true">
        <volume id="00f55d7b4440923f944c" enterpriseid="-1841327028"
            root="C:\apps\tc\tc12\TV" priority="0" />
        <transientvolume id="8409518771a8915f7f0d8264fd06e68a"
            enterpriseid="-1841327028" root="C:\\apps\\tc\\tc12\\TRV" />
      </fsc>
```

<!-- Make the changes for adding the cache server configuration to the FMS master configuration as follows -->

```
<fsc id="FSC_vc6s004_yytcadm_1"
      address="http:{IP_of_cache_server_machine}:4544" ismaster="false">
      <fccdefaults>
      <property name="FCC_EnableDirectFSCRouting" value="false"
        overridable="false"
      </fccdefaults>
   </fsc>
```

<!-- End of changes -->

```
      <clientmap subnet="127.0.0.1" mask="0.0.0.0">
        <assignedfsc fscid="FSC_vc6s004_yytcadm" priority="0" />
      </clientmap>
    </fscgroup>
  </fmsenterprise>
</fmsworld>
```

2. Define the configuration using the nearest FSC for the clients. In the above example, the **clientmap** should point to the nearest FSC.

```
        <clientmap subnet="127.0.0.1" mask="0.0.0.0">
                <assignedfsc fscid="FSC_vc6s004_yytcadm" priority="0" />
              </clientmap>
```

3. Save the file.

4. To copy the FMS master configuration file from the Teamcenter server to the standalone cache server:

   a. Restart the Teamcenter FSC service.

   b. Restart the FCC service from the *TC_ROOT***tccs\bin** directory.

Modify the FCC file on the client machine to point to the standalone cache server

1. On the client machine, edit the *TC_ROOT***tccs\fcc.xml** file.

   Example:

```
<!DOCTYPE fccconfig SYSTEM "fccconfig.dtd">

<!--
bcprt
This software and related documentation are proprietary to UGS Corp.
COPYRIGHT 2007 UGS CORP.  ALL RIGHTS RESERVED
ecprt
-->
<fccconfig version="1.3.2">
  <fccdefaults>
    <!-- general -->
    <!-- <property name="FCC_LogFile" value="$HOME\fcc.log|/tmp/$USER/
fcc.log"
        overridable="true"/> -->
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  </fccdefaults>
  <!-- default parentfsc - this is a marker that will be overwritten
       by the installer -->
  <parentfsc address="http://vc6s004:4544/" priority="0" />
  <assignment mode="clientmap" />
</fccconfig>
```

2. Edit the **parentfsc address** property to point the address to the standalone cache server.

3. If FSC cache server is used for **Teamcenter Tool Launcher Client on Active Workspace**, edit the **Fms_BootStrap_Urls** preference value to include a value for the standalone cache server.

Example:

http://*Machine_Name***:***Port_Number*

The values for the machine name and port number are the same as those specified in the step 4.

4. Restart the FCC service from the *TC_ROOT***tccs\bin** directory.

Define simulation tools

Create a simulation tool

You can configure simulation tools using the **Simulation Tool Configuration** view. While configuring simulation tools, you can clone tools, change ownership of selected tools, discard and reset tool changes, and create tool separator objects.

1. In CAE Manager, click the **CAE Configuration** toolbar menu, and choose **Simulation Tool Configuration**.

2. To create a simulation tool, click **Create Simulation Tool** in the view toolbar.

   This is the root tool configuration and you can create child tool configurations by selecting the root tool.

   Child tools inherit the values of the parent tool. You can clear the **Inherited** check box in a child tool to make configuration changes specific to the child tool.

3. Click **Assign** in the **Simulation Tool ID** box to automatically assign an ID.

4. Click **Assign** in the **Simulation Tool Revision** box to automatically assign a revision ID.

5. Specify a name in the **Simulation Tool Name** box.

6. Specify tool-specific information in the tabs in the **Simulation Tool Configuration** view.

7. To save the simulation tool configuration, click **Save Simulation Tool** in the view toolbar.

8. (Optional) To create a tool separator object for your child tools, right-click and choose **Add Tool Separator**.

   You may configure different categories of simulation tools for preprocessors, solvers, postprocessors, and other tools. You can use tool separators to separate tools while creating different categories of tools.

> Tip:
>
> You cannot create a tool separator for a root tool.

9. (Optional) To change the ownership of a tool object, click **Change Ownership of Selected Tool** in the view toolbar, and choose a new owning user.

   If a user with DBA privileges creates the root tool and child tools, this user can transfer the ownership of the child tools to different simulation administrators for creating different categories of tools.

   For example, you might have different simulation administrators for preprocessor, solver, and postprocessor tools. In such cases, a user with DBA privileges creates a root tool and then creates child tools for the preprocessor, solver, and postprocessor tools and transfers ownership of these child tools to the respective simulation administrators. The simulation administrators can then create multiple tool categories within the child tools.

10. (Optional) To copy all the information from the source tool and to create a new tool by cloning the source tool, click **Clone Simulation Tool** in the view toolbar.

    When you clone tools, the name of the child tool is the same as that of the source and the new tool is added as the last child of the parent tool.

11. (Optional) To create a new root simulation tool by removing the current tool tree and to clear the configuration in all panels, click **Clear Simulation Tool Structure** in the view toolbar.

12. To release a simulation tool, click **Release Simulation Tool**.

13. To release all simulation tools, click **Release all Tools**.

14. To undo the release of a simulation tool (already released), click **Unrelease Simulation Tool**.

Configure general properties for simulation tool launch

You can configure general properties after you have configured a process in the CAE Manager application. These include:

- Specifying an icon file for a tool launch and optionally sharing the icon file with more than one process.

- Specifying a tool launch dialog display file for configured simulation tools. The launch dialog is used to render the simulation tool launch dialog box for analysts.

- Checking or clearing an option to display or not display the **Simulation Tool Progress Monitor** dialog box for analysts to view the status of the tool launch.

- Checking or clearing an option to display or not display the **Launch Simulation Tool** dialog box. If you clear this option, the **Launch Simulation Tool** dialog box is not displayed, and analysts can start the tool launch directly.

1. In CAE Manager, click the **CAE Configuration** toolbar menu, and choose **Simulation Tool Configuration**.

2. Click the **General** tab.

   > Note:
   >
   > You must have at least one simulation tool configured to view the **General** and other tabs for a simulation process.

3. Specify a help document for generic tools.

   Simulation analysts may not be familiar with some generic tools. You, as simulation administrator, can upload a help document about how to use this tool using the **Detail Description Dataset** box. After launching the preconfigured tool, the simulation analyst can open the help related to this tool and view it.

   a. In the **Detail Description Dataset** box, click the **Add** button.

   b. To add a description file, click the **Browse** button in the **Configure File Description** dialog box and select a file.

   c. (Optional) Specify a dataset name.

   d. Select the dataset type.

4. To specify an icon file, click the **Browse** button in the **Icon File** area.

   You can specify an icon file (for example, png or jpeg) and this icon file is shared for more than one process that you define.

   If you specify a full path to the icon file, the system imports the file to the dataset and displays only the icon file name and not the full path to the icon after you apply the changes. If you specify only the icon file name, the system assumes that the file already exists in the dataset and does not import the file.

   > Note:
   >
   > All icon files are translated into ***.jpg** files before being imported into the dataset. Therefore, to use the same icon file for multiple tools, specify the file with the ***.jpg** extension.

If you specify a child process and you select the **Inherited** check box for the child process, the child process inherits the icon file details of the parent process.

5. To specify a display file for the tool launch dialog, click the **Browse** button in the **Tool Launch Dialog Display File** area and select the file.

6. To display a progress monitor for analysts to view the status of the tool launch, check the **Display Progress Monitor on Launch** check box.

   If you clear this check box, analysts cannot see the status for the tools they have launched.

7. To display a simulation tool dialog box when an analyst starts a simulation tool launch, check the **Show Launch Simulation Tool Dialog** check box.

   If you clear this check box, the **Launch Simulation Tool** dialog box is not displayed and analysts can start the tool directly. This is especially useful for tools that require no additional inputs from the analyst when starting.

8. To allow analysts to select inputs from multiple views, select the **Allow Input Selection from Multiple Views** check box.

   By default, analysts can select inputs only from a single active view.

9. (Applicable to Active Workspace only) To allow analysts to launch simulation tools without selecting any CAE item revision, select the **Launch Without Any Input** check box.

   Normally, the analyst selects a CAE item revision as the primary input to launch most of the simulation tools. With this option, the analyst can launch the tool in the context of a folder.

   In addition, it also supports launching tools from CAE items that are configured as primary input types in the **Input Configuration** tab.

Configure launch properties to specify a launch method and launch script location

You can use the **Launch Properties** tab to specify a launch method (local, local detached, remote, or server), specify a system temporary location for different launch tools or categories, and specify a launch script location.

You can set up a scratch location with different directories for different tools or categories, and this is especially useful when you have to specify a network directory location for a particular tool. The scratch location you specify is the category or tool level directory for all named references of datasets, datasets associated with item revisions, or item revisions created by Teamcenter.

You specify a scratch location at the root level while configuring a process. While creating sub tools or categories, the system uses the root scratch location if you do not separately specify a scratch location at the sub tools or category level.

1. In CAE Manager, click the **CAE Configuration** toolbar menu, and choose **Simulation Tool Configuration**.

2. Click the **Launch Properties** tab.

3. Specify a launch method.

Option	Requires Teamcenter Dispatcher?	Allows analysts to
Local Launch	No	Launch simulation tools on Teamcenter clients. The analyst must keep the Teamcenter client running until the simulation process is completed.
Local Detached Launch	Yes	Launch simulation tools on Teamcenter clients. The analyst can exit the Teamcenter client after launching the simulation process. The results are stored in Teamcenter.
Remote Launch	Yes	Launch simulation tools on a different machine or module. This machine does not have the Teamcenter server installed on it. However, it has simulation tools and Dispatcher components installed and configured to allow remote launch of simulation tools from Teamcenter clients. Remote launch is ideal for compute-intensive operations, such as meshing, solve execution, and post processing. Such operations do not require interactive user input and analysts can execute them as a batch process on remote machines with load balancing capability and the ability to monitor and administer request queues. When you launch simulation tools, the system exports the data (primary input and additional input files) to the Dispatcher staging directory. After launching simulation tools on a different module, the analyst can exit the Teamcenter client or even shut down the Teamcenter client. The results are stored in Teamcenter.
Server Launch	No	Launch simulation tools on the Teamcenter server. After launching simulation tools on the server, the analyst can exit the Teamcenter client or even shut down the Teamcenter client. The results are stored in Teamcenter.

> Note:
>
> The tool launch job gets stuck in the **Initializing** or **In Progress** state if the **PROCESS_MAX** value is very low in the **serverPool.properties** file and large files have to be exported to the staging directory. For more information about server manager pool-specific configuration tuning, see *Setting the PROCESS_MAX parameter*.
>
> For a remote launch, the tool name must match the service attribute name in the **translator.xml** file on the module from which you want to launch the remote tool.

4. (Optional) For assigning a high performance computing (HPC) configuration, select the **HPC Launch** check box, select a profile and connection, and specify a solver name. This is supported only for local or server launch methods.

   An administrator with DBA privileges **configures HPC parameters for simulation tool launch**.

5. Specify the default scratch location.

   The tool launch framework uses the default scratch location as the location to store all the input and output files. However, if the user staging location is configured (see **step 8**), the tool launch framework uses the staging location as the location for all input and output files for the **Local Launch** and **Server Launch** options.

   To specify the default scratch location at the category or tool level, select the appropriate operating system, click the **Browse** button, and select a location.

   > Note:
   >
   > This is applicable for server launch only.
   >
   > Shared paths can be used for the default scratch location and the user staging location.
   >
   > The OS user who is running the Teamcenter server should have the full control to share access for the shared path.
   >
   > Windows services do not support mapped drives. You must use a UNC path or a path that is local to the machine for the shared staging location.

   If the Dispatcher module is installed on:

   - Linux: Linux and Windows platform must be the same as the Linux path.
   - Windows: Linux and Windows platform must be the same as the Windows path.

6. Specify a launch script location.

   You can specify both the **System File Path** and the **Dataset Specification** options. For both these options, if the Dispatcher module is installed on:

> Note:
>
> For a server launch, the launch script should not output anything to the console. For example, in the case of bat files, the file should have **@echo off** at the beginning of the file.

- Linux: Linux and Windows platform must be the same as the Linux path.
- Windows: Linux and Windows platform must be the same as the Windows path.

Sample startup scripts to launch tools are available in the **CAE_ESP_Sample_Startup_Scripts.zip** file in the *TC_DATA* directory. You can use these sample scripts to create your own startup scripts to launch tools and specify other parameters as appropriate.

a. To specify the location of the launch script, select **System File Path**, click the cell in the **Path** column for the appropriate operating system, and click **Browse** to choose the launch script.

   Create an item in Teamcenter and include a dataset to hold your launch script.

   You cannot upload the script file with the **.bat** file extension directly on a dataset. You can remove this restriction from the dataset using Business Modeler IDE.

   As a work around, upload any text file to the dataset and check in the dataset. Then, check out the dataset, rename the named reference to **.bat**, and check in the dataset.

b. To select a dataset for the tool launch:

   A. Select **Dataset Specification**, click the cell in the **Path** column for the appropriate operating system, and click **Browse** to choose the launch script.

   B. In the **Search** dialog box, search for an item revision and select a dataset from the **Folders** list.

   C. In the **Selected Dataset** box, click the **Add** button to add the selected dataset.

      This is the same item revision and dataset you created in the previous step.

   D. Click **OK** in the **Search** dialog box to add the item revision and dataset information in the respective columns.

   E. (Optional) **Edit the launch script for minor changes** after you have uploaded the dataset.

7. (Optional) Select the **Use Parameter File** check box to create an ASCII text file in the working directory that contains all the command line arguments. Pass this file name as an argument to the

launch script rather than the individual parameters. This option is useful when the argument list to the script is very lengthy.

8. Select the **Use Staging Directory** check box to use shared locations.

   The staging directories are configured in the **Options dialog box** in Teamcenter or CAE Manager.

   This is supported only for the **Local Launch** and **Server Launch** options.

Remove the file restriction from the dataset using the Business Modeler IDE

Only a DBA can perform the following procedures.

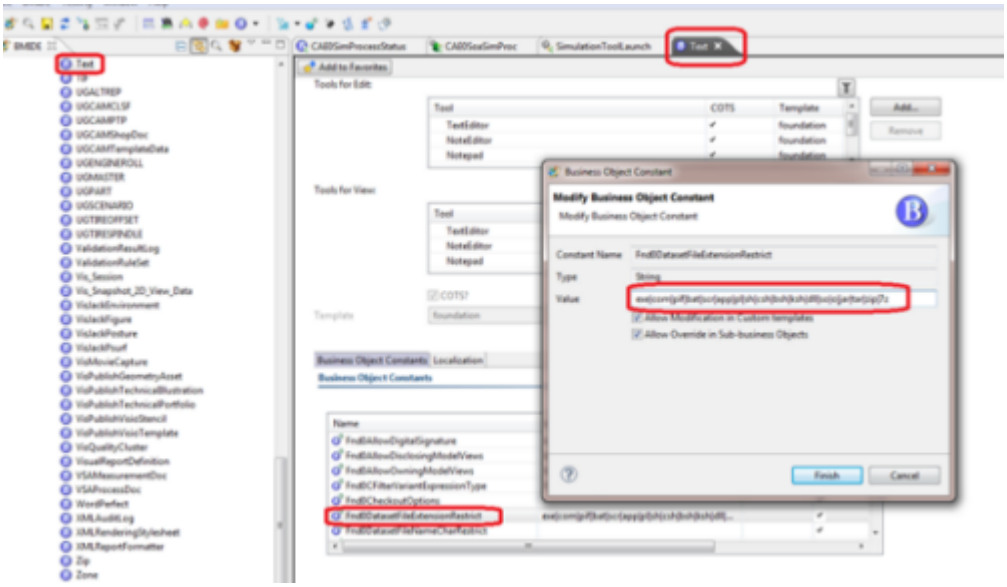
1. Start the Business Modeler IDE.

2. In the **Business Objects** folder, right-click the business object and choose **Open**.

   The constant appears in the **Business Object Constants** table on the **Main** tab.

3. In the **Business Object Constants** table, select the **Fnd0DatasetFileExtensionRestric** constant and click the **Edit** button.

4. In the **Modify Business Object Constant** dialog box, remove the **.bat** value from the **Value** box.



5. Click **Finish**.

6. To save the changes to the data model, choose **BMIDE→Save Data Model**, or click the **Save Data Model** button on the main toolbar.

7. Deploy the data model changes.

Configure simulation tools using sample scripts

While configuring simulation tools, you can specify a tool launch script. To create a tool launch script, you can refer to the sample scripts provided in the *TC_DATA***CAE_ESP_Sample_Files.zip** file. After you extract the compressed file, the Perl scripts for UNIX and Linux platforms and the batch files for the Windows platform are available from the **UNX** or **WNT** directories, respectively.

The following arguments are used in the sample scripts and they are listed alphabetically in the following table. You can refer the respective scripts for the exact sequence of arguments.

Argument	Description and values
set dataset_creation	Specifies the appropriate value for creating the dataset. Valid values: **As_Needed**, **Always**, or **Never**. Default value: **Always**. Example: **As_Needed**
set dataset_or_url_name	Specifies the input dataset or the URL name. Example: **002004/A**
set host_path	Specifies the host path. The host path is sent only in the case of local launch from Active Workspace. In all other launch types, the host path is null.
set input_file	Specifies the input file with the fully qualified path. Example: **D:\Scratch\ESP_Root_1426108216296\ESP_1426108216305\cad001.dat**
set input_xml_file	Specifies the input XML file. Valid values are as follows. • **Local Launch**: **-jobid** • **Local Detached Launch**, **Workflow Local Launch**, **Remote Launch**, **Workflow Remote Launch**, **Server Launch**, and **Workflow Server Launch**: **-inputXmlName** Examples: • **Local Launch**: **cae1429176454**

Argument	Description and values
	• All other types of launches: **D:\Scratch\ESP_Root_552f8c6c0000159000005a89\simprocess552f8c6c0000159000004717**
set item_creation	Specifies the appropriate value for creating the item. Valid values: **As_Needed**, **Always**, and **Never**. Default value: **Always**. Example: **As_Needed**
set item_id	Specifies the input item ID. Example: **002004**
set itemKeys_string_quotes_string	Specifies the input item key.
set rev_id	Specifies the input revision ID. Example: **A**
set security_token	Specifies the security token. You can use the security token instead of the password. However, it is valid only for certain duration.
set shared_staging_dir	Specifies the shared staging directory.
set tool_input_args	Specifies the tool input parameters concatenated with **?**. Default value is **noValue** Example: **-memory=ESTIMATE?-append=no?-scratch=yes?-Submit to Cluster=no**
set user_group	Specifies the user group of the logged in user.
set user_id	Specifies the user ID of the logged in user.
set user_staging_dir	Specifies the user staging directory.
set working_dir	Specifies the fully qualified working directory path.

Configure environment variables for the tool launch script

You can use the **Environment Variables** pane to define a set of environment variables and their associated values used for launching the script configured for the tool.

1. In CAE Manager, click the **CAE Configuration** toolbar menu, and choose **Simulation Tool Configuration**.

2. Click the **Environment Variables** tab.

3. In the **Environment Variables** pane, click the **Add** button, enter an environment variable name, and add the appropriate platform value (Windows or Linux) of the CAE authoring tool.

   If the Dispatcher module is installed on:

   - Linux: Linux and Windows platform must be the same as the Linux path.
   - Windows: Linux and Windows platform must be the same as the Windows path.

   Example:

   - **Name**: **SPLM_LICENSE_SERVER**
   - **Windows Platform Value**: **28000@***hostname*

Define input types and create input rules for the tool launch process

Define input types

1. In CAE Manager, click the **CAE Configuration** toolbar menu, and choose **Simulation Tool Configuration**.

2. Click the **Input Configuration** tab.

3. In the **Minimum Input** and **Maximum Input** boxes, type the minimum and maximum number of item revisions the process can accept as input.

4. Select a launch type.

   - **Single Launch** option to launch a single process regardless of the number of item revisions the user selects for input. Teamcenter passes a list of the selected item revisions to the process being launched.

     Select this option if analysts want to run the **CAE BOM Compare** report. This is useful for comparing files available in the same working directory. The **Multiple Launch** option exports files to different working directory for each input selected by the analyst and is not recommended for the launch of comparison tools.

     If you select this option, the simulation analyst *can* override the **User Staging Location** box while launching the simulation tool.

   - **Multiple Launch** option to launch multiple processes.

     For multiple processes, one process is selected for each item revision when the process is launched.

> Note:
>
> This option is applicable only if the **Maximum Input** value is greater than one.

If you select this option, the simulation analyst *cannot* override the **User Staging Location** box while launching the simulation tool.

5. (Optional) Select the **Export Materials** check box to define a **simulation tool for exporting material revisions**.

   This option is available only if Materials Management is installed.

6. Specify a primary input type.

   You can choose item revision types, datasets, or folders. You cannot configure a combination of these object types.

   The primary input types you include here are added to the primary type row options in the **Traversal Path** table for creating primary input rules and additional input rules.

   You can specify multiple dataset types or folder types as a primary input for a single simulation tool.

   If you choose datasets or folders, you cannot choose the **File Input** option and create traversal rules or define additional input rules using the file input method.

7. Specify a primary input rule by using the **file input method** or the **PLM XML export method**.

   You can create multiple primary input rules to download the named references of an item revision. Teamcenter evaluates the rules in the order listed. The first rule satisfied by locating an appropriate input file is chosen for the launch script and no more rules are evaluated. For example, you can create a rule for a primary item type such as **CAEAnalysisRevision** by defining a traversal path from the item revision to the solver-specific data deck (**CAESolver**) and from **CAESolver** to the file to be exported. Each of the rules, defined as primary input rules, is executed when the process is launched until a corresponding file is found. After a corresponding file is found, all remaining rules are ignored. Additionally, the file name is passed to the launch script as a parameter.

   > Note:
   >
   > You can add any number of **File Input** rules or **PLMXML Export** rules, but not *both* types of rules together.
   >
   > After you create file input rules, you can select multiple rules, copy, and paste them in the **File Output** box in the **Output Configuration** tab.

8. Specify an additional input rule by using the **file input method** or the **PLM XML export method**.

You can *optionally* create additional input rules to download named references of related item revisions to the **temp** directory. All of the rules, defined as additional input rules, are executed when the process is launched and all corresponding files are placed in the **temp** directory.

9. Specify optional inputs.

   While running preconfigured simulation tools, you (as a simulation administrator) cannot practically link all the optional inputs to the primary input item revision and predefine the traversal path for all the inputs that the simulation analyst wants to export in the tool configuration. For example, in addition to the results file (primary input), the simulation analyst may want to select results templates (optional inputs) and send them to a post processor. The results templates are not linked to the analysis item revision that holds the results file. In such cases, the simulation administrator can configure optional inputs to allow simulation analysts to select additional inputs such as results templates.

   The simulation administrator can configure item revisions for each optional input, the default value, the export directory, and the availability of the optional input in the simulation tool launched by the analyst by setting the visible value to **true**.

   During the tool launch process, the simulation analyst can select the desired item revision for each available optional input and use these templates to export only specific types of results.

   If you want to **configure style sheets for simulation tools launch**, the **SampleLaunchSimToolDialogHTML.html** sample HTML file includes the **Optional Inputs** section.

   a. Click the **Add** button in the **Runtime Inputs** area.

   b. Type a description for the runtime input.

      The description you specify is displayed in the simulation tool launched by the analyst.

   c. Select an item revision type.

      You can select only the specified type or any of its subtypes.

   d. To specify a default value for the runtime input, click the **Browse** button in the **Default Value** column and specify the search criteria.

      You can specify only objects of the selected item revision type or any of its subtypes (previous step).

   e. Specify an export directory for creating a directory within the **ESP** directory. For example, if you specify a **Results Template** directory, the system exports all the files under each optional input item revision to the **ESP\Results Template** directory.

      You can use the same directory for multiple optional inputs. If you do not specify a directory, the system exports the files to the **ESP** directory.

f. To make this runtime input item revision available in the simulation tool launched by the analyst, select **true** or **false** in the **Visible** column.

If you set it to **false**, the optional input is not displayed in the **Launch Simulation Tool** dialog box, but the system continues to export files from the configured item revision with the default value.

Define a primary input rule using the file input method

You can use the **File Input** option to:

- Specify an input rule (using a traversal rule) and specify dataset name patterns and file name patterns for the input.
- Define a relationship type at each step of the traversal rule to traverse to the next object if the secondary object is an item revision.
- Specify an input type and a file naming pattern for the input type.
- Create web link definitions to the result files in a shared network folder.

  In some cases, instead of storing large result files in the database, organizations prefer storing the generated result files on shared network folders. In such cases, the simulation administrator configures input rules for the system to create web links to the shared network folders.

1. Select the **File Input** option in the **Primary Input** box.
2. Click the **Add** button to open the **Primary Input Details** dialog box.
3. In the **Rule Name** box, type a rule name.
4. In the **Traversal Path** table, click the **Add** button to add a row and activate all options.
5. In the **Originating Type** column, click a cell and select an appropriate option. For example, select an item revision class such as **CAE 3D Model Revision**.

   The primary type options available here depend on the input types you have added in the **Primary Input Type** box.
6. In the **Relation** column, click a cell and select an appropriate option.
7. In the **Destination Object** column, click a cell and select an appropriate option.
8. In the **Relation Direction** column, click a cell and choose the appropriate option.

> Note:
>
> This option is available only if you choose the **Item Revision** option in the **Destination Object** column.

- **Primary to Secondary** indicates the path from an item revision to another and that the origination item revision is the primary contributor to the relationship.
- **Secondary to Primary** indicates the path from an item revision to another and that the secondary item revision class is the primary contributor to the relationship.

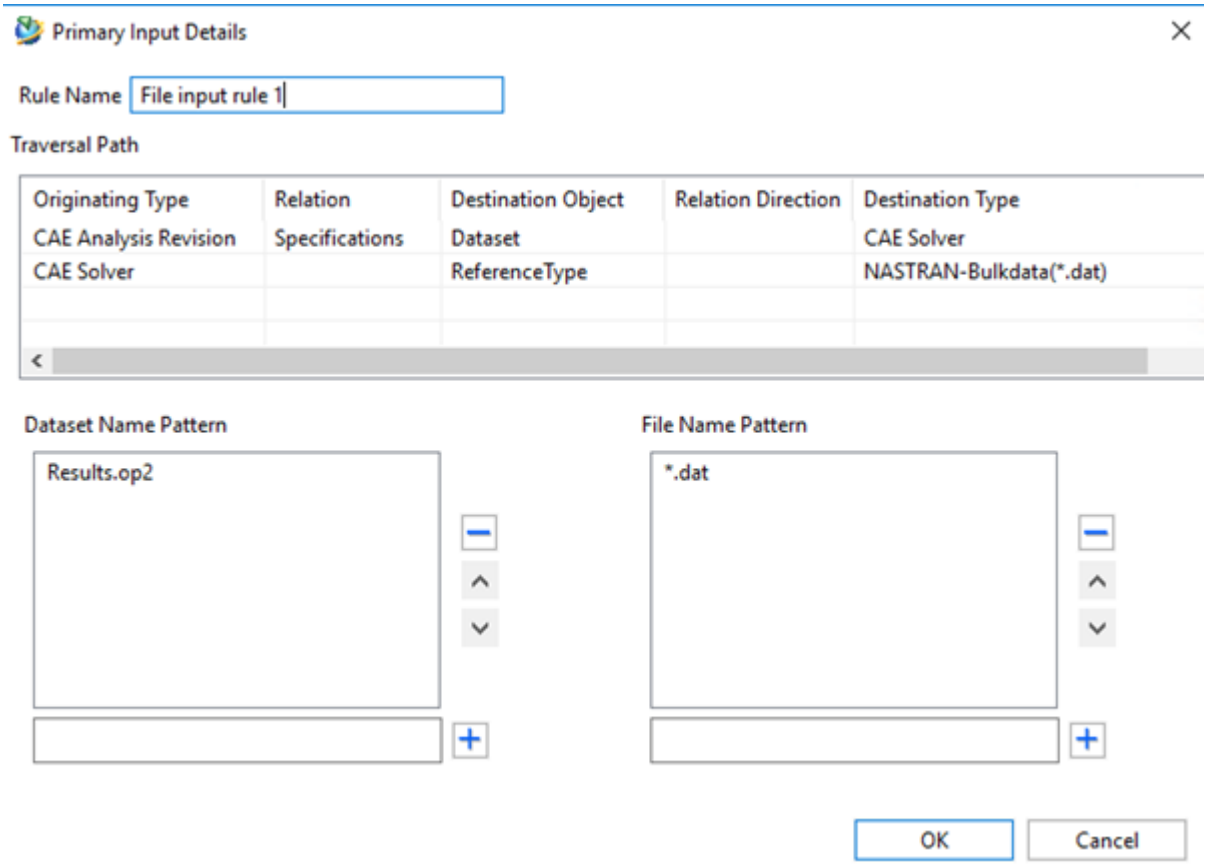
9. In the **Destination Type** column, click a cell and select an appropriate option. The options available here depend on the **Destination Object** type you have chosen in step 7.

   The **Destination Type** column, depending on the choices you have made in the previous columns, includes the file extension type in the menu and provides guidance to the simulation administrator for specifying the filename pattern in the **File Name Pattern** box.

10. Click the **Add** button to add another row.

    The **Originating Type** option in the second row is automatically populated with the value from the option you have selected in the **Destination Type** option from the first row.

    Primary rule example:



Traversal rule example for web link definition:

Originating Type	Relation	Destination Object	Relation Direction	Destination Type
CAE 3D Analysis Revision	**Specifications**	**Web Link**	None	**Web Link**

11. (Optional) In the **Dataset/URL Name Pattern** box, click the **Add** button to add a dataset name pattern, for example, *datasetname*, **datasetname*, or *datasetname**.

    You can use:

    - A constant string surrounded by double quotes, for example **"Input Deck"** or **"Results.op2"**.

      Tip:

      You can add spaces by including the constant string in double quotes.

    The system processes only datasets matching the exact string name.

- Wildcards and strings surrounded by double quotes, for example **"Input*"** or **"*Thermal*key*"**.

  The system treats them as a wildcard and processes the matching datasets.

- The **ITEMREVID** keyword.

  The system replaces the keyword with the **ITEMREVID** of the item revision that contains the dataset. You can use this keyword in combination with strings, for example, **ITEMREVID"-FEMap"**.

> Note:
>
> Keywords are not accepted.

Dataset/URL name pattern example for web link definition:

You can use a combination of wildcards and strings surrounded by quotes, for example, **ITEMREVID**.

12. In the **File Name Pattern** box, click the **Add** button to add a file name pattern you want to include. For example, ***.dat** or ***.op2**.

Define a primary input rule using the PLM XML export method

1. Select the **PLMXML Export** option in the **Primary Input** box.
2. Click the **Add** button to open the **PLMXML Export Details** dialog box.
3. In the **Rule Name** and **File Name** boxes, type a rule name and file name, respectively.
4. From the **Transfer Mode** menu, select an appropriate option.

> Tip:
>
> Traversal rules are not required to export the PLM XML file for the primary input object. However, you can configure traversal rules to **export the PLM XML file for the related object**.

Define additional input rules using the file input method

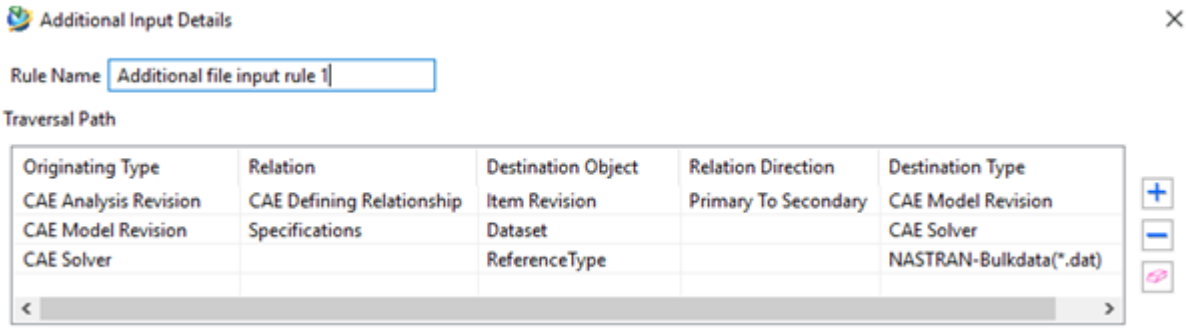
During the tool launch, if the input configuration has additional inputs, then all the named references related to the additional inputs are exported to the **ESP** folder. The **ESP_additional_inputs_list.txt** file contains a list of the exported files and the **ESP_*nn*_LOGS** directory contains the log files.

1. Select the **File Input** option in the **Additional Input** box.
2. Click the **Add** button to open the **Additional Input Details** dialog box.

3. (Optional) Create additional input rules.

   The procedure for creating additional input rules is similar to creating **primary input rules using the file input method**.

   Example of additional file input rules:



Define additional input rules using the PLM XML export method

During the tool launch, if the input configuration has additional inputs, all the named references related to the additional inputs and the top level PLM XML file are exported to the **ESP** folder. The **ESP_additional_inputs_list.txt** file contains a list of the exported files and the name of the top level PLM XML file. The **ESP_*nn*_LOGS** directory contains the log files.

1. Select the **PLMXML Export** option in the **Additional Input** box.

2. Click the **Add** button to open the **PLM XML Export Details** dialog box.

3. In the **Rule Name** and **File Name** boxes, type a rule name and a file name, respectively.

4. From the **Transfer Mode** menu, select an appropriate option.

5. In the **Traversal Path** table, click the **Add** button to add a row and activate all options.

6. In the **Originating Type** column, click a cell and select an appropriate option. For example, select an item revision class such as **CAEModelRevision**.

   The primary type options available here depend on the input types you have added in the **Primary Input Type** box.

7. In the **Relation** column, click a cell and select an appropriate option.

8. In the **Destination Object** column, click a cell and select an appropriate option.

9. In the **Relation Direction** column, click a cell and choose the appropriate option.

- **Primary to Secondary** indicates the path from an item revision to another and that the origination item revision is the primary contributor to the relationship.

- **Secondary to Primary** indicates the path from an item revision to another and that the secondary item revision class is the primary contributor to the relationship.

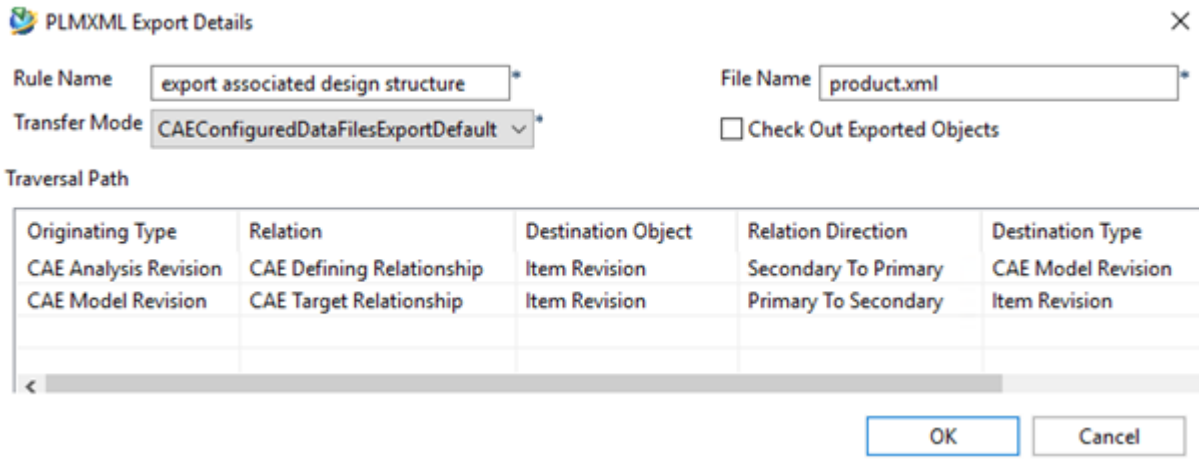
10. In the **Destination Type** column, click a cell and select an appropriate option. The options available here depend on the **Destination Object** type you choose.

    The **Destination Type** column, depending on the choices you have made in the previous columns, includes the file extension type in the menu and provides guidance to the simulation administrator for specifying the filename pattern in the **File Name Pattern** box.

11. Click the **Add** button to add another row.

    The **Originating Type** option in the second row is automatically populated with the value from the option you have selected in the **Destination Type** option from the first row.

    Example for exporting an associated design structure:



Specify output types and create output rules for the tool launch process

Specify output types and create output rules

You can use the **Output Configuration** pane to define the output for the launch process. Process output includes named references of datasets, datasets associated with item revisions, or item revisions created by Teamcenter. Simulation Process and Data Management imports suitable named references configured by you to the **temp** directory.

1. In CAE Manager, click the **CAE Configuration** toolbar menu, and choose **Simulation Tool Configuration**.

2. Click the **Output Configuration** tab.

3. Select one of the following options in the **Outputs** section:

   - **PLM/XML Import** option to specify a PLM XML file as the output for the launch process.

     This option allows the launch process to generate a PLM XML file. The PLM XML file is imported back into Teamcenter using PLM XML import procedures after the launch process is complete.

   - **File Output** option to add a **file output rule**.

4. Select the **Add** button in the **Output Item Name** section to **add an output item**.

5. In the **Create Items** section, select one of the following options:

   - **As Needed** to create items as needed when the tool is launched by the analyst.

     This is the default option.

   - **Always** to always create items when the tool is launched by the analyst.

   - **Never** to never create items when the tool is launched by the analyst.

6. In the **Create Datasets** section, select one of the following options:

   - **As Needed** to create datasets as needed when the tool is launched by the analyst.

     This is the default option.

   - **Always** to always create datasets when the tool is launched by the analyst.

   - **Never** to never create datasets when the tool is launched by the analyst.

   > Note:
   >
   > The analyst can override your selection for the options while launching the simulation tool.

7. (Optional) To allow analysts to select files to import to Teamcenter after successfully launching a simulation tool, check the **Selective Data Import** check box.

   > Note:
   >
   > This is applicable only for the **Local Launch** option.

8. To allow analysts to import only modified files to Teamcenter after running a simulation tool again, check the **Import Only Modified Files** check box.

9. To prevent data from being automatically uploaded and versioned when a tool launch is complete, you can specify file upload conflict options.

   After the tool launch is complete, based on the output rules, the system verifies each output file against existing files in the database. All new output files are uploaded to the database. For each file that has a similar file in the database, based on the file output rules, the system processes the files based on the following options:

   - **Prompt User** to open the **File Upload Conflicts** dialog if the same output file exists in the database. In such cases, the simulation analyst can select the **Upload**, **Rename and Upload**, or **Skip** option.
   - (Default option) **Upload** for the system to upload output files and create new revisions for existing files with the same name.
   - **Rename and Upload** for the system to automatically rename the output files with a suffix, if similar filenames exist, and upload them to the database.
   - **Skip** to avoid uploading the output files, if similar files exist in the database.

Define a file output rule

You can use the **File Output** option to:

- Specify an output (using a traversal rule) and specify dataset name patterns and file name patterns for the output.
- Define a relationship type at each step of the traversal rule to traverse to the next object if the secondary object is an item revision.
- Specify an output type and a naming pattern for the output type.
- Select an option to create items (as needed, always, or never) when the tool is launched by the analyst.
- Select an option to create datasets (as needed, always, or never) when the tool is launched by the analyst.
- Create web link definitions.

  In some cases, instead of storing large result files in the database, organizations prefer storing the generated result files on shared network folders. In such cases, the simulation administrator configures output rules for the system to create web links to result files in the shared network folders.

This option allows the process to generate one or more standard files. These files are uploaded back into Teamcenter as named references of defined datasets after the launch process is complete. In the case

of web link definitions, the files are not imported back to Teamcenter. Instead, the web link object is created or updated with the URL of the result file.

> Note:
>
> This procedure is applicable only if you select the **File Output** option in the **Output Files** pane.

1. Select the **File Output** option.
2. Click the **Add** button to open the **File Output Details** dialog box.
3. In the **Rule Name** box, type a rule name.
4. In the **Traversal Path** table, click the **Add** button to add a row and activate all options.
5. Specify a traversal path as follows:

   Example: Output rule (primary to secondary)

Originating Type	Relation	Destination Object	Relation Direction	Destination Type
CAE 3D Analysis Revision	**CAE Result Specifications**	**Item Revision**	**Primary to Secondary**	**CAE 3D Result Revision**
CAE 3D Result Revision	**Specifications**	**Dataset**		**CAE 3D Result**
CAE 3D Result	None	**ReferenceType**		**Nastran_results_log** (***.f06** file to be exported to the **temp** directory)

   Example: Output rule (secondary to primary)

Originating Type	Relation	Destination Object	Relation Direction	Destination Type
CAE 3D Model Revision	**CAE Defining**	**Item Revision**	**Secondary To Primary**	**CAE 3D Analysis Revision**
CAE 3D Analysis Revision	**Specifications**	**Dataset**		**CAE 3D Result**
CAE 3D Result	None	**ReferenceType**		**Nastran_results_log** (***.f06** file to be exported to the **temp** directory)

Traversal rule example for web link definitions:

Originating Type	Relation	Destination Object	Relation Direction	Destination Type
CAE 3D Model Revision	**Specifications**	**Web Link**		**Web Link**

6. In the **Dataset/URL Name Pattern** box, click the **Add** button to add a dataset name pattern you want to include.

> Tip:
>
> You can create multiple dataset naming patterns.

You can define dataset naming patterns as a:

- Constant string surrounded by quotes, for example, **"NXNastran"**.
- Combination of wildcards and strings surrounded by quotes, for example, ***"FEMap"*"Report"***.
- Keyword: **ITEMREVID**, **FILENAME**, and **BASEFILENAME**.

  If the simulation administrator configures the dataset name to this keyword:

  - **ITEMREVID**: Teamcenter places the output file in a dataset name that is identical to the **ITEMREVID** of the item revision that contains the dataset. For example, Teamcenter places the **000987/A** file in the **000987/A;1-***SampleItem* dataset.
  - **FILENAME**: Teamcenter places the output file in a dataset name that is identical to the full file name. For example, Teamcenter places the **excavator.op2** file in the **excavator.op2** dataset. (The file name path is not included in the dataset name.)
  - **BASEFILENAME**: Teamcenter places the output file in a dataset name that is identical to the file name without its extension. For example, Teamcenter places the **excavator.op2** file in the **excavator** dataset. (The file name path is not included in the dataset name.)
- Counter keyword, for example, **NNNN** for a 4-digit counter.

> Note:
>
> You can add spaces by embedding them in quotes, for example, **ITEMREVID" "NNNN**. If you add spaces without embedding them in quotes, Teamcenter displays an invalid naming pattern error.

You can define dataset naming patterns by using any combination of the above patterns. Exceptions are as follows:

- If the file extension pattern is not matched or specified, Teamcenter does not export any of the files.
- Counter keyword and **ITEMREVID** can only appear once in the naming pattern.
- Any unquoted series of characters containing something other than keywords are interpreted as a single quoted string to support earlier versions.

Valid naming pattern examples	Invalid naming pattern examples
ITEMREVID”-FEMap”	**“FEMap”ITEMREV**. When quoted strings exist, all characters outside quotes must be part of a keyword.
“FEMap-“NNN	**ITEMREVID”FEMap” NNN**. No spaces are allowed except inside quotes.
NNN”-FEMap”	**ITEMREVID”-“ITEMREVID** **ITEMREVID** keyword cannot appear more than once.
”FEMap-“NNNNNN	**NNN”-“NNN** Counter keyword cannot appear more than once.
ITEMREVID”-“NNNN	**nnn”-FEMap”** Keywords are case sensitive.
ITEMREVIDNNNNN	
”FEMap”*”-“ITEMREVID	
***”FEMap 10”NNN**	
***”FEMap 10 “*NNN**	
“FEMap-“NNN”-10”	
NXNASTRAN. This is considered the same as **“NXNASTRAN”**.	
FEMAPNNITEMREVID. This is considered the same as **“FEMAPNNITEMREVID”**. However, the keywords in this example are not recognized.	

Dataset/URL name pattern example for web link definitions:

You can use a combination of wildcards and strings surrounded by quotes, for example, **ITEMREVID"_"FILENAME"_"BASEFILENAME**.

7. In the **File Name Pattern** box, click the **Add** button to add a file name pattern you want to include, for example ***.op2**.

8. Click **OK** to create a file output rule.

Define an output item

You can optionally define an output item to import the file back to Teamcenter.

1. Click the **Add** button in the **Output Item Name** box to select an output item name.

2. To specify a naming pattern, click **Modify** and type a value in the **Name Pattern** box.

   - "*constant-name*_"**ITEMNAME**, for example **"femap_"ITEMNAME**.
   - "*constant-name*_"**ITEMID**, for example **"femap_"ITEMID**.

   ITEMNAME and **ITEMID** are the valid keywords for **ItemRevision**.

3. To allow the analyst to override the naming pattern you have specified, select the **Allow run time modification** check box.

4. Specify the maximum file size in the **Advanced Output Rules** section.

   When this option is specified and a simulation analyst launches the tool, the system imports the file bigger than the maximum file size as an URL. The system creates the URL (weblink) with the specification relationship to the item revision as per the configured output dataset rule for files bigger than the maximum file size. If the dataset naming pattern is not provided, then the name of the URL object is as per the file name.

   This option is not supported for remote launch methods. It is enabled only if the **Use Staging Directory** is selected from the **Launch Properties** tab and file output rules are defined.

Specify input parameters and HPC scheduling parameters for the tool launch script

You can use the **Input Parameter** pane to define the parameters that are passed to the launch script when the launch process is used by an analyst. These parameters can be static or dynamic. Dynamic parameters can either be extracted from the object attributes or the analyst can enter them at run time while launching the process.

You can also include scheduling parameters for HPC profiles.

1. In CAE Manager, click the **CAE Configuration** toolbar menu, and choose **Simulation Tool Configuration**.

2. Click the **Input Parameter** tab.

3. Define static or dynamic input parameters for the tool launch script.

   a. Click the **Add** button to add a row and activate all the options in the table.

   b. In the **Name** column, click a cell and specify a name for the input parameter.

      This name is used when passing values to the launch script at runtime when configured to pass **Name value pair**. It is also used to label the field on the launch script dialog when the default launch dialog is displayed.

   c. In the **Type** column, click a cell and select the type from the list.

      Use **String**, **Double**, **Boolean**, or **ListofValues** to include runtime parameters.

      For example, if you specify a parameter such as memory for **ListofValues**, you can specify valid values such as 100, 200, and 300 and select 100 as the default. When the tool is launched, the analyst can choose any of these values and 100 is the default value as set in this example.

      Use **Item**, **ItemRevision**, or **Form** to include dynamic parameters. Dynamic attributes can be extracted from the object attributes.

   d. (Optional) Select the **Runtime Parameter** column, and click a cell to select **true** or **false** values.

      If you select **true**, analysts can change the parameter values at run time if a tool launch dialog is displayed and appropriately configured to accept this input. The default tool launch dialog provides the capability to enter this information.

   e. In the **Script Input Option** column, click a cell and select an appropriate option from the list.

      If a tool is configured with a parameter using the **Name value pair** script input option, the parameter is passed to the launch script as **Name=Value** when the process is launched.

      If a tool is configured with a parameter using the **Value only** script input option, only **Value** is passed as a parameter to the launch script when the process is launched. In this case, the script relies on the order of the parameters passed to interpret their meanings.

      If a tool is configured with a parameter using the **Name only** script input option, only **Name** is passed as a parameter to the launch script when the process is launched.

   f. In the **Object Type** column, click a cell and select an appropriate option.

   g. In the **Value** column, click a cell and type an appropriate option.

      The value entered here is provided as a default value for runtime parameters, and is the provided value to the launch script for non-runtime parameters.

Analysts can use **Edit** button in this field to enter the allowable values for both **List of Values** and **Boolean** types. For **Boolean** types, only the first two entered values are considered.

In the default launch dialog, the **List of Values** fields are presented as a dropdown box and the **Boolean** fields are presented as a check box. The checked state corresponds to the first value in the **Boolean** definition.

4. Include **scheduling parameters for HPC profiles**.

   An administrative user with DBA privileges identifies business needs for high performance computing (HPC) configurations for specific simulation tools. Based on the needs, an HPC configuration is created that includes HPC connections and HPC profiles. The scheduling parameters specified for the HPC profile can be imported to the **Input Parameter** tab of the simulation tools for which HPC configuration is enabled.

   a. To include HPC scheduling parameters, choose the **Load Parameters from HPC Profile** button and click **Yes** to load the standard scheduler parameters.

      The system appends these parameters to the existing columns in the **Input Parameter** tab.

   b. (Optional) To discard the scheduling parameter changes, click the **Discard and Reset changes current tab** toolbar button.

   c. To save the scheduling parameter changes, click the **Save Simulation Tool** toolbar button.

Set attributes on items, item revisions, and forms for the tool launch process

You can set attributes on an **Item**, **ItemRevision**, or **Form**. The simulation analyst launches simulation tools and the launch tool provides values for the attributes configured by the simulation administrator. The process is as follows:

- The administrator specifies an attribute value transfer file.

  Teamcenter creates this file in the working directory of the launch tool. This file contains a list of configured attributes.

- The administrator sets attributes by creating a traversal rule and specifying the object attributes.

- The analyst launches the simulation tool.

  The simulation tool updates the attributes in the attribute value transfer file.

  The analyst can also edit the attribute XML file manually to specify attribute values as appropriate. Teamcenter creates this file in the working directory of the launch tool.

  A **schema file is used to validate the attribute XML file** in the tool launch process.

Teamcenter reads the attributes in the attribute value transfer file and applies the values to the configured Teamcenter objects using the traversal path.

1. In CAE Manager, click the **CAE Configuration** toolbar menu, and choose **Simulation Tool Configuration**.

2. Click the **Attribute Configuration** tab.

3. In the **Attribute Transfer File Name** box, specify a file name.

   Teamcenter appends **.xml** to the file name if the extension is not specified and creates this file in the working directory of the launch tool.

4. Specify a traversal rule for specifying that attributes.

   a. Click the **Add** button in the **Attributes** box to open the **Attribute Details** dialog box.

   b. In the **Rule Name** box, type a rule name.

   c. In the **Traversal Path** table, click the **Add** button to add a row and activate all options.

   d. In the **Originating Type** column, click a cell and select an appropriate option. For example, select an item revision class such as **CAEModelRevision**.

      The primary type options available here depend on the input types you have added in the **Primary Input Type** box in the **Input Configuration** pane.

   e. In the **Relation** column, click a cell and select an appropriate option.

   f. In the **Destination Object** column, click a cell and select an appropriate option.

   g. In the **Relation Direction** column, click a cell and choose one of the following options:

      These options are available only if you select the **ItemRevision** attribute in the **Destination Object** column.

      - **Primary to Secondary** to indicate the path from an item revision to another and that the origination item revision is the primary contributor to the relationship.

      - **Secondary to Primary** to indicate the path from an item revision to another and that the secondary item revision class is the primary contributor to the relationship.

      Traversal rules allow you to define a path from the input item revision to an item revision that ultimately holds the various output files. The **Relation Direction** option allows you to indicate that the path from one item revision to another item revision is through a specified

relationship and that the originating item revision is the primary contributor or the secondary contributor to the relationship.

If no relationship type is specified for traversing from one item revision to another, the system considers any item revision class of the specified type and attaches it to the item revision regardless of the relationship type used to attach it.

h. In the **Destination Type** column, click a cell and select an appropriate option. The options available here depend on the **Destination Object** type you have chosen in the previous step.

i. (Optional) Click the **Add** button to add another row.

The **Originating Type** option in the second row is automatically populated with the value from the option you have selected in the option from the first row.

5. Specify object attributes.

a. Click the **Add the selected type/property to the list** button to populate the list of attributes in the **Object Attributes** box.

b. In the **Object Attributes** box, click the **Add** button to add the selected attribute (for example, **Name** or **Location Code**) to the **Selected Attributes** box.

c. Click **OK** to complete adding attribute details in the **Attribute Details** dialog box.

Schema attributes for validating the attribute transfer XML file

The analyst can edit the attribute XML file manually to specify values as appropriate. Teamcenter creates this file in the working directory of the launch tool. This file contains a list of configured attributes.

A schema file is used to validate the attribute XML file in the tool launch process. The **tcsim_tool_attribute_xfer.xsd** schema file is located in the *TC_DATA* directory.

The attribute values are updated in Teamcenter if the XML file is valid. If the XML file is not valid, the system logs the errors in the log file and the attribute values are not updated in Teamcenter.

The schema marks the **name** and **value** attributes as **required**. Other attributes such as **max_length** are marked as **optional**. This means that any file specified in the attribute transfer configuration that does not have the required attributes of **name** and **value** results in an error that is logged in the syslog and the attribute value is not updated through the tool launch.

- **Example 1: Valid XML file**

  The following is an example of a valid XML file that has attributes with values. In addition, it has the **max_length** attribute that specifies the maximum number of characters to be included.

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<attribute_config>

  <AttributeConfigName name="AttributeRule1">
    <attribute max_length="128" name="object_name"
value="Modified_Object_Name"/>
    <attribute max_length="240" name="object_desc"
value="Modified_Object_Description"/>
  </AttributeConfigName>

</attribute_config>
```

- **Example 2: Valid XML file**

  The following is an example of a valid XML file that has attributes with no values. In addition, it has the **max_length** attribute that specifies the maximum number of characters to be included.

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<attribute_config>

  <AttributeConfigName name="AttributeRule1">
    <attribute max_length="128" name="object_name" value=""/>
    <attribute max_length="240" name="object_desc" value=""/>
  </AttributeConfigName>

</attribute_config>
```

- **Example 3: Valid XML file**

  The following is an example of a valid XML file that has attributes with values. However, it does not have the **max_length** attribute that specifies the maximum number of characters to be included.

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<attribute_config>

  <AttributeConfigName name="AttributeRule1">
    <attribute name="object_name" value="Modified_Object_Name"/> /* note
the missing max_length */
    <attribute name="object_desc" value="Modified_Object_Description"/>
  </AttributeConfigName>

</attribute_config>
```

- **Example 4: Invalid XML file**

  The following is an example of an invalid XML file where the attribute value is not available.

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<attribute_config>

  <AttributeConfigName name="AttributeRule1">
    <attribute max_length="128" name="object_name" /> /* the value attribute
is not available */
    <attribute max_length="240" name="object_desc"
value="Modified_Object_Description"/>
  </AttributeConfigName>

</attribute_config>
```

Define a notification list for tool launch completion

The **Feedback** pane is used to create a notification list about the status of processes and remove or retain temporary files created by processes.

> Note:
>
> Temporary files are removed for local launch only. Simulation Process and Data Management uses Dispatcher Server components to launch the simulation processes remotely. Dispatcher Client has it own mechanism to cleanup temporary files. You can configure this using the **RequestCleanup** properties in the **Service.properties** file in the **DispatcherClient\conf** directory.

1. In CAE Manager, click the **CAE Configuration** toolbar menu, and choose **Simulation Tool Configuration**.

2. Click the **Feedback** tab.

3. Select the appropriate option for removing temporary files.

   Temporary files are created in the **temp** you have specified.

   - **Always** to always remove temporary files after completion of the process.
   - **Never** to never remove temporary files.
   - **On Success** to remove temporary files only if the process is successful.

4. To notify an invoking user about completion of the tool execution, select the **Notify Invoking User** check box. (This check box is selected by default.)

> Note:
>
> Clearing this check box only turns off notification of the invoking user. If other notifications are configured using the **User Notification List** or the **Group Notification List**, they continue to be honored.

5. In the **User Notification List** box, select the appropriate users to send notifications.

6. In the **Group Notification List** box, select the appropriate users to send notifications.

Edit the launch script for minor changes

You can edit the launch script for minor changes after you have uploaded the dataset containing the script.

Procedure

1. Click the **Launch Script Editor** tab.

   You configure launch properties to specify a launch method and launch script location in the Launch Properties tab.

   You can make small changes to the launch script after you have selected a dataset and uploaded it without having to check it out, download the script, make the change, and upload it again.

2. Select the appropriate operating system such as Unix or Windows for the configured script.

3. Select the appropriate launch script you want to edit if multiple files are available.

   The launch script is displayed in an editable box.

4. Make the required changes and click the **Save Simulation Tool** view toolbar button.

   You can use standard Windows commands such as **Ctrl + X** to cut selected text and save it to the clipboard and **Ctrl + V** to paste from the clipboard while making changes.

5. (Optional) To discard the changes, click the **Discard and Reset changes current tab** view toolbar button.

Enable or disable access to simulation tools

The simulation administrator or a user with DBA privileges can set access control. By enabling or disabling access, you can control access to simulation tools depending on the user, group, or role.

- If you enable or disable access to a group, the configuration applies to users whose context is set to that group.
- If you enable or disable access to a user or role, then that configuration applies to the user regardless of group or role, or applies to role regardless of group.

1. In CAE Manager, click the **CAE Configuration** toolbar menu, and choose **Simulation Tool Configuration**.
2. Click the **Access Control** tab.
3. Select one of the following options:
   - **Enable Access** to allow access for the selected users, groups, or roles to simulation tools; that is, access is disabled for those users, groups, or roles who are not listed.
   - **Disable Access** to disallow access for the selected users, groups, or roles to simulation tools; that is, access is enabled for those users, groups, or roles who are not listed.
4. Select one of the following options:
   - To allow access, select the desired users, groups, or roles from the **Organization** list and click the **Add** button to add them to the **Access** list.
   - To disallow access, select the desired users, groups, or roles from the **Access** list and click the **Remove** button to move them to the **Organization** list.
5. Click **OK** to save the configuration.

Run the quick set up script to include preconfigured simulation tools

As a simulation administrator, you can run the quick set up script to include preconfigured simulation tools.

Procedure

1. Set up the sample configurations.

   You must set up the **sample configurations by running the tcsim_quick_setup.pl script** to include preconfigured simulation tools.

   a. From the Teamcenter command prompt, browse to the *TC_DATA***tcsim\setup** directory.

   b. To set up sample configurations, run the following command:

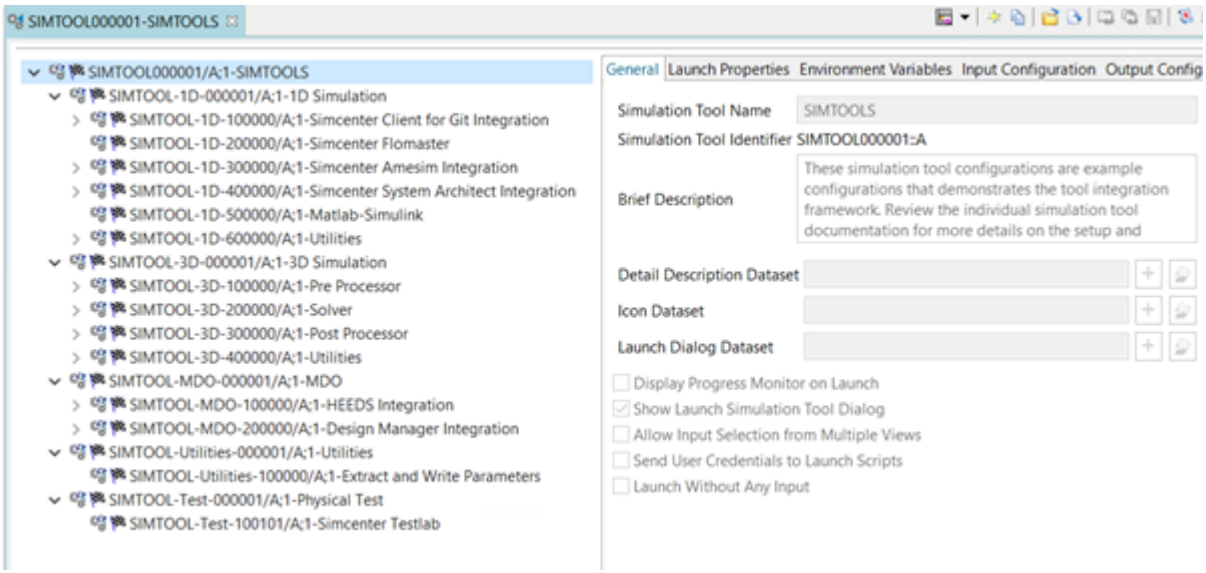
tcsim_quick_setup.pl -u=*DBAUserName* **-p=***DBAUserPassword* **-pf=***DBAUserPasswordFile*

> Note:
>
> These are sample configurations and you should modify them as per your requirements.

2. View the preconfigured simulation tools.

   a. Log on to Teamcenter rich client as an administrative user.

   b. In CAE Manager, click the **CAE Configuration** toolbar menu, and choose **Simulation Tool Configuration**.



   c. Select a preconfigured simulation tool and click the **Launch Properties** tab.

      The **Dataset Specification** section contains some scripts.

      By default, these files are available from the *TC_DATA\tcsim\example_configs\SimulationTool\SIMTOOL000001* directory.

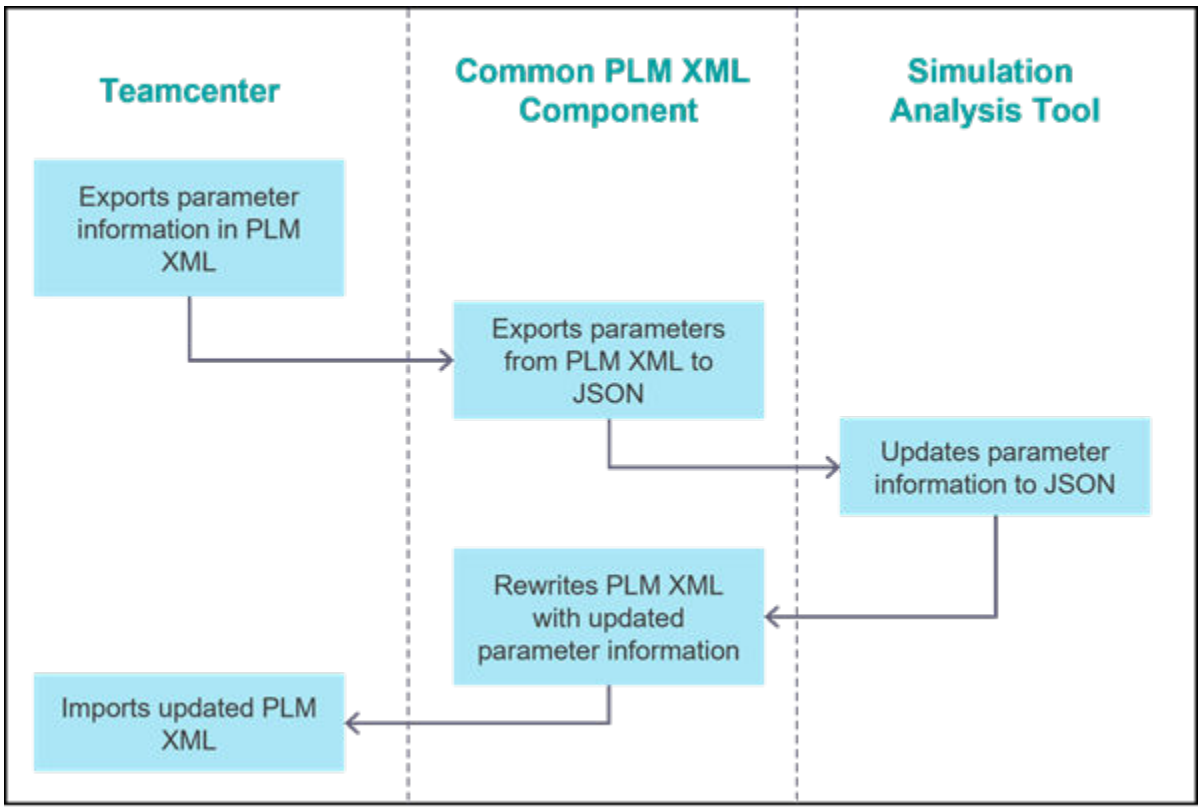
For more information see the video on how to configure simulation tools in **Why configure simulation tools?**

Configure a simulation tool to extract data for Simcenter Flomaster

Simulation analysts can launch preconfigured simulation tools such as preprocessors, solvers, postprocessors, and other tools. During the simulation tool launch, Teamcenter generates a PLM XML file and the data from it is used by the simulation analysis tools as the input to create the model. The

simulation analyst performs the analysis, and the system sends back the results by updating the PLM XML file. The results are imported back to Teamcenter from the PLM XML file.

All simulation tools do not have the capability to read and modify the PLM XML data. The conversion of MBSE parameters information from PLM XML is now done using JSON.



Simulation analysts can now use the default Simulation Flomaster tool to extract the PLM XML data to JSON for managing 1D data. For more information, see Process flow for managing 1D data in Simcenter Flomaster.

As a simulation administrator, you can set up sample configurations by running the **tcsim_quick_setup.pl script**. The sample simulation configurations include tools such as **Simcenter Flomaster**. This is the default tool for capturing model and analysis data for integrations such as Simcenter Flomaster.

After running the quick setup script, in CAE Manager, on the main toolbar, choose **CAE Configuration** → **Simulation Tool Configuration**. The **SIMTOOLS** configuration view displays the default preprocessors, solvers, postprocessors, and other utilities. Navigate to the **Simcenter Flomaster** tool. You can **modify this tool** as per your requirements.

This tool supports **Create Model**, **Update Model**, **Create Analysis**, and **Execute Analysis** launch types.

The **Simcenter Flomaster** tool imports and exports the **Study** tab properties in Active Workspace. Depending on the feature type installed at your site, the source type for importing and exporting

is different. If the **Simulation Process Management** feature is installed, the source type is **ATTRIBUTEXML**, whereas if the **Simulation Process Management with Parameter Management** feature is installed, the source type is **PLMXML_MBSE**.

For more information about installing simulation features, see **Install Simulation Process and Data Management features using TEM** or **Install Simulation Process and Data Management using Deployment Center**.

When the **Simulation Process Management with Parameter Management** feature is installed, the **Simcenter Flomaster** tool uses the JSON format for importing the MBSE properties to the **Study** tab in Active Workspace.

The JSON schema is available in the *TC_DATA***tcsim\example_configs\SimulationTool\SIMTOOL000001\Cae1DJsonSchema.json** directory. It supports primitive data types such as double, integer, string, boolean, and point for MSBE parameters. Currently, there is no validation available for the JSON file.

The JSON example is available in the *TC_DATA***tcsim\example_configs\SimulationTool\SIMTOOL000001\Cae1DJsonExampleFile.json** directory.

The PLM XML component of the tool generates the JSON file in the following format:

```
{
"NewParameter": "Y",
"Direction": "output",
"ParameterType": "Double",
"ParameterID": "",
"ParameterName": "DoubleParam",
"ParameterDescription": "Pwewrwerwer",
"Goal": "20",
"Maximum": "25",
"MaximumOperator": "",
"Minimum": "15",
"MinimumOperator": "",
"Resolution": "",
"InitialValue": "12",
"ValueType": "",
"Formula": "",
"Methodology": "",
"SourceType": "",
"MeasuredValue": "16",
"ProductId": "033760",
"ProductType": "CAE01DAnalysis",
"ProductRevisionName": "1dAnalysis001",
"ProductToParameterRelationType": "Att0HasParamValue",
"ParameterThreadName": "P004928",
"ParameterDefinitionName": "ATM_AttrDef_Double",
```

```
"ParameterDefinitionRevision": "A",
"AccessReferencePLMXMLId": "id3",
"ParameterRelationPLMXMLId": "",
"ProductRevisionPLMXMLId": "id2",
"ParameterThreadRefPLMXMLId": "",
"ParameterDefinitionRefPLMXMLId": ""
}
```

Define Simcenter Flomaster tool

As a simulation administrator, you can define the **Simcenter Flomaster** tool.

Procedure

1. Set up the sample configuration.

   You must set up the **sample configurations by running the tcsim_quick_setup.pl script** as a prerequisite for making the **Simcenter Flomaster** tool available.

   a. From the Teamcenter command prompt, browse to the *TC_DATA***tcsim\setup** directory.

   b. To set up sample configurations, run the following command:

      tcsim_quick_setup.pl -u=*DBAUserName* **-p=***DBAUserPassword* **-pf=***DBAUserPasswordFile*

2. Define the simulation tool for extracting the data for Simcenter Flomaster.

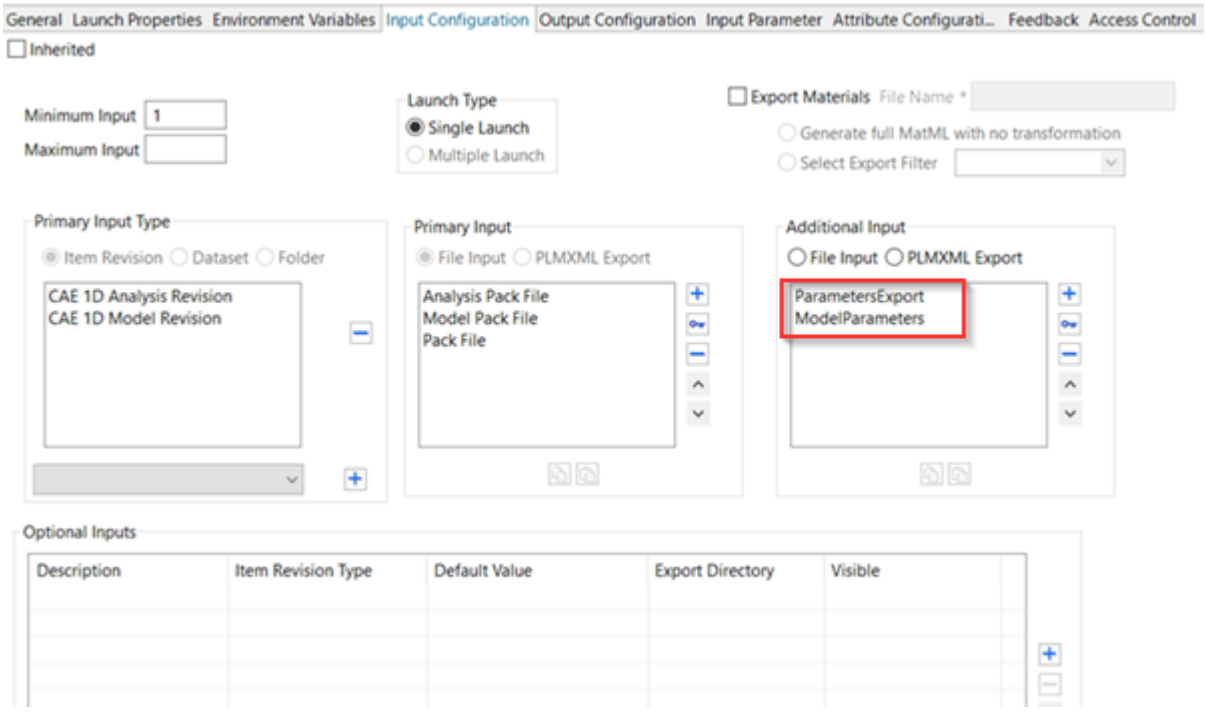
   a. In CAE Manager, click the **CAE Configuration** toolbar menu, and choose **Simulation Tool Configuration**.

   b. From **SIMTOOLS→SIMTOOL 1D Simulation**, select **Simcenter Flomaster**. This is the default tool used to extract data for Simcenter Flomaster.

   c. Choose the **Input Configuration** tab.

      The **Additional Input** section has two rules defined for exporting parameters:

      - **ParametersExport** to export the MBSE parameters.
      - **ModelParameters** to export the model parameters when the tool is launched using **CAE 1D Analysis** revision.

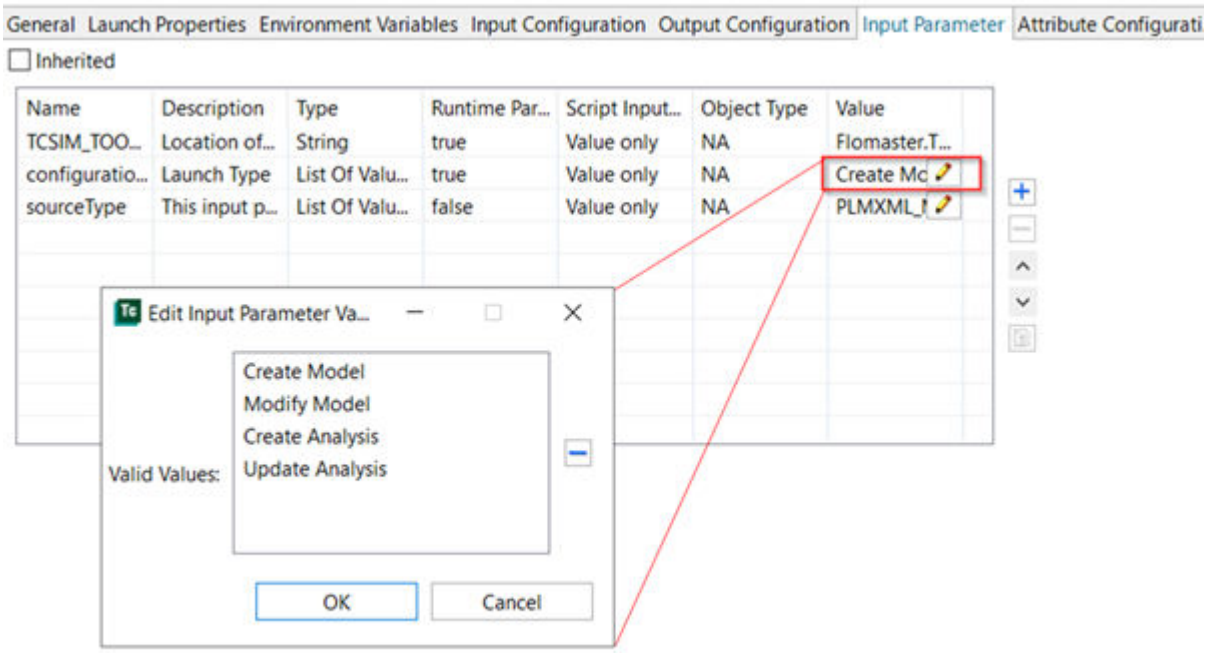


d. Choose the **Output Configuration** tab.

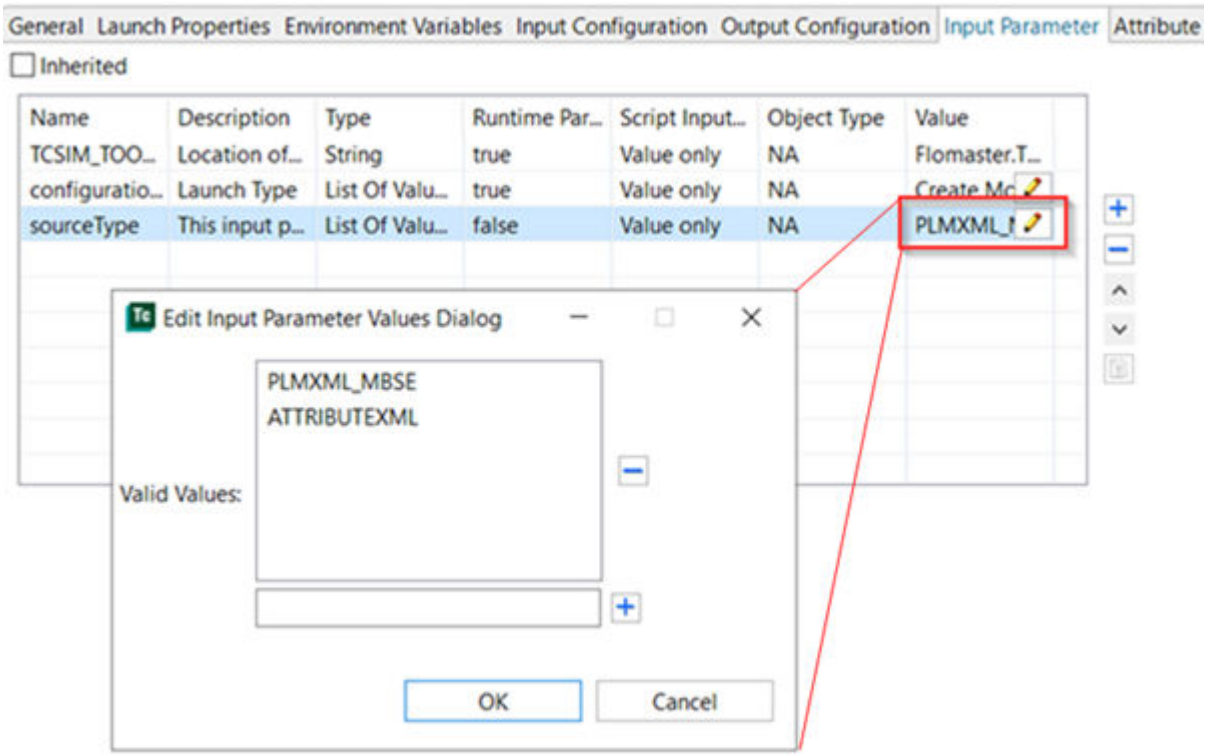
   This includes the **ImportParameterXML** rule to import the parameter XML file.

e. Choose the **Input Parameter** tab.

   The configuration type input parameter has four values: **Create Model**, **Modify Model**, **Create Analysis**, and **Update Analysis**.



The **sourceType** input parameters decides the type of parameter that is being exchanged, that is, **PLMXML_MBSE** for MBSE parameters or **ATTRIBUTEXML** for table row parameters.



Configure a simulation tool to extract the KPI values automatically

As a simulation administrator, you can set up sample configurations by running the **tcsim_quick_setup.pl** script. The sample simulation configurations include the **Extract KPI from Result** tool. This is the default tool for extracting KPI values from industry standard CAE results formats such as OP2 and BUN files automatically. It uses the NX Open utility to load the result files into Simcenter 3D Results Viewer before performing the extraction. The results from the NX Open utility are then populated in the **KPI** table.

The following is an example based on the default simulation tool used to extract KPI values. As a prerequisite, you must set up the **sample configurations by running the tcsim_quick_setup.pl script**.

After the simulation analyst launches the preconfigured simulation tool, the system exports the results files, an attribute XML file containing the KPI table attributes, the launch script, and the NX Open utility to the staging directory. The system also creates the **KPITemplateFile** subdirectory to which it exports the template XML file. The template XML file is an input XML file containing information about the KPI values that need to be extracted. The NX Open utility reads the template file before it performs the extraction. The system then calls the launch script, which internally calls the NX Open utility with the path to the results file, the path to the template XML file and the path to the output directory path as arguments. The NX Open utility extracts the required KPI values from the results file based on the inputs provided in the template XML file and generates the output KPI text file. The system then updates the KPI table in the attribute XML file and populates the **KPI** table of the analysis revision.

Set up the sample configuration

You must set up the **sample configurations by running the tcsim_quick_setup.pl script** as a prerequisite for making the **Extract KPI from Result** tool available.

1. From the Teamcenter command prompt, browse to the *TC_DATA***tcsim\setup** directory.

2. To set up sample configurations, run the following command:

   tcsim_quick_setup.pl -u=*DBAUserName* **-p=***DBAUserPassword* **-pf=***DBAUserPasswordFile*

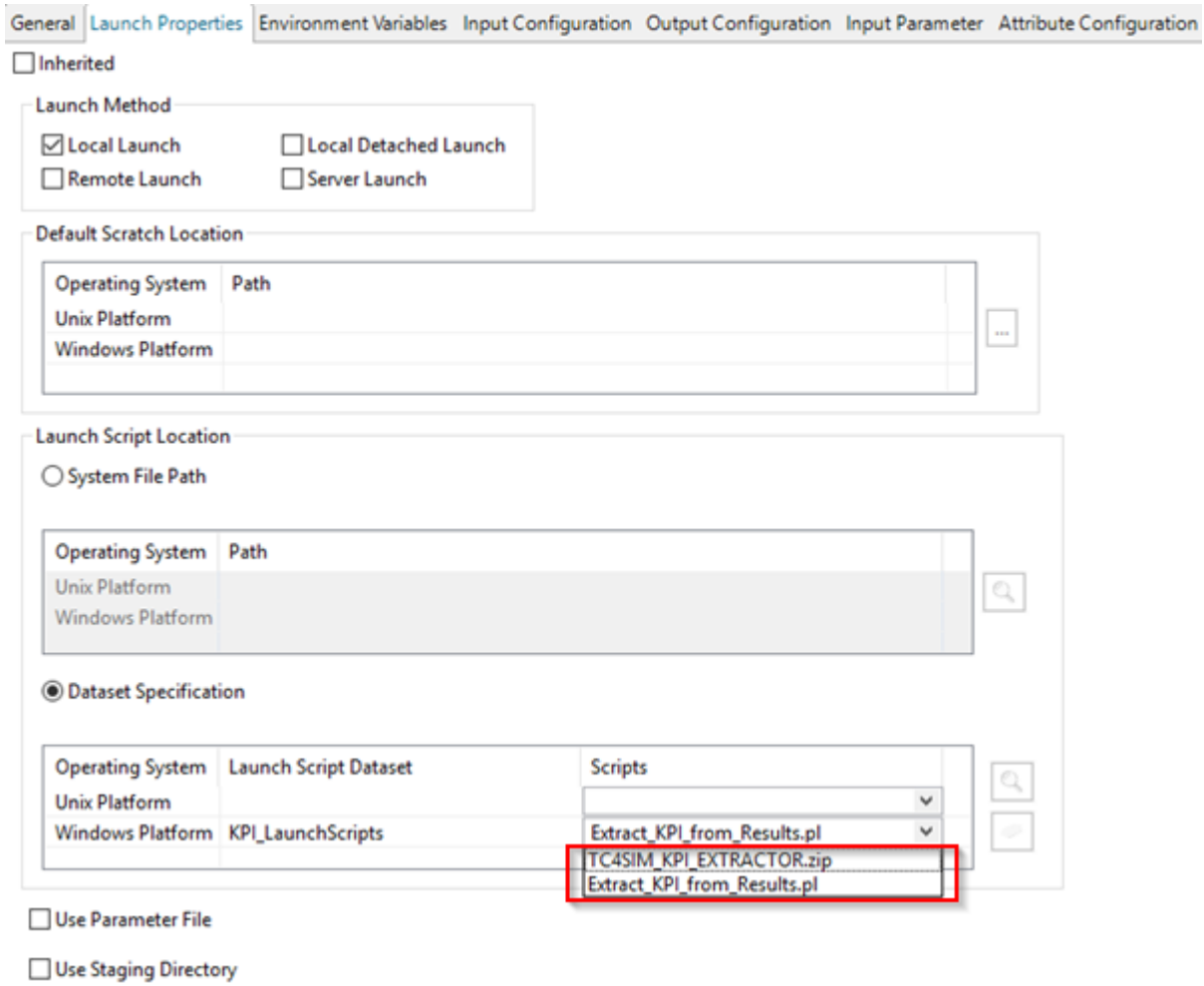
Define the simulation tool for extracting the KPI values automatically

1. In CAE Manager, click the **CAE Configuration** toolbar menu, and choose **Simulation Tool Configuration**.

2. Choose **SIMTOOLS→SIMTOOL-3D Simulation→SIMTOOL-3D Utilities→Extract KPI from Result**. This is the default tool used to extract KPI values.

3. In the **Launch Properties** tab, the **KPI_LaunchScripts** dataset is set to the **Extract_KPI_from_Results.pl** script and the **TC4SIM_KPI_EXTRACTOR.zip** compressed file that contains the NX Open utilities, including the documentation for these utilities.

By default, these files are available from the *TC_DATA\tcsim\example_configs\SimulationTool\SIMTOOL000001* directory.



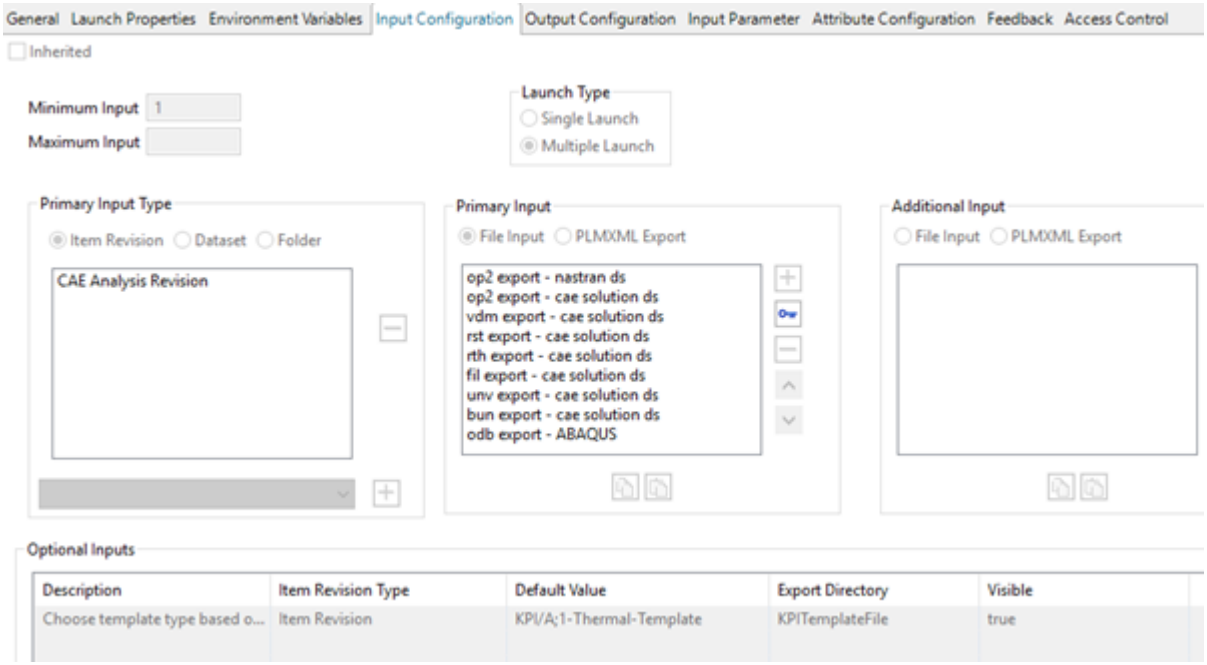
> Tip:
>
> Ensure that the correct version of Perl is installed. For more information, see *TC_DATA***tcsim\setup\example_configs\SimulationTool\SIMTOOL000001\-Tool_Integrations_Notes-Extract_KPI_from_Result.pdf**.

4. (Optional) Click the **Input Configuration** tab.

   In this tab, the primary input type is configured as **CAE 3D Analysis** revision and the configuration contains some primary input rules by default. In addition, the configuration contains the **KPI/A;1-Thermal-Template** as an optional input.

   You can create your own custom template XML file and specify it as an optional input for extracting KPI values.

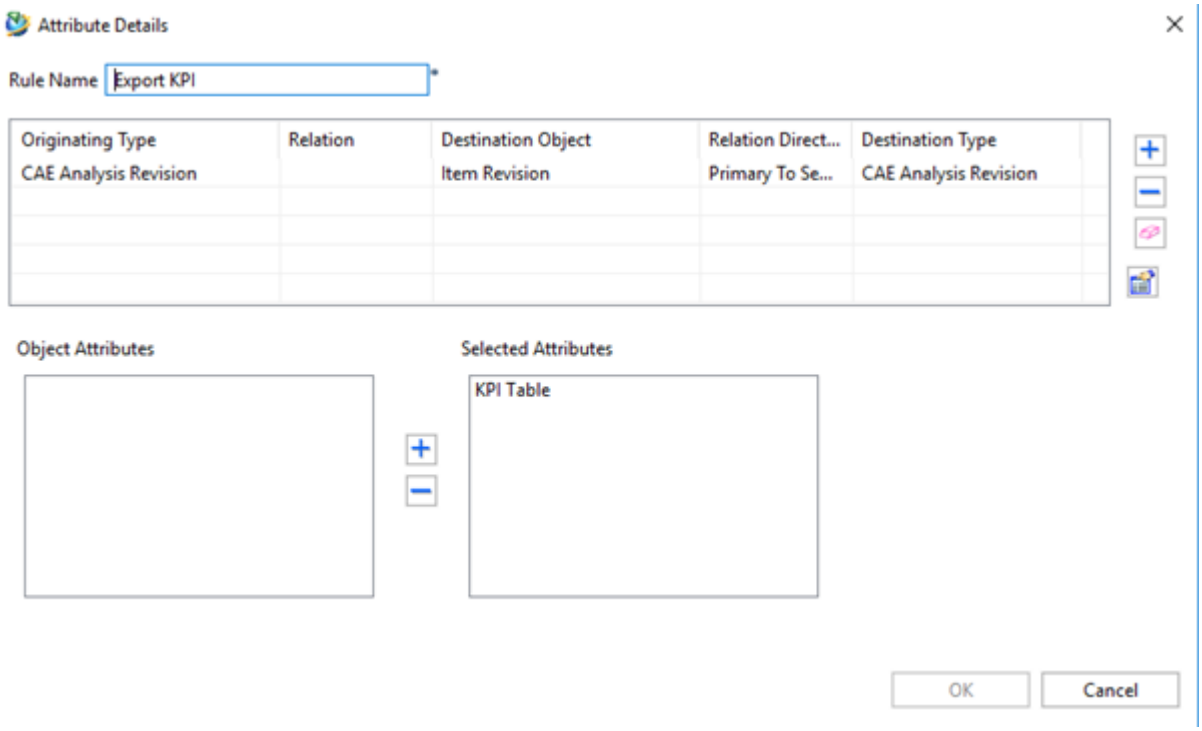
By default, the *ThermalTemplate.xml* file is available from the *TC_DATA\tcsim\example_configs\SimulationTool\SIMTOOL000001* directory.



5. (Optional) Click the **Attribute Configuration** tab.

   You can optionally provide a filename for the attribute XML file in this tab. By default, it is set to **KPI_attribute.xml**.

   This attribute has a default rule to export the KPI table attributes as follows:



Configure a simulation tool for connected mode

As a simulation administrator, you can configure a simulation tool using the connected mode for integrations such as STAR-CCM+.

The simulation administrator can configure simulation tools using the connected mode for integrations such as STAR-CCM+. This allows simulation analysts to launch tools without any input. The connected mode is best suited for tools that have a user interface. Additionally, the connected mode opens the Teamcenter Tool Launcher Client (TTLC) window with Active Workspace embedded in it. Simulation analysts can then complete the analysis and import the files back to Teamcenter by performing all the tasks in the embedded viewer.

Procedure

1. Specify a tool name.

   a. In CAE Manager, click the **CAE Configuration** toolbar menu, and choose **Simulation Tool Configuration**.

   b. To create a simulation tool, click **Create Simulation Tool** in the view toolbar.

   c. Click **Assign** to automatically assign an ID and revision.

   d. Specify a name for the tool and click **OK**.

2. Select launch properties for the tool.

   a. Choose the **General** tab and select the **Launch With Any Input** check box.

   b. Choose the **Launch Properties** tab.

   c. Select the **Connected Mode** radio button.

      When you select this option, the system selects the following options by default:

      - **Local Launch** and the **Use Staging Directory** check boxes in the **Launch Properties** tab.
      - **Single Launch** radio button in the **Input Configuration** tab.

   d. To include the launch script location, select the **Windows Platform** and specify the path to the **.bat** file for launching the integration tool such as STAR-CCM+. This file defines the simulation tool that can be launched by the simulation analyst.

      The connected mode is supported only on the Windows platform.

      You can configure simulation tools **using sample scripts**. To create a tool launch script, you can refer to the sample scripts provided in the *TC_DATA***CAE_ESP_Sample_Files.zip** file.

      When the simulation analyst launches the connected mode tool, the **Open** action exports the data from the selected object and executes the simulation tool launch script. The simulation administrator while configuring the launch script must ensure that the exported data opens in the already running session.

3. Specify input parameters.

   a. Choose the **Input Parameter tab**.

   b. Specify the path to the executable in the **Value** column.

4. Select the input rule for the tool.

   You must configure the primary input by specifying an item revision and create an input rule containing a traversal rule to associate the JT file.

   a. Choose the **Input Configuration** tab.

   b. In the **Primary Input Type** section, specify **Item Revision** and **CAE 3D Analysis Revision** as the input types.

      The item revision is required for the input rule and the analysis revision for the output rule.

c. In the **Primary Input** section, create a traversal rule to associate a JT file by selecting **File Input** and clicking **Add**.

d. Specify a rule name and create a traversal rule as follows:

Originating Type	Relation	Destination Object	Relation Direction	Destination Type
Item revision	Rendering	Dataset		Direct Model
Direct Model		ReferenceType		JTPART(*.jt)

e. In the **File Name Pattern** section, specify ***.jt**.

5. Specify the output rule for the tool.

You must configure an output rule to import the data with the 3D analysis revision by creating a traversal rule to capture the ***.sim** file.

a. Choose the **Output Configuration** tab.

b. To specify an output rule, select **File Output** and click **Add**.

c. Specify a rule name and create a traversal rule as follows:

Originating Type	Relation	Destination Object	Relation Direction	Destination Type
CAE 3D Analysis Revision	Specifications	Dataset		CAE STAR-CCM+
CAE STAR-CCM+		ReferenceType		CAE0SIM(*.sim)

If the simulation administrator creates multiple output rules, only the common objects defined for the rules are created by the system. Let us assume a scenario where the first rule has a traversal path from an analysis revision to a model revision and from a model revision to a dataset type of text. The second rule has a traversal path from an analysis revision to a result revision and from a result revision to a dataset type of text. Additionally, the third rule has a traversal path from an analysis revision to a model revision and from a model revision to a dataset type of test model file. In such a case, the system creates only the model revisions when the analyst launches the tool in the context of a folder. However, when the analyst chooses an analysis revision to launch the tool, the system honors all three rules and creates the model revision and the result revision.

d. In the **File Name Pattern** section, specify ***.sim**.

6. Save the tool and click the **Release Simulation Tool** toolbar button.

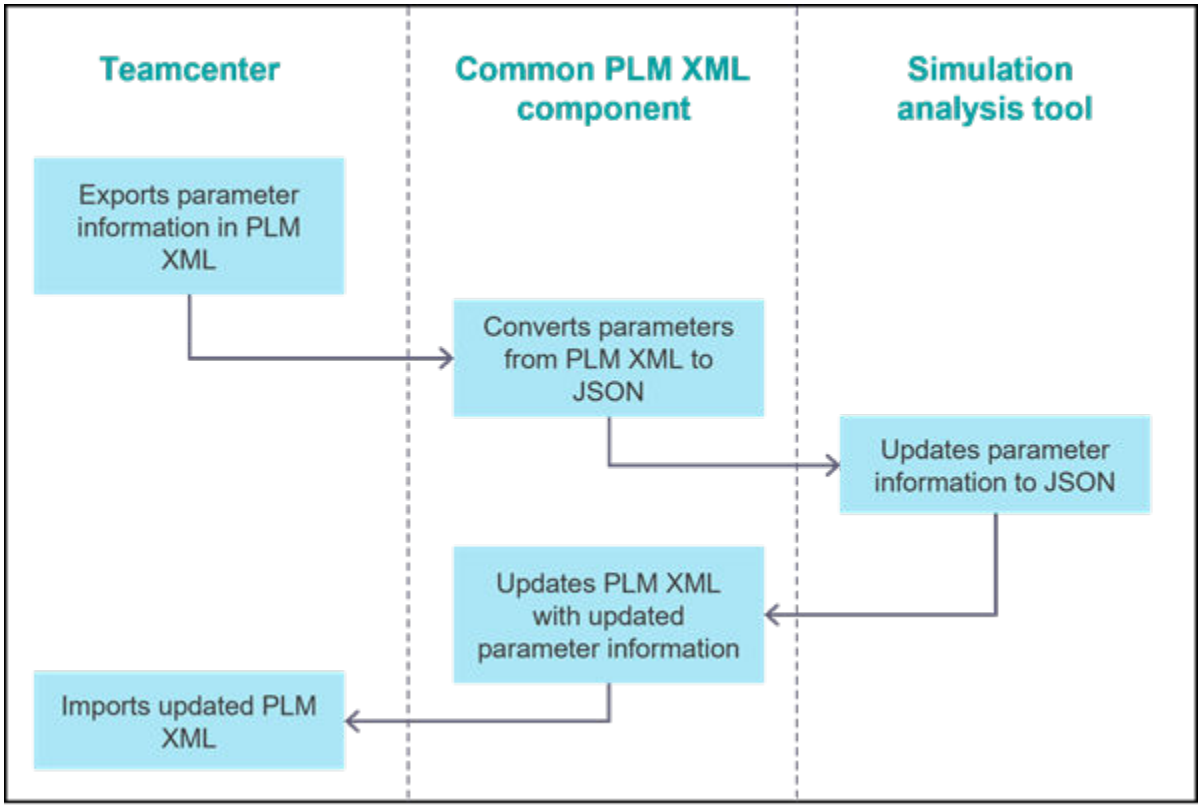
Configure a simulation tool to extract data for STAR-CCM+ Design Manager

Simulation analysts can launch preconfigured simulation tools such as preprocessors, solvers, postprocessors, and other tools. During the simulation tool launch, Teamcenter generates a PLM XML file and the data from it is used by the simulation analysis tools as the input to create the model. The simulation analyst performs the analysis, and the system sends back the results by updating the PLM XML file. The results are imported back to Teamcenter from the PLM XML file.

All simulation tools cannot read and modify the PLM XML data. The conversion of MBSE parameters information from PLM XML is now done using JSON schema.

The JSON schema is available in the *TC_DATA***tcsim\example_configs\SimulationTool\SIMTOOL000001\CaeMdoJsonSchema.json** directory. The compressed file also contains two Python scripts for converting PLM XML to JSON and JSON to PLM XML.

The PLM XML component of the tool generates the JSON file. An example of this file is available in *TC_DATA***tcsim\example_configs\SimulationTool\SIMTOOL000001\CaeMdoJsonExample.json**.



As a simulation administrator, you can set up sample configurations by running the **tcsim_quick_setup.pl script**. The sample simulation configurations include tools such as **Design Manger Integration**. This is the default tool for capturing model and analysis data for integrations such as Design Manager.

After running the quick setup script, in CAE Manager, on the main toolbar, choose **CAE Configuration** → **Simulation Tool Configuration**. The **SIMTOOLS** configuration view displays the default preprocessors, solvers, postprocessors, and other utilities. Navigate to the **Design Manger Integration** tool to view various tools for different launch types.

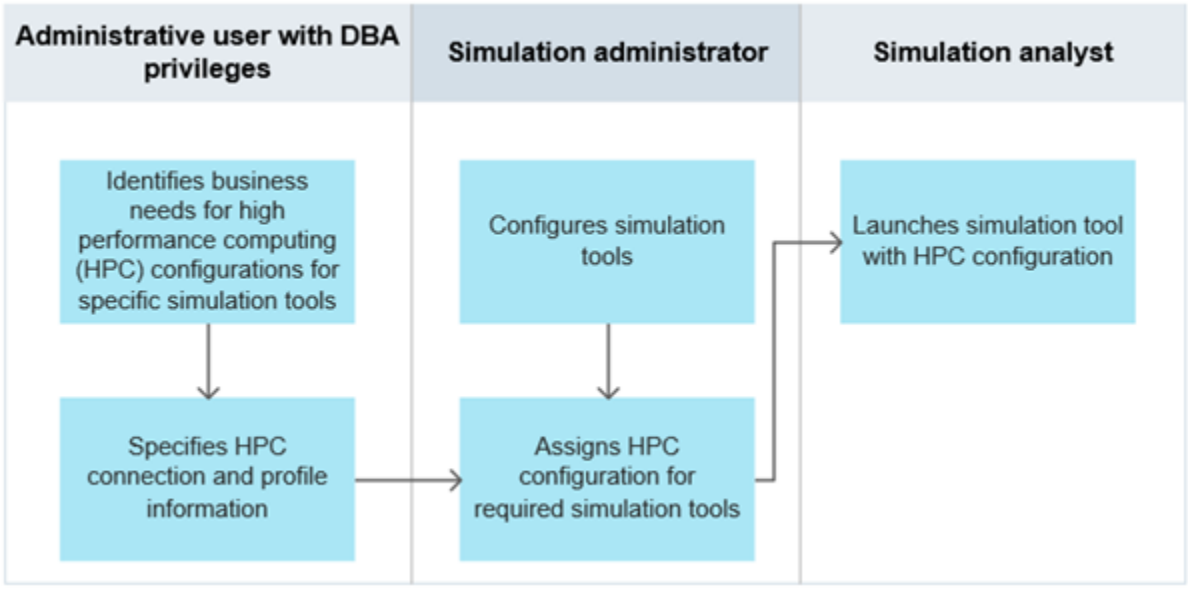
When the **Simulation Process Management with Parameter Management** feature is installed, the **Design Manger Integration** tool uses the JSON format for importing the MBSE parameters in Active Workspace.

The process is as follows:

1. The simulation analyst launches the preconfigured tool.
2. The launch script and a zip file are exported to the ESP directory. The compressed files contain the JSON schema and two Python scripts for converting PLM XML to JSON and JSON to PLM XML.
3. The launch script is executed, and the system creates log files in the **ESP\ESP_LOGS** directory.
4. The system creates a backup directory in the ESP directory where the exported PLM XML file and converted JSON files are copied for reference. The system extracts the zip file to the ESP directory. The zip file is also copied to the **ESP\backup** directory.
5. The system checks whether the Simulation Process Management with Parameter Management feature is installed and gathers some additional arguments provided by the user for the Design Manager launch.
6. The system executes the first Python script to convert the PLM XML to JSON. The script looks for the Design Manager project (**.dmprj**) file.
7. The system creates a macro file through the launch script to start Design Manager. This macro file also contains the path to the JSON file.
8. The simulation analyst creates a design study in Design Manager and exits the software.
9. The second Python script converts the JSON to PLM XML.
10. The system imports the appropriate data from the PLM XML file to Teamcenter.

Managing HPC connection and profile information for tool launch

Specify HPC connection and profile information



The administrator identifies business needs for high performance computing (HPC) configurations for specific simulation tools. Based on the needs, an HPC configuration is created that includes:

- Connection information: Name, host or URL, authentication protocol and authentication key location.
- Profile information: Name, submission command, other submission arguments, monitoring command, job cancel command, and connection object.

As an administrative user with DBA privileges, you can configure HPC parameters for simulation tool launch. You can create connections and profiles in the **Simulation HPC Configuration** view. It is also possible to quickly create connections or profiles by cloning existing ones.

You can also use the sample HPC configurations containing standard scheduler parameters if the simulation administrator has run the quick setup script (**tcsim_quick_setup.pl**) to **create sample configurations** at your site.

You can migrate profiles and connections by using the **cae_migrate_configurations** utility. For more information, type *utility_name* **-h** on the Teamcenter command prompt.

> Example:
>
> You can create connection and profile information for the Nastran solver on HPC clusters and this information can be used while configuring the tool launch for the Nastran solver. When the simulation analyst launches the Nastran tool with the HPC configuration, the connection and

profile information are passed on to the launch script as arguments. The launch script moves the input files from the operating system staging directory to the HPC staging directory. After the Nastran solve is complete, the output files are moved back to Teamcenter.

Create a new HPC connection

An administrator user with DBA privileges creates new HPC connections to specify connection information. This includes the name, host or URL, authentication protocol, and authentication key location.

Procedure

1. In CAE Manager, click **CAE Configuration** → **Simulation HPC Configuration**.
2. Create a new connection.
    a. Click the **Create HPC Configuration** icon in the **Simulation HPC Configuration** view and select **Connection**.
    b. (Mandatory) Specify a name for the connection and click **OK**. ID and revision are automatically assigned.
    c. Select the connection you created.
    d. (Mandatory) Specify the host name or the URL.
    e. (Optional) Specify the authentication protocol and the authentication key location as appropriate.
    f. To save the connection details, click **Save**.
3. Clone an existing connection to create a new one quickly.
    a. Select the connection and click the **Clone HPC Configuration** button in the **Simulation HPC Configuration** view.
    b. (Mandatory) Specify a name and click **OK**. ID and revision are automatically assigned.
    c. Repeat **step d** through **step f.**
4. To delete a connection, select the connection and click the **Delete HPC Configuration** icon in the **Simulation HPC Configuration** view.
5. To search for connections, click **Search** → **Advanced** → **Select a Search** → **More** → **CAE Configuration Revision** and click **OK**. Specify the name and click **Search**.

Create a new HPC profile

An administrator user with DBA privileges specifies HPC profile information. This includes details such as the name, submission command, other submission arguments, monitoring command, job cancel command, and connection objects.

Procedure

1. In CAE Manager, click **CAE Configuration** → **Simulation HPC Configuration**.

2. Create a new profile.

    a. Click the **Create HPC Configuration** icon in the **Simulation HPC Configuration** view and select **Profile**.

    b. (Mandatory) Specify a name for the profile and click **OK**. ID and revision are automatically assigned.

    c. Select the profile you created.

    d. (Mandatory) Specify the submission command.

    e. (Optional) Specify other submission arguments, monitoring command, and job cancel command as appropriate.

    f. (Mandatory) Select a connection object. This is the connection you created in **Create a new HPC conection**.

    You can add multiple connection objects. When you include multiple connections, the simulation analyst while launching the simulation tool has the discretion to select the appropriate connection.

    g. To save the profile details, click **Save**.

3. Specify scheduling parameters for profiles.

    The scheduling parameters you specify can be imported to the **Input Parameter** tab of the simulation tools for which HPC configuration is enabled. Some of these parameters are passed to the simulation tools with HPC configuration launched by simulation analysts.

    a. Select a profile and click the **Standard Scheduler Parameter** tab.

    b. To specify a scheduling parameter, click **Add** and specify a name and description.

    c. From the **Type** column, select an appropriate value such as **String**, **Double**, **Boolean**, or **List of Values**.

d. From the **Runtime Parameter** column, select **true** or **false** values.

e. From the **Script Input Option** column, select an appropriate option.

When **String**, **Double**, or **List of Values** is selected, you can choose **Name value pair** or **Value only**. When **Boolean** is selected, you can choose you can choose **Name value pair** or **Value only** or **Name only**.

f. In the **Value** column, specify a value for the **Name value pair** or **Value only** option.

This column is disabled for the **Name only** option.

4. Clone an existing profile to create a new one quickly.

a. Select the profile and click the **Clone HPC Configuration** button in the **Simulation HPC Configuration** view.

b. (Mandatory) Specify a name and click **OK**. ID and revision are automatically assigned.

c. Repeat **step d** through **step g.**

5. To search for profiles, click **Search** → **Advanced** → **Select a Search** → **More** → **CAE Configuration Revision** and click **OK**. Specify the name and click **Search**.

6. (Optional) To delete a profile, select the profile and click the **Delete HPC Configuration** icon in the **Simulation HPC Configuration** view.

Configure style sheets to allow analysts to launch simulation tools

Configure style sheets for simulation tool launch

The simulation administrator can create a launch dialog style sheet for configured simulation tools. The launch dialog style sheet is used to render the **Launch Simulation Tool** dialog box for analysts. If you do not specify a style sheet, the default style sheet is used to render this dialog box.

The default style sheet, the **SampleLaunchSimToolDialogHTML.html** file, is available from the **CAE_ESP_Sample_Files.zip** file in the *TC_DATA* directory. The default style sheet template is created using Knockout, an open source Java script library. For more information about using this Java script library, see the **Knockout website**.

The location of the view model java script (**CaeToolLaunchDialogViewModel.js**) file is as follows:

- 4 Tier: *staging***webapp_root\teamcenter\html_xrt\tcsim_scripts\CaeToolLaunchDialogViewModel.js**

- 2 Tier: *rich client log file***commonclient****localserver_localhost***Teamcenter_ userid***teamcenter****html_xrt****tcsim_scripts****CaeToolLaunchDialogViewModel.js**

  You can see look up the **Rich Client Log File** path from **Teamcenter→Help→About**.
  For example, **C:\Users***Teamcenter_ userid***Teamcenter****RAC****20180709123338***Teamcenter_ userid*_**TcRAC_20180910095723.log**

You can share the launch dialog style sheet with any number of configured simulation tool categories or simulation tools even if they are from different categories. You can select the same launch dialog style sheet filename for multiple tools either by selecting a launch display file in the UI or selecting the **Inheritance** check box for sub processes.

After you specify the full path for the tool launch dialog style sheet in the **Simulation Tool Configuration** view, the file is imported and the configured parameter in the XML configuration file is recorded as only a filename with an extension. If you specify only a filename with extension, it is assumed that the filename already exists in the **TCCAESimToolConfig102009** default dataset (indicated by the **CAE_simulation_tool_config_dsname** preference) and is not imported.

Modify style sheets for simulation tool launch

All CAE Manager rich client application views and dialog boxes are rendered based on XML style sheets. You use style sheets to change the layout of pages such as forms, dialog boxes, and views. Style sheets are XML documents stored in Teamcenter **XMLStylesheetRendering** datasets.

You can modify Teamcenter style sheets and specify a link to allow analysts to launch simulation tools defined in CAE Manager.

1. In My Teamcenter, click **Open Search View** on the main toolbar, click **Select a Search** in the view toolbar and choose the **General** search type.

2. To locate the style sheet you want to modify, click **Clear all search fields** in the view toolbar, type *baseitemrev** in the **Name** box, and perform the search.

   > Warning:
   >
   > There is no dedicated **CAEItemRevSummary** style sheet. The **BaseItemRevSummary** is a summary style sheet type and is registered to the **Item Revision** type. Therefore, you may want to create a custom style sheet based on an existing style sheet because when you edit the **BaseItemRevSummary** style sheet, you change all summary style sheets for the **Item Revision** type.

3. Add the following command action key and parameter value in the first section titled **tc_xrt_actions**:

   Example:

```
<section titleKey="tc_xrt_actions" commandLayout="vertical">
  <command actionKey = "copyAction" commandId = "com.teamcenter.rac.copy"/>
  <command actionKey = "reviseAction" commandId =
"com.teamcenter.rac.revise"/>
  <command actionKey = "saveAsAction" commandId =
"org.eclipse.ui.file.saveAs"/>
  <command actionKey = "newProcessAction" titleKey = "tc_xrt_newProc"

commandId = "com.teamcenter.rac.newProcess"/>

<! -- Add a link to launch simulation tools -->
   <command commandId="com.teamcenter.rac.tcsim.commands.launchsimproc"
 title="Launch_Tool_Name">
   <parameter value="com.teamcenter.rac.tcsim.commands.SimTool-00017"
   name="com.teamcenter.rac.tcsim.LaunchSimulationProcess"/>
 </command>
</section>
```

> Note:
>
> The tool ID is required for the style sheet configuration. When the tool is invoked from the style sheet, the latest released revision of the corresponding simulation tool is launched.
>
> You can use blank characters in the `com.teamcenter.rac.tcsim.commands.`*itemID* line for the parameter value.

4. Replace *Launch_Tool_Name* with the name of the simulation tool you want analysts to launch, for example, **Launch NxNastran**.

> Tip:
>
> The link you specify is available from the **Summary** tab of CAE Manager; it is not enabled from the **Summary** tab of My Teamcenter.

5. Specify the tool ID of the simulation tool you want the analyst to launch. In the above example (step 3), the tool ID is *SimTool-00017*.

6. To modify the style sheet, click **Apply**.

Configure workflow templates for launching simulation tools

Configure workflow templates

You can use Teamcenter Workflow Designer to configure workflow handlers to allow analysts to launch simulation tools from Simulation Process and Data Management.

- After you create a workflow template, an analyst cannot modify the template at the time of launching the workflow.
- Analysts can use the workflow template to launch only the simulation tools associated with the tasks defined in the template.
- For remote launch, you must install and configure the Dispatcher Management functionality.
- After successful completion of a workflow launch:
  - Objects are created in Teamcenter and appropriate relationships created between the objects depending on the simulation tool configuration.
  - Output files from the simulation tool are imported into appropriate Teamcenter datasets.
  - New output files are uploaded to the database. For each output filename that has a similar filename in the database, the system processes the files based on the **file upload conflict options** specified by the simulation administrator.
  - Operating system folders created to support the execution of the launch tool are cleaned according to the simulation tool configuration.

    Simulation Process and Data Management uses Dispatcher Server components to launch the simulation tools remotely. Dispatcher Client has its own mechanism to cleanup temporary files. You can configure this using the **RequestCleanup** properties in the **DispatcherClient\conf\Service.properties** file.
- Notifications are sent to analysts after successful completion or failure of a workflow process.

To allow analysts to launch simulation tools from a workflow, you must define an appropriate workflow template that leverages the **CAE-simulation-process-launch-handler** action handler option.

> Note:
>
> You can create process templates from workflow templates for local, remote, or server launch; workflow templates are not supported for local detached launch.

The following are the high-level procedures for creating or modifying a workflow process template.

1. Create a new workflow process template using the Workflow Designer.

   For more information about creating workflow process templates, see *Workflow Designer*.
2. From the **Action Handler** menu, choose **CAE-simulation-process-launch-handler**.
3. Specify arguments and values as appropriate.

For more information about arguments and values, see **CAE-simulation-process-launch-handler** in *Workflow Designer*.

The **–tool** argument is mandatory and requires the simulation tool ID value. The rest of the arguments are optional and can be specified without any values.

Tool names and revisions are no longer supported. The tool is now launched with the latest released revision. If you have an existing action handler with a tool name and revision values, you must modify them and use only the tool ID value.

Export and import workflow templates

1. Choose **Tools→Export** to export the template you created. Specify a directory and name for the file you want to export.

2. In Teamcenter, choose **Tools→Import→From PLMXML** to import the workflow template (XML file) you exported. Ensure that you choose the **workflow_template_import** option from the **Transfer Mode Name** menu.

Set up Dispatcher to launch simulation tools from different machines

Prerequisites for setting up Dispatcher on modules

You can set up Dispatcher on different machines or modules that have simulation tools. Analysts can launch simulation tools as remote or local detached launch after you configure simulation tools and set up Dispatcher on the required modules.

- Install Dispatcher components on the Teamcenter server using TEM.

- If you use the RMI mode for Dispatcher:

  - The Dispatcher clients and Dispatcher Scheduler should point to the common staging directory and they both should have read/write permissions to that directory for file transfers.

    This eliminates the need for file transfers from clients and module locations. Clients write the translator input files to the common staging directory and modules write the translator result files to the staging directory in the results directory. For example, if **d:\StagingDir** is the common staging directory for the module machine, then **\\ModuleMachine\StagingDir** on the client machine (can be a different machine) should be the same directory, that is, **StagingDir**.

  - The Dispatcher clients and Dispatcher Scheduler should be installed by the same user to avoid directory access issues since the client and Dispatcher modules use the same staging location.

  - In the common staging directory configuration, the Dispatcher module should have directory-create permission and write access to files on the client staging location using the network.

This means that when you access the staging location from the module machine, you can access or list (in Linux systems) the directory contents of the staging location. The staging location can be a network-shared location.

- Share and map the path to the *TC_ROOT* and *TC_DATA* directory on the Teamcenter server from all modules.
- Set up the required Java version on all modules.

Set up Dispatcher on different machines

To reduce the time required to set up Dispatcher on modules, copy the required Dispatcher components from the Teamcenter server to the module and perform the following essential configuration procedures:

1. Copy the *TC_ROOT***Dispatcher\Module** directory from the Teamcenter server to each module.
2. Set the **MODULEBASE** and **JAVABIN** entities for modules by editing the **Dispatcher\Module\conf\translator.xml** file on all modules to make the following changes:

   a. Set the **MODULEBASE** entity to the current location of the module directory on the module.

   Example:

   ```
   <?xml version="1.0" ?>

   <!DOCTYPE Translators[
   <!ENTITY  MODULEBASE "\\machine_name\Module">
   ```

   b. Set the **JAVABIN** entity to the bin location of the JRE installation.

   Example:

   ```
   <!ENTITY  JAVABIN "C:\apps\Java\jre7x64\bin">
   ```

   c. Enable the required services such as **nxnastran** and **simprocess** by setting the **isactive** attribute to **true**.

   > Note:
   >
   > You can also use TEM to configure these services instead of manually editing the **translator.xml** file.

   nxnastran example:

```
<!-- Configuration of the Simulation Process - nxnastran -->
  <NxNastran provider="SIEMENS" service="nxnastran"
isactive="true">
    <TransExecutable dir="&MODULEBASE;/Translators/simprocess"
      name="tc_launch_sim_process_ts_pl.bat"/>
    <Options>
      <Option name="inputpath" string=""
              description="Full path to the input file."/>
      <Option name="outputdir" string=""
          description="Full path to the output directory."/>
    </Options>
    <FileExtensions>
      <InputExtensions nitem="1">
        <InputExtension extension=".xml"/>
      </InputExtensions>
    </FileExtensions>
  </NxNastran>
```

3. Set the staging directory, log volume location, and scheduler URL by editing the **Module\conf\transmodule.properties** file on all modules:

> Note:
>
> If the Dispatcher components are installed on different machines, ensure that the mapped drive is used for the staging directory and the log location in the Dispatcher configuration. Users should have read write access to this location. Also, Dispatcher should be run using a command prompt (cmd) instead of as a service.
>
> Use forward slashes (**/**) as path separators while editing all properties in the **transmodule.properties** file.

- To specify the staging directory, set the **Staging.Dir** property to the mapped drive path.

  Example:

  Staging.Dir=//*machine_name_of_Teamcenter_server*/*TC_ROOT*/**Dispatcher**

  where the staging directory is a directory inside the Dispatcher root directory, that is, *TC_ROOT***Dispatcher\staging**.

- To specify the log location, set the **LogVolumeLocation** property to a desired location on the module.

  Example:

  Change **LogVolumeLocation=C:/PROGRA~1/Siemens/TEAMCE~1/DISPAT~1/Logs** to **LogVolumeLocation=***module***/Dispatcher/Logs**

- To specify the scheduler URL, set the **Scheduler.URL** property to the Teamcenter server or any other host name on which Scheduler is running.

  Example:

  Change **Scheduler.URL=rmi://localhost:2001** to **Scheduler.URL=rmi://** *machine_name_of_Teamcenter_server*:**2001**

4. Set the *TC_ROOT* and **JAVA_HOME** properties by editing the **Dispatcher\Module\bin\runmodule.bat** file on all modules:

   - Set the *TC_ROOT***install** directory to the mapped drive.

     Example:

     Change **call "C:\Program Files\Siemens\Teamcenter10\install\tem_init.bat"** to **call "** *machine_name_of_Teamcenter_server**TC_ROOT***install\tem_init.bat"**

   - Set the **JAVA_HOME** property to path of the Java installation directory.

     Example:

     Change **set JAVA_HOME=%TC_JRE_HOME%** to **set JAVA_HOME=C:\apps\Java\jre7x64\bin**

5. To copy translators or services, copy the *TC_ROOT***Dispatcher\Module\Translators** directory to the **Module** directory on all modules.

Enable SimProcess and NxNastran services for remote launch

You can enable the **SimProcess** and **NxNastran** services on Dispatcher to launch a simulation process or NX Nastran remotely by using Simulation Process and Data Management.

1. To specify the Perl installation location for the **SimProcess** service, edit the *Module***Translators\simprocess\tc_launch_sim_process_ts_pl.bat** file.

   When you copy the translators or services to a module, share and map the path to the *TC_ROOT***perl\bin\perl.exe** file on the Teamcenter server. By default, the batch files for the **SimProcess** and **NxNastran** services point to the default Perl location of Teamcenter, and not the mapped path required for modules.

   Example:

   Change **%TC_ROOT%\perl\bin\perl -I%TC_ROOT%\perl\lib %0 %*** to *Z*:*TC_ROOT***perl\bin\perl**

   where *Z* is the shared and mapped path to the Teamcenter installation Perl directory.

2. Configure remote launch.

   a. Open the *Module***conf\translator.xml** file.

   b. Copy the **SimProcess** or **NxNastran** service tags to another area of the **translator.xml** file.

      SimProcess service example:

```
<!-- Configuration of the Simulation Process - simprocess -->
  <SimProcess provider="SIEMENS" service="simprocess" isactive="true">
    <TransExecutable dir="&MODULEBASE;/Translators/simprocess"
name="tc_launch_sim_process_ts_pl.bat"/>
    <Options>
      <Option name="inputpath" string=""
              description="Full path to the input file."/>
      <Option name="outputdir" string=""
          description="Full path to the output directory."/>
    </Options>
    <FileExtensions>
      <InputExtensions nitem="1">
        <InputExtension extension=".xml"/>
      </InputExtensions>
    </FileExtensions>
  </SimProcess>
```

   c. Make the following changes to the service tags you have copied.

      SimProcess service example for remote launch:

```
<!-- Configuration of the Simulation Process - simprocess -->
  <SimProcess provider="SIEMENS" service="simprocess" isactive="true">
    <TransExecutable dir="&MODULEBASE;/Translators/simprocess"
name="tc_launch_sim_process_ts_pl.bat"/>
    <Options>
      <Option name="inputpath" string=""
              description="Full path to the input file."/>
      <Option name="outputdir" string=""
          description="Full path to the output directory."/>
    </Options>
    <FileExtensions>
      <InputExtensions nitem="1">
        <InputExtension extension=".xml"/>
      </InputExtensions>
    </FileExtensions>
  </SimProcess>
```

   d. Activate a service by setting the **isactive** property to **true**. By default, it is set to **false**.

Example:

```
<SimProcess provider="SIEMENS" service="simprocess"
isactive="true">
```

e. Add the **TaskPrep** and **DatabaseOperation** classes by editing the *Module***Dispatcher\DispatcherClient\conf\Service.properties** file for each service you added in the **translator.xml** file.

Example:

```
#  Translator.<provider name>.<translator name>.Prepare=
#  Translator.<provider name>.<translator name>.Load=
```

Add the following lines:

Translator.SIEMENS.simprocess
.Prepare=com.teamcenter.dc.simprocess.TaskPrep

Translator.SIEMENS.simprocess
.Load=com.teamcenter.dc.simprocess.DatabaseOperation

f. Repeat steps **d** and **e** for each process you want to configure.

If you want to configure a process which has spaces in its name, for example **Launch Heeds** process, then use the following format:

```
<Launch_Heeds provider="SIEMENS" service="Launch Heeds"
isactive="true">
  <TransExecutable dir="&MODULEBASE;/Translators/simprocess"
name="tc_launch_sim_process_ts_pl.bat"/>
  <Options>
      <Option name="inputpath" string="" description="Full path
to the input file."/>
      <Option name="outputdir" string="" description="Full path
to the output directory."/>
  </Options>
  <FileExtensions>
      <InputExtensions nitem="1">
          <InputExtension extension=".xml"/>
      </InputExtensions>
  </FileExtensions>
</Launch_Heeds>

# Provider Name: SIEMENS
# Translator Name: Launch\ Heeds
Translator.SIEMENS.Launch\ Heeds.Prepare =
com.teamcenter.dc.simprocess.TaskPrep
```

```
Translator.SIEMENS.Launch\ Heeds.Load =
com.teamcenter.dc.simprocess.DatabaseOperation
```

3. Configure local detached launch.

> Note:
>
> The local detached launch option has been deprecated and will be removed in a future version of Teamcenter.

a. Open the *Module***conf****translator.xml** file.

b. Copy the **SimProcess** or **NxNastran** service tags to another area of the **translator.xml** file.

SimProcess service example:

```
<!-- Configuration of the Simulation Process - simprocess -->
  <SimProcess provider="SIEMENS" service="simprocess" isactive="true">
    <TransExecutable dir="&MODULEBASE;/Translators/simprocess"
           name="tc_launch_sim_process_ts_pl.bat"/>
    <Options>
      <Option name="inputpath" string=""
              description="Full path to the input file."/>
      <Option name="outputdir" string=""
          description="Full path to the output directory."/>
    </Options>
    <FileExtensions>
      <InputExtensions nitem="1">
        <InputExtension extension=".xml"/>
      </InputExtensions>
    </FileExtensions>
  </SimProcess>
```

c. Make the following changes to the service tags you have copied.

- Append the **simprocess** tag name with the machine name of the module, that is, **simprocess_***module_name*
- Append the **service="simprocess"** tag with the machine name of the module, that is, **service="simprocess_***module_name***"**.

> Note:
>
> You must specify the *module_name* value in uppercase for both Linux and Windows.

SimProcess service example for local detached launch:

```
<!-- Configuration of the Simulation Process - simprocess_module_name -->
  <SimProcess provider="SIEMENS" service="simprocess_module_name"
isactive="true">
    <TransExecutable dir="&MODULEBASE;/Translators/simprocess"
           name="tc_launch_sim_process_ts_pl.bat"/>
    <Options>
      <Option name="inputpath" string=""
              description="Full path to the input file."/>
      <Option name="outputdir" string=""
          description="Full path to the output directory."/>
    </Options>
    <FileExtensions>
      <InputExtensions nitem="1">
        <InputExtension extension=".xml"/>
      </InputExtensions>
    </FileExtensions>
  </SimProcess>
```

d. Activate a service by setting the **isactive** property to **true**. By default, it is set to **false**.

Example:

```
<SimProcess provider="SIEMENS" service="simprocess_module_name"
isactive="true">
```

e. Add the **TaskPrep** and **DatabaseOperation** classes by editing the *Module***Dispatcher\DispatcherClient\conf\Service.properties** file for each service you added in the **translator.xml** file.

Example:

```
#  Translator.<provider name>.<translator name>.Prepare=
#  Translator.<provider name>.<translator name>.Load=
```

Add the following lines:

Translator.SIEMENS.simprocess_*module_name*
.Prepare=com.teamcenter.dc.simprocess.TaskPrep

Translator.SIEMENS.simprocess_*module_name*
.Load=com.teamcenter.dc.simprocess.DatabaseOperation

f. Repeat steps **d** and **e** for each process that you want to configure.

Configure the simulation dashboard

Why use the simulation dashboard?

The simulation dashboard provides a clear view of the status of all the models and analyses carried out by simulation analysts at the program, milestone, group, or individual user level. It allows decision makers to access the latest information and make correct decisions.

The simulation dashboard:

- Displays the list and status of all models and analyses based on a query.
- Displays the status of models and analyses based on variants of the vehicle or master structure based on the pedigree information.
- Presents the status of the results corresponding to the models and analyses.
- Monitors key performance indicator (KPI) values such as maximum stress, minimum temperature, or maximum displacement from analysis revisions or analysis templates.

Override the group administrator restriction to create configuration objects

Only a group administrator can create configuration objects at the group level. To avoid this restriction, you can select the **Bypass Group Administrator** option.

1. In CAE Manager, click **Edit→Options→CAE→General**.
2. In the **Options** dialog box, hover over a field to find the preference name associated with it.

   For information about retrieving a list of preferences, see *Where can I get a list of preferences?* in *Teamcenter Preferences*.
3. Click the **CAE Configuration** tab.
4. Select the **Bypass Group Administrator** check box.

   By default, it is set to **false**.

Configure an analysis or a model dashboard

After you create dashboards, the system saves them in the database as **CAE Configuration** objects with the configuration type as **Analysis Dashboard** or **Model Dashboard**.

You can use an existing analysis or model dashboard, clone it, and quickly create a new one. While creating a new one using a cloning method, you cannot modify the type after you clone it. For example, if you select an analysis dashboard and clone it, you cannot change it later to a model dashboard.

After creating dashboards, a user with DBA privileges can use the **cae_migrate_configurations** utility to export the dashboard configurations to a different site or machine and import them again using the same utility. To view the command line help for this utility, type **cae_migrate_configurations -h** on the Teamcenter command prompt.

1. In CAE Manager, click the **CAE Configuration** toolbar menu, and choose **Simulation Dashboard Configuration**.

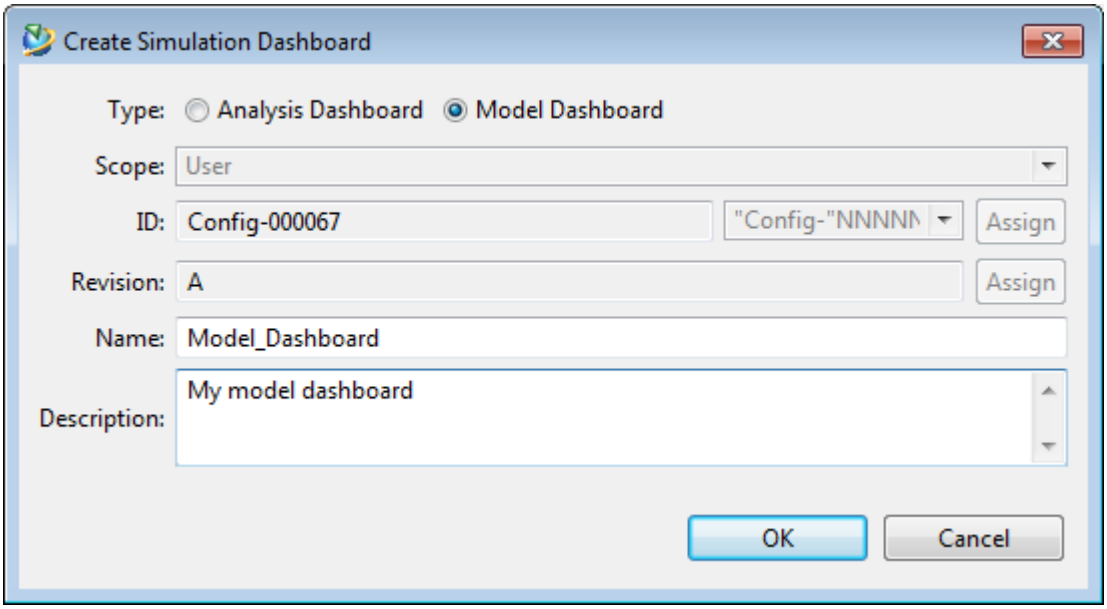
2. Select the **Site**, **Group**, or **User** scope option.

3. To create a simulation dashboard, click **Create Simulation Dashboard** in the view toolbar.

4. Select **Analysis Dashboard** or **Model Dashboard**.

5. Assign an ID and a revision.

6. Type a dashboard name in the **Name** box.



7. (Optional) Specify a description in the **Description** box.

8. To create the dashboard, click **OK**.

Specify a dashboard name and the object types to monitor

1. Click the **General** tab.

2. Specify an object name in the **Object Title** box.

   This is the name that appears in the first column header when the simulation analyst opens the dashboard.

3. To monitor all the components of the selected model BOM, including the sub-assemblies and the root, or only the leaf level components, select the **All Component Types** option.

   OR

4. To monitor only specific component types, select the **Specific Component Types** option.

   a. After selecting this option, select the component types you want to monitor from the **All Available Types** box and move them to the **Selected Types** box. You must select at least one component type.

   b. Select the component type you have moved and click the **Filter Component Types** filter.

      This dialog box allows you to specify the attribute values by which you want to filter the components to monitor.

   c. In the **Type** column, click the **Add** button and select the component type you want to filter from the list.

   d. Click the **Add** button to specify the attribute values.

   e. Specify an **And** or **Or** value as appropriate.

   f. From the **Attribute Name** list, select the appropriate value that you want to filter.

   g. From the **Condition** list, select the appropriate condition you want to use as a filter. Conditions are available specific to the selected property data type. For example, for a property of data type **String**, filter conditions such as **Equal To**, **Not Equal To**, **Starts With**, **Not Starts With**, **Contains**, **Is Null**, or **Is Not Null** are available.

   h. In the **Value** field, specify the appropriate value.

   i. To specify attributes for the other specific component types you have added, repeat **Step c** through **Step h**.

   j. Click **OK** to specify the attribute values.

Specify attributes, classification attributes, files, and variant options you want to monitor

Configure attributes to monitor

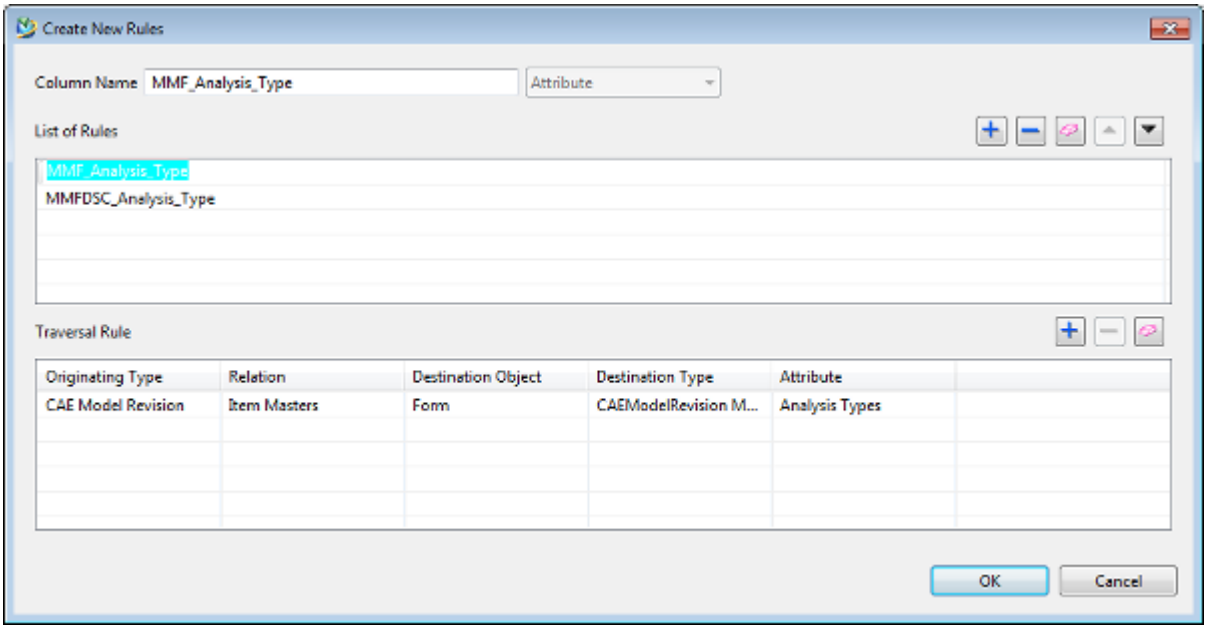
While configuring attributes to monitor, you can:

- Specify multiple rules against each analysis type in a configuration with multiple analysis types.
- Include the same attribute name for each rule you specify as long as they have different traversal paths (see examples that follow).
- Specify multiple attribute names for each rule you specify (see examples that follow).

> Note:
>
> You cannot create duplicate rules with the same traversal path for the selected analysis type.

1. Click the **Columns to Monitor** tab.
2. To add a column to monitor, click the **Add** button.
3. Specify a column name for the file you want to monitor, and then select **Attribute**.
4. Create a list of rules.
   a. In the **List of Rules** section, click the **Add** button.
   b. Specify a name for the rule.

   After you create rules, you can remove rules, clear all rules, or specify an order for the rules.



5. Create traversal rules for each of the rules you created earlier.

   a. Select a rule in the **List of Rules** section.

   b. To create a traversal rule, in the **Traversal Rules** section, click the **Add** button.

Each of these rules can have the same attribute name or the same file extension if they have *different* traversal paths.

Attribute name example	File extension example
CAE Analysis Revision → FormX → AttX	CAE Analysis Revision → Nastran → *.op2
CAE Analysis Revision → FormY → AttX	CAE Analysis Revision → CAE Result → *.op2
CAE Analysis Revision → FormZ → AttX	CAE Analysis Revision → CustomDataset → *.op2

Each of the rules can have multiple attribute names or multiple file extensions if they have *different* traversal paths.

Attribute name example	File extension example
CAE Analysis Revision → FormX → AttX	CAE Analysis Revision → CAE Result → *.op2
CAE Analysis Revision → FormY → AttY	CAE Analysis Revision → AnsysResults → *.odb
CAE Analysis Revision → FormZ → AttZ	CAE Analysis Revision → AnsysThermalResults → *.rth

Traversal rule example for **Model User data1**:

Originating Type	Relation	Destination Object	Destination Type	Attribute
CAE 3D Analysis Revision	CAE Defining	ItemRevision	CAE 3D Model Revision	None
CAE 3D Model Revision	Item Masters	Form	CAE Model Revision Master	User Data 1

Traversal rule example for **Results User data1**:

Originating Type	Relation	Destination Object	Destination Type	Attribute
CAE 3D Analysis Revision	CAE Defining	ItemRevision	CAE 3D Model Revision	None
CAE 3D Model Revision	Item Masters	Form	CAE Model Revision Master	User Data 1

Configure classification attributes to monitor

You can configure dashboards to display the classification attributes inherited from the parent item. Within the traversal rule, you can set the following:

- **Destination Object** as **Item**.
- **Destination Type** as the parent item type.
- Specify the classification ID of the attribute in the **Classification Attribute ID** column.

In the dashboard, you can display any number of classification attributes from all the available component types by configuring separate rules for each attribute.

1. Click the **Columns to Monitor** tab.
2. To add a column to monitor, click the **Add** button.
3. Specify a column name for the file you want to monitor and then select **Classification Attribute**.
4. Create a list of rules.
    a. In the **List of Rules** section, click the **Add** button.
    b. Specify a name for the rule.

After you create rules, you can remove rules, clear all rules, or specify an order for the rules.

5. Create traversal rules for each of the rules you created earlier.

   a. Select a rule from the **List of Rules** section.

   b. To create a traversal rule, in the **Traversal Rules** section, click the **Add** button.

Traversal rule example 1:

Originating Type	Relation	Destination Object	Destination Type	Destination Type
CAE 3D Model Revision	None	**Item**	**CAE 3D Model**	**10050**

Traversal rule example 2:

Originating Type	Relation	Destination Object	Destination Type	Destination Type
CAE 3D Model Revision	**CAE Source**	**Item Revision**	**CAE 3D Geometry Revision**	None
CAE 3D Geometry Revision	**CAE Target**	**Item Revision**	**Item Revision**	**1001**

Configure files to monitor

1. Click the **Columns to Monitor** tab.

2. To add a column to monitor, click the **Add** button.

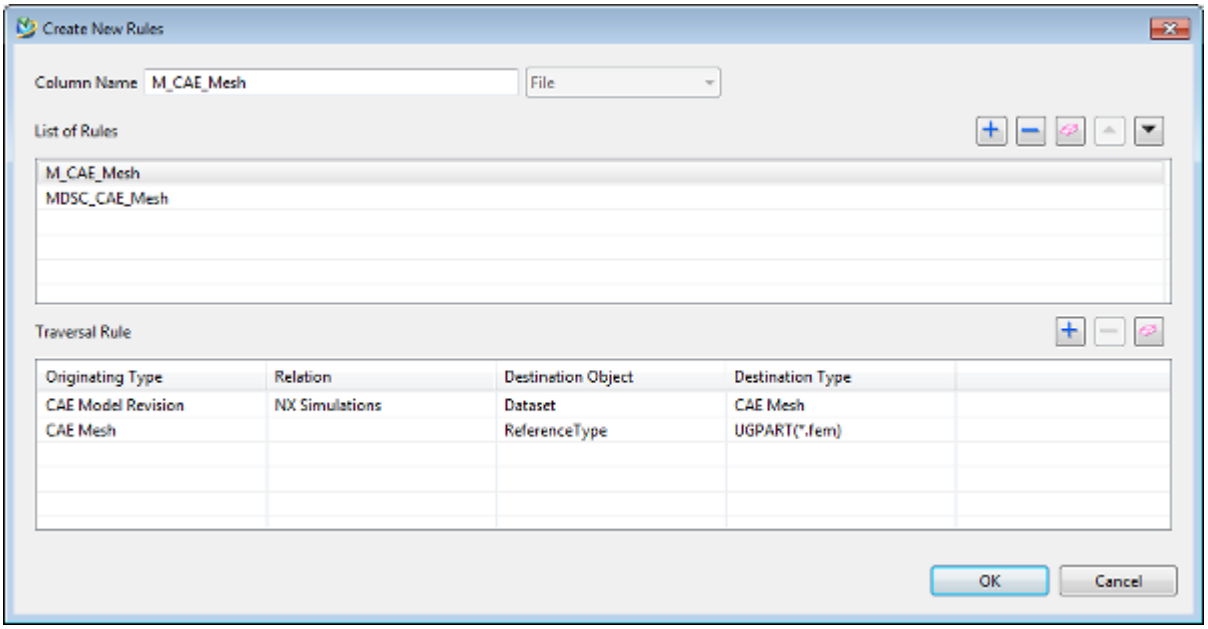
3. Specify a column name for the file you want to monitor, and then select the **File** attribute.

4. Create a list of rules.

   a. In the **List of Rules** section, click the **Add** button.

   b. Specify a name for the rule.

   After you create rules, you can remove rules, clear all rules, or specify an order for the rules.



5. Create traversal rules for each of the rules you created earlier.

   a. Select a rule in the **List of Rules** section.

   b. To create a traversal rule, in the **Traversal Rules** section, click the **Add** button.

Traversal rule example for **NX CAD**:

Originating Type	Relation	Destination Object	Destination Type
CAE 3D Analysis Revision	**CAETarget**	**ItemRevision**	**ItemRevision**
ItemRevision	**Specifications**	**Dataset**	**UGMaster**
UGMaster	None	**ReferenceType**	**UGPART(*.prt)**

Traversal rule example for **CAD JT**:

Originating Type	Relation	Destination Object	Destination Type
CAE 3D Analysis Revision	**CAETarget**	**ItemRevision**	**ItemRevision**
ItemRevision	**Rendering**	**Dataset**	**Direct Model**
Direct Model	None	**ReferenceType**	**JTPART(*.jt)**

Configure variant options to monitor

While configuring variant options, you can:

- Specify the variant option values (captured in the model or analysis pedigree objects) to monitor.
- Monitor one or more variant option values.
- Specify traversal rules for any variant option value captured in the analysis pedigree or model pedigree with reference to the selected item revision type you want to monitor.

> Note:
>
> If the product structure data is configured by using classic variants, such data cannot be monitored in the simulation dashboard when the Product Configurator mode is enabled. In such cases, the system displays the **Not Found** value in the dashboard.

1. Click the **Columns to Monitor** tab.
2. To add a column to monitor, click the **Add** button.
3. Specify a column name for the file you want to monitor, and then select the **Variant Option** attribute.
4. Create a list of rules.
   a. In the **List of Rules** section, click the **Add** button.
   b. Specify a name for the rule.
5. Create traversal rules for each of the rules you created earlier.
   a. Select a rule in the **List of Rules** section.
   b. To create a traversal rule, in the **Traversal Rules** section, click the **Add** button.

   Traversal rule example:

Originating Type	Relation	Destination Type	Variant Option Name
CAE 3D Analysis Revision	CAE Defining	CAE 3D Model Revision	Engine Style

Create a simulation dashboard and configure the KPI attributes to monitor

In this example, you create a column attribute for KPI Stress that originates from the analysis revision and another column attribute for KPI Strain that originates from the analysis template.

Create a simulation dashboard to monitor KPI values

1. In CAE Manager, click the **CAE Configuration** toolbar menu, and choose **Simulation Dashboard Configuration**.
2. Select the **Site**, **Group**, or **User** scope option.
3. To create a simulation dashboard, click **Create Simulation Dashboard** in the view toolbar.
4. Select the **Analysis Dashboard**.
5. Assign an ID and a revision.
6. Type a dashboard name in the **Name** box.
7. (Optional) Specify a description in the **Description** box.
8. To create the dashboard, click **OK**.

Specify KPI attributes to monitor

1. Create a simulation dashboard to monitor KPI values.
2. Click the **General** tab.
3. Specify an object name in the **Object Title** box.

   This is the name that appears in the first column header when the simulation analyst opens the dashboard.
4. Select the object types you want to monitor. Select **Specific Component Types** and select **CAE 3D Analysis Revision** from the **All Available Types** box and move them to the **Selected Types** box.
5. Click the **Columns to Monitor** tab.
6. To add a column to monitor, click the **Add** button.
7. Specify a column name for the file you want to monitor, for example, **KPI Stress**, and then select **Attribute**.
8. Create the first rule to monitor stress.
   a. In the **List of Rules** section, click the **Add** button.
   b. Specify a name for the rule, for example, **Stress**.

9. Create a traversal rule for the first rule.

    a. Select the rule in the **List of Rules** section.

    b. To create a traversal rule, in the **Traversal Rules** section, click the **Add** button to add the first row of the rule.

    c. In the **Originating Type** column, select **CAE 3D Analysis Revision**.

    d. In the **Attribute** column, select **KPI Table**.

    e. Click the **Add** button to add the second row of the rule.

    The system automatically populates the second row of the **Originating Type** column with the value from the **Attribute** column of the first row.

    f. In the **Attribute** column of the second row, click the **Add** button. Select appropriate values from the **Input Table Property** and **Table Property to Monitor** menus, and type the appropriate value in the **Input Table Property Value** box.

    g. To create the new rules, click **OK**.

10. Create the second rule.

    a. In the **List of Rules** section, click the **Add** button.

    b. Specify a name for the rule, for example **Strain**.

11. Create another traversal rule for the second rule.

    a. Select the second rule in the **List of Rules** section.

    b. Repeat steps b to g.

After configuring the dashboard

1. To save the simulation dashboard configuration, click **Save Simulation Dashboard Configuration** in the view toolbar.

2. To clone the simulation dashboard configuration, click **Clone Simulation Dashboard** in the view toolbar.

3. To delete the simulation dashboard configuration, click **Delete Simulation Dashboard Configuration** in the view toolbar.

    You can delete a dashboard only if you have write access to the rule set.

Customize the simulation template used to display the dashboard reports

After viewing the results in the simulation dashboard, simulation analysts can generate a report of the dashboard. The dashboard report is exported to Microsoft Excel format.

As a simulation administrator or a designated user, you can include a new Microsoft Excel template or update the existing one that is used to display the dashboard report.

In the rich client, you can add any number of templates. However, Active Workspace supports only **CAE_default_Simulation_Dashboard_template**.

> Note:
>
> Only template revisions with names starting with CAE are displayed as valid templates for the export of simulation dashboard results. The naming convention starting with CAE is not case sensitive.

Add a new template

1. In the **Home** tab of My Teamcenter or CAE Manager, select a folder and click **File→New→Item**.
2. Select **Item** and click **Next**.
3. Specify the name as **cae_analysis_simulation_dashboard** and click **Finish**.

   The naming convention is not case sensitive.
4. To associate a file type to the item revision, you must create a dataset. In the **Home** tab, select the item revision you created and click **File→New→Dataset**.
5. (Optional) Specify a name for the dataset. If you do not specify a name, the system populates the ID of the item revision you have selected.
6. Click **More** and select **MS ExcelX** as the file associate type.
7. Click **Import** to select the Microsoft Excel file you want to associate and click **OK**.

Alternatively, use the **cae_add_report_templates** utility to import multiple templates. To view the command line help for this utility, type **cae_add_report_templates -h** on the Teamcenter command prompt.

Update an existing template

1. To search for the existing template, in My Teamcenter, click **Advanced** search.
2. From **Select a Search**, select **Item**.

3. In the **Name** box, type *cae**.

4. In the **Type** box, replace **Item** with **ExcelTemplate**, and click **Search**.

5. Double-click the search result to open the excel template in My Teamcenter and expand the **Excel Template Revision**.

6. Right-click the **MS Excel** type, select **Check-In/Out** and select **Check Out**.

7. Right-click the **MS Excel** type and select **Named References**. Select the default template and remove it.

   Click **Upload** and select the Microsoft Excel template you want to import and close the **Named References** dialog.

8. Right-click the **MS Excel** type, select **Check-In/Out** and select **Check In**.

Alternatively, use the **cae_add_report_templates** utility to update an existing template by selecting the **-update** option. To view the command line help for this utility, type **cae_add_report_templates -h** on the Teamcenter command prompt.

Configure units of measure for simulation integration applications

As a simulation administrator, you can configure the units of measure (UOM) for simulation integration applications by running the quick set up utility. The UOM values required for simulation integration applications are included in the *TC_DATA***tcsim\unit_definitions.csv** file. The quick set up utility calls a script to import the values from the CSV file to the Teamcenter database.

The CSV file for simulation is based on the **unit_definitions.csv** file used by Teamcenter administrators to configure UOM in the Teamcenter environment. For more information, see *Working with the unit of measure definitions file* in *Teamcenter Administration*.

To configure units of measure for simulation integration applications, run the **tcsim_quick_setup.pl script** from the *TC_DATA***tcsim\setup** directory.

Enable indexing for analysts to find recommended simulations

For a simulation analyst, creating a new verification request requires extensive manual review of requirements, CAD data, and input or output parameters before starting the simulation. Finding the most relevant data based on simulation history is challenging, leading to varying success rates among analysts.

Simulation analysts can use the **Recommended Simulations** tab to quickly find the simulation data with the parameters that are most relevant to the verification request you are creating. This ensures that they have a consistent set of comparable results to start their work.

When verification requests are created, the input and output parameters are automatically included based on the selected object. These parameters have data types such as integer, double, boolean, string, or point and properties such as goal, minimum, and maximum values. These properties are indexable.

As an administrator, to index these properties, you must ensure that the **Simulation Process Management with Measurable Attribute for Active Workspace** feature is installed. When this feature is installed, the **SWA_IsSwa0simattrmgmtawTemplateInstalled** site preference is set to **true** by default to make these properties indexable.

Convert NX managed mode files to native mode

NX is used in managed mode at most sites to create and store CAD assemblies directly in Teamcenter. Subsequently, these designs are then imported into simulation tools so they can be modified or used as a starting point to run simulations. Since the assemblies are saved in NX managed mode, they cannot be directly imported into simulation tools unless the assembly is translated from managed to native.

The process is as follows:

- (Teamcenter administrator) Enables the **CAE_Export_NX_Native_Files** site preference to **true** to export the NX files (**.prt**, **.sim**, and **.fem**) to native mode. By default, the preference is set to **false**.

  Ensure that the **PIE_IMF_FOR_CAD** site preference is not enabled when you enable this preference.

- (Simulation administrator) Configures simulation tools to specify additional input and optional input rules for **.prt**, **.sim**, and **.fem** files. For more information, see **Create a simulation tool**.

- (Simulation analyst) Launches the preconfigured simulation tool to convert NX managed mode files to native mode, imports the native mode files for other simulation tools, and continues the analysis work.

Set relations for changing the status of boundary conditions

You can set relations for changing the status of boundary conditions by using site preferences.

Procedure

1. Search for the **CAE_significant_relation_types_Excluded_for_CAEBoundaryCondition** site preference.

   For information about retrieving a list of preferences, see *Where can I get a list of preferences?* in *Teamcenter Preferences*.

   This preference is to set the relations that will *not* cause the CAE status to become out of date on the **CAE 3D Boundary Condition** revision. For example, when a **CAE 3D Analysis** revision contains an input boundary condition, the CAE status should not become out of date on the **CAE Boundary Condition** revision when the **CAE 3D Analysis** is modified.

By default, this value of this preference is set to **TC_CAE_Include** relation type.

2. Search for the **CAE_significant_relation_types_Included_for_CAEBoundaryCondition** site preference.

   This preference is to set the relations that cause the CAE status to become out of date on the **CAE 3D Boundary Condition** revision. For example, when a **CAE 3D Analysis** revision causes its extracted boundary condition to go out of date.

   By default, the value of this preference is set to **CAE0Extract** relation type.

Capture the status of released objects

You can capture the status of released objects by setting the **CAE_allow_cae_status_on_released_objects** site preference.

Procedure

1. Search for the **CAE_allow_cae_status_on_released_objects** site preference.

   For information about retrieving a list of preferences, see *Where can I get a list of preferences?* in *Teamcenter Preferences*.

   This is an optional preference to capture the CAE status changes on released objects.

2. Set to **true** for the system to capture the CAE status changes on released objects.

   By default, it is set to **false**.

3. Save your changes.

Map material revisions from the source structure to the target structure

Map material revisions from the product to the model

Prerequisite

Teamcenter materials management must be installed and ready to use for mapping material revisions.

Process flow for exporting material information

Simulation administrator	1. **Maps material revisions from the product structure** to the model structure by modifying the **datamapping.xml** file. 2. **Defines a simulation tool to export the material information**. This tool is configured for a specific type of solver, for example, Simcenter STAR-CCM+.
Simulation analyst	1. Generates the model structure from a product structure by using data map rules or structure map rules. The generated model structure references the material revisions from the product structure. 2. Selects the generated model structure and runs the preconfigured simulation tool. The material information is exported. 3. Performs the analysis by running a preconfigured analysis tool, for example, Simcenter STAR-CCM+. The analysis tool uses the exported material information for running the analysis. 4. Captures the results from the solver, verifies the results, and sends recommendations.

Map material revisions from the product to the model

You (as a user with DBA privileges) can *uncomment* the **datamapping.xml** and **NodeXMLConfig** files to map material revisions from the product revision (source) to the model revision (target). The simulation analyst can then generate model structures with the mapped material objects using data mapping and structure map rules.

By default, the material revision mapping is not enabled and is commented in the **datamapping.xml** and **NodeXMLConfig** files. You can uncomment these files to specify:

- All associated **Mat1MaterialRevision** objects with the **Mat1UsesMaterial** relation from the product revision (source) you want to map and reference to the corresponding model revision (target) with the same relation.

- (Optional) The material objects you want to map to secondary objects or **CAE 3D Geometry** revisions that are created through data mapping and structure map rules.

1. Locate the **datamapping.xml** file.

   The **datamapping.xml** sample file is located in the *TC_DATA* directory. However, the file used at runtime is managed in a **CAEStructureMap** dataset attached to the item revision indicated by the **CAE_datamap_files_location** preference.

   a. Open the **datamapping.xml** sample file from the *TC_DATA* directory.

b. Search for **MaterialRevision** in this file.

```
<!-- Example: To Map all attached Materials of type Mat1MaterialRevision to
input item revisions of relationship Materials to be referenced to output
CAE 3D Model revisions with same ( Materials ) relationship -->
     <!--Begin MATERIAL_MAP-->
     <!--<xsl:for-each select="$node/smn:NODES/smn:NODE/
         smn:NODE_LINE[@CLASS=&apos;Mat1MaterialRevision&apos; and
         @TYPE=&apos;Mat1MaterialRevision&apos;]">
       <xsl:if test="smn:NODE_LINKS/smn:NODE_LINK/@RELATIONSHIP_TYPE=&apos;
         Mat1UsesMaterial&apos;">
       <xsl:call-template name="MaterialRevision">
       <xsl:with-param name="operationtype">REFERENCE</xsl:with-param>
       <xsl:with-param name="outrelationshiptype">Mat1UsesMaterial
            </xsl:with-param>
       <xsl:with-param name="parenttype">CAEModelRevision</xsl:with-param>
       <xsl:with-param name="materialtype" select="@TYPE"/>
          </xsl:call-template>
          </xsl:if>
          </xsl:for-each>-->
<!--End MATERIAL_MAP-->

<!-- Example: To Map all attached Materials of type
Mat1CompMaterialRevision to
input item revisions of relationship Materials to be referenced to output
CAE
Model revisions with same ( Materials ) relationship -->
     <!--Begin MATERIAL_MAP-->
     <!--<xsl:for-each select="$node/smn:NODES/smn:NODE/
smn:NODE_LINE[@CLASS=
       &apos;Mat1CompMaterialRevision&apos; and
         @TYPE=&apos;Mat1CompMaterialRevision&apos;]">
       <xsl:if test="smn:NODE_LINKS/smn:NODE_LINK/@RELATIONSHIP_TYPE=
           &apos;Mat1UsesMaterial&apos;">
       <xsl:call-template name="CompoundMaterialRevision">
       <xsl:with-param name="operationtype">REFERENCE</xsl:with-param>
       <xsl:with-param name="outrelationshiptype">Mat1UsesMaterial
            </xsl:with-param>
       <xsl:with-param name="parenttype">CAEModelRevision</xsl:with-param>
       <xsl:with-param name="materialtype" select="@TYPE"/>
       </xsl:call-template>
             </xsl:if>
             </xsl:for-each>-->
<!--End MATERIAL_MAP-->
```

c. Uncomment both the material maps.

d. Save the **datamapping.xml** file, and upload it to the dataset attached to the item revision.

2. Locate the **NodeXMLConfig.xml** file.

   The **NodeXMLConfig.xml** sample file is located in the *TC_DATA* directory. However, the file used at runtime is managed in a **CAEStructureMap** dataset attached to the item revision indicated by the **CAE_datamap_files_location** preference.

   a. Open the **NodeXMLConfig.xml** file from the *TC_DATA* directory.

   b. Search for **MaterialRevision** in this file.

```
<!--<smn:NODE_LINE CLASS="Mat1MaterialRevision"
TYPE="Mat1MaterialRevision">
     <smn:ATTR_NODES>
          <smn:ATTR_NODE INTERNAL_NAME="object_name" NAME="Name"
               TYPE="Attribute"/>
          <smn:ATTR_NODE INTERNAL_NAME="object_desc" NAME="Description"
               TYPE="Attribute"/>
          <smn:ATTR_NODE INTERNAL_NAME="item_id" NAME="ID" TYPE="Runtime"/>
          <smn:ATTR_NODE INTERNAL_NAME="structure_revisions"
               NAME="BOM View Revisions" TYPE="Reference"/>
          <smn:ATTR_NODE INTERNAL_NAME="project_ids" NAME="Project IDs"
               TYPE="Runtime"/>
          </smn:ATTR_NODES>
          <smn:NODE_LINKS>
             <smn:NODE_LINK CLASS="ItemRevision"
               RELATIONSHIP_TYPE="Mat1UsesMaterial" TYPE="ItemRevision"/>
          </smn:NODE_LINKS>
</smn:NODE_LINE>-->
<!--<smn:NODE_LINE CLASS="Mat1CompMaterialRevision"
     TYPE="Mat1CompMaterialRevision">
     <smn:ATTR_NODES>
        <smn:ATTR_NODE INTERNAL_NAME="object_name" NAME="Name"
            TYPE="Attribute"/>
        <smn:ATTR_NODE INTERNAL_NAME="object_desc" NAME="Description"
             TYPE="Attribute"/>
        <smn:ATTR_NODE INTERNAL_NAME="item_id" NAME="ID" TYPE="Runtime"/>
          <smn:ATTR_NODE INTERNAL_NAME="structure_revisions"
             NAME="BOM View Revisions" TYPE="Reference"/>
          <smn:ATTR_NODE INTERNAL_NAME="project_ids" NAME="Project IDs"
             TYPE="Runtime"/>
          </smn:ATTR_NODES>
          <smn:NODE_LINKS>
            <smn:NODE_LINK CLASS="ItemRevision" RELATIONSHIP_TYPE=
                "Mat1UsesMaterial" TYPE="ItemRevision"/>
          </smn:NODE_LINKS>
</smn:NODE_LINE>-->
```

   c. Uncomment both the **Mat1MaterialRevision** and **Mat1CompMaterialRevision** nodes.

d. Save the **NodeXMLConfig.xml** file, and upload it to the dataset attached to the item revision.

Define a simulation tool for exporting material revisions

To map material revisions from the product to the model:

1. In CAE Manager, from the main tool bar, click **CAE Configuration→ Simulation Tool Configuration**.

2. In the **SimTool-***tool-ID*-**Root_Tool** panel, click **Create Simulation Tool** and specify a name for the tool you want to configure, for example, *MaterialsMgmtConfigTool*.

3. (Optional) Click the **Input Configuration** tab and select **CAE 3D Model Revision** as the primary input type.

   This is required only if you want to specify additional input rules.

4. Select the **Export Materials** check box, and specify a file name, for example, *mat.xml*.

   The file type can be of any format supported by the solver for exporting the material information.

   This option is available only if Materials Management is installed.

5. Select one of the following options:

   - **Generate full MatML with no transformation** to export a complete **MatML** export.

     MatML is an extensible markup language (XML) developed especially for the interchange of material information.

   - **Select Export Filter** to export to a solver-specific format, for example, Catia or Ansys.

     The default values in this menu are determined by the **IMM_EXPORT_FILTERS** preference.

     > Note:
     >
     > Only XSLT based translators from the preference are supported in the **Select Export Filter**.

6. Create a primary input rule for PLM XML export. Use the **CAEConfiguredDataFilesExportDefault** transfer mode to export a PLM XML file of the model structure along with the associated material IDs.

> Tip:
>
> You can add multiple primary input rules and additional input rules to gather all the information. The system collects material IDs for all the rules and combines them in a single export file without repeating the material ID. This is the export file type you specified in **step 4**.

a. Select **PLMXML Export** in the **Primary Input** section and click the **Add** button to create a traversal rule for exporting the primary inputs.

b. Specify a rule name and select the **CAEConfiguredDataFilesExportDefault** option from the **Transfer Mode** menu.

c. In the **File Name** box, specify a name for the export file, for example, **exp.xml**.

d. To create the traversal rule, click **OK**.

7. (Optional) Specify additional inputs as appropriate. The following is only an example.

You can specify additional inputs only if you have selected an option as the primary input type. In the following example, **CAE 3D Model Revision** is specified as the primary input type in **step 3**.

a. Select the **PLMXML Export** option as the additional input and click **Add** to create a traversal rule for exporting the additional inputs.

b. Specify a rule name in the **Rule Name** box, for example, **addl_rule1**.

c. Specify a file name for the export file in the **File Name** box, for example, **addl_rule1.xml**.

d. From the **Transfer Mode** menu, select **CAEConfiguredDataFilesExportDefault**.

e. Specify a traversal path as follows:

Originating Type	Relation	Destination Object	Relation Direction	Relation Direction
CAE 3D Model Revision	CAE Source	Item Revision	Primary to Secondary	CAE 3D Geometry Revision
CAE 3D Geometry Revision	Materials	Item Revision	Primary to Secondary	Material Revision

f. To create this rule, click **OK**.

8. (Optional) To copy all the information from the source tool and to create a new tool by cloning the source tool, click **Clone Simulation Tool** in the view toolbar.

   When you clone tools, the name of the child tool is the same as that of the source and the new tool is added as the last child of the parent tool.

9. (Optional) To export the tool configuration, from the Teamcenter command prompt, run the **cae_migrate_tool_configuration** utility. To view the command line help for this utility, type **cae_migrate_tool_configuration -h** on the Teamcenter command prompt.

   To import a tool configuration you have exported, run the same utility.

To create a simulation tool, follow the procedures outlined in **Define simulation tools**.

Create a model structure using a workflow process

Configure the workflow process for running structure maps

You (as a user with DBA privileges) can configure a workflow process to allow simulation analysts to create model structures by using structure map rules. The workflow process uses the Dispatcher **AsyncService** to process asynchronous requests from Teamcenter.

The process for configuring a workflow process is as follows:

1. **Configure the Dispatcher AsyncService to activate the translator, create a Dispatcher client access rule, and set the SOA URL field in the site object of your local site and each remote site to which you need to send asynchronous requests**.

2. **Create a workflow process to specify the structure map item ID and set the execution type as async (default value)**.

After the workflow process is configured, a simulation analyst can open a product structure and select **File→New→Workflow Process** in CAE Manager to create a model structure using predefined structure map rules.

Configure Dispatcher for the async mode

1. Open the **translator.xml** file from the *Dispatcher_Root***Module\conf** directory and search for **AsyncService**.

2. Set the **isactive** attribute to **true** to activate this translator.

3. Edit the *CHANGE_ME* tags in the **asyncservice.bat** (Windows) or **asyncservice.sh** (Linux) file in the *Dispatcher_Root***Module\Translators\asyncservice** directory.

   Example:

```
set TC_ROOT=CHANGE_ME
```

Change to:

```
set TC_ROOT=D:\Siemens\Teamcenter11
```

If you have installed Teamcenter in the **D:\Siemens\Teamcenter11** directory.

Similarly, change all *CHANGE_ME* tags in the **asyncservice.bat** or **asyncservice.sh** file.

4. Create a dispatcher client access rule.

   After installing Dispatcher, you must add a rule to the access rule tree permitting the translation service proxy user to update attributes of a **DispatcherRequest** object. If this access rule is not created correctly, the dispatcher client reports errors.

5. Set **SOA URL** for your site.

   Use the Organization application to create a site and add the middle tier system's URL to the **SOA URL** field in the site object of your local site and each remote site to which you need to send asynchronous requests. This should have the same base value as that used to set up and configure the middle tier and to run Active Workspace and Rich Client.

   Example:

   http://localhost:7001/tc

   If you use SSL to access Teamcenter, that is, if the address starts with **"https:"**, ensure that a proper certificate is stored in the business logic server's trust certificate store file.

6. In the Organization application, create a user in the **Teamcenter dba** group with the same *OS name* as your user name on the machine you are running as a Dispatcher module. This user name is used by **asyncservice** for auto login.

Configure a workflow process for running structure maps

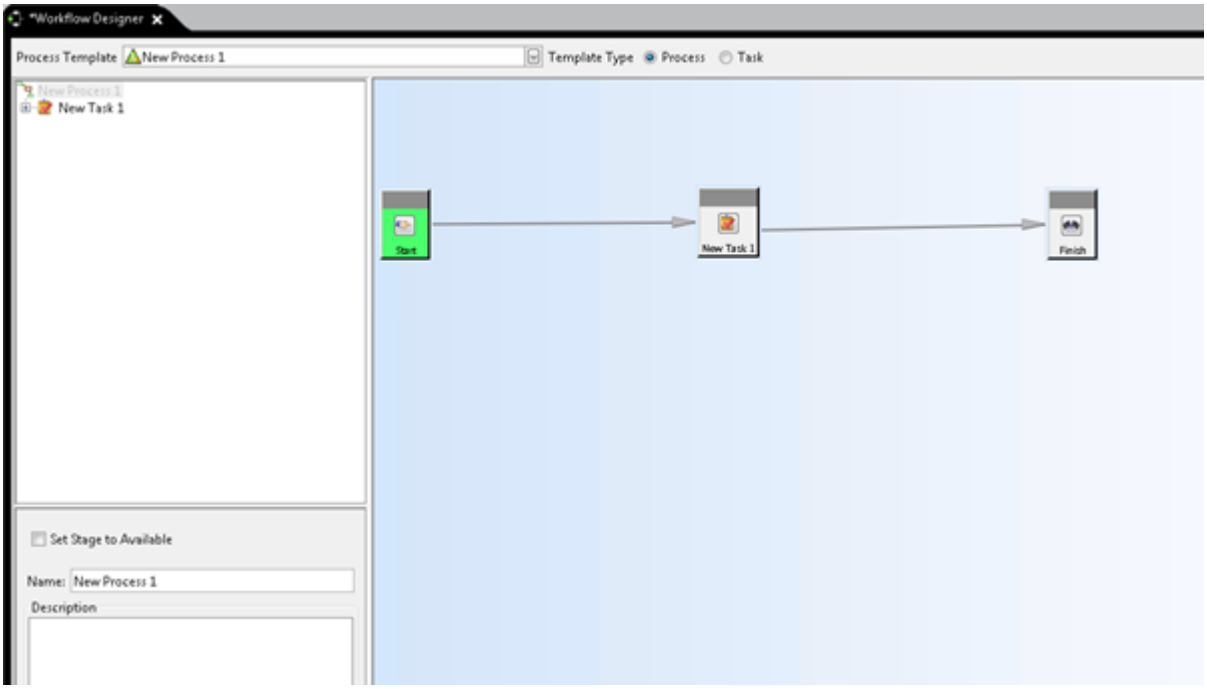
An administrator (user with DBA privileges) can configure a workflow process for simulation analysts to create a model structure by using structure maps rules.

1. In Workflow Designer, log on as a user with DBA privileges.

2. Click **File** → **New Root Template**.

3. Specify a template name and choose **Process** as the template type.

4. Create a simple task and join it from **Start** and end it with **Finish**.



5. Right-click the new task you created and select **Task Properties**, and click **Display the Task Handlers Panel**.

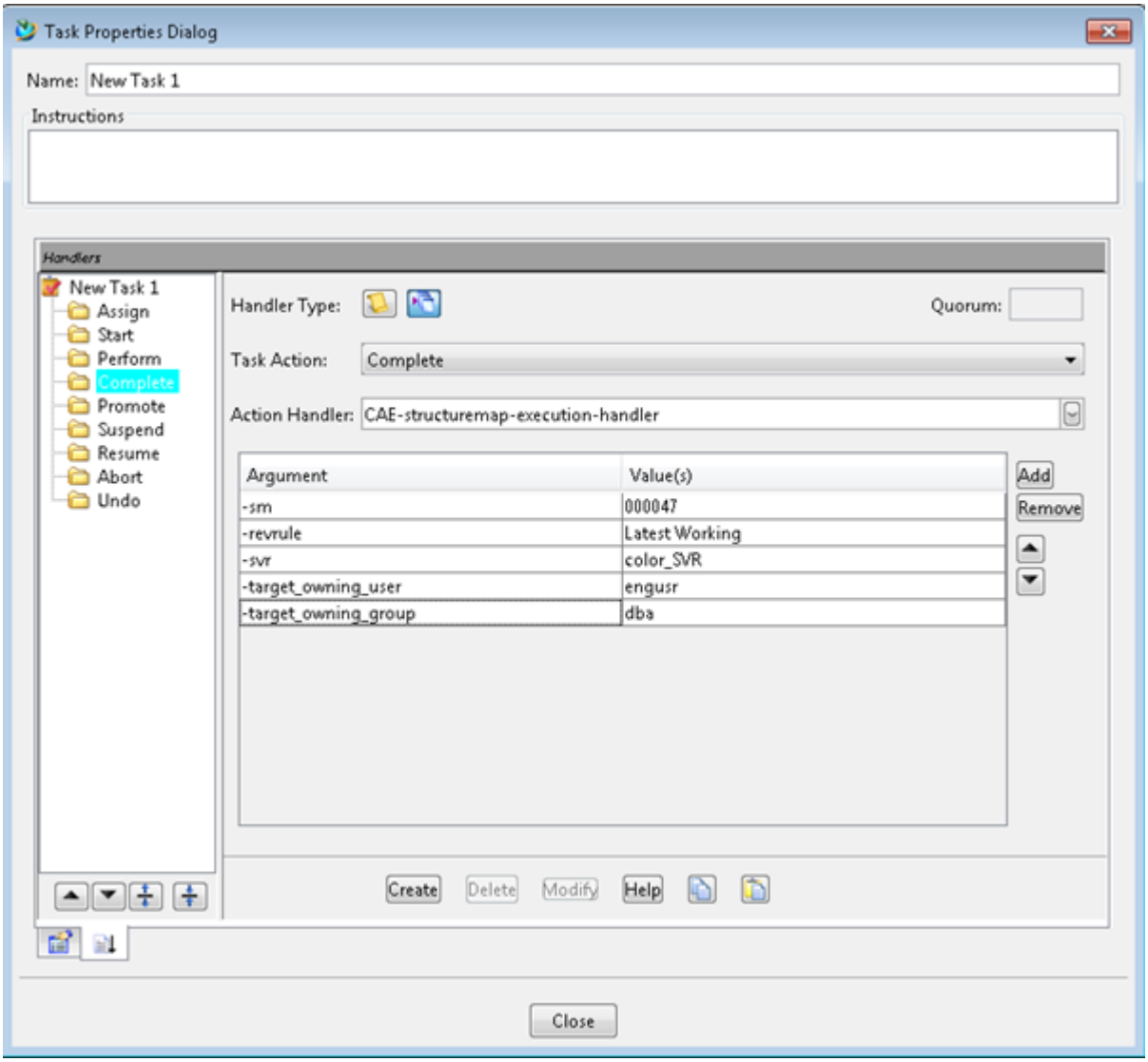
6. (Optional) In the **Attributes** panel, select the **Process in Background** option. This option is available only if the **EPM_task_execution_mode** preference is set to **CONFIGURABLE**.

   If you set the **EPM_task_execution_mode** preference to **BACKGROUND**, all tasks run in the background.

   By default, it is sync mode.

7. In the **Task Properties Dialog**, click **Display the Task Handlers Panel**.

8. Select the **Complete** task action and select **CAE-structuremap-execution-handler** as the action handler.



Specify arguments and values as follows.

Argument	Description
-sm	(Mandatory) Specifies the structure map item ID.
-revrule	(Optional) Specifies the revision rule, for example, **Latest Working**.
-svr	(Optional) Specifies the saved variant rule (SVR). If there are multiple SVRs, use a comma (**,**) as a separator. If the SVR is not available on the root of the input product item revision you have opened, you can define the value in the following format: • *item_id***::svr**: The system considers the SVR available on the latest working revision of the item to which the *item_id* corresponds.

Argument	Description
	• *item_id*::*rev_id*::**svr**: The system considers the SVR available on the revision of the item to which the *item_id* and *rev_id* corresponds.
-target_owning_user	(Optional) Specifies the owner of the workflow process for the resulting CAE structure. If not specified, the current workflow process owner is the owner of the CAE BOM.
-target_owning_group	(Optional) Specifies the group name of the owning user of the workflow process for the resulting CAE structure. If not specified, the current workflow process owner's group is the owning group of the CAE BOM.

9. To create the action handler, click **Create**.

> Tip:
>
> You can specify multiple action handlers for each structure map you want to execute and change the value for the structure map item ID or **-sm** argument.

Configure file upload rules to support analysis on local desktops

Why configure file upload rules?

At some sites, not all simulation tools are integrated with Teamcenter. Simulation analysts may prefer to run the simulation tools on their local desktops and periodically upload and download the data to or from Teamcenter as needed.

A user with DBA privileges configures the file upload rules at the site level, the group administrator configures them at the group level, and the simulation analyst configures them at the user level. All types of users can use the **File Upload Rules Configuration** view to create file upload rules for analysts to upload or download the analysis files to or from Teamcenter. After configuring the file upload rules, all users can use the **File Explorer** view to upload or download analysis files to or from Teamcenter based on predefined upload rules available at the site, group, or user level.

You can specify or create:

- Site name, group name, or user name.
- Primary input types, for example, item revisions.
- Dataset creation options.
- File upload rules by specifying a traversal rule. This defines the path from the input item revision to the dataset that holds the various output files.

Specify File Explorer options

1. In CAE Manager, click **Edit→Options→CAE→File Explorer**.

2. In the **Options** dialog box, hover over a field to find the preference name associated with it.

   For information about retrieving a list of preferences, see *Where can I get a list of preferences?* in *Teamcenter Preferences*.

3. To specify a default working directory for analysts to upload or download files, click **Browse** in the applicable platform and select the appropriate directory.

4. To specify the default file upload conflict resolution option, select one of the following:

   - **Upload** to upload the file to Teamcenter and overwrite the file with the same file name.
   - **Rename and Upload** to rename the file and upload it to Teamcenter.
   - **Skip** to skip the upload of the file to Teamcenter.

5. To specify the default file download conflict resolution option, select one of the following:

   - **Overwrite** to download the file to a local machine and overwrite the file with the same file name.
   - **Rename and Upload** to rename the file and download it to a local machine.
   - **Skip** to skip the download of the file to the local machine.

6. To specify an alternate location folder for the file upload or download rule set configuration, click **Browse** and select an alternate folder.

   The default location is the **Newstuff** folder in Teamcenter.

Override the group administrator restriction to create configuration objects

Only a group administrator can create configuration objects at the group level. To avoid this restriction, you can select the **Bypass Group Administrator** option.

1. In CAE Manager, click **Edit→Options→CAE→General**.

2. In the **Options** dialog box, hover over a field to find the preference name associated with it.

   For information about retrieving a list of preferences, see *Where can I get a list of preferences?* in *Teamcenter Preferences*.

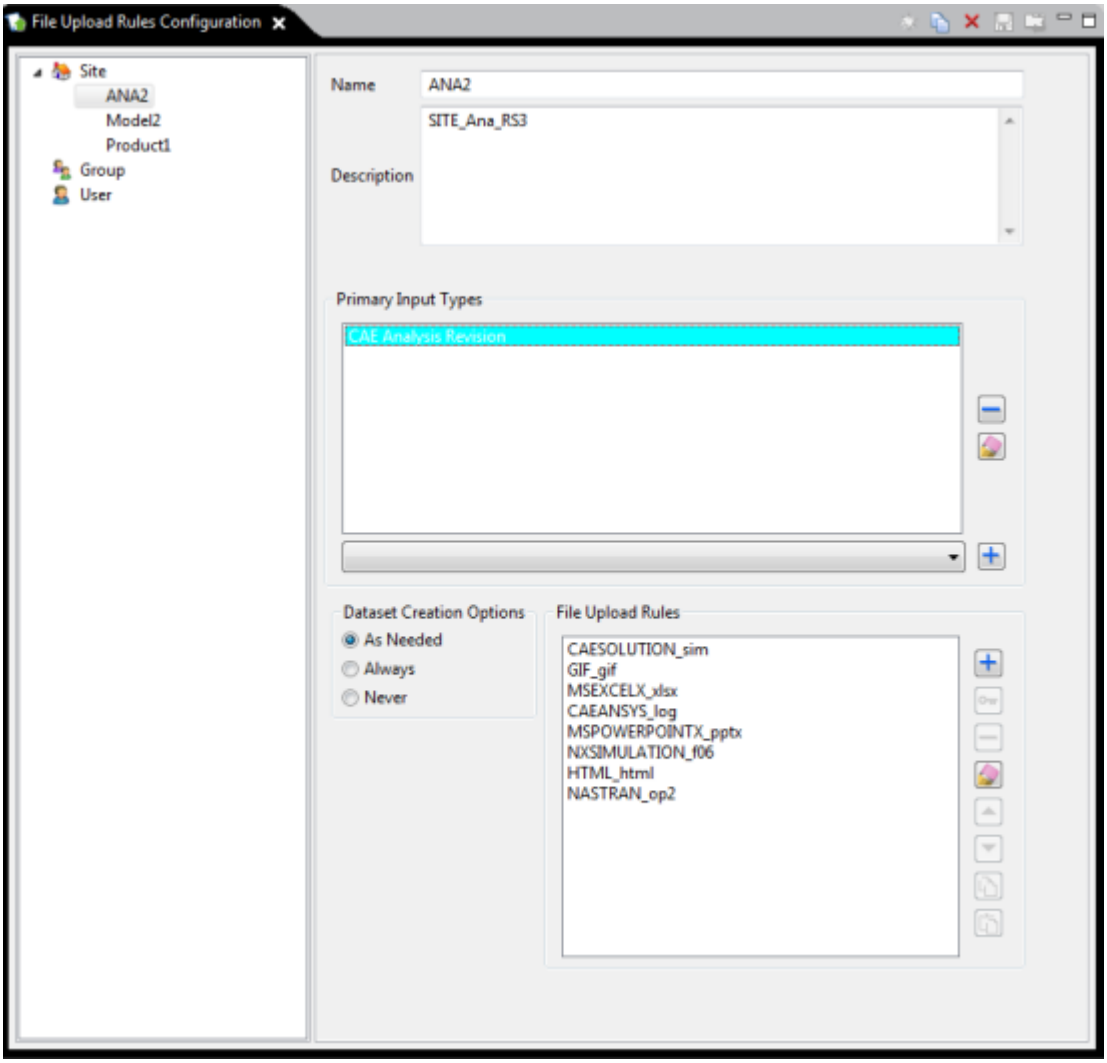
3. Click the **CAE Configuration** tab.

4. Select the **Bypass Group Administrator** check box.

   By default, it is set to **false**.

Configure and deploy file upload rules

1. In CAE Manager, click the **CAE Configuration** toolbar menu, and choose **File Upload Rules Configuration**.



2. Select the **Site**, **Group**, or **User** scope option.

3. To create a file upload rule, click **Create New File Upload Rule Set** in the view toolbar.

4. Specify a file upload rule name in the **Name** box.

This is the name that appears in the **File Upload Rule Set** menu when the simulation analyst opens the **File Upload** dialog box.

5. (Optional) Specify a description in the **Description** box.

6. To add a primary input type, in the **Primary Input Types** area, select a primary input type and click **Add** +.

7. In the **Dataset Creation Options** area, select one of the following options:

   - **As Needed** to create datasets as needed while uploading files, using the **File Explorer** view.

     This is the default option.

   - **Always** to create datasets each time while uploading files, using the **File Explorer** view.

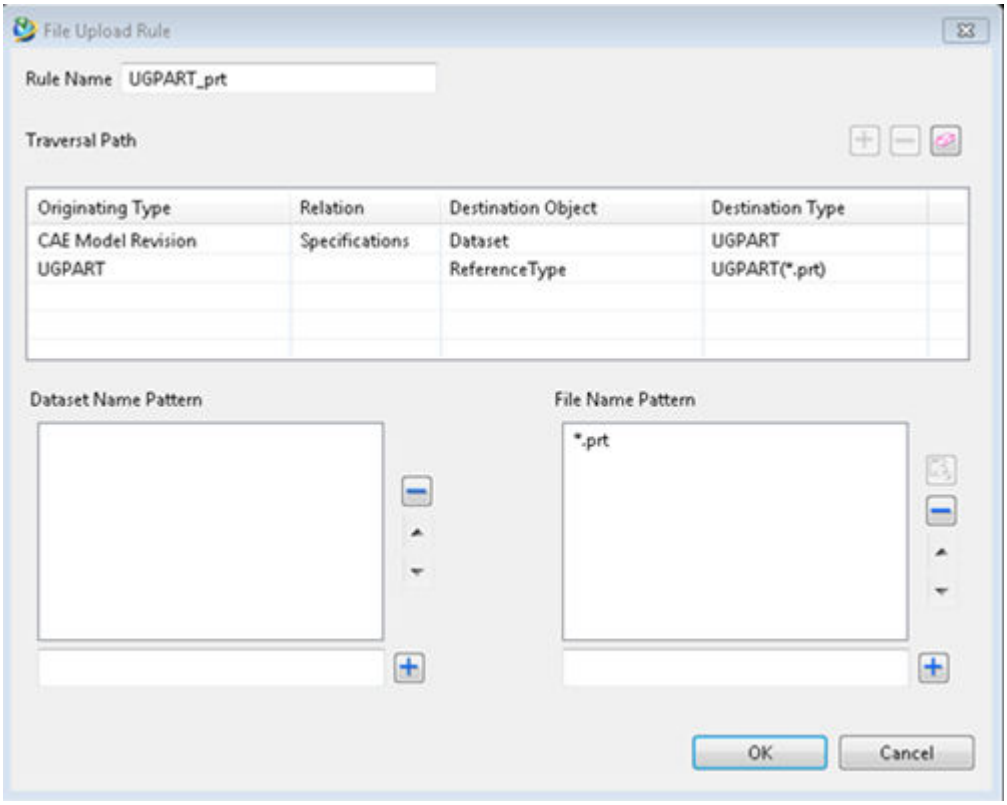
   - **Never** to specify that datasets should not be created, but use existing ones that match the configuration while uploading files, using the **File Explorer** view.

8. To create a file upload rule, in the **File Upload Rules** area, click **Add** + to open the **File Upload Rule** dialog box.

9. Specify a name for the rule in the **Rule Name** box.

10. Create a traversal rule to define the path from the input item revision to an item revision that holds the various output files.

    a. In the **Traversal Path** area, click **Add** + to activate all options.



b. In the **Originating Type** column, click a cell and select an appropriate option, for example, **CAE 3D Analysis Revision**.

> Note:
>
> The options available here depend on the input types you have added in the **Primary Input Types** box.

c. In the **Relations** column, click a cell and select an appropriate option, for example, **Specifications**.

d. In the **Destination Object** column, click a cell and select an appropriate option, for example, **Dataset**.

e. In the **Destination Type** column, click a cell and select an appropriate option, for example, **Nastran**.

The following is an example of a traversal path to create a destination type from a **CAE 3D Analysis Revision** item revision (**Originating Type**) to a Nastran file containing data (**Destination Type**):

Originating Type	Relation	Destination Object	Destination Type
CAE 3D Analysis Revision	**Specifications**	**Dataset**	**Nastran**
Nastran	Not applicable	**Reference Type**	**CAE0OP2 (*.op2)**

f. In the **Dataset Name Pattern** box, click the **Add** button **to add a dataset name pattern you want to include**, for example, **FILENAME"-nastran"**.

> Note:
>
> If you do not specify a dataset-naming pattern, a new dataset is created to hold the resulting output files. Simulation Process and Data Management assigns a default name comprising the item revision's **itemID** and **itemRev** to the new dataset.

g. In the **File Name Pattern** box, click the **Add** button to add a file name pattern you want to include, for example, ***.fem**.

11. When files are uploaded to multiple item revisions of the same type, the **Filename to Item Revision Mapping** option helps the system resolve any ambiguities. When files are uploaded to a single item revision, this option is ignored.

    (Optional) To specify multiple mapping definitions for a given file name pattern, click the **Filename to Item Revision Mapping** option.

    You can specify mapping information for each file extension.

    The keywords supported are **ITEMID**, **ITEMNAME**, **REVID**, **REVNAME**, **SEQNUM**, and **REVDESC**.

    You can use a combination of the above keywords and fixed strings in double quotes. For example, **"Mesh-"ITEMID"-"REVID**.

    You can use the following operators in a keyword search:

    - **Contains ()**. For example, **Contains (ITEMID)**.
    - **Begins with ()**
    - **Ends with ()**

    Example for item and item revision:

    002868-Blade Analysis

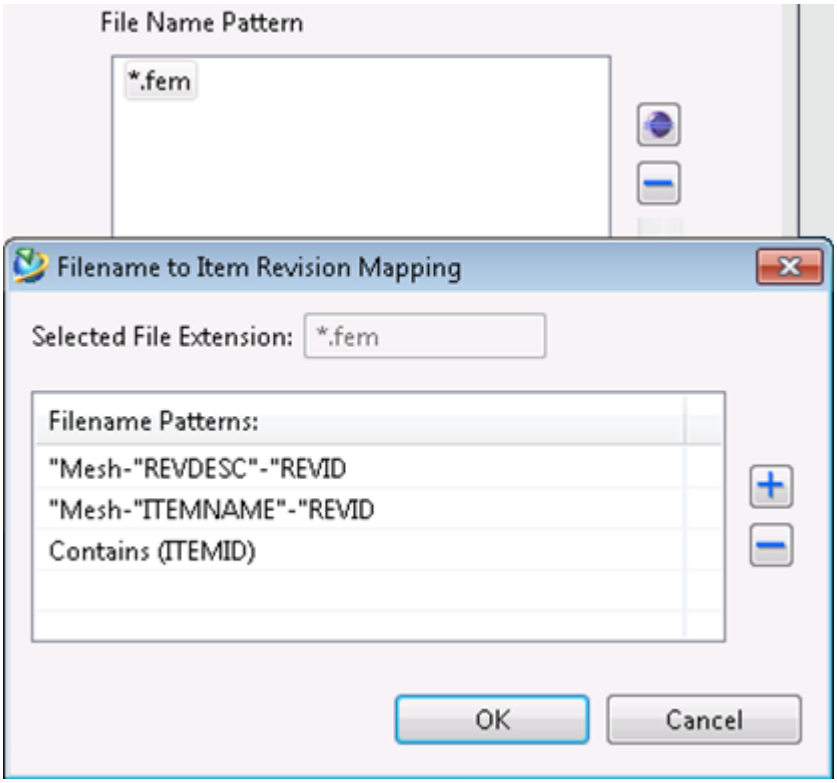
    002868/B;2-Blade Analysis Revised (Revision Description: Thermal Analysis)

The keywords resolve to the following values:

- **ITEMID=002868**
- **ITEMNAME=Blade Analysis**
- **REVID=B**
- **REVNAME=Blade Analysis Revised**
- **SEQNUM=2**
- **REVDESC=Thermal Analysis**

Examples:

- **“Mesh-"REVDESC"-"REVID**
- **"Mesh-"ITEMNAME"-"REVID**
- **Contains (ITEMID)**



> Note:
>
> The keyword **ITEMREVID** is not supported.

12. To save the file upload rule set, click **Save File Upload Rule Set** in the view toolbar.

13. To clone the file upload rule set, click **Clone File Upload Rule Set** in the view toolbar.

14. To delete the file upload rule set, click **Delete File Upload Rule Set** in the view toolbar.

15. To reset the file upload rule set to the last saved state, click **Reset File Upload Rule Set** in the view toolbar.

Create a file upload rule for folders

You can configure file upload rules to import a folder structure into Teamcenter.

1. In CAE Manager, click the **CAE Configuration** toolbar menu, and choose **File Upload Rules Configuration**.

2. Select the **Site**, **Group**, or **User** scope option.

3. To create a file upload rule, click **Create New File Upload Rule Set** in the view toolbar.

4. Specify a file upload rule name in the **Name** box, for example, **Map Folders**.

   This is the name that appears in the **File Upload Rule Set** menu when the simulation analyst opens the **File Upload** dialog box.

5. (Optional) Specify a description in the **Description** box.

6. To add a primary input type, in the **Primary Input Types** area, select a primary input type and click **Add**.

7. In the **Dataset Creation Options** area, select:

   - **As Needed** to create datasets as needed while uploading files, using the **File Explorer** view.

     This is the default option.

   - **Always** to create datasets each time while uploading files, using the **File Explorer** view.

   - **Never** to specify that new datasets must not be created and that existing ones that match the configuration while uploading files must be used, using the **File Explorer** view.

8. To create a file upload rule, in the **File Upload Rules** area, click **Add** + to open the **File Upload Rule** dialog box.

9. Specify a file upload rule name in the **Rule Name** box.

   This is the name that appears in the **File Upload Rule Set** menu when the simulation analyst opens the **File Upload** dialog box.

10. Click **Add** + to create a traversal path.

    Example:

Originating Type	Relation	Destination Object	Destination Type
CAEAnalysisRevision	**Specifications**	**CAE Folder**	**CAE Folder**

    The system attaches a **CAE Folder** to another **CAE Folder** or to an item revision using the **Has CAE Folders** relationship.

    When you configure the rule, you must set the destination object and the destination type columns in the traversal path as **CAE Folder** type. The file name pattern defaults to ***.*** for the **CAE Folder** type. You cannot change this.

    No additional lines are required for the traversal path. A file upload rule can contain only one rule that maps to a given item revision type and a CAE folder.

Configure the workflow process to release CAE folders

As a simulation administrator, you can configure a workflow process by using the **CAE-attach-related-cae-folder-objects** action handler to allow simulation analysts to release the item revision containing the CAE folder structure and it contents. Analysts use the CAE folder structure to manage different file types from different simulation tools when they work in offline mode.

1. Create a new workflow process template using the Workflow Designer.

   For more information about creating workflow process templates, see *Workflow Designer*.

2. From the **Action Handler** menu, choose **CAE-attach-related-cae-folder-objects**.

3. Specify arguments and values as appropriate.

   For more information about arguments and values, see **CAE-attach-related-cae-folder-objects** in *Workflow Designer*.

Configure properties in dialog boxes

Configure properties in dialog boxes

You can configure the properties in the **Multiple Object Save As** and **Multiple Object Revise** dialog boxes using the **Operation Descriptor** tab in Business Modeler IDE.

In BMIDE, make the following changes:

- Set the **performDeepCopy** property on the **Item Revision** to **false** to hide this unused property from being displayed in the **Multiple-Replace Item Revision** dialog box.
- Specify a display name for the **fnd0CheckOutOnRevise** property in the **Modify OperationInput Property** dialog box to show some user friendly text in the **Multiple-Replace Item Revision** dialog box.

Configure properties in the Multi-Replace Item Revision dialog box

You can customize the properties in the **Multiple-Replace Item Revision** dialog box using the **Options** dialog box. This dialog box is available from the **View** menu in CAE Manager.

1. In CAE Manager, click **Edit→Options→CAE→UI Configurations**.
2. In the **Options** dialog box, hover over a field to find the preference name associated with it.

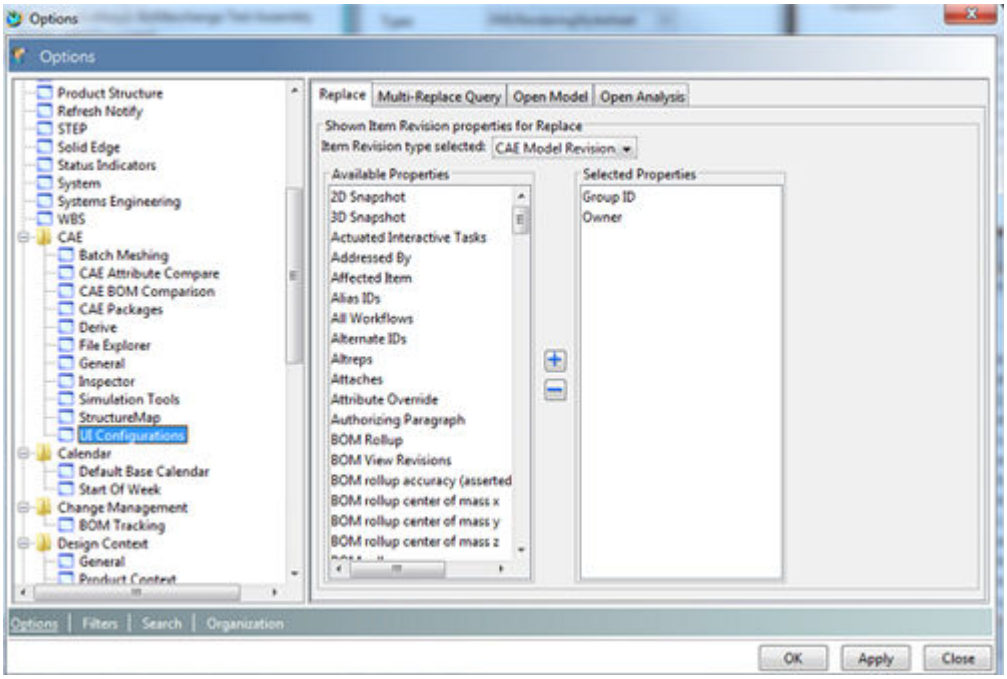
   For information about retrieving a list of preferences, see *Where can I get a list of preferences?* in *Teamcenter Preferences*.
3. Customize columns in the **Selected and Matching Item Revisions** section of the **Multiple-Replace Item Revision** dialog box.

   a. Click the **Replace** tab.

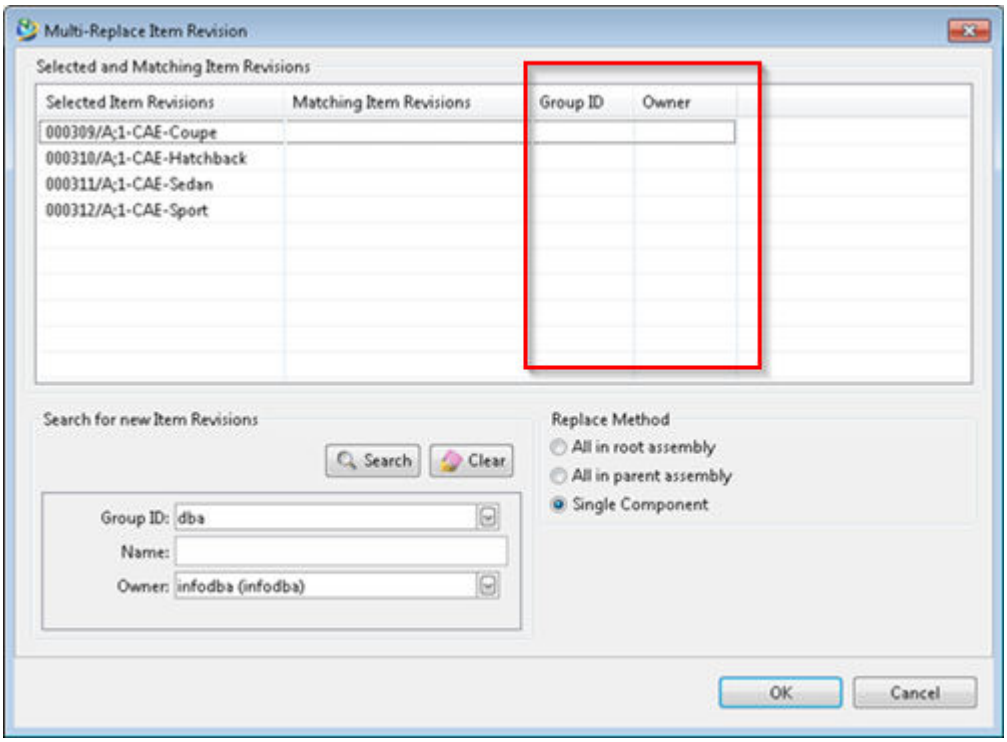
   b. From the **Item Revision type selected** menu, select the appropriate revision type.

      By default, it is the **CAE 3D Model Revision** type.

   c. To customize properties in the **Multi-Replace Item Revision** dialog box in CAE Manager, select the properties you want to include from the **Available Properties** box and move them to the **Selected Properties** dialog box.



These default properties are displayed as columns in the **Selected and Matching Item Revisions** section of the **Multiple-Replace Item Revision** dialog box.



4. Customize options in the **Search for new Item Revisions** section of the **Multiple-Replace Item Revision** dialog box.

a. Click the **Multi-Replace Query** tab.

b. From the **Item Revision type selected** menu, select the appropriate revision type.

   By default, it is **CAE 3D Model Revision** type.

c. To customize properties in the **Multi-Replace Item Revision** dialog box in CAE Manager, select the properties you want to include from the **Available Properties** box and move them to the **Selected Properties** dialog box.

   The properties that are configured for search are displayed in the **Selected and Matching Item Revisions** section of the **Multiple-Replace Item Revision** dialog box.

Run scripts to quickly set up sample configurations

Set up sample configurations

After you install the Simulation Process and Data Management template on a virtual machine or a sandbox environment, you can run scripts to quickly set up sample configurations. This provides an easy setup for presales or customers who want to explore the Simulation Process and Data Management capabilities.

These scripts are available from the *TC_DATA***tcsim****setup** directory. You can run them from the Teamcenter command prompt by switching to that directory:

Script name and usage example	Description
tcsim_quick_setup.pl -u=*DBAUserName* **-p=***DBAUserPassword* **-pf=***DBAUserPasswordFile*	This script calls the following scripts: • **tcsim_setup_user.pl** • **tcsim_setup_config.pl** • **tcsim_setup_example_configs.pl** Note: A user with system administrator or group administrator privileges must run this script.
tcsim_setup_user.pl -u=*DBAUserName* **-p=***DBAUserPassword* **-pf=***DBAUserPasswordFile* **-simusername=***UserNameToCreate* **-simuserpw=***UserPasswordForCreatedUser*	Creates a **Sam Analyst** user under the **Engineering** group with the **Analyst** role. The **User ID** of this user is **sam**.

Script name and usage example	Description
	Note: A user with system administrator or group administrator privileges must run this script.
tcsim_setup_config.pl -u=*UserName* **-p=***UserPassword* **-pf=***UserPasswordFile*	This script imports: • Data map configurations. • Tool launch client configurations.
tcsim_setup_example_configs.pl -u=*UserName* **-p=***UserPassword* **-pf=***UserPasswordFile*	This script imports: • Sample structure map configurations. • Sample tool configurations • Sample CAE package configurations. • Sample derivative rules at the site level. • Sample model and analysis dashboard configurations at the site level. • Sample file upload rule configurations at the site level.

For more information on running these scripts, from the Teamcenter command prompt, type the script name followed by **-h** or **-help**, for example, **tcsim_quick_setup.pl -h**.

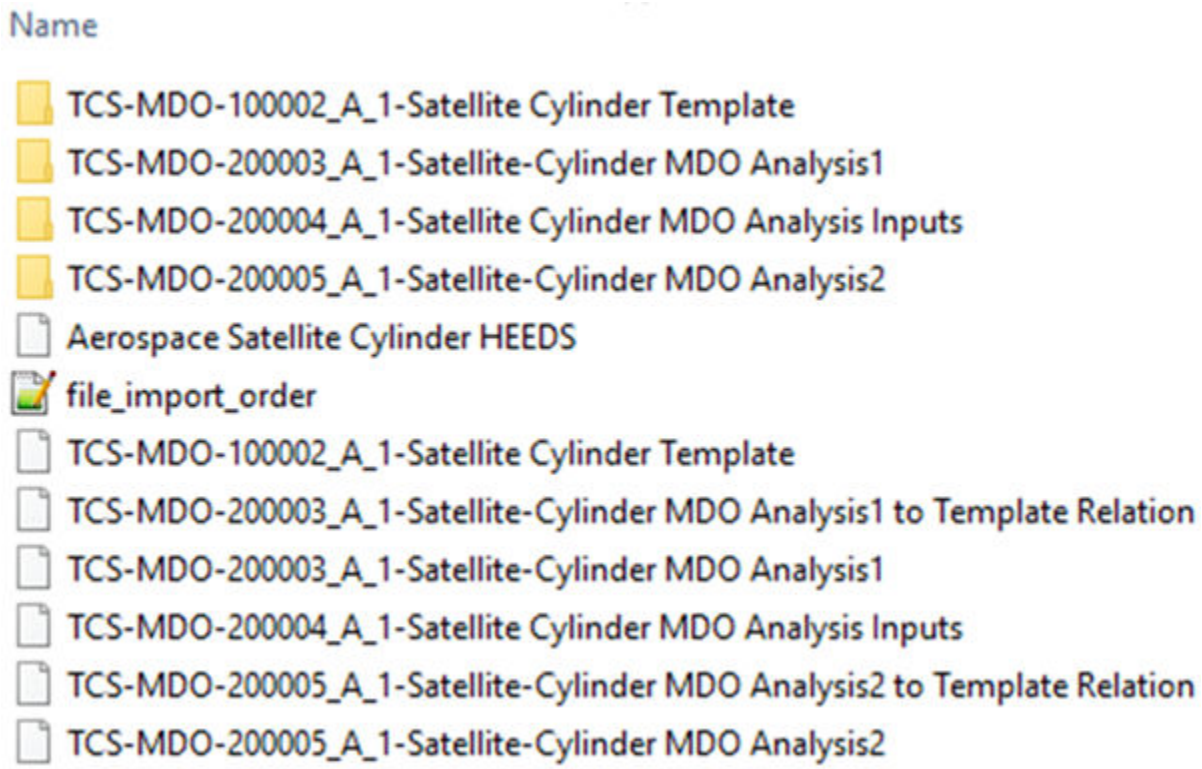
After running these scripts, you can log on to the Teamcenter rich client as a **sam** user and use the sample configuration to run the Simulation Process and Data Management application.

After running the quick setup script, in CAE Manager, on the main toolbar, choose **CAE Configuration** → **Simulation Tool Configuration**. The **SIMTOOLS** configuration view displays the default preprocessors, solvers, postprocessors, and other utilities.

You can also log on to Active Workspace as a **sam** user to open the **Analyst** workspace. This is the default workspace for this user.

How is the CAE sample data packaged?

The CAE data package contains the PLM XML files you want to import. The package contains the CAD and other data and a **file_import_order.txt** file in a folder.



When you **import the CAE data using rich client or Active Workspace**, the package folders are compressed separately and selected as input.

The first line of the **file_import_order.txt** file is **TopLevelFolder**:*Top Level Folder Name*. The top level folder is one of the folders imported as a part of the CAE data package and is pasted under the **Home** folder of the user or the selected folder in Teamcenter or Active Workspace in the case of UI import.

The **file_import_order.txt** file has the PLM XML file names in the sequence in which they are imported. The last PLM XML file specified is always the file with the folder structure data.

```
TopLevelFolder:TcSim Sample Data
TCS-MDO-100002_A_1-Satellite Cylinder Template.xml:CAE_incremental_import
TCS-MDO-200004_A_1-Satellite Cylinder MDO Analysis Inputs.xml:CAE_incremental_import
TCS-MDO-200003_A_1-Satellite-Cylinder MDO Analysis1.xml:CAE_incremental_import
TCS-MDO-200003_A_1-Satellite-Cylinder MDO Analysis1 to Template Relation.xml:CAE_incremental_import
TCS-MDO-200005_A_1-Satellite-Cylinder MDO Analysis2.xml:CAE_incremental_import
TCS-MDO-200005_A_1-Satellite-Cylinder MDO Analysis2 to Template Relation.xml:CAE_incremental_import
Aerospace Satellite Cylinder HEEDS.xml:CAE_incremental_import
```

You can use different transfer modes to export the data and the folder structure, respectively. To export:

- The PLM XML files while creating the CAE data package, use the **CAEConfiguredDataFilesExportDefault** transfer mode.
- The folder structure in PLM XML, use the **CAEExportFolderWith1LevelContent** transfer mode.

If you use the **cae_import_data utility**, the system imports the CAE data packages to the directory you specify in the **import_directory** parameter.

Import the sample CAE data

Using the rich client

1. In CAE Manager, select a folder from the **Home** view.
2. Click **Tools→Import CAE Data**.
3. To import the data package, in the **Import CAE Data** dialog box, click **Add** and browse to the location containing the sample data.

   The data package is imported asynchronously. The system displays a notification sent through email with the log details when the import is complete.
4. Access your **Mailbox** to find the email notification with the log details.

   The data is imported into the folder you selected in step 1.

Using Active Workspace

1. Click **FOLDERS** and select a folder.
2. Click **New** > **Import CAE Data**.
3. Click **Choose file** and browse to the location containing the sample data.

   > Tip:
   >
   > If you use Google Chrome, you can drag and drop the file. This action is not supported in Internet Explorer.
4. Click **Alerts** to find the notification with the log details.

   The data is imported into the folder you selected in step 1.

Using the cae_import_data utility

You can import sample CAE data from the operating system to Teamcenter using the **cae_import_data** utility. To view the command line help for this utility, type **cae_import_data -h** on the Teamcenter command prompt.

Import updated GIT integration files

Presales and implementation teams can quickly import updated GIT integration files using the **tcsim_import_int_def_files.pl** script as per customer requirements.

For more information about prerequisites and running this script, open a Teamcenter command prompt and type **tcsim_import_int_def_files.pl -h**.

After running this script, the system imports integration definition files to existing datasets. If the datasets do not exist in Teamcenter, the script creates the datasets and imports the files. After modifying the files as per your processes, you can run the script again to import the changed files.

Import integration definition files

You can use the **tcsim_import_int_def_files** utility in the *TC_DATA***tcsim\setup** directory to import integration definition files to existing datasets. For more information, browse to this directory from a command prompt with administrative privileges and type **tcsim_import_int_def_files -h**. This utility is primarily used by presales.

Quick start for configuring Simcenter HEEDS

Process flow for HEEDS analysis

This quick start is designed to help the lead or the expert HEEDS user configure simulation tools and the HEEDS analysts to launch the preconfigured simulation tools.

These topics provide only a high-level overview of how to configure and launch simulation tools. For more information, see the respective topics.

- **Configure simulation tools to launch preprocessors, solvers, and postprocessors**
- For more information about launching preconfigured simulation tools, see *Launch simulation tools* in *Simulation Process and Data Management on Rich Client — Usage* in the Teamcenter help.

Simulation administrator	*Sets up the sample tool configurations by running the* **tcsim_quick_setup.pl script**. The lead or an expert HEEDS user can use these default tools as a starting point to configure tools.
Lead or an expert HEEDS user	**Creates a HEEDS project from scratch**. 1. Creates the **CAE MDAO Model** business object in CAE Manager. 2. Creates the HEEDS project by running the **Simcenter HEEDS - Create or Modify Template** tool.

HEEDS analyst	**Updates the metadata of the existing CAE MDAO template in Teamcenter**. 1. Creates the **CAE MDAO Model** business object in CAE Manager. 2. Updates the HEEDS project by running the **Simcenter HEEDS - Extract Template Info** tool. **Releases the CAE MDAO Model business object.**. 1. Opens the **CAE MDAO Model** business object in CAE Manager. 2. Uses the **TCM Release Process** to start the workflow process and releases the business object. **Extracts the information for the analysis**. 1. Creates the **CAE MDAO Analysis** business object in CAE Manager in the context of the released **CAE MDAO Model** business object 2. Extracts information by running the preconfigured **Simcenter HEEDS - Extract Study Info** tool. 3. (After the tool launch is complete) Searches for the input file and attaches it to the **CAE MDAO Analysis** business object you created. **Executes the analysis in an interactive mode or a non-interactive mode (batch process)**. 1. Opens the **CAE MDAO Analysis** business object. 2. Executes the analysis in an interactive mode by running the preconfigured **Simcenter HEEDS - Execute Analysis** tool. OR Executes the analysis in a non-interactive mode by running the preconfigured **Simcenter HEEDS - Execute Analysis (Batch)** tool. **Post processes the results**. 1. Opens the **CAE MDAO Analysis** business object. 2. Post processes the results by running the preconfigured **Simcenter HEEDS - Post Processing** tool.

Create or modify a HEEDS project

The following procedures are to be performed by a lead or an expert HEEDS user.

- Create the CAE MDAO template
- Run a preconfigured tool to update the CAE MDAO template
- Run a preconfigured tool to update the metadata of the existing CAE MDAO template
- Release the CAE MDAO template

Create the CAE MDAO template

1. In CAE Manager, select a folder in the **Home** view, for example, **Newstuff**.
2. Click **File** → **New** → **CAE Item**.
3. In the **New CAE Item Wizard** dialog box, select **CAE MDAO Model**, and click **Next**.
4. Specify a name and click **Finish**.

Run a preconfigured tool to update the CAE MDAO template

You can use the **Simcenter HEEDS - Create or Modify Template** tool when you want to create a HEEDS project from scratch. This tool launches the HEEDS application and creates a project (***.heeds**) file in the HEEDS application.

You can use also this tool to modify an existing template. It launches the HEEDS application and modifies the configuration. After the user saves the configuration in HEEDS, all the information is stored in the ***.heeds** file. When you exit the HEEDS application, the CAE MDAO template objects are updated in Teamcenter. You must not update the MDAO template information from Teamcenter.

For more information about this tool, click **Simulation Tools** → **Simulation Tool Help** → **MDAO** → **Create or Modify Template**, and click **Show Detailed Description**.

1. Click **CAE Configuration** → **Simulation Tool Configuration** on the main toolbar.
2. Select **SIMTOOL***version***MDAO**, expand it, select the **SIMTOOL***version***Simcenter HEEDS - Create or Modify Template** tool, and click **Release Simulation Tool** on the **SIMTOOLS** configuration view.
3. Open the **CAE MDAO Model** you created in Create the CAE MDAO template in a separate view by double-clicking it.
4. Select the item revision of the **CAE MDAO Model**, click **Simulation Tools** → **MDAO** → **Simcenter HEEDS - Create or Modify Template**.
5. In the **Launch Simulation Tool** dialog box, specify the appropriate information or select the appropriate options.

For more information, see *Launch simulation tools* in *Simulation Process and Data Management on Rich Client — Usage*.

Run a preconfigured tool to update the metadata of the existing CAE MDAO template

Update the metadata of an existing template in Teamcenter.

If you have already created a HEED project (***.heeds** file) and you want to manage the data in Teamcenter, you can use the **Simcenter HEEDS - Extract Template Info** tool.

For more information about this tool, click **Simulation Tools → Simulation Tool Help → MDAO → Extract Template Info**, and click **Show Detailed Description**.

1. Click **CAE Configuration → Simulation Tool Configuration** on the main toolbar.

2. Select **SIMTOOL***version***MDAO**, expand it, select the **SIMTOOL***version***Simcenter HEEDS - Extract Template Info** tool, and click **Release Simulation Tool** on the **SIMTOOLS** configuration view.

3. Open the **CAE MDAO Model** you created in **Create the CAE MDAO template** in a separate view by double-clicking it.

4. Select the item revision of the **CAE MDAO Model**, click **Simulation Tools → MDAO →Simcenter HEEDS - Extract Template Info**.

5. In the **Launch Simulation Tool** dialog box, specify the appropriate information or select the appropriate options.

   For more information, see *Launch simulation tools* in *Simulation Process and Data Management on Rich Client — Usage*.

Release the CAE MDAO template

1. Open the **CAE MDAO Model** you created in **Create the CAE MDAO template** in a separate view by double-clicking it.

2. Choose **File→New→Workflow Process** to open the **New Process** dialog box.

   The **Process Name** box contains the revision name automatically. You can accept the default name or enter a different name for this process.

3. In the **Process Template** list, select **TCM Release Process**.

   Your action is not required for the **Add Status** task on the **Process Template** tab. The system performs the task, which assigns the **TCM Released** status.

4. Click **OK** to initiate the workflow process.

Extract the information required for the analysis and perform the HEEDS analysis

The following procedures are to be performed by a HEEDS analyst.

- **Create the CAE MDAO analysis business object**
- **Extract the information required for the analysis**
- **Perform the HEEDS analysis in an interactive mode or a non-interactive mode (batch process)**
- **Post process the results**

Create the CAE MDAO analysis business object

You must create the **CAE MDAO Analysis** business object based on the **CAE MDAO Model** released by the lead or an expert HEEDS user.

1. In CAE Manager, select a folder in the **Home** view, for example, **Newstuff**.
2. Click **File** → **New** → **CAE Item**.
3. In the **New CAE Item Wizard** dialog box, select **CAE MDAO Analysis** and click **Next**.
4. Specify a name and click **Finish**.

Extract the information required for the analysis

You can use the **Simcenter HEEDS - Extract Study Info** tool to extract the information required for the analysis.

After the tool is run, the **CAE MDAO Analysis** business object is updated in Teamcenter.

For more information about this tool, click **Simulation Tools** → **Simulation Tool Help** → **MDAO** → **Extract Study Info**, and click **Show Detailed Description**.

1. Click **CAE Configuration** → **Simulation Tool Configuration** on the main toolbar.
2. Select **SIMTOOL***version***MDAO**, expand it, select the **SIMTOOL***version***Simcenter HEEDS - Extract Study Info** tool, and click **Release Simulation Tool** on the **SIMTOOLS** configuration view.
3. Open the **CAE MDAO Analysis** business object you created in **Create the CAE MDAO analysis business object** in a separate view by double-clicking it.

4. Select the item revision of the **CAE MDAO Model** business object, click **Simulation Tools** → **MDAO** → **Simcenter HEEDS - Extract Study Info**.

5. In the **Launch Simulation Tool** dialog box, specify the appropriate information or select the appropriate options.

   For more information, see *Launch simulation tools* in *Simulation Process and Data Management on Rich Client —Usage*.

6. After the tool launch is complete, search for the input file and attach the input file to the **CAE MDAO Analysis** business object you created in **Create the CAE MDAO analysis business object**.

   The location of this file depends on the scratch location you provided while launching simulation tools.

Perform the HEEDS analysis in an interactive mode or a non-interactive mode (batch process)

You can perform the HEEDS analysis in an interactive mode by using the **Simcenter HEEDS - Execute Analysis** tool or in a non-interactive mode by using the **Simcenter HEEDS - Execute Analysis (Batch)** tool.

For more information about this tool, click **Simulation Tools** → **Simulation Tool Help** → **MDAO** → **Execute Analysis** or **Execute Analysis (Batch)**, and click **Show Detailed Description**.

1. Click **CAE Configuration** → **Simulation Tool Configuration** on the main toolbar.

2. Select **SIMTOOL***version***MDAO**, expand it, and select one of the following tools, and click **Release Simulation Tool** on the **SIMTOOLS** configuration view.

   - To perform the HEEDS analysis in interactive mode: **SIMTOOL***version***Simcenter HEEDS - Execute Analysis**

   - To perform the HEEDS analysis in non interactive mode: **SIMTOOL***version***Simcenter HEEDS - Execute Analysis (Batch)**

3. Open the **CAE MDAO Analysis** business object you created in **Create the CAE MDAO analysis business object** in a separate view by double-clicking it.

   Ensure that the input file after running **Simcenter HEEDS - Extract Study Info** is attached to the **CAE MDAO Analysis** business object as described in **step 6**.

4. Select the item revision of the **CAE MDAO Model** business object, click **Simulation Tools** → **MDAO** → **Simcenter HEEDS - Execute Analysis** or **Simcenter HEEDS - Execute Analysis (Batch)**.

5. In the **Launch Simulation Tool** dialog box, specify the appropriate information or select the appropriate options.

For more information, see *Launch simulation tools* in *Simulation Process and Data Management on Rich Client — Usage* in the Teamcenter documentation.

6. After executing the HEEDS analysis by running a simulation tool, select the **CAE MDAO Analysis** business object, click **Open with → Summary**, and click the **Study Information** tab.

   The system populates the **Best Value** column based on the best design.

Post process the results

You can post process the results by using the **Simcenter HEEDS - Post Processing** tool.

The system updates the **CAE MDAO Analysis** business object after the user exits the HEEDS application.

For more information about this tool, click **Simulation Tools → Simulation Tool Help → MDAO → Post Processing**, and click **Show Detailed Description**.

1. Click **CAE Configuration → Simulation Tool Configuration** on the main toolbar.

2. Select **SIMTOOL***version***MDAO**, expand it, select the **SIMTOOL***version***Simcenter HEEDS - Post Processing** tool, and click **Release Simulation Tool** on the **SIMTOOLS** configuration view.

3. Open the **CAE MDAO Analysis** business object you created in **Create the CAE MDAO analysis business object** in a separate view by double-clicking it.

4. Select the item revision of the **CAE MDAO Model** business object, click **Simulation Tools → MDAO → Simcenter HEEDS - Post Processing**.

5. In the **Launch Simulation Tool** dialog box, specify the appropriate information or select the appropriate options.

   For more information, see *Launch simulation tools* in *Simulation Process and Data Management on Rich Client — Usage*.

5. Configuring and customizing Simulation Process and Data Management for Active Workspace

Create workspaces for different user roles

Simulation analysts and physical test engineers are exposed to all the commands in the **Default** workspace and some of these commands are not relevant to mainstream CAE workflows. Instead, they can use the default workspace for their user role.

User role	Default workspace
Simulation analyst	**Analyst**
Physical test engineer	**Physical Test Engineer**

Depending on the user role, the system automatically assigns the workspace by default. In the case of simulation analyst, it is **Analyst** workspace and for physical test engineer, **Physical Test Engineer** workspace by default.

A user with system administrative privileges can create or assign workspaces. For more information about workspaces, see *Learn about workspaces* in *Active Workspace Customization*.

For more information about adding the workspace or importing custom workspace definitions, see *Create or update workspace mappings* in *Active Workspace Customization*.

Configure the Simulation-related objects table

You (as a system administrator) can configure the fields in the **Related Simulation Objects** table of the **Simulation** tab in Active Workspace.

You can edit style sheets in the Teamcenter rich client to add new properties and expose the new fields in the graphical user interface (GUI) for the following types:

- **CAE 3D Analysis Revision**
- **CAE 3D Model Revision**
- **CAE 3D Geometry Revision**

By default, the **Related Simulation Objects** section displays the **Object**, **Type**, **Relation**, **Release Status**, **Date Released**, and **Owner** fields.

The style sheets referred to here are XML documents stored in Teamcenter **XMLStylesheetRendering** datasets. The following is an example of how to edit *two* style sheets for the **CAE 3D Analysis Revision** type to add the **object_name** property and expose the **Name** field in the **Related Simulation Objects** section.

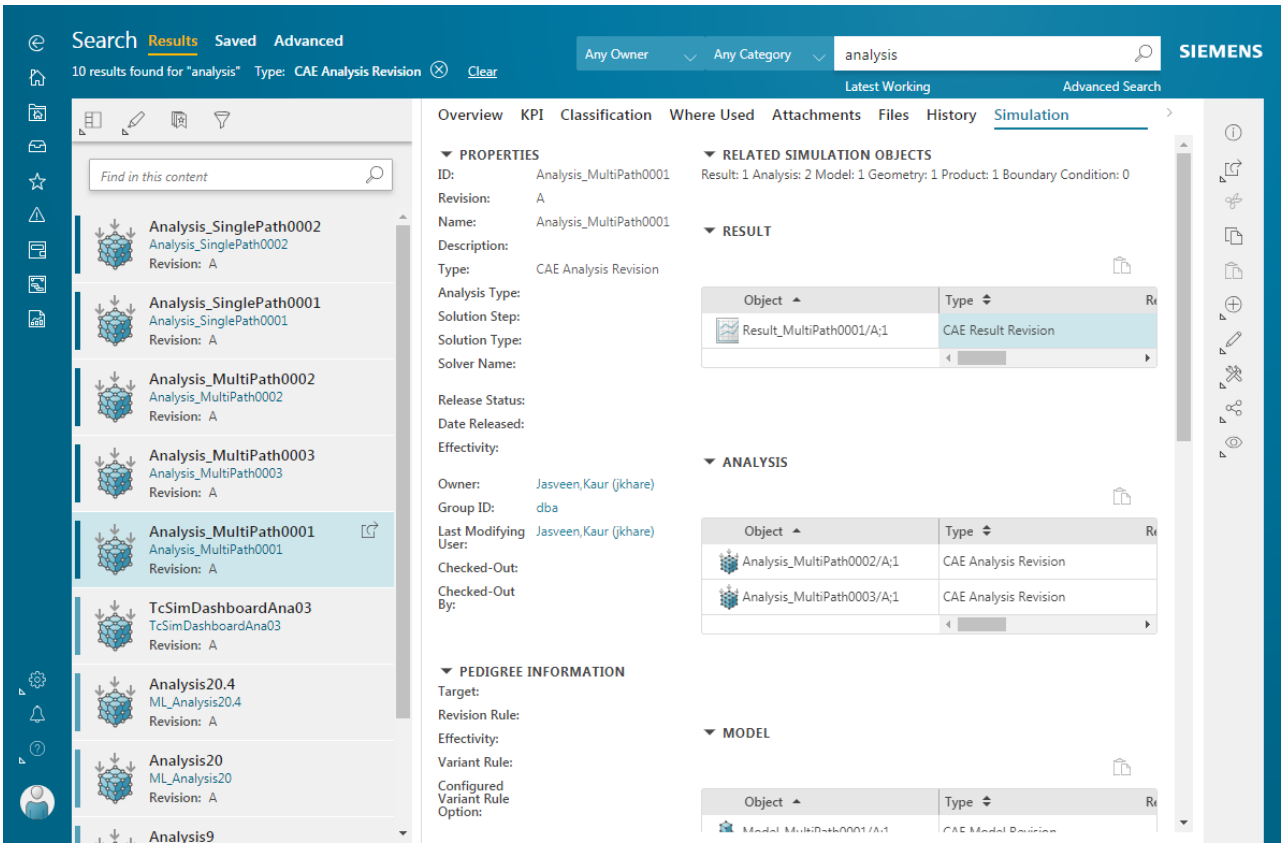
You can also add custom properties for the **CAE 3D Analysis Revision** type.

1. To search for style sheets related to the **CAE 3D Analysis Revision** type, in My Teamcenter (rich client), type *Cae1CAEAna**, select **Dataset** type, and perform a search.

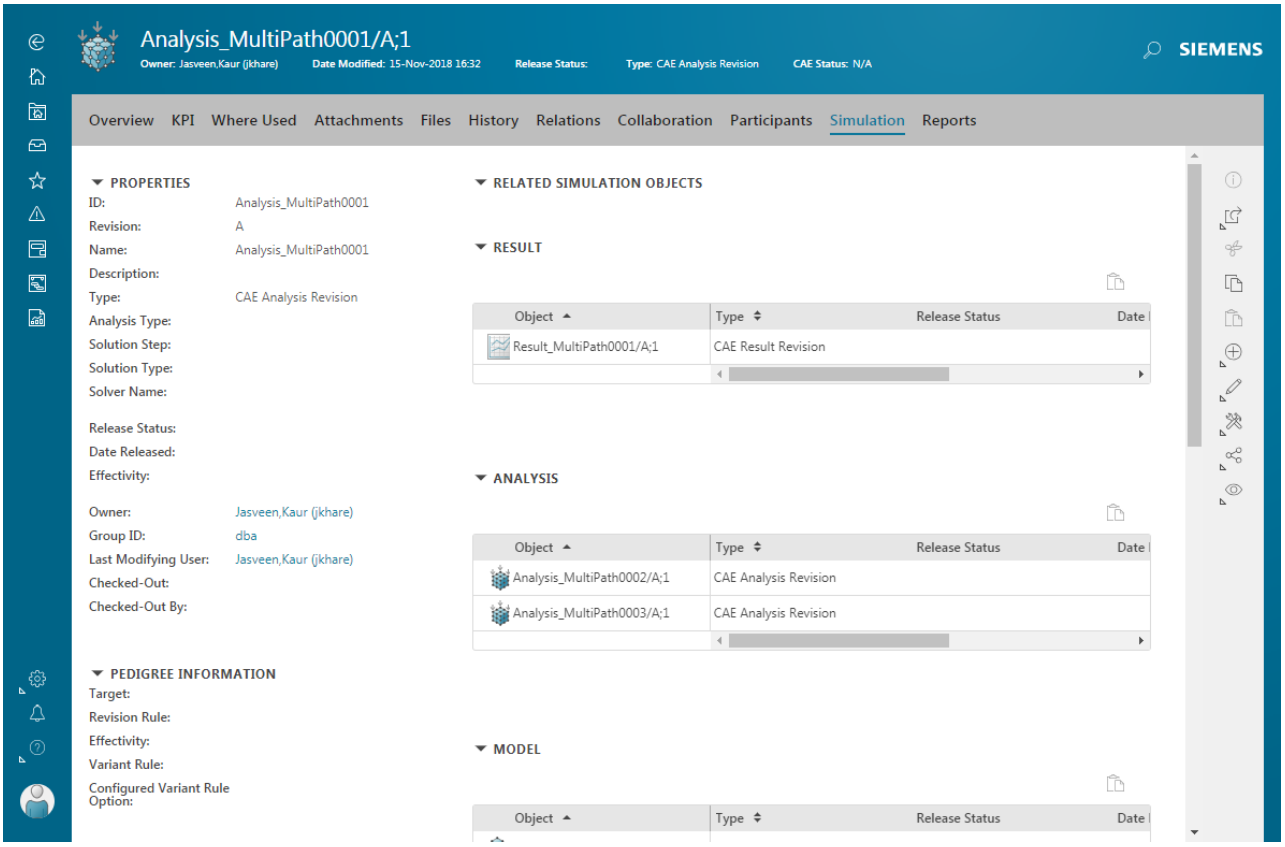
   > Note:
   >
   > Only a DBA user can modify style sheets.

   The search results show **Cae1CAEAnalysisRevSummary** and **Cae1CAEAnalysisRevSummaryForShowObjectLocation**.

   In Active Workspace, users can select a **CAE 3D Analysis Revision** type from the **Results** page and click the **Simulation** tab to view the details.



   Alternatively, they can select a **CAE 3D Analysis Revision** type from the **Results** page, click **Open** to open it separately, and click the **Simulation** tab to view the details.



The **Cae1CAEAnalysisRevSummary** and the **Cae1CAEAnalysisRevSummaryForShow-ObjectLocation** style sheets determine the fields displayed in the **Related Simulation Objects** section for the respective selection.

2. To view the style sheet, select **Cae1CAEAnalysisRevSummary** from the **Search Results** view, and click the **Viewer** tab.

3. Edit the **tc_xrt_Analysis** section in the style sheet to add the new **object_name** property and expose the **Name** field in the GUI.

```
<section titleKey="tc_xrt_RelatedSimulationObjects" >
   <htmlPanel id="com.siemens.splm.client.tcsim.xrtSimulationSublocation" />
</section>
<section titleKey="tc_xrt_Analysis">
   <objectSet source="TC_CAE_Include.CAEAnalysisRevision,
                      S2P:TC_CAE_Include.CAEAnalysisRevision"
              sortdirection="ascending" sortby="object_string"
                  defaultdisplay="listDisplay">
       <tableDisplay>
           <property name="object_string"/>
           <property name="object_type"/>
      <!-- ******** Your customization goes here ******** -->
           <property name="object_name"/>
      <!-- ******** End of customization ******** -->
           <property name="relation"/>
           <property name="release_status_list"/>
```

```
            <property name="date_released"/>
            <property name="owning_user"/>
        </tableDisplay>
   </objectSet>
</section>
<section titleKey="tc_xrt_Result">
    <objectSet source="TC_CAE_Results.CAEResultRevision"
             sortdirection="ascending" sortby="object_string"
                 defaultdisplay="listDisplay">
        <tableDisplay>
            <property name="object_string"/>
            <property name="object_type"/>
    <!-- ******** Your customization goes here ******** -->
            <property name="object_name"/>
    <!-- ******** End of customization ******** -->
            <property name="relation"/>
            <property name="release_status_list"/>
            <property name="date_released"/>
            <property name="owning_user"/>
        </tableDisplay>
    </objectSet>
</section>
```

4. To save your changes, click **Apply**.

In Active Workspace, select a **CAE 3D Analysis Revision** type from the **Results** page and click the **Simulation** tab to view the details.

The **Related Simulation Objects** table displays the new **Name** field.

Similarly, to search for style sheets related to the **CAE 3D Analysis Revision** type, in My Teamcenter (rich client), select the **Cae1CAEAnalysisRevSummaryForShowObjectLocation** sheet from the **Search Results** view. Click the **Viewer** tab and repeat steps 3 and 4.

Configure the traversal paths in the related simulation objects table

Configure the traversal paths in the related simulation objects table

When you open a simulation object in Active Workspace, the **Overview** tab displays the related simulation objects. For example, if you open an analysis revision, related objects such as Result, Analysis, Model, Geometry, Product, and Boundary Condition are displayed.

In the **Overview** tab, prior to 2406, related simulation objects were displayed as multiple tables based on the object type. The following are examples for an analysis revision from the default style sheet for **CAE 3D Analysis** revision:

```
Cae1SimulationSearchProvider.CAEResultRevision
Cae1SimulationSearchProvider.CAEAnalysisRevision
```

All simulation objects are consolidated and displayed in a single table now. The following is an example for an analysis revision from the default style sheet for CAE 3D Analysis Revision:

```
Cae1SimulationSearchProvider.BusinessObject
```

Example: Configure the traversal paths for analysis revisions

When analysts open an analysis revision in the **Overview** tab, all the related objects such as Result, Analysis, Model, Geometry, Product, and Boundary Condition are displayed in the **Related Simulation Objects** table. If they want to view only analysis revisions, they can filter them using multiple selections. To avoid multiple selections, as the administrator, you can customize the style sheet to show only analysis revisions and their traversal paths to the related simulation objects.

To customize the style sheet using the XRTEditor:

1. Log on to Active Workspace and open an analysis revision.

2. Open another tab, log on with administrator privileges, and open the XRTEditor.

   The editor in the secondary tab is now linked to your primary tab. The editor will follow your navigation in the primary tab, displaying the style sheet used to render each page when applicable.

3. To view only analysis objects in the **Related Simulation Objects** table, edit the **ObjectSet** source of the style sheet to add only analysis revisions.

   Default:

   ```
   <objectSet source="Cae1SimulationSearchProvider.BusinessObject"
       sortdirection="ascending" sortby="object_string"
       defaultdisplay="tableDisplay" maxRowCount="5">
   ```

   This displays all related simulation objects since **Cae1SimulationSearchProvider.BusinessObject** is used.

   To display only analysis revisions, add the following:

   ```
   <objectSet source="Cae1SimulationSearchProvider.
       BusinessObject.(CAEAnalysisRevision)"
       sortdirection="ascending" sortby="object_string"
       defaultdisplay="tableDisplay" maxRowCount="5">
   ```

4. Save the style sheet.

5. Refresh the primary tab where you opened the analysis revision. The **Related Simulation Objects** table displays only analysis revisions.

6. To expose only analysis object types with configurable traversal paths in the user interface, edit the style sheet as follows.

   Let us consider that two geometry revisions are related to the analysis revision. The first geometry revision has a direct relation using the source relationship. The second geometry revision is related to a model revision using the source relationship and the model relation is related to the analysis relation using the defining relationship. This is as follows:

   - `CAE Analysis Revision (TC_CAE_Source)-> CAE Geometry Revision`
   - `CAE Analysis Revision (TC_CAE_Defining)-> CAE Model Revision (TC_CAE_Source)-> CAE Geometry Revision`

   Default:

   ```
   <objectSet source="Cae1SimulationSearchProvider.BusinessObject"
       sortdirection="ascending" sortby="object_string"
       defaultdisplay="tableDisplay" maxRowCount="5">
   ```

   To expose only the first geometry revision with the direct relation visible in the user interface, edit as follows:

   ```
   <objectSet source="
   Cae1SimulationSearchProvider.CAEGeometryRevision[TC_CAE_Source]"
       sortdirection="ascending" sortby="object_string"
       defaultdisplay="tableDisplay" maxRowCount="5">
   ```

7. Save the style sheet.

8. Refresh the primary tab where you opened the analysis revision. Only the first geometry revision with the direct relation is visible. The traversal path is displayed in the **Traceability Information** section.

9. To expose both the geometry revisions in the user interface, edit as follows:

   ```
   <objectSet source=" Cae1SimulationSearchProvider.CAEGeometryRevision.
       (TC_CAE_Defining:CAEModelRevision#TC_CAE_Source^^TC_CAE_Source"
       sortdirection="ascending" sortby="object_string"
       defaultdisplay="tableDisplay" maxRowCount="5">
   ```

10. Save the style sheet.

11. Refresh the primary tab where you opened the analysis revision. Both the geometry revisions are visible. The traversal path is displayed in the **Traceability Information** section when you select each geometry revision.

Add traversal paths for related models, geometry, analysis, and product revisions in the context of a result revision

As a user with DBA privileges, you can edit style sheets to add traversal paths for related models, geometry, analysis, and product revisions in the context of a result revision and expose them in the user interface.

You can edit the **objectSet** source in the style sheet to add the traversal paths as described in the following procedures. If you want to retain the default behavior, do not change `<objectSet source="Cae1SimulationSearchProvider.BusinessObject">`.

Procedure

1. Log on to Active Workspace and open a **CAE 3D Result Revision**.

   The **Related Simulation Objects** table is displayed in the **Overview** tab.

2. Open another tab, log on with administrator privileges, and open the XRTEditor.

   The editor in the secondary tab is now linked to your primary tab. The editor will follow your navigation in the primary tab, displaying the style sheet used to render each page when applicable.

3. Add a traversal path for the related model revisions in the context of a result revision and expose the travel path in the user interface.

   a. Create a construct to get model revisions.

   ```
   /* Construct Query to get Model
   (
   CAE Result Revision <-(TC_CAE_Results) CAE Analysis Revision
   (TC_CAE_Defining)->
       CAE Model Revision
   )
   */
   ```

   b. To add a traversal path for the above construct, edit the style sheet, and specify as follows:

   ```
   <objectSet source="Cae1SimulationSearchProvider.BusinessObject.
   (CAEModelRevision
       [S2P$TC_CAE_Results:CAEAnalysisRevision#TC_CAE_Defining])">
   ```

   You can use the following annotations:

- **S2P$** before a relation to specify that the traversal is from a secondary to a primary object.
- **^^** as an AND operator between two paths.
- **#** as a separator between two segments of the path.

  Example: `S2P$TC_CAE_Target:CAEGeometryRevision#S2P$TC_CAE_Source`

  This means that the input of the `S2P$TC_CAE_Source` segment is the output of the `S2P$TC_CAE_Target:CAEGeometryRevision` segment.

4. Add traversal paths for related geometry revisions in the context of a result revision and expose the travel paths in the user interface.

   a. Create a construct to get geometry revisions.

```
/* Construct Query to get Geometry
 (
 CAE Result Revision <-(TC_CAE_Results) CAE Analysis Revision
(TC_CAE_Defining)->
    CAE Model Revision (TC_CAE_Source)-> CAE Geometry Revision,
 CAE Result Revision <-(TC_CAE_Results) CAE Analysis Revision
(TC_CAE_Source)->
    CAE Geometry Revision
 )
 */
```

   b. To add traversal paths for the above construct, edit the style sheet, and specify as follows:

```
<objectSet source="Cae1SimulationSearchProvider.BusinessObject.
(CAEGeometryRevision

[S2P$TC_CAE_Results:CAEAnalysisRevision#TC_CAE_Defining:CAEModelR
evision

#TC_CAE_Source^^S2P$TC_CAE_Results:CAEAnalysisRevision#TC_CAE_Sou
rce])">
```

5. Add a traversal path for the related analysis revisions in the context of a result revision and expose the travel path in the user interface.

   a. Create a construct to get analysis revisions.

```
/* Construct Query to get Analysis
(
CAE Result Revision <-(TC_CAE_Results) CAE Analysis Revision
```

```
)
*/
```

b. To add traversal paths for the above construct, edit the style sheet, and specify as follows:

```
<objectSet source="Cae1SimulationSearchProvider.BusinessObject.
    (CAEAnalysisRevision[S2P$TC_CAE_Results])">
```

6. Add traversal paths for related product revisions in the context of a result revision and expose the travel paths in the user interface.

a. Create a construct to get product revisions.

```
/* Construct Query to get Product
 (
 CAE Result Revision <-(TC_CAE_Results) CAE Analysis Revision
(TC_CAE_Defining)->
    CAE Model Revision (TC_CAE_Source)->
    CAE Geometry Revision (TC_CAE_Source/TC_CAE_Target)-> Item
Revision,
 CAE Result Revision <-(TC_CAE_Results) CAE Analysis Revision
(TC_CAE_Defining) ->
    CAE Model Revision (TC_CAE_Source/TC_CAE_Target)-> Item
Revision,
 CAE Result Revision <-(TC_CAE_Results) CAE Analysis Revision
(TC_CAE_Source)->
    CAE Geometry Revision (TC_CAE_Source/TC_CAE_Target)-> Item
Revision,
 CAE Result Revision <-(TC_CAE_Results) CAE Analysis Revision
(TC_CAE_Target)->
    Item Revision
 )
 */
```

b. To add traversal paths for the above construct, edit the style sheet, and specify as follows:

```
<objectSet source="Cae1SimulationSearchProvider.BusinessObject.
(ItemRevision

[S2P$TC_CAE_Results:CAEAnalysisRevision#TC_CAE_Defining:CAEModelR
evision
    #TC_CAE_Source:CAEGeometryRevision
    #TC_CAE_Source^^S2P$TC_CAE_Results:CAEAnalysisRevision

#TC_CAE_Defining:CAEModelRevision#TC_CAE_Source:CAEGeometryRevisi
on

#TC_CAE_Target^^S2P$TC_CAE_Results:CAEAnalysisRevision#TC_CAE_Def
```

```
ining:

CAEModelRevision#TC_CAE_Source^^S2P$TC_CAE_Results:CAEAnalysisRev
ision#

TC_CAE_Defining:CAEModelRevision#TC_CAE_Target^^S2P$TC_CAE_Result
s:
    CAEAnalysisRevision#TC_CAE_Source:CAEGeometryRevision
    #TC_CAE_Source^^S2P$TC_CAE_Results:CAEAnalysisRevision

#TC_CAE_Source:CAEGeometryRevision#TC_CAE_Target^^S2P$TC_CAE_Resu
lts:
    CAEAnalysisRevision#TC_CAE_Target])">
```

7. Save the style sheet.

8. Refresh the **Overview** tab where you opened the **CAE 3D Result Revision** and verify the changes.

Add traversal paths for related models, results, geometry, analysis, and product revisions in the context of an analysis revision

As a user with DBA privileges, you can edit style sheets to add traversal paths for related models, results, geometry, analysis, and product revisions in the context of an analysis revision and expose them in the user interface.

You can edit the **objectSet** source in the style sheet to add the traversal paths as described in the following procedures. If you want to retain the default behavior, do not change `<objectSet source="Cae1SimulationSearchProvider.BusinessObject">`.

Procedure

1. Log on to Active Workspace and open a **CAE 3D Analysis Revision**.

   The **Related Simulation Objects** table is displayed in the **Overview** tab.

2. Open another tab, log on with administrator privileges, and open the XRTEditor.

   The editor in the secondary tab is now linked to your primary tab. The editor will follow your navigation in the primary tab, displaying the style sheet used to render each page when applicable.

3. Add a traversal path for related model revisions in the context of an analysis revision and expose the traversal path in the user interface.

   a. Create a construct to get model revisions.

```
/* Construct Query to get Model
(
CAE Analysis Revision (TC_CAE_Defining)-> CAE Model Revision
)
*/
```

b. To add a traversal path for the above construct, edit the style sheet, and specify as follows:

```
<objectSet source="Cae1SimulationSearchProvider.BusinessObject.
    (CAEModelRevision[TC_CAE_Defining])">
```

You can use the following annotations:

- **S2P$** before a relation to specify that the traversal is from a secondary to a primary object.
- **^^** as an AND operator between two paths.
- **#** as a separator between two segments of the path.

  Example: `S2P$TC_CAE_Target:CAEGeometryRevision#S2P$TC_CAE_Source`

  This means that the input of the `S2P$TC_CAE_Source` segment is the output of the `S2P$TC_CAE_Target:CAEGeometryRevision` segment.

4. Add a traversal path for related result revisions in the context of an analysis revision and expose the traversal path in the user interface.

   a. Create a construct to get result revisions.

```
/* Construct Query to get Result
(
CAE Analysis Revision (TC_CAE_Results)-> CAE Result Revision
)
*/
```

   b. To add a traversal path for the above construct, edit the style sheet, and specify as follows:

```
<objectSet source="Cae1SimulationSearchProvider.BusinessObject.
    (CAEResultRevision[TC_CAE_Results])">
```

5. Add traversal paths for related geometry revisions in the context of an analysis revision and expose the traversal paths in the user interface.

   a. Create a construct to get geometry revisions.

```
/* Construct Query to get Geometry
 (
```

```
 CAE Analysis Revision (TC_CAE_Defining)-> CAE Model Revision
(TC_CAE_Source)->
    CAE Geometry Revision,
    CAE Analysis Revision (TC_CAE_Source)-> CAE Geometry Revision
 )
 */
```

b. To add traversal paths for the above construct, edit the style sheet, and specify as follows:

```
<objectSet source="Cae1SimulationSearchProvider.BusinessObject.
    (CAEGeometryRevision[TC_CAE_Defining:CAEModelRevision
    #TC_CAE_Source^^TC_CAE_Source])">
```

6. Add traversal paths for related analysis revisions in the context of another analysis revision and expose the traversal paths in the user interface.

a. Create a construct to get analysis revisions.

```
/* Construct Query to get Analysis
 (
 CAE Analysis Revision (TC_CAE_Include)-> CAE Analysis Revision,
 CAE Analysis Revision (TC_CAE_Include)<- CAE Analysis Revision
 )
 */
```

b. To add traversal paths for the above construct, edit the style sheet, and specify as follows:

```
<objectSet source="Cae1SimulationSearchProvider.BusinessObject.
(CAEAnalysisRevision
    [TC_CAE_Include^^S2P$TC_CAE_Include])">
```

7. Add traversal paths for related product revisions in the context of an analysis revision and expose the traversal paths in the user interface.

a. Create a construct to get product revisions.

```
/* Construct Query to get Product
 (
 CAE Analysis Revision (TC_CAE_Defining)-> CAE Model Revision
(TC_CAE_Source)->
    CAE Geometry Revision (TC_CAE_Source/TC_CAE_Target)-> Item
Revision,
 CAE Analysis Revision (TC_CAE_Defining) ->
    CAE Model Revision (TC_CAE_Source/TC_CAE_Target)-> Item
Revision,
 CAE Analysis Revision (TC_CAE_Source)->
    CAE Geometry Revision (TC_CAE_Source/TC_CAE_Target)-> Item
Revision,
```

```
CAE Analysis Revision (TC_CAE_Target)-> Item Revision
)
*/
```

b. To add traversal paths for the above construct, edit the style sheet, and specify as follows:

```
<objectSet source="Cae1SimulationSearchProvider.BusinessObject.
    (ItemRevision[TC_CAE_Defining:CAEModelRevision
    #TC_CAE_Source:CAEGeometryRevision
    #TC_CAE_Source^^TC_CAE_Defining:CAEModelRevision
    #TC_CAE_Source:CAEGeometryRevision
    #TC_CAE_Target^^TC_CAE_Defining:CAEModelRevision
    #TC_CAE_Source^^TC_CAE_Defining:CAEModelRevision
    #TC_CAE_Target^^TC_CAE_Source:CAEGeometryRevision
    #TC_CAE_Source^^TC_CAE_Source:CAEGeometryRevision
    #TC_CAE_Target^^TC_CAE_Target])">
```

8. Save the style sheet.

9. Refresh the **Overview** tab where you opened the **CAE 3D Analysis Revision** and verify the changes.

Add traversal paths for related results, geometry, analysis, and product revisions in the context of a model revision

As a user with DBA privileges, you can edit style sheets to add traversal paths for related results, geometry, analysis, and product revisions in the context of a model revision and expose them in the user interface.

You can edit the **objectSet** source in the style sheet to add the traversal paths as described in the following procedures. If you want to retain the default behavior, do not change `<objectSet source="Cae1SimulationSearchProvider.BusinessObject">`.

Procedure

1. Log on to Active Workspace and open a **CAE 3D Model Revision**.

   The **Related Simulation Objects** table is displayed in the **Overview** tab.

2. Open another tab, log on with administrator privileges, and open the XRTEditor.

   The editor in the secondary tab is now linked to your primary tab. The editor will follow your navigation in the primary tab, displaying the style sheet used to render each page when applicable.

3. Add a traversal path for related results revisions in the context of a model revision and expose the traversal path in the user interface.

a. Create a construct to get result revisions.

```
/* Construct Query to get Result
(
CAE Model Revision <-(TC_CAE_Defining) CAE Analysis Revision
(TC_CAE_Results)->
    CAE Result Revision
)
*/
```

b. To add a traversal path for the above construct, edit the style sheet, and specify as follows:

```
<objectSet source="Cae1SimulationSearchProvider.BusinessObject.
(CAEResultRevision
    [S2P$TC_CAE_Defining:CAEAnalysisRevision#TC_CAE_Results])">
```

You can use the following annotations:

- **S2P$** before a relation to specify that the traversal is from a secondary to a primary object.
- **^^** as an AND operator between two paths.
- **#** as a separator between two segments of the path.

  Example: `S2P$TC_CAE_Target:CAEGeometryRevision#S2P$TC_CAE_Source`

  This means that the input of the `S2P$TC_CAE_Source` segment is the output of the `S2P$TC_CAE_Target:CAEGeometryRevision` segment.

4. Add a traversal path for related geometry revisions in the context of a model revision and expose the traversal path in the user interface.

a. Create a construct to get geometry revisions.

```
/* Construct Query to get Geometry
(
CAE Model Revision (TC_CAE_Source)-> CAE Geometry Revision
)
*/
```

b. To add a traversal path for the above construct, edit the style sheet, and specify as follows:

```
<objectSet source="Cae1SimulationSearchProvider.BusinessObject.
    (CAEGeometryRevision[TC_CAE_Source])">
```

5. Add a traversal path for related analysis revisions in the context of a model revision and expose the traversal path in the user interface.

a. Create a construct to get analysis revisions.

```
/* Construct Query to get Analysis
(
CAE Model Revision <-(TC_CAE_Defining) CAE Analysis Revision
)
*/
```

b. To add a traversal path for the above construct, edit the style sheet, and specify as follows:

```
<objectSet source="Cae1SimulationSearchProvider.BusinessObject.
    (CAEAnalysisRevision[S2P$TC_CAE_Defining])">
```

6. Add traversal paths for related product revisions in the context of a model revision and expose the traversal paths in the user interface.

a. Create a construct to get product revisions.

```
/* Construct Query to get Product
 (
 CAE Model Revision (TC_CAE_Source)-> CAE Geometry Revision
    (TC_CAE_Source/TC_CAE_Target)-> Item Revision,
 CAE Model Revision (TC_CAE_Source/TC_CAE_Target)-> Item Revision
 )
 */
```

b. To add traversal paths for the above construct, edit the style sheet, and specify as follows:

```
<objectSet source="Cae1SimulationSearchProvider.BusinessObject.
    (ItemRevision[TC_CAE_Source:CAEGeometryRevision
    #TC_CAE_Source^^TC_CAE_Source:CAEGeometryRevision
    #TC_CAE_Target^^TC_CAE_Source^^TC_CAE_Target])">
```

7. Save the style sheet.

8. Refresh the **Overview** tab where you opened the **CAE 3D Model Revision** and verify the changes.

Add traversal paths for related models, results, analysis, and product revisions in the context of a geometry revision

As a user with DBA privileges, you can edit style sheets to add traversal paths for related models, results, analysis, and product revisions in the context of a geometry revision and expose them in the user interface.

You can edit the **objectSet** source in the style sheet to add the traversal paths as described in the following procedures. If you want to retain the default behavior, do not change `<objectSet source="Cae1SimulationSearchProvider.BusinessObject">`.

Procedure

1. Log on to Active Workspace and open a **CAE 3D Geometry Revision**.

   The **Related Simulation Objects** table is displayed in the **Overview** tab.

2. Open another tab, log on with administrator privileges, and open the XRTEditor.

   The editor in the secondary tab is now linked to your primary tab. The editor will follow your navigation in the primary tab, displaying the style sheet used to render each page when applicable.

3. Add a traversal path for the related model revisions in the context of a geometry revision and expose the traversal path in the user interface.

   a. Create a construct to get model revisions.

   ```
   /* Construct Query to get Model
   (
   CAEGeometryRevision <-(TC_CAE_Source) CAEModelRevision
   )
   */
   ```

   b. To add a traversal path for the above construct, edit the style sheet, and specify as follows:

   ```
   <objectSet source="Cae1SimulationSearchProvider.BusinessObject.
       (CAEModelRevision[S2P$TC_CAE_Source])">
   ```

   You can use the following annotations:

   - **S2P$** before a relation to specify that the traversal is from a secondary to a primary object.
   - **^^** as an AND operator between two paths.
   - **#** as a separator between two segments of the path.

     Example: `S2P$TC_CAE_Target:CAEGeometryRevision#S2P$TC_CAE_Source`

     This means that the input of the `S2P$TC_CAE_Source` segment is the output of the `S2P$TC_CAE_Target:CAEGeometryRevision` segment.

4. Add traversal paths for the related result revisions in the context of a geometry revision and expose the traversal paths in the user interface.

a. Create a construct to get result revisions.

```
/* Construct Query to get Result
 (
 CAE Geometry Revision <-(TC_CAE_Source) CAE Analysis Revision
(TC_CAE_Results)->
    CAE Result Revision,
 CAE Geometry Revision <-(TC_CAE_Source) CAE Model Revision <-
(TC_CAE_Defining)
    CAE Analysis Revision (TC_CAE_Results)-> CAE Result Revision
 )
 */
```

b. To add traversal paths for the above construct, edit the style sheet, and specify as follows:

```
<objectSet source="Cae1SimulationSearchProvider.BusinessObject.
    (CAEResultRevision[S2P$TC_CAE_Source:CAEAnalysisRevision
    #TC_CAE_Results^^S2P$TC_CAE_Source:CAEModelRevision
    #S2P$TC_CAE_Defining:CAEAnalysisRevision#TC_CAE_Results])">
```

5. Add traversal paths for related analysis revisions in the context of a geometry revision and expose the traversal paths in the user interface.

a. Create a construct to get analysis revisions.

```
/* Construct Query to get Analysis
 (
 CAE Geometry Revision <-(TC_CAE_Source) CAE Analysis Revision,
 CAE Geometry Revision <-(TC_CAE_Source) CAE Model Revision
    <-(TC_CAE_Defining) CAE Analysis Revision
 )
 */
```

b. To add traversal paths for the above construct, edit the style sheet, and specify as follows:

```
<objectSet source="Cae1SimulationSearchProvider.BusinessObject.
    (CAEAnalysisRevision[S2P$TC_CAE_Source^^S2P$TC_CAE_Source:
    CAEModelRevision#S2P$TC_CAE_Defining])">
```

6. Add traversal paths for related product revisions in the context of a geometry revision and expose the traversal paths in the user interface.

a. Create a construct to get product revisions.

```
/* Construct Query to get Product
(
CAE Geometry Revision (TC_CAE_Source/TC_CAE_Target) -> Item
```

```
Revision
)
*/
```

b. To add traversal paths for the above construct, edit the style sheet, and specify as follows:

```
<objectSet source="Cae1SimulationSearchProvider.BusinessObject.
    (ItemRevision[TC_CAE_Source^^TC_CAE_Target])">
```

7. Save the style sheet.

8. Refresh the **Overview** tab where you opened the **CAE 3D Geometry Revision** and verify the changes.

Add traversal paths for related models, results, geometry, and analysis revisions in the context of a product revision

As a user with DBA privileges, you can edit style sheets to add traversal paths for related models, results, geometry, and analysis revisions in the context of a product revision and expose them in the user interface.

You can edit the **objectSet** source in the style sheet to add the traversal paths as described in the following procedures. If you want to retain the default behavior, do not change `<objectSet source="Cae1SimulationSearchProvider.BusinessObject">`.

Procedure

1. Log on to Active Workspace and open a **Item Revision**.

   The **Related Simulation Objects** table is displayed in the **Overview** tab.

2. Open another tab, log on with administrator privileges, and open the XRTEditor.

   The editor in the secondary tab is now linked to your primary tab. The editor will follow your navigation in the primary tab, displaying the style sheet used to render each page when applicable.

3. Add traversal paths for related model revisions in the context of a product revision and expose the traversal paths in the user interface.

   a. Create a construct to get model revisions.

```
/* Construct Query to get Model
 (
 Item Revision <-(TC_CAE_Source/TC_CAE_Target) CAE Model
Revision,
 Item Revision <-(TC_CAE_Source/TC_CAE_Target) CAE Geometry
```

```
Revision
    <-(TC_CAE_Source) CAE Model Revision
 )
 */
```

b. To add traversal paths for the above construct, edit the style sheet, and specify as follows:

```
<objectSet source="Cae1SimulationSearchProvider.BusinessObject.

(CAEModelRevision[S2P$TC_CAE_Target^^S2P$TC_CAE_Source^^S2P$TC_CA
E_Target:

CAEGeometryRevision#S2P$TC_CAE_Source^^S2P$TC_CAE_Source:CAEGeome
tryRevision
    #S2P$TC_CAE_Source])">
```

You can use the following annotations:

- **S2P$** before a relation to specify that the traversal is from a secondary to a primary object.
- **^^** as an AND operator between two paths.
- **#** as a separator between two segments of the path.

  Example: `S2P$TC_CAE_Target:CAEGeometryRevision#S2P$TC_CAE_Source`

  This means that the input of the `S2P$TC_CAE_Source` segment is the output of the `S2P$TC_CAE_Target:CAEGeometryRevision` segment.

4. Add traversal paths for related result revisions in the context of a product revision and expose the traversal paths in the user interface.

   a. Create a construct to get result revisions.

```
/* Construct Query to get Result
(
 Item Revision <-(TC_CAE_Source/TC_CAE_Target) CAE Geometry
Revision
    <-(TC_CAE_Source) CAE Model Revision
    <-(TC_CAE_Defining) CAE Analysis Revision (TC_CAE_Results)
    -> CAE Result Revision,
 Item Revision <-(TC_CAE_Source/TC_CAE_Target) CAE Model
Revision
    <-(TC_CAE_Defining) CAE Analysis Revision (TC_CAE_Results)
    -> CAE Result Revision,
 Item Revision <-(TC_CAE_Source/TC_CAE_Target) CAE Geometry
Revision
```

```
    <-(TC_CAE_Source) CAE Analysis Revision (TC_CAE_Results)
    -> CAE Result Revision,
 Item Revision <-(TC_CAE_Target) CAE Analysis Revision
(TC_CAE_Results)
    -> CAE Result Revision
)
*/
```

b. To add traversal paths for the above construct, edit the style sheet, and specify as follows:

```
<objectSet source="Cae1SimulationSearchProvider.BusinessObject.
    (CAEResultRevision[S2P$TC_CAE_Source:CAEGeometryRevision
    #S2P$TC_CAE_Source:CAEModelRevision
    #S2P$TC_CAE_Defining:CAEAnalysisRevision
    #TC_CAE_Results^^S2P$TC_CAE_Target:CAEGeometryRevision
    #S2P$TC_CAE_Source:CAEModelRevision
    #S2P$TC_CAE_Defining:CAEAnalysisRevision
    #TC_CAE_Results^^S2P$TC_CAE_Source:CAEModelRevision
    #S2P$TC_CAE_Defining:CAEAnalysisRevision
    #TC_CAE_Results^^S2P$TC_CAE_Target:CAEModelRevision
    #S2P$TC_CAE_Defining:CAEAnalysisRevision
    #TC_CAE_Results^^S2P$TC_CAE_Source:CAEGeometryRevision
    #S2P$TC_CAE_Source:CAEAnalysisRevision
    #TC_CAE_Results^^S2P$TC_CAE_Target:CAEGeometryRevision
    #S2P$TC_CAE_Source:CAEAnalysisRevision

#TC_CAE_Results^^S2P$TC_CAE_Target:CAEAnalysisRevision#TC_CAE_Res
ults])">
```

5. Add traversal paths for related geometry revisions in the context of a product revision and expose the traversal paths in the user interface.

a. Create a construct to get geometry revisions.

```
/* Construct Query to get Geometry
(
Item Revision <-(TC_CAE_Source/TC_CAE_Target) CAE Geometry
Revision
)
*/
```

b. To add traversal paths for the above construct, edit the style sheet, and specify as follows:

```
<objectSet source="Cae1SimulationSearchProvider.BusinessObject.
    (CAEGeometryRevision[S2P$TC_CAE_Source^^S2P$TC_CAE_Target])">
```

6. Add traversal paths for related geometry revisions in the context of a product revision and expose the traversal paths in the user interface.

a. Create a construct to get analysis revisions.

```
/* Construct Query to get Analysis
(
 Item Revision <-(TC_CAE_Source/TC_CAE_Target) CAE Geometry
Revision
    <-(TC_CAE_Source) CAE Model Revision
    <-(TC_CAE_Defining) CAE Analysis Revision,
 Item Revision <-(TC_CAE_Source/TC_CAE_Target) CAE Geometry
Revision
    <-(TC_CAE_Source) CAE Analysis Revision,
 Item Revision <-(TC_CAE_Source/TC_CAE_Target) CAE Model
Revision
    <-(TC_CAE_Defining) CAE Analysis Revision,
 Item Revision <-(TC_CAE_Target) CAE Analysis Revision
)
*/
```

b. To add traversal paths for the above construct, edit the style sheet, and specify as follows:

```
<objectSet source="Cae1SimulationSearchProvider.BusinessObject.
    (CAEAnalysisRevision[S2P$TC_CAE_Source:
    CAEGeometryRevision#S2P$TC_CAE_Source:CAEModelRevision
    #S2P$TC_CAE_Defining^^S2P$TC_CAE_Target:CAEGeometryRevision
    #S2P$TC_CAE_Source:CAEModelRevision
    #S2P$TC_CAE_Defining^^S2P$TC_CAE_Source:CAEGeometryRevision
    #S2P$TC_CAE_Source^^S2P$TC_CAE_Target:CAEGeometryRevision
    #S2P$TC_CAE_Source^^S2P$TC_CAE_Source:CAEModelRevision
    #S2P$TC_CAE_Defining^^S2P$TC_CAE_Target:CAEModelRevision
    #S2P$TC_CAE_Defining^^S2P$TC_CAE_Target])">
```

7. Save the style sheet.

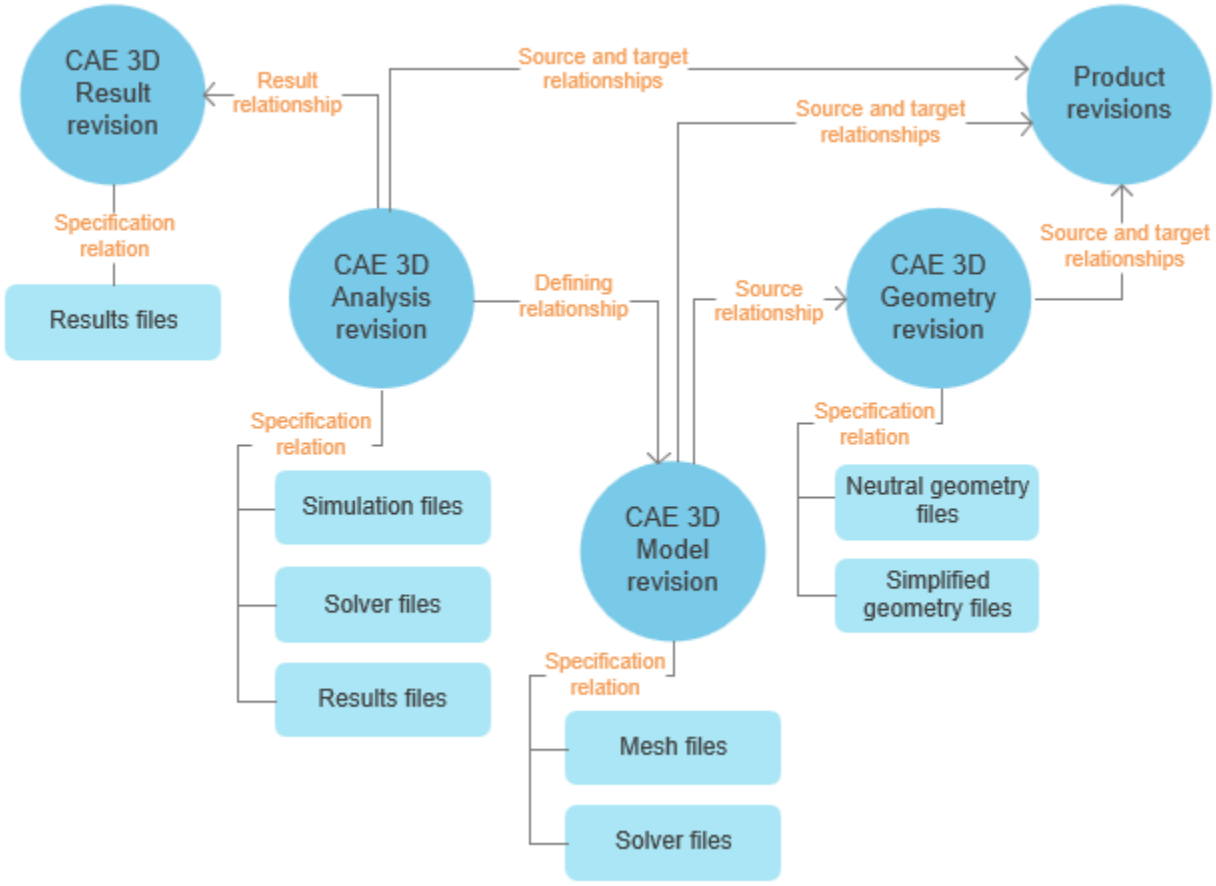
8. Refresh the **Overview** tab where you opened the **Item Revision** and verify the changes.

Edit style sheets to expose custom revision types in the Simulation tab

Expose custom revision types in the context of the product revision

You (as a user with **dba** privileges) can edit style sheets to expose custom revision types in the context of the product revision and configure traversal paths to define how to traverse the data structure and specify which relationships are of interest and what should be done when these relationships are encountered. For example, you can define a traversal path from a primary item type, such as an analysis revision, to the solver-specific data deck (**CAESolver**) and from **CAESolver** to the file to be exported.

The following is an example of the generic simulation data model.



In Active Workspace, users can select an item revision type from the **Results** page and view the details in the **Simulation** tab. Alternatively, they can select an item revision type from the **Results** page, click **Open** to open it separately, and view the details in the **Simulation** tab. The **Cae1ItemRevSummary** and the **Cae1ItemRevSummaryForShowObjectLocation** style sheets determine the objects displayed in the **Related Simulation Objects** table for the respective selection.

You can edit style sheets to add a new traversal path and expose the new objects or edit the default traversal path to hide some objects in the graphical user interface (GUI). The style sheets referred to here are XML documents stored in Teamcenter **XMLStylesheetRendering** datasets and you can edit them using the **Viewer** tab in Teamcenter rich client.

To expose custom revision types in the context of the product revision:

- Edit the **Cae1ItemRevSummary** style sheet and add custom geometry, model, analysis, and result revisions. In addition, create construct queries for each of these custom revisions to specify traversal paths.

- Edit the **Cae1ItemRevSummaryForShowObjectLocation** style sheet and add custom geometry, model, analysis, and result revisions. In addition, create construct queries for each of these custom revisions to specify traversal paths.

The following is an example of how to edit the style sheet to customize the GUI to display model, result, geometry, and analysis revisions in the **Related Simulation Objects** table.

Edit style sheets using the Viewer tab in Teamcenter rich client

1. In Teamcenter rich client, sign in as a **dba** user.

2. Edit the **Cae1ItemRevSummary** style sheet.

    a. To search for style sheets related to the item revision type, in My Teamcenter (rich client), type *Cae1ItemRevSummary**, select **Dataset** type, and perform a search.

    The search results show **Cae1ItemRevSummary** and **Cae1ItemRevSummaryForShowObjectLocation**.

    b. To view the style sheet, select **Cae1ItemRevSummary** from the **Search Results** view, and click the **Viewer** tab.

3. Add model revisions to the style sheet.

    a. Create a construct query.

    Example:

```
/* Construct Query to get Model

 (

 Item Revision
 <-(TC_CAE_Source/TC_CAE_Target) CAE 3D Model Revision,

 Item Revision
 <-(TC_CAE_Source/TC_CAE_Target) CAE 3D Geometry Revision
 <-(TC_CAE_Source) CAE 3D Model Revision

 )

 */

Cae1SimulationSearchProvider.CAEModelRevision.
(S2P$TC_CAE_Target^^S2P$TC_CAE_Source
^^S2P$TC_CAE_Target:CAEGeometryRevision#S2P$TC_CAE_Source
^^S2P$TC_CAE_Source:CAEGeometryRevision#S2P$TC_CAE_Source)
```

    b. Edit the **objectSet source** properties in the **tc_xrt_Model** section of the style sheet.

    Default object source:

```
<section titleKey="tc_xrt_Model">
  <objectSet source="Cae1SimulationSearchProvider.CAEModelRevision"
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

Change to:

```
<section titleKey="tc_xrt_Model">
<!-- ********** Start of customization ********** -->
  <objectSet source="Cae1SimulationSearchProvider.CAEModelRevision.
    (S2P$TC_CAE_Target ^^S2P$TC_CAE_Source
    ^^S2P$TC_CAE_Target:CAEGeometryRevision#S2P$TC_CAE_Source
    ^^S2P$TC_CAE_Source:CAEGeometryRevision#S2P$TC_CAE_Source)"
<!-- ********** End of customization ********** -->
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

> Tip:
>
> You can specify **S2P$** as the prefix before a relation to specify that the traversal path is from the secondary object to the primary object, **^^** as an *AND* operator between two paths, and **#** as the separator between the two segments of the traversal path.

4. Add result revisions to the style sheet.

   a. Create a construct query.

      Example:

```
/* Construct Query to get Result

(

 Item Revision
 <-(TC_CAE_Source/TC_CAE_Target) CAE 3D Geometry Revision
 <-(TC_CAE_Source) CAE 3D Model Revision
 <-(TC_CAE_Defining) CAE 3D Analysis Revision (TC_CAE_Results)
 -> CAE 3D Result Revision,

 Item Revision
 <-(TC_CAE_Source/TC_CAE_Target) CAE 3D Model Revision
 <-(TC_CAE_Defining) CAE 3D Analysis Revision (TC_CAE_Results)
 -> CAE 3D Result Revision,

 Item Revision
 <-(TC_CAE_Source/TC_CAE_Target) CAE 3D Geometry Revision
 <-(TC_CAE_Source) CAE 3D Analysis Revision (TC_CAE_Results)
 -> CAE 3D Result Revision,

 Item Revision
 <-(TC_CAE_Target) CAE 3D Analysis Revision (TC_CAE_Results)
 -> CAE 3D Result Revision

)

*/

Cae1SimulationSearchProvider.CAEResultRevision.
(S2P$TC_CAE_Source:CAEGeometryRevision
#S2P$TC_CAE_Source:CAEModelRevision
#S2P$TC_CAE_Defining:CAEAnalysisRevision
#TC_CAE_Results

^^S2P$TC_CAE_Target:CAEGeometryRevision
#S2P$TC_CAE_Source:CAEModelRevision
#S2P$TC_CAE_Defining:CAEAnalysisRevision
#TC_CAE_Results

^^S2P$TC_CAE_Source:CAEModelRevision
#S2P$TC_CAE_Defining:CAEAnalysisRevision
#TC_CAE_Results

^^S2P$TC_CAE_Target:CAEModelRevision
#S2P$TC_CAE_Defining:CAEAnalysisRevision
#TC_CAE_Results

^^S2P$TC_CAE_Source:CAEGeometryRevision
#S2P$TC_CAE_Source:CAEAnalysisRevision
```

```
#TC_CAE_Results

^^S2P$TC_CAE_Target:CAEGeometryRevision
#S2P$TC_CAE_Source:CAEAnalysisRevision
#TC_CAE_Results

^^S2P$TC_CAE_Target:CAEAnalysisRevision
#TC_CAE_Results)
```

b. Edit the **objectSet source** properties in the **tc_xrt_Result** section of the style sheet.

Default object source:

```
<section titleKey="tc_xrt_Result">
  <objectSet source="Cae1SimulationSearchProvider.CAEResultRevision"

 sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

Change to:

```
<section titleKey="tc_xrt_Result">
<!-- ********** Start of customization ********** -->
  <objectSet source="Cae1SimulationSearchProvider.CAEResultRevision.
(S2P$TC_CAE_Source:CAEGeometryRevision
#S2P$TC_CAE_Source:CAEModelRevision
#S2P$TC_CAE_Defining:CAEAnalysisRevision
#TC_CAE_Results

^^S2P$TC_CAE_Target:CAEGeometryRevision
#S2P$TC_CAE_Source:CAEModelRevision
#S2P$TC_CAE_Defining:CAEAnalysisRevision
#TC_CAE_Results

^^S2P$TC_CAE_Source:CAEModelRevision
#S2P$TC_CAE_Defining:CAEAnalysisRevision
#TC_CAE_Results

^^S2P$TC_CAE_Target:CAEModelRevision
```

```
#S2P$TC_CAE_Defining:CAEAnalysisRevision
#TC_CAE_Results

^^S2P$TC_CAE_Source:CAEGeometryRevision
#S2P$TC_CAE_Source:CAEAnalysisRevision
#TC_CAE_Results

^^S2P$TC_CAE_Target:CAEGeometryRevision
#S2P$TC_CAE_Source:CAEAnalysisRevision
#TC_CAE_Results

^^S2P$TC_CAE_Target:CAEAnalysisRevision
#TC_CAE_Results)"
<!-- ********** End of customization ********** -->
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

5. Add geometry revisions to the style sheet.

   a. Create a construct query.

      Example:

```
/* Construct Query to get Geometry

(

Item Revision
<-(TC_CAE_Source/TC_CAE_Target) CAE 3D Geometry Revision

)

*/

Cae1SimulationSearchProvider.CAEGeometryRevision.
(S2P$TC_CAE_Source^^S2P$TC_CAE_Target)
```

   b. Edit the **objectSet source** properties in the **tc_xrt_Geometry** section of the style sheet.

Default object source:

```
<section titleKey="tc_xrt_Geometry">
  <objectSet source="Cae1SimulationSearchProvider.CAEGeometryRevision"
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

Change to:

```
<section titleKey="tc_xrt_Geometry">
<!-- ********** Start of customization ********** -->
  <objectSet source="Cae1SimulationSearchProvider.CAEGeometryRevision.
(S2P$TC_CAE_Source^^S2P$TC_CAE_Target)"
<!-- ********** End of customization ********** -->
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

6. Add analysis revisions to the style sheet.

   a. Create a construct query.

      Example:

```
/* Construct Query to get Analysis

(

 Item Revision
 <-(TC_CAE_Source/TC_CAE_Target) CAE 3D Geometry Revision
 <-(TC_CAE_Source) CAE 3D Model Revision
```

```
 <-(TC_CAE_Defining) CAE 3D Analysis Revision,

 Item Revision
 <-(TC_CAE_Source/TC_CAE_Target) CAE 3D Geometry Revision
 <-(TC_CAE_Source) CAE 3D Analysis Revision,

 Item Revision
 <-(TC_CAE_Source/TC_CAE_Target) CAE 3D Model Revision
 <-(TC_CAE_Defining) CAE 3D Analysis Revision,

 Item Revision
 <-(TC_CAE_Target) CAE 3D Analysis Revision

)

*/

Cae1SimulationSearchProvider.CAEAnalysisRevision.
(S2P$TC_CAE_Source:CAEGeometryRevision
#S2P$TC_CAE_Source:CAEModelRevision
#S2P$TC_CAE_Defining

^^S2P$TC_CAE_Target:CAEGeometryRevision
#S2P$TC_CAE_Source:CAEModelRevision#S2P$TC_CAE_Defining

^^S2P$TC_CAE_Source:CAEGeometryRevision#S2P$TC_CAE_Source
^^S2P$TC_CAE_Target:CAEGeometryRevision#S2P$TC_CAE_Source
^^S2P$TC_CAE_Source:CAEModelRevision#S2P$TC_CAE_Defining
^^S2P$TC_CAE_Target:CAEModelRevision#S2P$TC_CAE_Defining
^^S2P$TC_CAE_Target)
```

b. Edit the **objectSet source** properties in the **tc_xrt_Analysis** section of the style sheet.

Default object source:

```
<section titleKey="tc_xrt_Analysis">
  <objectSet source="Cae1SimulationSearchProvider.CAEAnalysisRevision"
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

Change to:

```
<section titleKey="tc_xrt_Analysis">
<!-- ********** Start of customization ********** -->
  <objectSet source="Cae1SimulationSearchProvider.CAEAnalysisRevision.
(S2P$TC_CAE_Source:CAEGeometryRevision
#S2P$TC_CAE_Source:CAEModelRevision
#S2P$TC_CAE_Defining

^^S2P$TC_CAE_Target:CAEGeometryRevision
#S2P$TC_CAE_Source:CAEModelRevision#S2P$TC_CAE_Defining

^^S2P$TC_CAE_Source:CAEGeometryRevision#S2P$TC_CAE_Source
^^S2P$TC_CAE_Target:CAEGeometryRevision#S2P$TC_CAE_Source
^^S2P$TC_CAE_Source:CAEModelRevision#S2P$TC_CAE_Defining
^^S2P$TC_CAE_Target:CAEModelRevision#S2P$TC_CAE_Defining
^^S2P$TC_CAE_Target)"
<!-- ********** End of customization ********** -->
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

7. To save your changes, click **Apply**.

8. To view the customized revisions, search for an item revision type in Active Workspace, select it from the **Results** page, and click the **Simulation** tab to view the details. The **Related Simulation Objects** table displays the customized model, result, geometry, and analysis revisions.

9. Edit the **Cae1ItemRevSummaryForShowObjectLocation** style sheet.

   a. To search for style sheets related to the item revision type, in My Teamcenter (rich client), type *Cae1ItemRevSummary**, select **Dataset** type, and perform a search.

      The search results show **Cae1ItemRevSummary** and **Cae1ItemRevSummaryForShowObjectLocation**.

   b. To view the style sheet, select **Cae1ItemRevSummaryForShowObjectLocation** from the **Search Results** view, and click the **Viewer** tab.

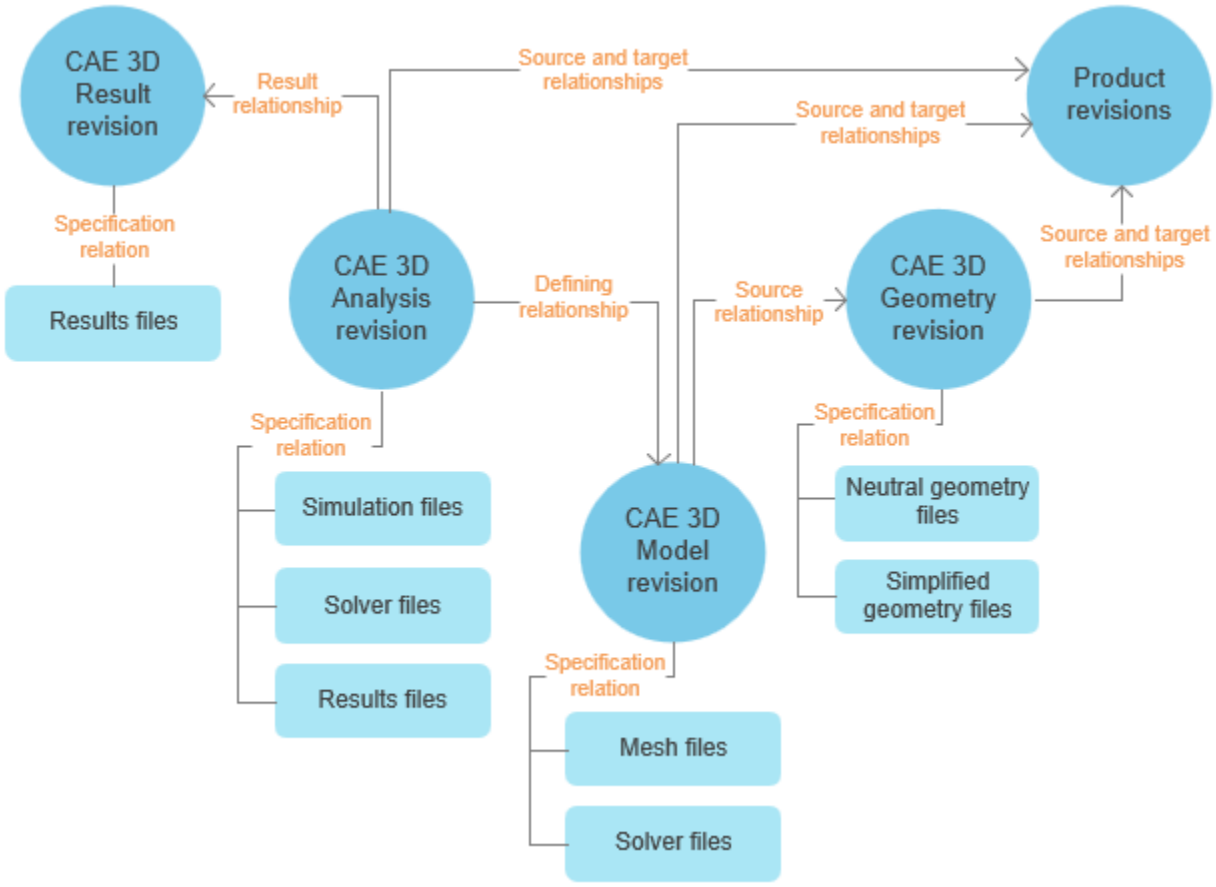
   c. Repeat steps 3 through 7.

d. To view the customized revisions, search for an item revision type in Active Workspace, select it from the **Results** page, click **Open** to open it separately, and click the **Simulation** tab to view the details. The **Related Simulation Objects** table displays the customized model, result, geometry, and analysis revisions.

Expose custom revision types in the context of the geometry revision

You (as a user with **dba** privileges) can edit style sheets to expose custom revision types in the context of the product revision and configure traversal paths to define how to traverse the data structure and specify which relationships are of interest and what should be done when these relationships are encountered. For example, you can define a traversal path from a primary item type, such as an analysis revision, to the solver-specific data deck (**CAESolver**) and from **CAESolver** to the file to be exported.

The following is an example of the generic simulation data model.



In Active Workspace, users can select a geometry revision from the **Results** page and view the details in the **Simulation** tab. Alternatively, they can select a geometry revision from the **Results** page, click **Open** to open it separately, and view the details in the **Simulation** tab. The **Cae1GeometryRevSummary** and the **Cae1GeometryRevSummaryForShowObjectLocation** style sheets determine the objects displayed in the **Related Simulation Objects** table for the respective selection.

You can edit style sheets to add a new traversal path and expose the new objects or edit the default traversal path to hide some objects in the graphical user interface (GUI). The style sheets referred to here are XML documents stored in Teamcenter **XMLStylesheetRendering** datasets and you can edit them using the **Viewer** tab in Teamcenter rich client.

The following is an example of how to edit the style sheet to customize the GUI to display product, model, analysis, and result revisions in the **Related Simulation Objects** table.

To expose custom revision types in the context of the geometry revision:

- Edit the **Cae1ItemRevSummary** style sheet and add customized product, model, analysis, and result revisions. In addition, create construct queries for each of these custom revisions to specify traversal paths.

- Edit the **Cae1ItemRevSummaryForShowObjectLocation** style sheet and add customized product, model, analysis, and result revisions. In addition, create construct queries for each of these custom revisions to specify traversal paths.

Edit style sheets using the Viewer tab in Teamcenter rich client

1. In Teamcenter rich client, sign in as a **dba** user.

2. Edit the **Cae1GeometryRevSummary** style sheet.

    a. To search for style sheets related to the item revision type, in My Teamcenter (rich client), type *Cae1GeometryRevSummary**, select **Dataset** type, and perform a search.

    The search results show **Cae1GeometryRevSummary** and **Cae1GeometryRevSummaryForShowObjectLocation**.

    b. To view the style sheet, select **Cae1GeometryRevSummary** from the **Search Results** view, and click the **Viewer** tab.

3. Add product revisions to the style sheet.

    a. Create a construct query.

    Example:

```
/* Construct Query to get Product

(

CAE 3D Geometry Revision (TC_CAE_Source/TC_CAE_Target)
-> Item Revision

)
```

```
*/

Cae1SimulationSearchProvider.ItemRevision.
(TC_CAE_Source^^TC_CAE_Target)
```

b. Edit the **objectSet source** properties in the **tc_xrt_Product** section of the style sheet.

Default object source:

```
<section titleKey="tc_xrt_Product">
  <objectSet source="Cae1SimulationSearchProvider.ItemRevision"
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

Change to:

```
<section titleKey="tc_xrt_Product">
<!-- ********** Start of customization ********** -->
  <objectSet source="Cae1SimulationSearchProvider.ItemRevision.
(TC_CAE_Source^^TC_CAE_Target)"
<!-- ********** End of customization ********** -->
 sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

4. Add model revisions to the style sheet.

a. Create a construct query.

Example:

```
/* Construct Query to get Model

(

CAEGeometryRevision
<-(TC_CAE_Source) CAEModelRevision

)

*/


Cae1SimulationSearchProvider.CAEModelRevision.
(S2P$TC_CAE_Source)
```

b. Edit the **objectSet source** properties in the **tc_xrt_Model** section of the style sheet.

Default object source:

```
<section titleKey="tc_xrt_Model">
  <objectSet source="Cae1SimulationSearchProvider.CAEModelRevision"
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

Change to:

```
<section titleKey="tc_xrt_Model">
<!-- ********** Start of customization ********** -->
  <objectSet source="Cae1SimulationSearchProvider.CAEModelRevision.
  (S2P$TC_CAE_Source)"
<!-- ********** End of customization ********** -->
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
```

```
      </tableDisplay>
    </objectSet>
</section>
```

> Tip:
>
> You can specify **S2P$** as the prefix before a relation to specify that the traversal path is from the secondary object to the primary object, **^^** as an *AND* operator between two paths, and **#** as the separator between the two segments of the traversal path.

5. Add analysis revisions to the style sheet.

   a. Create a construct query.

      Example:

```
/* Construct Query to get Analysis

 (

 CAE 3D Geometry Revision
 <-(TC_CAE_Source) CAE 3D Analysis Revision,

 CAE 3D Geometry Revision
 <-(TC_CAE_Source) CAE 3D Model Revision
 <-(TC_CAE_Defining) CAE 3D Analysis Revision

 )

 */

 Cae1SimulationSearchProvider.CAEAnalysisRevision.
 (S2P$TC_CAE_Source ^^S2P$TC_CAE_Source:CAEModelRevision
 #S2P$TC_CAE_Defining)
```

   b. Edit the **objectSet source** properties in the **tc_xrt_Analysis** section of the style sheet.

      Default object source:

```
<section titleKey="tc_xrt_Analysis">
  <objectSet source="Cae1SimulationSearchProvider.CAEAnalysisRevision"
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
```

```
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

Change to:

```
<section titleKey="tc_xrt_Analysis">
<!-- ********** Start of customization ********** -->
  <objectSet source="Cae1SimulationSearchProvider.CAEAnalysisRevision.
  (S2P$TC_CAE_Source^^S2P$TC_CAE_Source:CAEModelRevision
  #S2P$TC_CAE_Defining)"
<!-- ********** End of customization ********** -->
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

6. Add result revisions to the style sheet.

   a. Create a construct query.

      Example:

```
/* Construct Query to get Result

 (

 CAE 3D Geometry Revision
 <-(TC_CAE_Source) CAE 3D Analysis Revision
 (TC_CAE_Results)
 -> CAE 3D Result Revision,

 CAE 3D Geometry Revision
 <-(TC_CAE_Source) CAE 3D Model Revision
 <-(TC_CAE_Defining) CAE 3D Analysis Revision (TC_CAE_Results)
 -> CAE 3D Result Revision

 )

 */
```

```
 Cae1SimulationSearchProvider.CAEResultRevision.
 (S2P$TC_CAE_Source:
 CAEAnalysisRevision#TC_CAE_Results
 ^^S2P$TC_CAE_Source:CAEModelRevision
 #S2P$TC_CAE_Defining:CAEAnalysisRevision
 #TC_CAE_Results)
```

b. Edit the **objectSet source** properties in the **tc_xrt_Result** section of the style sheet.

Default object source:

```
<section titleKey="tc_xrt_Result">
  <objectSet source="Cae1SimulationSearchProvider.CAEResultRevision"
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

Change to:

```
<section titleKey="tc_xrt_Result">
<!-- ********** Start of customization ********** -->
  <objectSet source="Cae1SimulationSearchProvider.CAEResultRevision.
 (S2P$TC_CAE_Source:
 CAEAnalysisRevision#TC_CAE_Results
 ^^S2P$TC_CAE_Source:CAEModelRevision
 #S2P$TC_CAE_Defining:CAEAnalysisRevision
 #TC_CAE_Results)"
<!-- ********** End of customization ********** -->
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

7. To save your changes, click **Apply**.

8. To view the customized revisions, search for a geometry revision type in Active Workspace, select it from the **Results** page, and click the **Simulation** tab to view the details. The **Related Simulation Objects** table displays the customized product, model, result, and analysis revisions.

9. Edit the **Cae1GeometryRevSummaryForShowObjectLocation** style sheet.

    a. To search for style sheets related to the item revision type, in My Teamcenter (rich client), type *Cae1GeometryRevSummary**, select **Dataset** type, and perform a search.

    The search results show **Cae1GeometryRevSummary** and **Cae1GeometryRevSummaryForShowObjectLocation**.

    b. To view the style sheet, select **Cae1GeometryRevSummaryForShowObjectLocation** from the **Search Results** view, and click the **Viewer** tab.

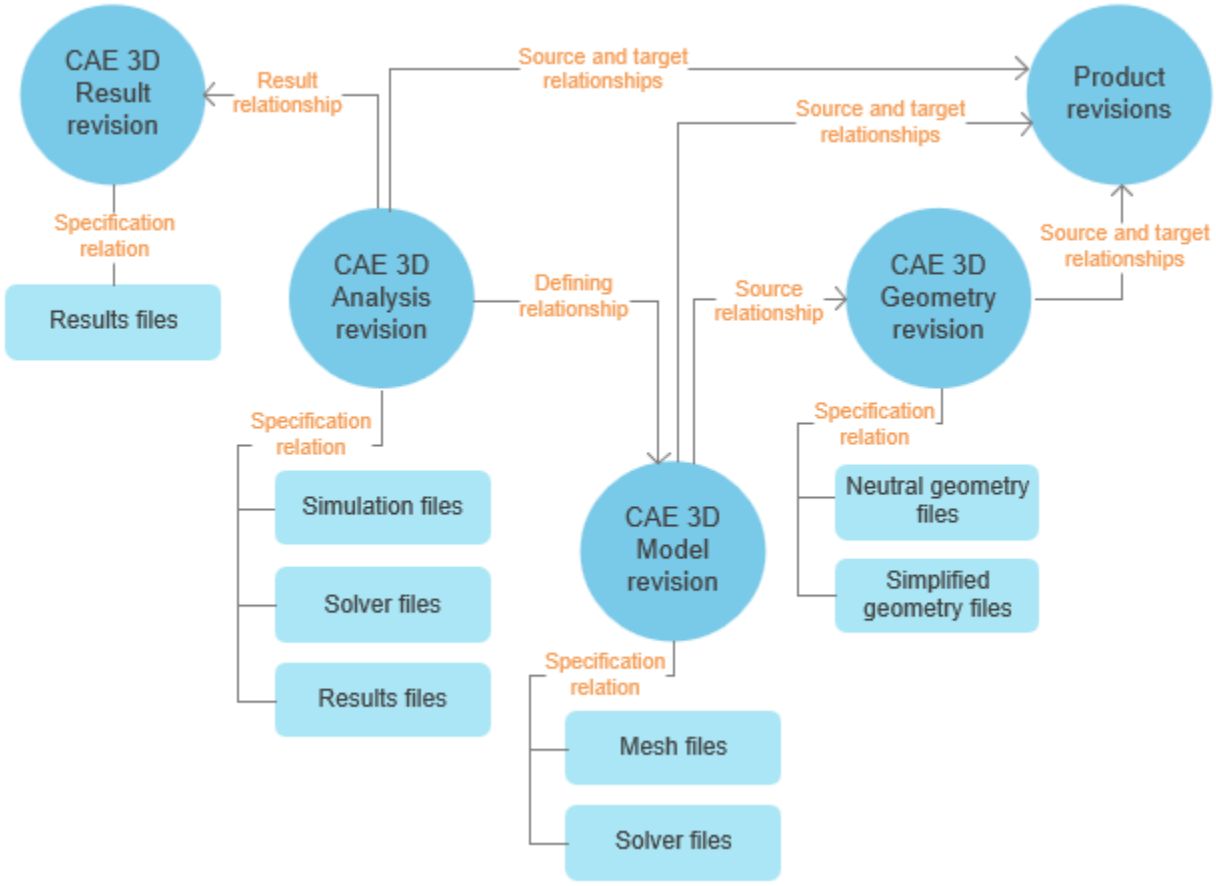
    c. Repeat steps 3 through 7.

    d. To view the customized revisions, search for a geometry revision type in Active Workspace, select it from the **Results** page, click **Open** to open it separately, and click the **Simulation** tab to view the details. The **Related Simulation Objects** table displays the customized product, model, result, and analysis revisions.

Expose custom revision types in the context of the model revision

You (as a user with **dba** privileges) can edit style sheets to expose custom revision types in the context of the product revision and configure traversal paths to define how to traverse the data structure and specify which relationships are of interest and what should be done when these relationships are encountered. For example, you can define a traversal path from a primary item type, such as an analysis revision, to the solver-specific data deck (**CAESolver**) and from **CAESolver** to the file to be exported.

The following is an example of the generic simulation data model.



In Active Workspace, users can select a model revision from the **Results** page and view the details in the **Simulation** tab. Alternatively, they can select a model revision type from the **Results** page, click **Open** to open it separately, and view the details in the **Simulation** tab. The **Cae1ModelRevSummary** and the **Cae1ModelRevSummaryForShowObjectLocation** style sheets determine the objects displayed in the **Related Simulation Objects** table for the respective selection.

You can edit style sheets to add a new traversal path and expose the new objects or edit the default traversal path to hide some objects in the graphical user interface (GUI). The style sheets referred to here are XML documents stored in Teamcenter **XMLStylesheetRendering** datasets and you can edit them using the **Viewer** tab in Teamcenter rich client.

The following is an example of how to edit the style sheet to customize the GUI to display product, geometry, analysis, and result revisions in the **Related Simulation Objects** table.

To expose custom revision types in the context of the model revision:

- Edit the **Cae1ItemRevSummary** style sheet and add customized product, geometry, analysis, and result revisions. In addition, create construct queries for each of these custom revisions to specify traversal paths.

- Edit the **Cae1ItemRevSummaryForShowObjectLocation** style sheet and add customized product, geometry, analysis, and result revisions. In addition, create construct queries for each of these custom revisions to specify traversal paths.

Edit style sheets using the Viewer tab in Teamcenter rich client

1. In Teamcenter rich client, sign in as a **dba** user.

2. Edit the **Cae1ModelRevSummary** style sheet.

   a. To search for style sheets related to the item revision type, in My Teamcenter (rich client), type *Cae1ModelRevSummary**, select **Dataset** type, and perform a search.

      The search results show **Cae1ModelRevSummary** and **Cae1ModelRevSummaryForShowObjectLocation**.

   b. To view the style sheet, select **Cae1ModelRevSummary** from the **Search Results** view, and click the **Viewer** tab.

3. Add product revisions to the style sheet.

   a. Create a construct query.

      Example:

```
/* Construct Query to get Product

 (

 CAE 3D Model Revision (TC_CAE_Source)
 -> CAE 3D Geometry Revision(TC_CAE_Source/TC_CAE_Target)
 -> Item Revision,

 CAE 3D Model Revision (TC_CAE_Source/TC_CAE_Target)
 -> Item Revision

 )

 */

Cae1SimulationSearchProvider.ItemRevision.
(TC_CAE_Source:CAEGeometryRevision#TC_CAE_Source
^^TC_CAE_Source:CAEGeometryRevision#TC_CAE_Target
^^TC_CAE_Source^^TC_CAE_Target)
```

   b. Edit the **objectSet source** properties in the **tc_xrt_Product** section of the style sheet.

Default object source:

```
<section titleKey="tc_xrt_Product">
  <objectSet source="Cae1SimulationSearchProvider.ItemRevision"
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

Change to:

```
<section titleKey="tc_xrt_Product">
  <!-- ************* Start of customization ************* -->
  "Cae1SimulationSearchProvider.ItemRevision.
  (TC_CAE_Source:CAEGeometryRevision#TC_CAE_Source
  ^^TC_CAE_Source:CAEGeometryRevision#TC_CAE_Target
  ^^TC_CAE_Source^^TC_CAE_Target)"
  <!-- ************* End of customization ************* -->
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

> Tip:
>
> You can specify **S2P$** as the prefix before a relation to specify that the traversal path is from the secondary object to the primary object, **^^** as an *AND* operator between two paths, and **#** as the separator between the two segments of the traversal path.

4. Add geometry revisions to the style sheet.

   a. Create a construct query.

      Example:

```
/* Construct Query to get Geometry

(

CAE 3D Model Revision (TC_CAE_Source)
-> CAE 3D Geometry Revision

)

*/

Cae1SimulationSearchProvider.CAEGeometryRevision.
(TC_CAE_Source)
```

b. Edit the **objectSet source** properties in the **tc_xrt_Geometry** section of the style sheet.

Default object source:

```
<section titleKey="tc_xrt_Geometry">
  <objectSet source="Cae1SimulationSearchProvider.CAEGeometryRevision"
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

Change to:

```
<section titleKey="tc_xrt_Geometry">
  <!-- ************* Start of customization ************* -->
  <objectSet source="Cae1SimulationSearchProvider.
  CAEGeometryRevision.(TC_CAE_Source)"
  <!-- ************* End of customization ************* -->
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
```

```
    </objectSet>
</section>
```

5. Add analysis revisions to the style sheet.

   a. Create a construct query.

      Example:

```
/* Construct Query to get Analysis

(

CAE 3D Model Revision
<-(TC_CAE_Defining) CAE 3D Analysis Revision

)

*/

Cae1SimulationSearchProvider.CAEAnalysisRevision.
(S2P$TC_CAE_Defining)
```

   b. Edit the **objectSet source** properties in the **tc_xrt_Analysis** section of the style sheet.

      Default object source:

```
<section titleKey="tc_xrt_Analysis">
  <objectSet source="Cae1SimulationSearchProvider.CAEAnalysisRevision"
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

      Change to:

```
<section titleKey="tc_xrt_Analysis">
  <!-- ************* Start of customization ************* -->
  <objectSet source="Cae1SimulationSearchProvider.CAEAnalysisRevision.
(S2P$TC_CAE_Defining)"
  <!-- ************* End of customization ************* -->
```

```
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

6. Add result revisions to the style sheet.

   a. Create a construct query.

      Example:

```
/* Construct Query to get Result

(

CAE 3D Model Revision
<-(TC_CAE_Defining) CAE 3D Analysis Revision
(TC_CAE_Results)
-> CAE 3D Result Revision

)

*/

Cae1SimulationSearchProvider.CAEResultRevision.
(S2P$TC_CAE_Defining:CAEAnalysisRevision
#TC_CAE_Results)
```

   b. Edit the **objectSet source** properties in the **tc_xrt_Result** section of the style sheet.

      Default object source:

```
<section titleKey="tc_xrt_Result">
  <objectSet source="Cae1SimulationSearchProvider.CAEResultRevision"
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
```

```
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

Change to:

```
<section titleKey="tc_xrt_Result">
<!-- ************* start of customization ************* -->
  <objectSet source="Cae1SimulationSearchProvider.CAEResultRevision.
  (S2P$TC_CAE_Defining:CAEAnalysisRevision#TC_CAE_Results)"
<!-- ************* end of customization ************* -->
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

7. To save your changes, click **Apply**.

8. To view the customized revisions, search for a model revision type in Active Workspace, select it from the **Results** page, and click the **Simulation** tab to view the details. The **Related Simulation Objects** table displays the customized product, result, geometry, and analysis revisions.

9. Edit the **Cae1ModelRevSummaryForShowObjectLocation** style sheet.

   a. To search for style sheets related to the item revision type, in My Teamcenter (rich client), type *Cae1ModelRevSummary**, select **Dataset** type, and perform a search.

      The search results show **Cae1ModelRevSummary** and **Cae1ModelRevSummaryForShowObjectLocation**.

   b. To view the style sheet, select **Cae1ModelRevSummaryForShowObjectLocation** from the **Search Results** view, and click the **Viewer** tab.

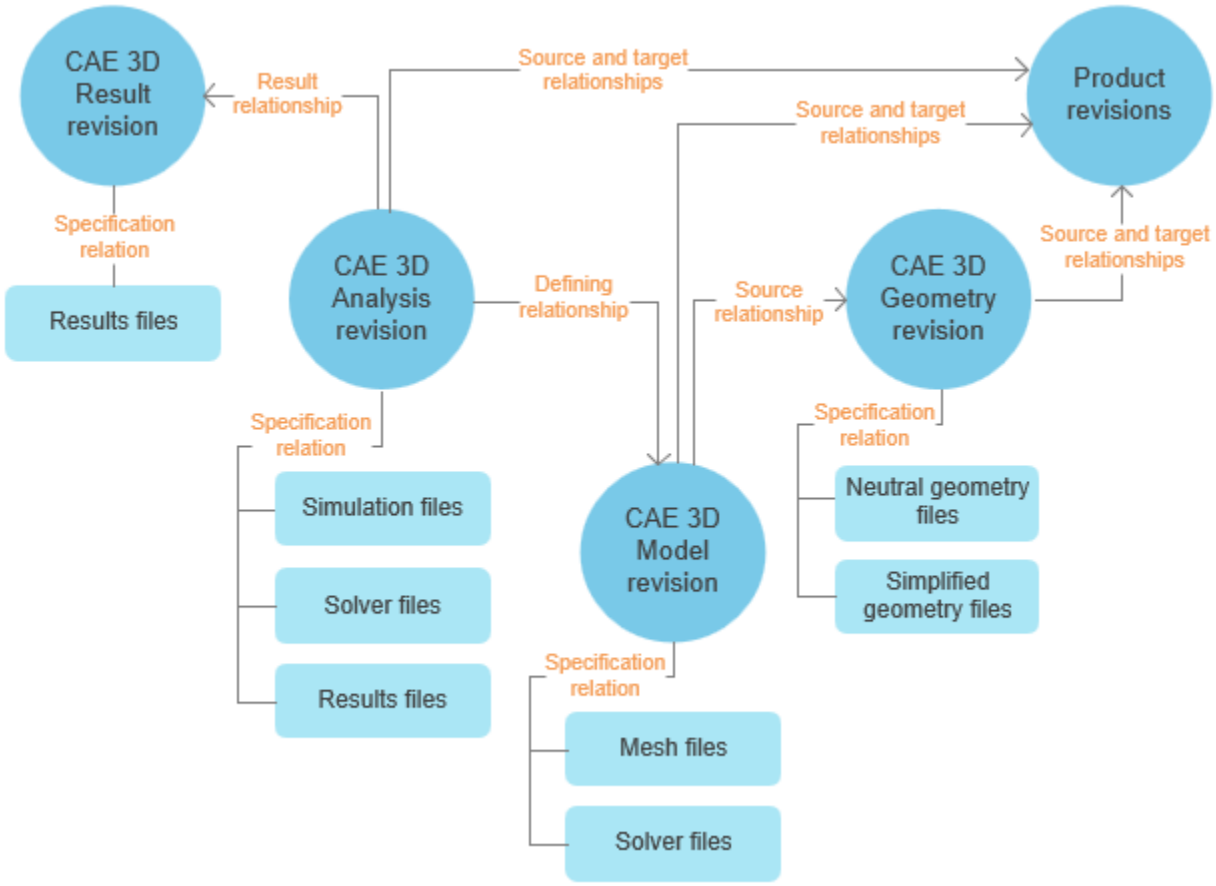
   c. Repeat steps 3 through 7.

   d. To view the customized revisions, search for a model revision type in Active Workspace, select it from the **Results** page, click **Open** to open it separately, and click the **Simulation** tab to view the details. The **Related Simulation Objects** table displays the customized product, result, geometry, and analysis revisions.

Expose custom revision types in the context of the analysis revision

You (as a user with **dba** privileges) can edit style sheets to expose custom revision types in the context of the product revision and configure traversal paths to define how to traverse the data structure and specify which relationships are of interest and what should be done when these relationships are encountered. For example, you can define a traversal path from a primary item type, such as an analysis revision, to the solver-specific data deck (**CAESolver**) and from **CAESolver** to the file to be exported.

The following is an example of the generic simulation data model.



In Active Workspace, users can select an analysis revision from the **Results** page and view the details in the **Simulation** tab. Alternatively, they can select an analysis revision from the **Results** page, click **Open** to open it separately, and view the details in the **Simulation** tab. The **Cae1AnalysisRevSummary** and the **Cae1AnalysisRevSummaryForShowObjectLocation** style sheets determine the objects displayed in the **Related Simulation Objects** table for the respective selection.

You can edit style sheets to add a new traversal path and expose the new objects or edit the default traversal path to hide some objects in the graphical user interface (GUI). The style sheets referred to here are XML documents stored in Teamcenter **XMLStylesheetRendering** datasets and you can edit them using the **Viewer** tab in Teamcenter rich client.

To expose custom revision types in the context of the analysis revision:

- Edit the **Cae1ItemRevSummary** style sheet and add customized product, geometry, model, and result revisions. In addition, create construct queries for each of these custom revisions to specify traversal paths.

- Edit the **Cae1ItemRevSummaryForShowObjectLocation** style sheet and add customized product, geometry, model, and result revisions. In addition, create construct queries for each of these custom revisions to specify traversal paths.

The following is an example of how to edit the style sheet to customize the GUI to display product, geometry, model, and result revisions in the **Related Simulation Objects** table.

Edit style sheets using the Viewer tab in Teamcenter rich client

1. In Teamcenter rich client, sign in as a **dba** user.

2. Edit the **Cae1AnalysisRevSummary** style sheet.

    a. To search for style sheets related to the item revision type, in My Teamcenter (rich client), type *Cae1AnalysisRevSummary**, select **Dataset** type, and perform a search.

    The search results show **Cae1AnalysisRevSummary** and **Cae1AnalysisRevSummaryForShowObjectLocation**.

    b. To view the style sheet, select **Cae1AnalysisRevSummary** from the **Search Results** view, and click the **Viewer** tab.

3. Add product revisions to the style sheet.

    a. Create a construct query.

    Example:

```
/* Construct Query to get Product

 (

 CAE 3D Analysis Revision (TC_CAE_Defining)
 -> CAE 3D Model Revision (TC_CAE_Source)
 -> CAE 3D Geometry Revision (TC_CAE_Source/TC_CAE_Target)
 -> Item Revision,

 CAE 3D Analysis Revision (TC_CAE_Defining)
 -> CAE 3D Model Revision (TC_CAE_Source/TC_CAE_Target)
 -> Item Revision,
```

```
 CAE 3D Analysis Revision (TC_CAE_Source)
 -> CAE 3D Geometry Revision (TC_CAE_Source/TC_CAE_Target)
 -> Item Revision,

 CAE 3D Analysis Revision (TC_CAE_Target)-> Item Revision

 )

 */

Cae1SimulationSearchProvider.ItemRevision.
(TC_CAE_Defining:CAEModelRevision#TC_CAE_Source:CAEGeometryRevision
#TC_CAE_Source^^TC_CAE_Defining:CAEModelRevision
#TC_CAE_Source:CAEGeometryRevision#TC_CAE_Target
^^TC_CAE_Defining:CAEModelRevision#TC_CAE_Source
^^TC_CAE_Defining:CAEModelRevision#TC_CAE_Target
^^TC_CAE_Source:CAEGeometryRevision#TC_CAE_Source
^^TC_CAE_Source:CAEGeometryRevision#TC_CAE_Target
^^TC_CAE_Target)
```

b. Edit the **objectSet source** properties in the **tc_xrt_Product** section of the style sheet.

Default object source:

```
<section titleKey="tc_xrt_Product">
  <objectSet source="Cae1SimulationSearchProvider.ItemRevision"
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

Change to:

```
<section titleKey="tc_xrt_Product">
  <!-- ************* Start of customization ************* -->
  <objectSet source="Cae1SimulationSearchProvider.ItemRevision.
(TC_CAE_Defining:CAEModelRevision#TC_CAE_Source:CAEGeometryRevision
#TC_CAE_Source^^TC_CAE_Defining:CAEModelRevision
#TC_CAE_Source:CAEGeometryRevision#TC_CAE_Target
^^TC_CAE_Defining:CAEModelRevision#TC_CAE_Source
^^TC_CAE_Defining:CAEModelRevision#TC_CAE_Target
```

```
^^TC_CAE_Source:CAEGeometryRevision#TC_CAE_Source
^^TC_CAE_Source:CAEGeometryRevision#TC_CAE_Target
^^TC_CAE_Target)"
  <!-- ************* End of customization ************* -->
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

> Tip:
>
> You can specify **S2P$** as the prefix before a relation to specify that the traversal path is from the secondary object to the primary object, **^^** as an *AND* operator between two paths, and **#** as the separator between the two segments of the traversal path.

4. Add geometry revisions to the style sheet.

   a. Create a construct query.

      Example:

```
/* Construct Query to get Geometry

 (

 CAE 3D Analysis Revision (TC_CAE_Defining)
 -> CAE 3D Model Revision (TC_CAE_Source)
 -> CAE 3D Geometry Revision,

 CAE 3D Analysis Revision (TC_CAE_Source)
 -> CAE 3D Geometry Revision

 )

*/

Cae1SimulationSearchProvider.CAEGeometryRevision.
(TC_CAE_Defining:CAEModelRevision#TC_CAE_Source
^^TC_CAE_Source)
```

   b. Edit the **objectSet source** properties in the **tc_xrt_Geometry** section of the style sheet.

Default object source:

```
<section titleKey="tc_xrt_Geometry">
  <objectSet source="Cae1SimulationSearchProvider.CAEGeometryRevision"
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

Change to:

```
<section titleKey="tc_xrt_Geometry">
  <!-- ************* Start of customization ************* -->
  <objectSet source="Cae1SimulationSearchProvider.CAEGeometryRevision.
(TC_CAE_Defining:CAEModelRevision
#TC_CAE_Source^^TC_CAE_Source)"
  <!-- ************* End of customization ************* -->
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

5. Add analysis revisions to the style sheet.

   a. Create a construct query.

      Example:

```
 /* Construct Query to get Analysis

 (

 CAE 3D Analysis Revision (TC_CAE_Include)
 -> CAE 3D Analysis Revision,
```

```
 CAE 3D Analysis Revision (TC_CAE_Include)
 <- CAE 3D Analysis Revision

 )

 */

Cae1SimulationSearchProvider.CAEAnalysisRevision.
(TC_CAE_Include^^S2P$TC_CAE_Include)
```

b. Edit the **objectSet source** properties in the **tc_xrt_Analysis** section of the style sheet.

Default object source:

```
<section titleKey="tc_xrt_Analysis">
  <objectSet source="Cae1SimulationSearchProvider.CAEAnalysisRevision"
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

Change to:

```
<section titleKey="tc_xrt_Analysis">
  <!-- ************* Start of customization ************* -->
  <objectSet source="Cae1SimulationSearchProvider.CAEAnalysisRevision.
  (TC_CAE_Include^^S2P$TC_CAE_Include)"
  <!-- ************* End of customization ************* -->
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

6. Add result revisions to the style sheet.

   a. Create a construct query.

      Example:

```
/* Construct Query to get Result

(

CAE 3D Analysis Revision (TC_CAE_Results)
-> CAE 3D Result Revision

)

*/

Cae1SimulationSearchProvider.CAEResultRevision.
(TC_CAE_Results)
```

   b. Edit the **objectSet source** properties in the **tc_xrt_Result** section of the style sheet.

      Default object source:

```
<section titleKey="tc_xrt_Result">
  <objectSet source="Cae1SimulationSearchProvider.CAEResultRevision"
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

      Change to:

```
<section titleKey="tc_xrt_Result">
<!-- ************* Start of customization ************* -->
  <objectSet source="Cae1SimulationSearchProvider.
  CAEResultRevision.(TC_CAE_Results)"
<!-- ************* End of customization ************* -->
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
```

```
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

7. To save your changes, click **Apply**.

8. To view the customized revisions, search for an analysis revision type in Active Workspace, select it from the **Results** page, and click the **Simulation** tab to view the details. The **Related Simulation Objects** table displays the customized product, model, result, and geometry revisions.

9. Edit the **Cae1AnalysisRevSummaryForShowObjectLocation** style sheet.

   a. To search for style sheets related to the item revision type, in My Teamcenter (rich client), type *Cae1AnalysisRevSummary**, select **Dataset** type, and perform a search.

      The search results show **Cae1AnalysisRevSummary** and **Cae1AnalysisRevSummaryForShowObjectLocation**.

   b. To view the style sheet, select **Cae1AnalysisRevSummaryForShowObjectLocation** from the **Search Results** view, and click the **Viewer** tab.

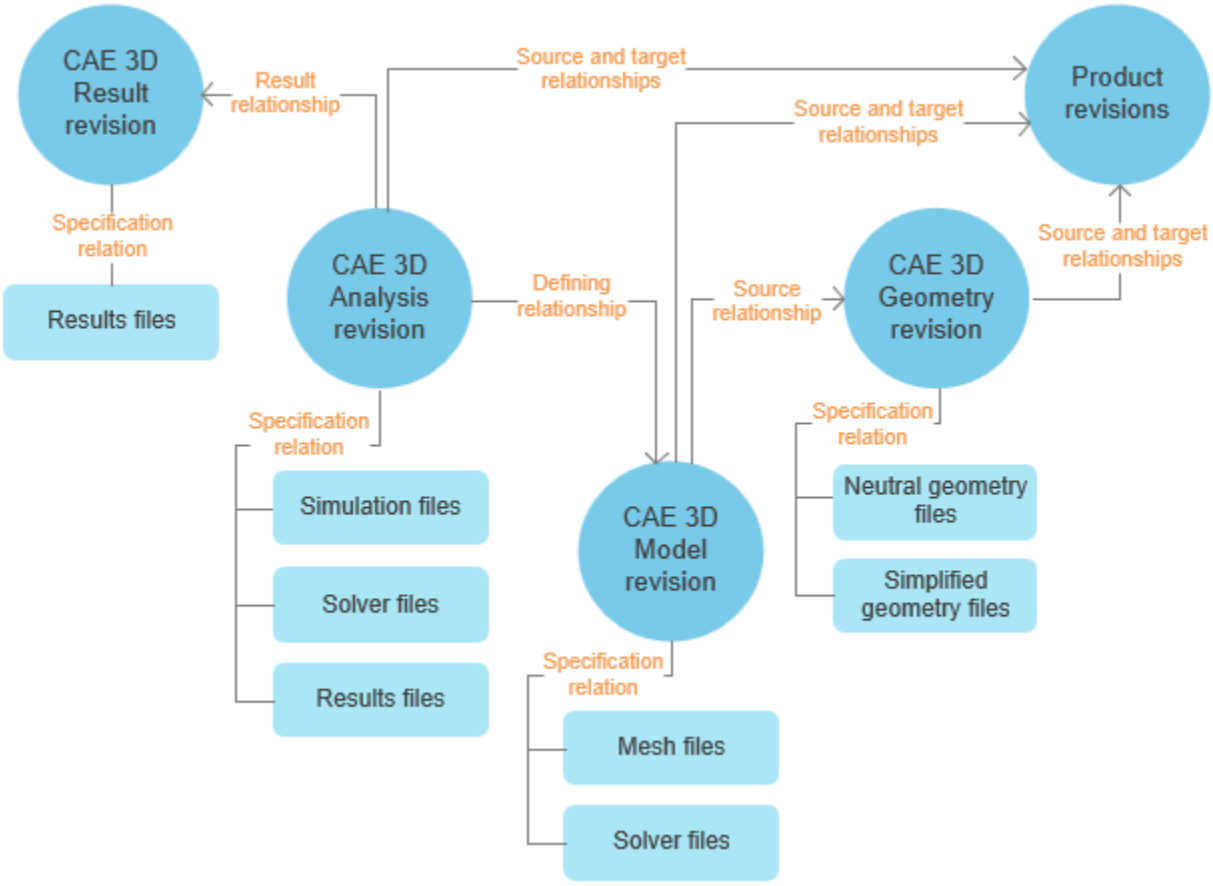
   c. Repeat steps 3 through 7.

   d. To view the customized revisions, search for an analysis revision type in Active Workspace, select it from the **Results** page, click **Open** to open it separately, and click the **Simulation** tab to view the details. The **Related Simulation Objects** table displays the customized product, model, result, and geometry revisions.

Expose custom revision types in the context of the result revision

You (as a user with **dba** privileges) can edit style sheets to expose custom revision types in the context of the product revision and configure traversal paths to define how to traverse the data structure and specify which relationships are of interest and what should be done when these relationships are encountered. For example, you can define a traversal path from a primary item type, such as an analysis revision, to the solver-specific data deck (**CAESolver**) and from **CAESolver** to the file to be exported.

The following is an example of the generic simulation data model.



In Active Workspace, users can select a result revision from the **Results** page and view the details in the **Simulation** tab. Alternatively, they can select a result revision from the **Results** page, click **Open** to open it separately, and view the details in the **Simulation** tab. The **Cae1ResultRevSummary** and the **Cae1ResultRevSummaryForShowObjectLocation** style sheets determine the objects displayed in the **Related Simulation Objects** table for the respective selection.

You can edit style sheets to add a new traversal path and expose the new objects or edit the default traversal path to hide some objects in the graphical user interface (GUI). The style sheets referred to here are XML documents stored in Teamcenter **XMLStylesheetRendering** datasets and you can edit them using the **Viewer** tab in Teamcenter rich client.

To expose custom revision types in the context of the result revision:

- Edit the **Cae1ItemRevSummary** style sheet and add customized product, geometry, model, and analysis revisions. In addition, create construct queries for each of these custom revisions to specify traversal paths.

- Edit the **Cae1ItemRevSummaryForShowObjectLocation** style sheet and add customized product, geometry, model, and analysis revisions. In addition, create construct queries for each of these custom revisions to specify traversal paths.

The following is an example of how to edit the style sheet to customize the GUI to display product, geometry, model, and analysis revisions in the **Related Simulation Objects** table.

Edit style sheets using the Viewer tab in Teamcenter rich client

1. In Teamcenter rich client, sign in as a **dba** user.

2. Edit the **Cae1ResultRevSummary** style sheet.

    a. To search for style sheets related to the item revision type, in My Teamcenter (rich client), type *Cae1ResultRevSummary**, select **Dataset** type, and perform a search.

    The search results show **Cae1ResultRevSummary** and **Cae1ResultRevSummaryForShowObjectLocation**.

    b. To view the style sheet, select **Cae1ResultRevSummary** from the **Search Results** view, and click the **Viewer** tab.

3. Add product revisions to the style sheet.

    a. Create a construct query.

    Example:

```
/* Construct Query to get Product

 (

 CAE 3D Result Revision
 <-(TC_CAE_Results) CAE 3D Analysis Revision (TC_CAE_Defining)
 -> CAE 3D Model Revision (TC_CAE_Source)
 -> CAE 3D Geometry Revision (TC_CAE_Source/TC_CAE_Target)
 -> Item Revision,

 CAE 3D Result Revision
 <-(TC_CAE_Results) CAE 3D Analysis Revision (TC_CAE_Defining)
 -> CAE 3D Model Revision (TC_CAE_Source/TC_CAE_Target)
 -> Item Revision,

 CAE 3D Result Revision
 <-(TC_CAE_Results) CAE 3D Analysis Revision (TC_CAE_Source)
 -> CAE 3D Geometry Revision (TC_CAE_Source/TC_CAE_Target)
 -> Item Revision,

 CAE 3D Result Revision
 <-(TC_CAE_Results) CAE 3D Analysis Revision (TC_CAE_Target)
 -> Item Revision
```

```
 )

 */


Cae1SimulationSearchProvider.ItemRevision.
(S2P$TC_CAE_Results:CAEAnalysisRevision
#TC_CAE_Defining:CAEModelRevision
#TC_CAE_Source:CAEGeometryRevision
#TC_CAE_Source^^S2P$TC_CAE_Results:CAEAnalysisRevision
#TC_CAE_Defining:CAEModelRevision
#TC_CAE_Source:CAEGeometryRevision
#TC_CAE_Target^^S2P$TC_CAE_Results:CAEAnalysisRevision
#TC_CAE_Defining:CAEModelRevision
#TC_CAE_Source^^S2P$TC_CAE_Results:CAEAnalysisRevision
#TC_CAE_Defining:CAEModelRevision
#TC_CAE_Target^^S2P$TC_CAE_Results:CAEAnalysisRevision
#TC_CAE_Source:CAEGeometryRevision
#TC_CAE_Source^^S2P$TC_CAE_Results:CAEAnalysisRevision
#TC_CAE_Source:CAEGeometryRevision
#TC_CAE_Target^^S2P$TC_CAE_Results:CAEAnalysisRevision
#TC_CAE_Target)
```

b. Edit the **objectSet source** properties in the **tc_xrt_Product** section of the style sheet.

Default object source:

```
<section titleKey="tc_xrt_Product">
  <objectSet source="Cae1SimulationSearchProvider.ItemRevision"
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

Change to:

```
<section titleKey="tc_xrt_Product">
  <!-- ************* start of customization ************* -->
  <objectSet source="Cae1SimulationSearchProvider.ItemRevision.
```

```
(S2P$TC_CAE_Results:CAEAnalysisRevision
#TC_CAE_Defining:CAEModelRevision
#TC_CAE_Source:CAEGeometryRevision
#TC_CAE_Source^^S2P$TC_CAE_Results:CAEAnalysisRevision
#TC_CAE_Defining:CAEModelRevision
#TC_CAE_Source:CAEGeometryRevision
#TC_CAE_Target^^S2P$TC_CAE_Results:CAEAnalysisRevision
#TC_CAE_Defining:CAEModelRevision
#TC_CAE_Source^^S2P$TC_CAE_Results:CAEAnalysisRevision
#TC_CAE_Defining:CAEModelRevision
#TC_CAE_Target^^S2P$TC_CAE_Results:CAEAnalysisRevision
#TC_CAE_Source:CAEGeometryRevision
#TC_CAE_Source^^S2P$TC_CAE_Results:CAEAnalysisRevision
#TC_CAE_Source:CAEGeometryRevision
#TC_CAE_Target^^S2P$TC_CAE_Results:CAEAnalysisRevision
#TC_CAE_Target)"
  <!-- ************* end of customization ************* -->
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

> Tip:
>
> You can specify **S2P$** as the prefix before a relation to specify that the traversal path is from the secondary object to the primary object, **^^** as an *AND* operator between two paths, and **#** as the separator between the two segments of the traversal path.

4. Add geometry revisions to the style sheet.

   a. Create a construct query.

      Example:

```
/* Construct Query to get Geometry

(

CAE 3D Result Revision
<-(TC_CAE_Results) CAE 3D Analysis Revision (TC_CAE_Defining)
-> CAE 3D Model Revision (TC_CAE_Source)
```

```
 -> CAE 3D Geometry Revision,

 CAE 3D Result Revision
 <-(TC_CAE_Results) CAE 3D Analysis Revision (TC_CAE_Source)
 -> CAE 3D Geometry Revision

 )

 */

Cae1SimulationSearchProvider.CAEGeometryRevision.
(S2P$TC_CAE_Results:CAEAnalysisRevision
#TC_CAE_Defining:CAEModelRevision
#TC_CAE_Source^^S2P$TC_CAE_Results:CAEAnalysisRevision
#TC_CAE_Source)
```

b. Edit the **objectSet source** properties in the **tc_xrt_Geometry** section of the style sheet.

Default object source:

```
<section titleKey="tc_xrt_Geometry">
  <objectSet source="Cae1SimulationSearchProvider.CAEGeometryRevision"
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

Change to:

```
<section titleKey="tc_xrt_Geometry">
  <!-- ************* start of customization ************* -->
  <objectSet source="Cae1SimulationSearchProvider.CAEGeometryRevision.
(S2P$TC_CAE_Results:CAEAnalysisRevision
#TC_CAE_Defining:CAEModelRevision
#TC_CAE_Source^^S2P$TC_CAE_Results:CAEAnalysisRevision
#TC_CAE_Source)"
  <!-- ************* end of customization ************* -->
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
```

```
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

5. Add model revisions to the style sheet.

   a. Create a construct query.

      Example:

```
/* Construct Query to get Model

(

CAE 3D Result Revision
<-(TC_CAE_Results) CAE 3D Analysis Revision (TC_CAE_Defining)
-> CAE 3D Model Revision

)

*/

Cae1SimulationSearchProvider.CAEModelRevision.
(S2P$TC_CAE_Results:CAEAnalysisRevision#TC_CAE_Defining)
```

   b. Edit the **objectSet source** properties in the **tc_xrt_Model** section of the style sheet.

      Default object source:

```
<section titleKey="tc_xrt_Model">
  <objectSet source="Cae1SimulationSearchProvider.CAEModelRevision"
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

      Change to:

```
<section titleKey="tc_xrt_Model">
<!-- ************* start of customization ************* -->
  <objectSet source="Cae1SimulationSearchProvider.CAEModelRevision.
 (S2P$TC_CAE_Results:CAEAnalysisRevision#TC_CAE_Defining)"
<!-- ************* end of customization ************* -->
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

6. Add analysis revisions to the style sheet.

   a. Create a construct query.

      Example:

```
/* Construct Query to get Analysis

(

CAE 3D Result Revision <-(TC_CAE_Results) CAE 3D Analysis Revision

)

*/

Cae1SimulationSearchProvider.
CAEAnalysisRevision.(S2P$TC_CAE_Results)
```

   b. Edit the **objectSet source** properties in the **tc_xrt_Analysis** section of the style sheet.

      Default object source:

```
<section titleKey="tc_xrt_Analysis">
  <objectSet source="Cae1SimulationSearchProvider.CAEAnalysisRevision"
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
```

```
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

Change to:

```
<section titleKey="tc_xrt_Analysis">
  <!-- ************* start of customization ************* -->
  <objectSet source="Cae1SimulationSearchProvider.
CAEAnalysisRevision.(S2P$TC_CAE_Results)"
  <!-- ************* end of customization ************* -->
    sortdirection="ascending" sortby="object_string"
    defaultdisplay="tableDisplay" maxRowCount="5">
      <tableDisplay>
        <property name="object_string"/>
        <property name="object_type"/>
        <property name="release_status_list"/>
        <property name="date_released"/>
        <property name="owning_user"/>
      </tableDisplay>
    </objectSet>
</section>
```

7. To save your changes, click **Apply**.

8. To view the customized revisions, search for a result revision type in Active Workspace, select it from the **Results** page, and click the **Simulation** tab to view the details. The **Related Simulation Objects** table displays the customized product, geometry, model, and analysis revisions.

9. Edit the **Cae1ResultRevSummaryForShowObjectLocation** style sheet.

   a. To search for style sheets related to the item revision type, in My Teamcenter (rich client), type *Cae1ResultRevSummary**, select **Dataset** type, and perform a search.

      The search results show **Cae1ResultRevSummary** and **Cae1ResultRevSummaryForShowObjectLocation**.

   b. To view the style sheet, select **Cae1ResultRevSummaryForShowObjectLocation** from the **Search Results** view, and click the **Viewer** tab.

   c. Repeat steps 3 through 7.

   d. To view the customized revisions, search for a result revision type in Active Workspace, select it from the **Results** page, click **Open** to open it separately, and click the **Simulation** tab to view the details. The **Related Simulation Objects** table displays the customized product, geometry, model, and analysis revisions.

Customizing the simulation tool launch page in Active Workspace

About customizing the tool launch page

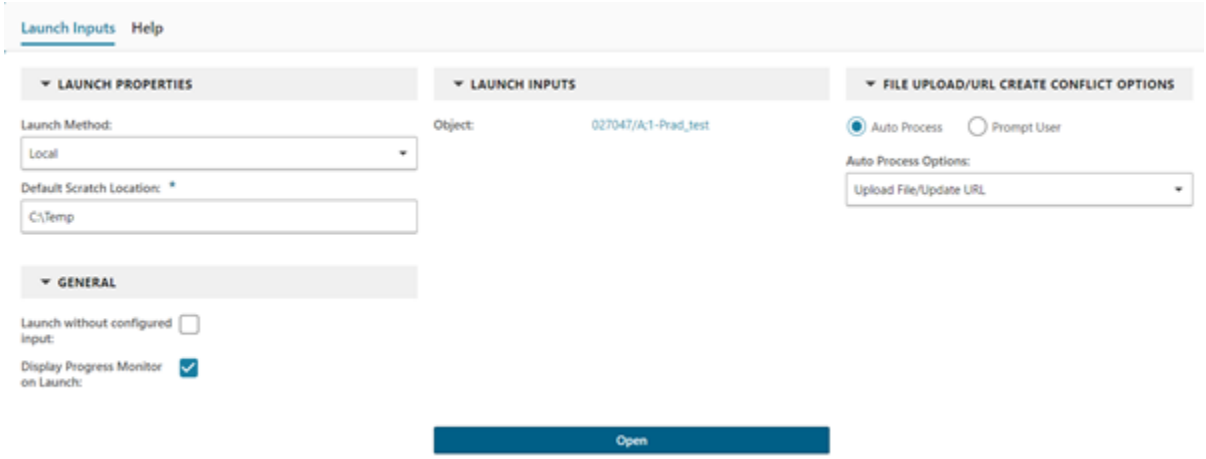
The simulation administrator configures simulation tools and the simulation analyst uses these preconfigured simulation tools to launch preprocessors, solvers, or postprocessors. Each tool might require different inputs based on the requirements of the tool. After configuring simulation tools, an administrative user with DBA privileges can create multiple customized launch pages for different launch tools and associate specific tools to the customized pages.

For more information, see The Active Workspace customization process in *Active Workspace Customization*.

Simulation administrator	1. **Configures the simulation tool**. 2. **Releases the simulation tool**.
An administrative user with DBA privileges	1. **Customizes the simulation tool launch page by using Panel Builder**. 2. **Exports the UI changes to a custom module to avoid overwriting the default simulation tool launch page**. 3. **Builds and publishes the UI changes from the STAGE directory to the file repository**. 4. **Associates a specific tool to the customized simulation tool launch page**. 5. **Make the customization available for upgrades**.
Simulation analyst	1. Opens a geometry, model, or analysis revision. 2. Launches the preconfigured simulation tool by using the customized tool launch page.

Customize the tool launch page

The default tool launch page in Active Workspace displays the following options. You can customize it to suit the requirements at your site.



As a prerequisite, you must install Active Architect. It is a collection of pages that allows you to easily modify existing command placements, visibility, icons, and so on and quickly create your own commands. For more information about installing it, see *What is Active Architect?* in *Configuration and Extensibility*.

> Note:
>
> The following procedures can be done by only an administrative user with DBA privileges.

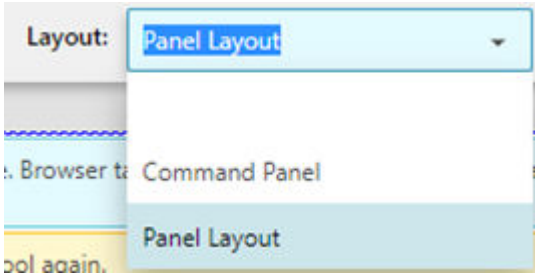
Customize the simulation tool launch page

You can customize the simulation tool launch page by using **Panel Builder**.

1. Log on to Active Workspace URL at your site with administrative privileges.
2. Switch to **Active Architect** workspace.
3. Open **Panel Builder**.

   For more information, see Opening Panel Builder in *Active Workspace Customization*.
4. To activate the development mode, add **devMode** to the URL. For example, you can use, *http://10.134.52.93:3000***/devMode**/#/*canvas?viewModelId=Untitled*.

   For more information, see Using developer mode in *Active Workspace Customization*.
5. Choose **Panel Layout** to see a broader layout of the view.



6. You can perform the following customizations:

   - Drag and drop containers, for example, **Command Panel** section, and organize the existing widgets.
   - Remove any of the HTML tags or widgets.

     If a field is not required, you can remove it. Similarly, if an option has a default value set in the case of a possible conflict, you can remove it.

     > Example:
     >
     > If the **File Upload/URL Create Conflict Options** is removed, the default value is selected. But do not remove the **Select Primary Input File** section because there is no default value for this.

   - Change the order of the HTML widgets.
   - Modify the default display name of the HTML tags or widgets.
   - Modify the CSS style for the widgets or the page.

     CSS styles are not handled in CAE code. You should use only those CSS styles set by Active Workspace administrator.

   - You can display the **Input Parameters** in the **Parameters** section differently using the **param** in the **data.inputParamList** list. It contains a list of input parameters defined in the **Input Parameters** tab of the tool configuration.

Name	Description	Type	Runtime Para...	Script Input O...	Object Type	Value
TC_USER	User Name	String	true	Value only	NA	
TC_PASS	Password	String	true	Value only	NA	
SUBMIT_CLUS...	Submit to Clu...	List Of Values	true	Value only	NA	Ivory
NUM_CORES	Number of Co...	List Of Values	true	Value only	NA	12
SW_VERSION	Software Versi...	List Of Values	true	Value only	NA	2019.1
PRIORITY	Priority	List Of Values	true	Value only	NA	Low

Param details:

Type	Description
param.type	Defines the type of input parameter. Supported types are **STRING**, **BOOLEAN**, and **DOUBLE**.
param.hasLov	Defines if the parameter has list of values. Valid values: • **0** if the parameter does not have a list of values • **1** if the parameter has a list of values.
param.propertyName	Defines the input parameter name defined in the tool configuration.
param.modelList	Contains the list of values defined for the input parameter.

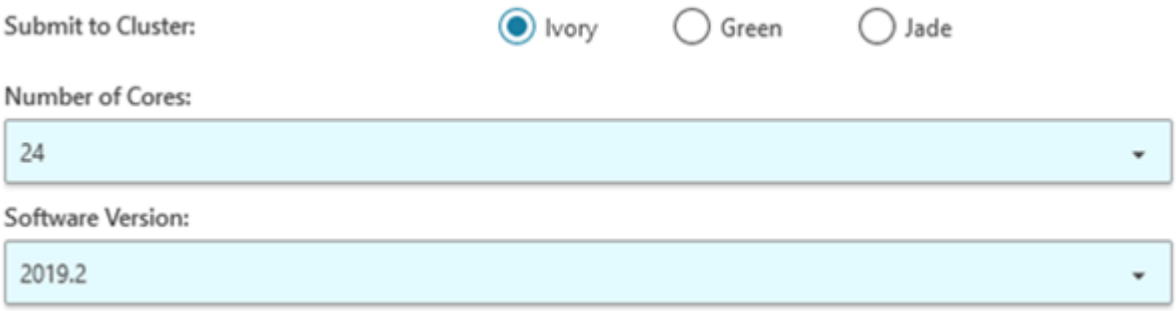
Example 1:

The text box for **User Name** and **Password** fields are displayed based on the **param.propertyName** property.


```
<li aw-repeat="param : data.inputParamList">
 <div exist-when="param.type=='STRING' && param.hasLov=='0' && param.propertyName=='TC_USER'">
  <aw-textbox prop="param">
  </aw-textbox>
 </div>
 <div exist-when="param.type=='STRING' && param.hasLov=='0' && param.propertyName=='TC_PASS'">
  <aw-password prop="param">
  </aw-password>
 </div>
```

Example 2:

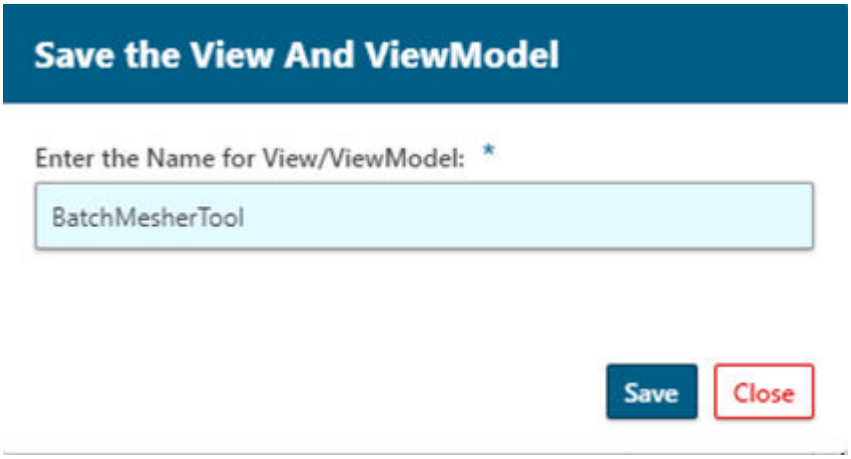
All the properties in example below are configured as LOV in Simulation Tool Configuration view in RAC, but can be displayed differently as radio options and dropdown.



You can make this change by using the **modelList** attribute of **param**. The **Submit to Cluster** radio buttons are displayed by using **param.modelList**.

```
<div exist-when="param.type=='STRING' && param.hasLov=='1' && param.propertyName=='SUBMIT_CLUSTER'">
 <aw-radiobutton prop="param" list="param.modelList">
 </aw-radiobutton>
</div>
<div exist-when="param.type=='STRING' && param.hasLov=='1' && (param.propertyName=='NUM_CORE' || param.propertyName=='SW_VERSION')">
 <aw-widget prop="param">
 </aw-widget>
```

7. After performing the customization, click the **Save Changes** button, specify a name, and click **Save**.



Any changes made to the UI using the UI builder tools are stored by the declarative artifact service in a separate section of the file repository as adds or deltas. No changes are made to the full site file repository.

Export the UI changes to a custom module

You can export the UI changes to a custom module to avoid overwriting the default simulation tool launch page.

To create custom Active Workspace components, you must work within an environment configured for Active Workspace development. Siemens Digital Industries Software provides several scripts to assist

in the development of component modules. Unless otherwise stated, all scripts must be run from the **stage** directory.

For more information about development scripts, see *Active Workspace Customization*.

1. To initialize the environment, run the *TC_ROOT***aws2\stage\initEnv.cmd** script.

2. To create a custom module, run the **generateModule** script from the Teamcenter command prompt. You do so by specifying the **npm run generateModule** command.

3. When the script prompts you to define the type, enter **module** and specify a module name.

Build and publish the UI changes from the stage directory to the file repository

You can publish your local site to the file repository services as the new full site. This makes it your new baseline for this environment. This is similar to a full commit for all your customizations.

For more information about publishing your local site, see *Active Workspace Customization*.

> Note:
>
> You need to build and publish the UI changes only once. However, you must perform these steps for each successive modification in the stylesheet.

1. Export the changes in the site file repository.

   To export the adds and deltas, that is, changes made to the view model to the site repository, run the **exportToSrc** script from the Teamcenter command prompt. You must change to the *TC_ROOT***aws2\stage** directory before running this script.

   You do so by specifying the **npm run exportToSrc** *Gateway Client URL* **--moduleName=***Name_of_the_Module* command. For example, you can specify **npm run exportToSrc** *http://10.134.52.93:3000* **--moduleName=***Name_of_the_module*.

2. Build and publish the changes by using the **awbuild** script to run **initenv**, **npm run build**, and **npm run publish**.

   Type the **awbuild.cmd** command:

Associate specific tools to the customized simulation tool launch pages

You can associate specific tools to the customized simulation tool launch pages by using the XRT editor.

For more information about modifying style sheets using the XRT Editor, see *Active Workspace Customization*.

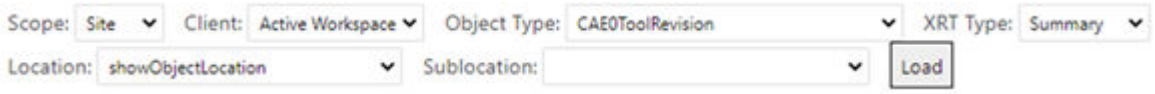
1. Log on to Active Workspace URL at your site by specifying the host and port (separated by a colon). For example, you can use, *http://10.134.52.93:3000*.

2. Select the **Active Architect** workspace and click the **XRT Editor** tile.

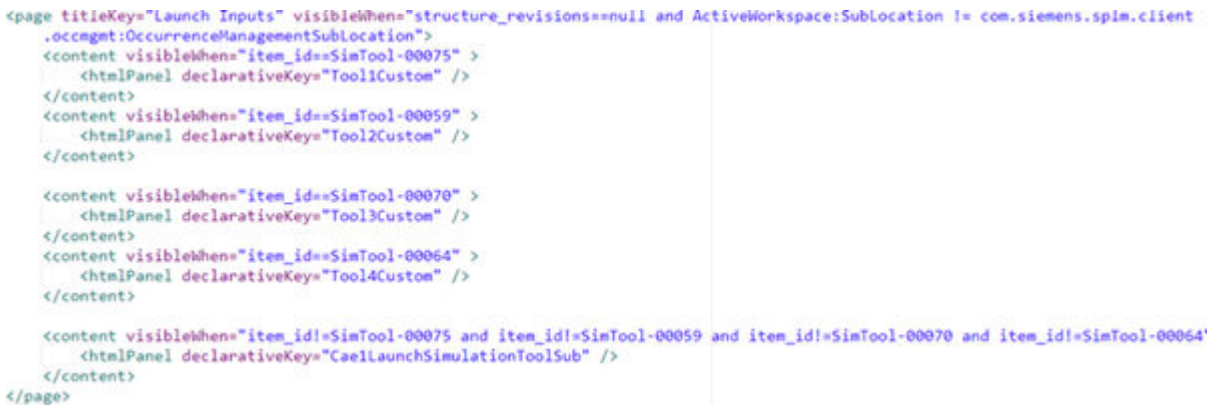
3. Select the following options:

   - **Scope**: **Site**
   - **Client**: **Active Workspace**
   - **Object Type**: **CAE0ToolRevision**
   - **XRT Type**: **Summary**
   - **Location**: **showObjectLocation**

4. Click **Load**.



5. To associate a customized view with a specific tool, use the **CAE0ToolRevision** object item ID to link to the **htmlPanel declarativeKey** as follows:



   Example:

```
<content visibleWhen="item_id==SimTool-00075" >
            <htmlPanel
declarativeKey="BatchMesherTool" />
        </content>
```

6. To associate a view with multiple tools, use the **not equals** condition.

   Example:

```
<content visibleWhen="item_id!=SimTool-00075 and item_id!
=SimTool-00059 and item_id!=SimTool-00070 and item_id!
```

```
=SimTool-00064" >
            <htmlPanel
declarativeKey="Cae1SimToolLaunchInputs" />
        </content>
```

Make the customization available for upgrades

The upgraded client kit does not automatically include the customized module. If you want to make your customization available in the upgraded environment:

1. Add the entry for your module in the **modules** section of the **STAGE\src\solution\kit.json** file.

   Example:

   ```
   "modules": [
   "tc-aw-solution",
   "testtcsim"
   ],
   ```

2. **Build and publish the UI changes from the stage directory to the file repository**.

3. **Associate specific tools to the customized simulation tool launch pages**.

Set the naming pattern for datasets and their related files at the site level

As a simulation administrator, you can use the **CAE_renaming_pattern_for_dataset_and_file_names** site preference to set the naming pattern for datasets and their related files at the site level in Active Workspace.

Procedure

1. Search for the **CAE_renaming_pattern_for_dataset_and_file_names** user preference.

   For information about retrieving a list of preferences, see *Where can I get a list of preferences?* in *Teamcenter Preferences*.

2. Set the preference value as appropriate.

   By default, the value of this preference is empty.

   Example:

   You can set the preference value to **"object_name"_"object_desc"_"item_revision_id"**.

Consider a CAE 3D model revision with **Revision** as **A**, **Name** as **Lower Arm 010**, and **Description** as **IMW_Lower_Arm_010**. In such a case, based on the preference value and the value of the respective properties on the parent CAE object, the datasets and their related files are renamed as **Lower Arm 010_IMW_Lower_Arm_010_A**.

The **Update Dataset and File Name** command is visible on the **Files** tab of the CAE object only when you set a preference value.

3. Save your changes.

6. CAE action handlers

Introduction to workflow

A *workflow* is the automation of business procedures in which documents, information or tasks are passed from one participant to another in a way that is governed by rules or procedures. Teamcenter workflows allow you to manage your product data processes. Typically, documents, information, or tasks are passed from one participant to another in a way that is governed by rules or procedures.

A *workflow process* is initiated by a user, and workflow tasks are assigned to users.

Workflow handlers are small ITK programs used to extend and customize workflow tasks. Action handlers perform actions, such as attaching objects and sending email; rule handlers can identify whether a rule has been satisfied.

For more information, see *Workflow Designer on Rich Client*.

CAE-attach-related-cae-folder-objects

DESCRIPTION

At some sites, not all simulation tools are integrated with Teamcenter. In such cases, simulation analysts can run the simulation tools on their local desktop and periodically upload or download the data to or from Teamcenter. The analysts can create a CAE folder structure within an item revision to manage the different types of files from different simulation tools.

Simulation administrators can configure a workflow process using the **CAE-attach-related-cae-folder-objects** action handler to allow simulation analysts to release the item revision containing the CAE folder structure and it contents.

This action handler attaches the specified related **CAE Folder** (**CAE0FileCollection**) objects of the target objects as target or reference attachments to the workflow process. It searches all the target objects, finds the **CAE Folder** objects recursively, and then adds them as a target or as reference attachments. If a **CAE Folder** object is already part of the target list, it is ignored.

SYNTAX

CAE-attach-related-cae-folder-objects -attachment=*target* | *reference*

ARGUMENTS

-attachment *target* | *reference*

The attachment type with which the objects are attached to the workflow process.

The **–tool** argument is mandatory and requires the simulation tool ID value. The rest of the arguments are optional and can be specified without any values.

PLACEMENT

It is typically placed on the **Start** action of the root task so that the list of target attachments is updated during the workflow process initiation.

RESTRICTIONS

Requires one or more target objects to find the related **CAE Folder** objects. The placement should allow at least one target object before the execution of this handler takes place.

EXAMPLES

This example attaches all the **CAE Folder** objects as target objects to the workflow process when a workflow process is initiated on an CAE item revision.

Argument	Values
-attachment	**target**

CAE-decode-token-and-update-model-attributes

DESCRIPTION

This action handler is for managing Simcenter Client for Git default workflows. The user must provide the Simcenter Client for Git token in the **Comments** box of the workflow process.

This action handler must be configured on the complete action of a **Do** task. It decodes the token and updates the **Model Identifier**, **Model Version**, and **Model Name** properties on the **CAE 1D Model** revision.

SYNTAX

CAE-decode-token-and-update-model-attributes

ARGUMENTS

None

PLACEMENT

It is typically placed on the complete action of the **Do** task only. It requires inputs and users should provide them in the **Comments** box of the workflow process.

RESTRICTIONS

Configure only for **CAE 1D Model** revisions.

CAE-mark-up-to-date

DESCRIPTION

In a complex product development environment, different analysts perform different tasks of the overall analysis. For example, abstractions are delivered by one group, models built by another group, and load cases defined by another group. In such scenarios, it becomes critical to know when the analysis data, possibly with multiple dependencies, is out-of-date. The analyst can then act on it and ensure that the analysis is built with the correct set of data to deliver accurate results.

When analysts complete their work, they have to mark the revisions they worked on as up-to-date. Instead of the analyst manually doing this, the simulation administrator can configure a workflow process using the **CAE-mark-up-to-date** action handler. This allows the system to automatically mark revisions as up-to-date when they are released through a workflow process.

SYNTAX

CAE-mark-up-to-date

ARGUMENTS

None

PLACEMENT

Typically, before the release action.

RESTRICTIONS

Configure only for CAE items.

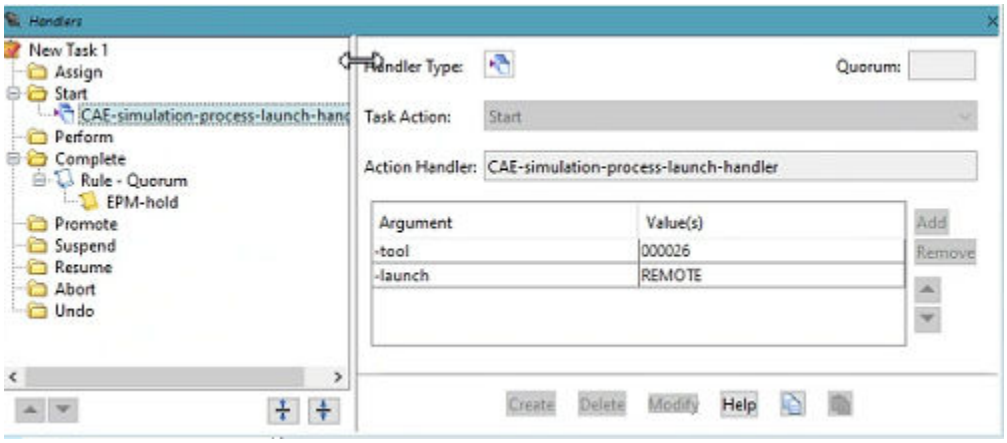
CAE-simulation-process-launch-handler

DESCRIPTION

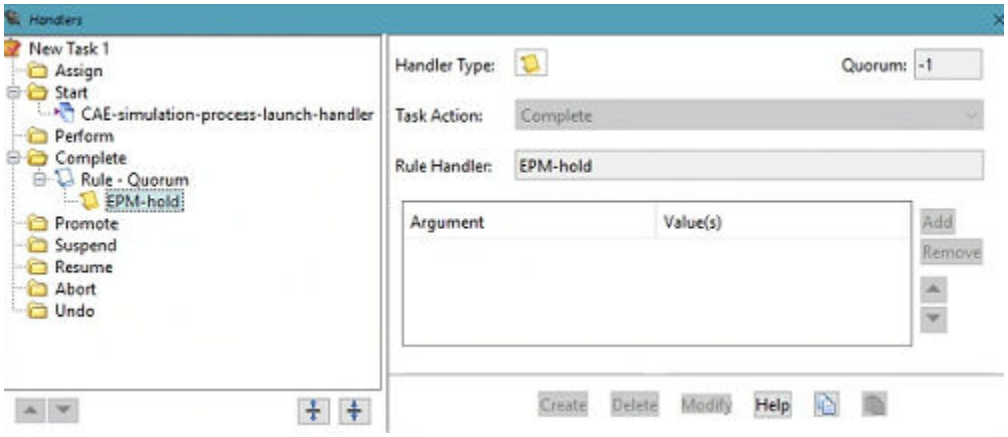
Launches the specified simulation tool.

PLACEMENT

- Place the **CAE-simulation-process-launch-handler** action handler on the **Start** action.



- Place the **EPM-hold** rule handler on the **Complete** action. This stops the task from automatically completing when started.



SYNTAX

CAE-simulation-process-launch-handler -tool=*tool_ID* **-launch=***LOCAL_OR_SERVER_OR_REMOTE* **-nosync -continue -noref -param::**

Note:

When you use **local** as the input for the **-launch** argument, the workflow initiates a server launch. The input means local to the server.

ARGUMENTS

-tool

The ID of the simulation tool to launch.

Note:

The simulation tool ID you specify here must match the simulation tool ID defined in the **Simulation Tool Configuration** dialog box in CAE Manager.

The **–tool** argument is mandatory and requires the simulation tool ID value. The rest of the arguments are optional and can be specified without any values.

Tool names and revisions are no longer supported. The tool is now launched with the latest released revision. If you have an existing action handler with a tool name and revision values, you must modify them and use only the tool ID value.

-launch

This argument is mandatory if you select the **Remote Launch** option in the **Simulation Tool Configuration** dialog box in CAE Manager.

Note:

If this value is not specified, the handler assumes the launch type to be local, this is, the machine on which Teamcenter server is running.

-nosync

If specified, a synchronous process running in the background does not inform the task about its completion. As a result, the control from the current task goes to the next task (if any) as soon as the current task starts.

If not specified, the system displays the following warning:

A simulation batch run is in progress. The task will complete offline after the process completes.

Note:

This argument is valid for local launch only. Remote launch is always run in non-synchronous mode.

This parameter is deprecated. To run the process in synchronous mode, use the **EPM-hold** rule handler on the **Complete action**. Do not use the rule handler if want to run the process in non-synchronous mode.

-continue

If specified, the current task moves to the next task after completion even if the current task fails.

If not specified, the task stops on failure.

> Note:
>
> This argument is valid for local launch only. Remote launch is always run in nonsynchronous mode.
>
> This argument is not valid if you specify the **-nosync** argument.

-noref

If specified, the handler does not add output objects as reference attachments.

If not specified, the handler adds output objects as reference attachments in the **Reference** folder.

> Note:
>
> This argument is valid for local launch only. Remote launch is always run in nonsynchronous mode and output objects are never added as reference attachments.
>
> This argument is not valid if you specify the **-nosync** argument.

-param::*paramName*

Used to assign run-time parameter values for any parameters already defined as part of the tool configuration in the **Simulation Tool Configuration** dialog box in CAE Manager.

Launches the tool with the *paramValue* value for the *paramName* parameter as defined in the tool configuration. The specified parameters are processed according to the defined configuration.

> Note:
>
> The *paramName* value must be defined as a run-time parameter for the tool configuration in the **Simulation Tool Configuration** dialog box. Any run-time parameters defined in the tool configuration that are not indicated as action handler arguments get the default values defined in the tool configuration. The *paramValue* value can be an empty string, in which case the default value of the corresponding *paramName* is overridden with an empty value.

RESTRICTIONS

None.

CAE-structuremap-execution-handler

DESCRIPTION

Performs the structure map execution on the target objects.

> Note:
>
> Users can configure multiple CAE-structuremap-execution-handler with different action handler configurations.

SYNTAX

CAE-structuremap-execution-handler -sm=*structure-map-ID* **-revrule=***Revision-Rule* **-svr=***Saved-Variant-Rule* **–target_owning_user=***owning_user* **-target_owning_group=***owning _group*

ARGUMENTS

-sm *target* | *reference*

(Mandatory) Structure Map Item ID.

-svr

(Optional) Saved variant rules to be applied on the target structure. For multiple SVRs, use comma (**,**) as a separator.

> Note:
>
> Users can configure the SVRs present on other item revisions.

If users provide only the SVR name, the system considers that SVR present on the input item revision, which is a target object.

If users provide the SVR name in the format **<item_id>::svr**, then the system considers the SVR present on the latest working revision of the item to which the **item_id** corresponds.

If users provide the SVR name in the format **<item_id>::<rev_id::>svr**, then the system considers the SVR present on the revision of the item to which the **item_id** and **rev_id** corresponds.

If multiple SVRs are provided the system applies those SVRs which are found. Not found SVRs are skipped.

-revrule

(Optional) Revision rule to be applied on the target structure.

-target_owning_user

(Optional) Owning user to be set for the resulting CAE structure. If not specified, then the current workflow process owner is the owner of the CAE BOM.

-target_owning_group

(Optional) Group name of the owning user to set. If not specified, then the current workflow process owner's group is the owning group of the CAE BOM.

PLACEMENT

Not applicable.

RESTRICTIONS

None.

EXAMPLES

Not applicable.

7. Troubleshooting

Simulation tools are not getting launched through TTLC due to the wrong association of the .tcsimxml file

Issue

Simulation tools are not getting launched through Teamcenter Tool Launcher Client (TTLC) due to the wrong association of the **.tcsimxml** file.

Issue details

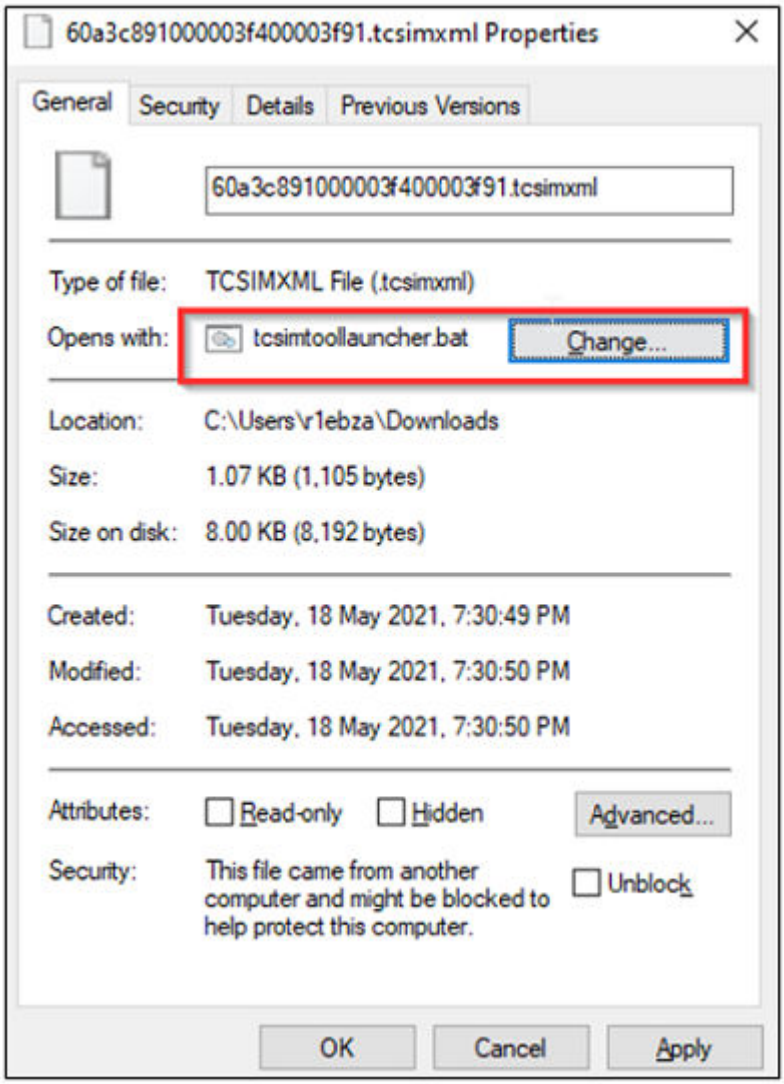
Initially, the customer had an error related to the incompatibility of the TTLC version with the Teamcenter server in the TTLC log. The correct TTLC version was reinstalled and then the tool launch was attempted through TTLC. The **.tcsimxml** file opened with the Notepad application instead of the **tcsimtoollauncher.bat** file. The **tcsimtoollauncher.bat** file calls TTLC and therefore the **.tcsimxml** file has to be associated with this bat file. This association gets created in the registry when TTLC is installed.

Solution or Workaround

If the association of **.tcsimxml** file is wrongly created with another application instead of **tcsimtoollauncher.bat**, then perform following steps to create the correct association:

1. Right click on any **.tcsimxml** file and go to properties.

2. Click the **Change...** button and select the **tcsimtoollauncher.bat** file from TTLC installation.



Perl issue while executing the Extract KPI from Result tool

Issue

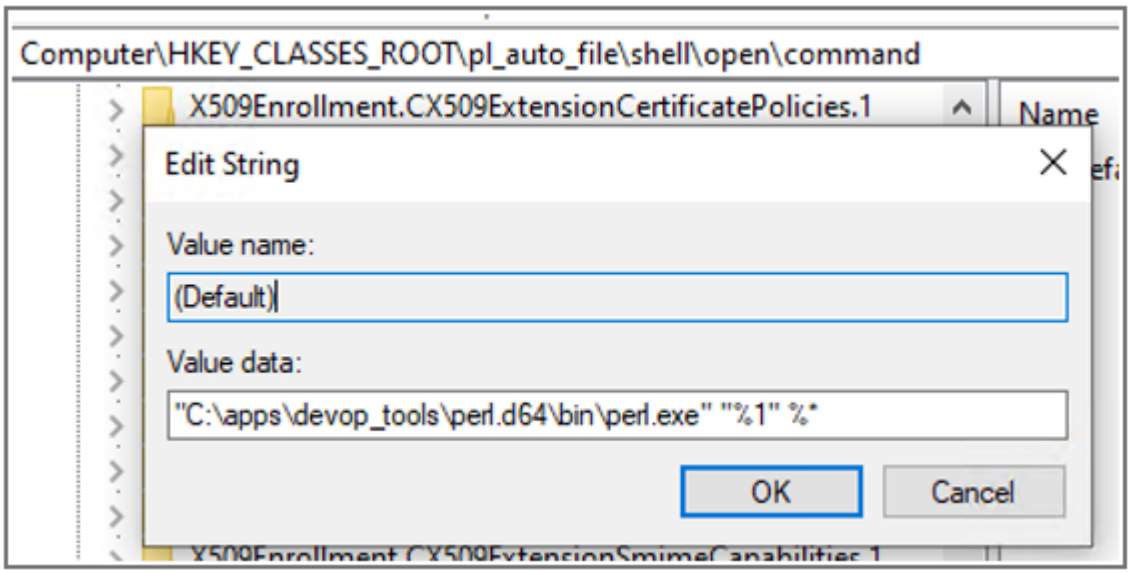
Perl issue while executing the **Extract KPI from Result** tool.

Issue details

A customer launched the **Extract KPI from Result** simulation tool from Active Workspace on **CAEResult** object through TTLC. The **sim_process_launch.log** file showed that the tool launch failed. The launch script output log showed the empty input arguments. The files were exported during tool launch, but the launch script execution failed and status of tool was failed.

Solution or Workaround

1. The launch script in this tool requires Perl installation. For Perl file to run, there is an entry in registry as follows: **HKEY_CLASSES_ROOT\pl_auto_file\shell\open\command**.



   The value of this entry should consist **%*** at the end. Otherwise, the input argument values are shown as empty. Adding **%*** at the end of the value resolves the issue of the empty input parameters.

2. If the above solution does not work, execute the following steps and verify if it resolves the issue:

   a. Open a **Command Prompt** and select **Run as Administrator**.

   b. To set up the correct registry values, type the following commands:

```
assoc .pl=PerlScript
ftype PerlScript=<path of the perl.exe> %1 %*
```

The **Perl.exe** path should be present inside the **PATH** system variable. The Perl installation should contain **XML::Simple** and **Archive::Zip folders**.

For the customer mentioned above, this value was correct and executing the association steps also did not work. The customer had multiple Perl installations on the machine. The Strawberry Perl installation that was used to execute the launch script was not correct as mentioned in the **executable path**.

The customer was asked to remove the Strawberry Perl from his machine and install the correct Perl. After installing ActiveState Perl and setting up the correct registry values as mentioned in **step b** and the **PATH** system variable value, the customer was able to launch the tool successfully.

For more information about setting up Perl to execute the **Extract KPI from Result** tool, see *Extract KPI from Result Tool Integration* PDF in the Teamcenter data (*TC_DATA*) directory.

Related objects are not visible on the Simulation page due to unconfigured relations in the style sheet

Issue

Related objects are not visible on the **Simulation** page due to unconfigured relations in the style sheet.

Issue details

The customer launched the Ansys simulation tool from Active Workspace on the **CAE 3D Model** revision. When the tool launch was completed, the **CAE 3D Analysis** revision got created in Teamcenter. The related **CAE 3D Analysis** was visible in **Where used** tab. Also, the **Simulation** relation between the **CAE 3D Analysis** and **CAE 3D Model** revisions was visible on the **Relations** tab. However, this was not visible in the **Simulation** tab of the **CAE 3D Model** revision.

Solution or Workaround

The related objects tables on the **Simulation** page are governed by style sheets. In the default style sheet, default relations between object types are used to show them in **Related** objects tables. To show the related objects with relationships other than the default relation, then they must be configured in the style sheet. For example, in the above customer scenario, the **Simulation** relation between the **CAE 3D Model** and **CAE 3D Analysis** revision is used, but the default relation between the **CAE 3D Analysis** and **CAE 3D Model** revision is **CAE Defining**. In such cases the traversal paths should be configured in the style sheet.

To configure the traversal path:

1. Log on to Teamcenter as a DBA user.

2. Search for **Cae1CAEModelRevSummaryForAnalystWS** and **Cae1CAEModelRevSummaryForShowObjectLocationForAnalystWS** style sheet datasets.

   These are the stylesheets for **CAE 3D Model** revision in the **Analyst Workspace** of Active Workspace.

3. In the style sheet, search for `<page titleKey="tc_xrt_Simulation" visibleWhen="structure_revisions==null">`. Under this page, you will find a section as follows:

   ```
   <section titleKey="tc_xrt_Analysis">
   <objectSet source="Cae1SimulationSearchProvider.CAEAnalysisRevision"
   sortdirection="ascending"
   sortby="object_string" defaultdisplay="tableDisplay" maxRowCount="5">
   ```

4. Change the section as follows:

```
<section titleKey="tc_xrt_Analysis">
<objectSet source="Cae1SimulationSearchProvider.CAEAnalysisRevision.
(S2P$Simulation)" sortdirection="ascending"
sortby="object_string" defaultdisplay="tableDisplay" maxRowCount="5">
```

Here, **S2P** is secondary to primary, specifying that **CAE 3D Model** revision is the secondary object related to **CAE 3D Analysis** revision. **Simulation** is the name of the relationship they are related to each other. You must use the internal name of the relationship.

5. Save both the style sheets.

6. Log on as a **Analyst** user and verify the **Analysis** table on **Simulation** page of the **CAE 3D Model** revision by selecting it or opening it.

   If the user is not **Analyst**, then search for the default workspace style sheets and perform **step 2** through **step 5**.

   For more information, see Configure the Simulation-related objects table.

TTLC causing Single Sign On (SSO) Error

Issue

Teamcenter Tool Launch Client (TTLC) is causing Single Sign On (SSO) Error. The error says: Rich client login cannot complete because the Security Services Applets are disabled.

Issue details

Customer had done the SSO setup and launched the tool through Active Workspace using TTLC. The SSO login was working for rich client and Active Workspace also. The above mentioned error was faced while logging on through TTLC only when launching the tool.

Solution or Workaround

The **.tcsimxml** file provided by the customer did not have **/sa** at the end of the SSO URL. **/sa** indicates the Security Services Session Agent in the SSO URL.

The Security Services Session Agent loads the required Teamcenter Security Services applets for SSO login through TTLC. As the customer had not installed Security Services Session Agent and since **/sa** was missing in the SSO URL, login was not happening through TTLC. This ultimately caused the issue while launching the tool. Active Workspace login does not require these applets that is why the customer did not have any issue while logging on to Active Workspace using SSO login.

The customer was asked to install Security Services Session Agent and update the SSO URL in the SSO setup by adding **/sa** at the end. After that, the SSO login issue in TTLC was resolved.

For more information, see installing the Session Agent in *Security Services Installation/Customization*.

Error 203454 is displayed while importing analysis dashboards configurations at the site level

Issue

Error 203454 is displayed while importing analysis dashboards configurations at the site level.

Issue details

Customers reported that running the **tcsim_quick_setup** utility throws an error when it tries to import dashboards. It happens only when the customer has manually modified the **tcadmin** person's name in the Organization application before running this utility. Below are the steps given for reproducing this issue.

1. In an environment where the **tcsim_quick_setup** utility is never run, change the **tcadmin** user name in the Organization application from **Tcadmin, testuser** to **Tcadmin, testuser001**, for example.

2. Run the **tcsim_quick_setup** utility. The following error occurs:

```
Error 203454: An error has occurred during the processing of some
objects.
Please view the log file using the "-log" option for more details.
 Error in importing Analysis dashboards configurations at SITE level
```

Solution or Workaround

Before running the **tcsim_quick_setup** utility, make sure that **tcadmin** user name is **Tcadmin, testuser**.

All bootstrap servers are unavailable error during tool launch from Active Workspace

Issue

All bootstrap servers are unavailable error during tool launch from Active Workspace.

Issue details

While launching the tool from Active Workspace with local launch type, customers reported the following error in the TTLC log file. They were unable to launch the tool.

```
[Session for user: 50114 with group: Simulation.TMG.HASETRI and role:
Engineer and Server: http://192.168.207.60:3001/tc/]
[ERROR] 2022-01-07 15:51:45 - InternalServerException: All bootstrap
servers
are unavailable See server syslog file: tcserver.exe28e418c2.syslog
[Session for user: 50114 with group: Simulation.TMG.HASETRI and role:
Engineer and Server: http://192.168.207.60:3001/tc/]
[ERROR] 2022-01-07 15:51:45 - Could not get 'simulation_author' license
which
is required for using 'Teamcenter Tool Launch Client' application.
```

The first error is due to bootstrap server. It this error is displayed, it means the FMS is not configured or running properly on the client machine.

Solution or Workaround

1. Verify whether the **FMS_HOME** variable is set properly.
2. Ensure that the FMS server is up and running.
3. Verify that the value assign to the **Fms_BootStrap_Urls** preference is reachable.

   You can do this by pinging the server, for example, **http://***server_name***:4544/Ping**
4. Ensure that **fcc.xml** inside the **tccs** folder is pointing to the correct FSC address.
5. Make sure that the **Host** file is modified correctly with the machine IP address and name.

   Example:

   IP address full computer name

Simulation tool visibility in Active Workspace

Issue

Simulation tool visibility in **Open in Simulation Tool** panel in Active Workspace.

Issue details

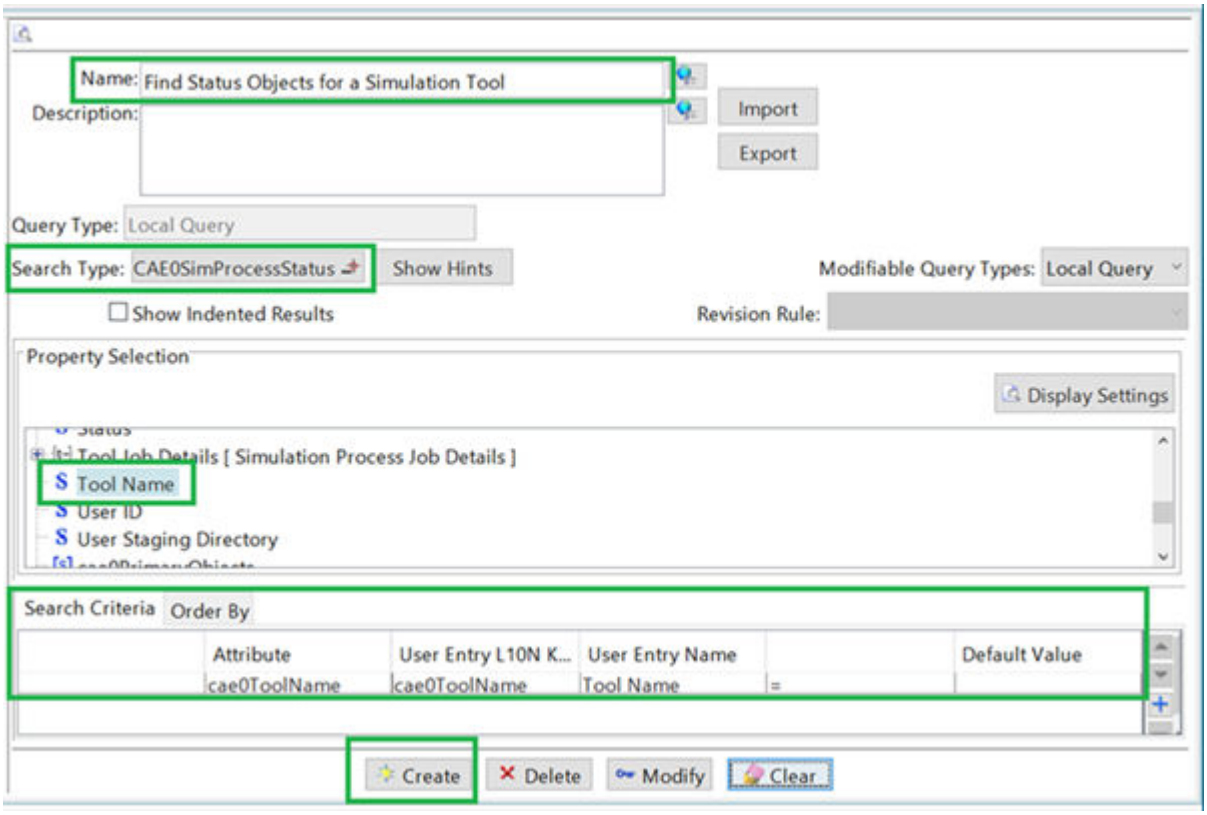
If a user runs the simulation tool on an object and another user wants to run the same simulation tool on the same object while the tool is running, the simulation tool does not appear in the **Open in Simulation Tool** panel.

Solution or Workaround

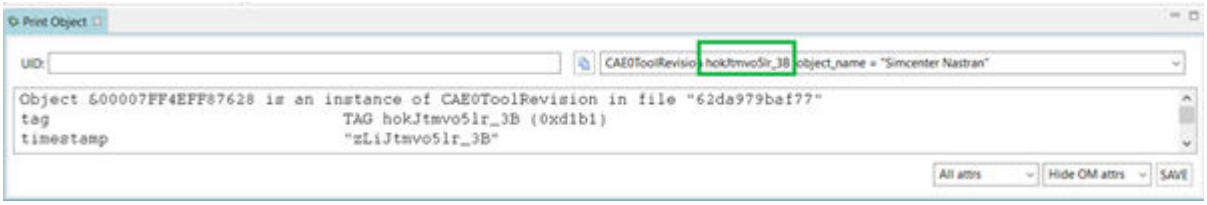
The workaround can be applied by a DBA user. If analyst or designer users face this issue, they should contact the DBA user.

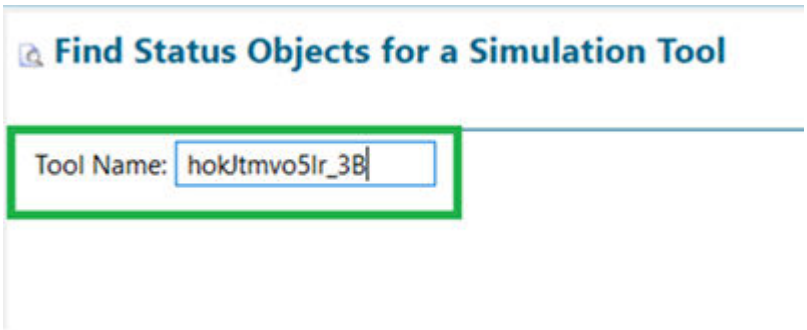
> Note:
>
> Only a DBA user can perform the following procedure.

1. Opens Query Builder application and creates the **Find Status Objects for a Simulation Tool** query.
2. Selects the search type as **CAE0SimProessStatus** for the query.
3. Adds the **Tool Name** search criteria for the query and clicks the **Create** button.



4. Opens CAE Manager application and opens **Simulation Tool Configuration** view.
5. Opens **Print Object** view and selects the simulation tool which needs to be searched.
6. Copies the UID of the simulation tool and pastes it in the query.





7. Searches and deletes the status objects or asks the owning user to complete the job.

   The simulation tool is now visible in the **Open in Simulation Tool** panel.

Teamcenter server launch UNIX access issue

Issue

Teamcenter server launch submitting jobs to the UNIX system results in an access issue and scratch directory not defined error.

Issue details

Teamcenter server running on windows system launches simulation processes on a high-performance cluster (HPC). The use case is as follows:

- Teamcenter runs on a Windows server and HPC runs on a UNIX machine.
- Teamcenter writes files from the database to a scratch directory on a UNIX share. These files are input files, batch files, and log files.
- Teamcenter executes a batch file that creates a simulation job on the HPC system.
- The **TcServer** process needs read/write/execute access for the UNIX share.
- The access to that scratch folder is done in Windows by using the authentication with UNIX credentials. These credentials may be different from the Windows credentials.

Solution or Workaround

1. On the corporate server stop the pool manager, FSC, and the process manager service if these services are running.

2. Open a Teamcenter command prompt as an administrator.

3. Map the UNC shared locations in the command prompt used in the tool launch script.

   Example:

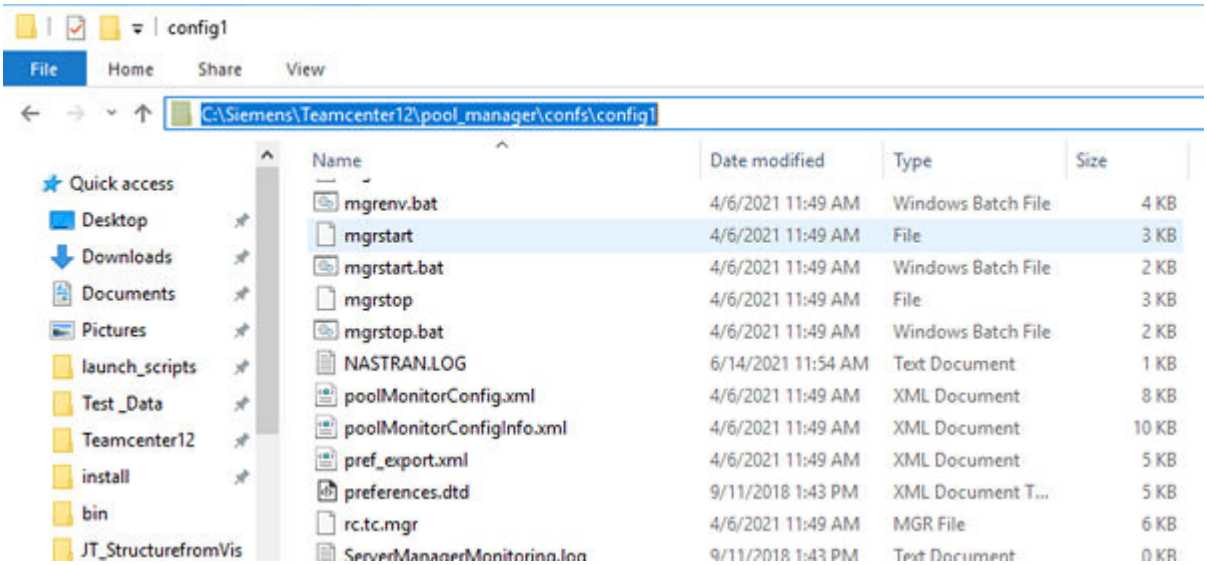
   ```
   C:\Windows\System32>net use \\<path>\work\tmp\tc_temp
   /user: tc-admin-user password=password:   /persistent: yes
   ```

   The mapping should be the same as tool configuration default tool scratch using the same case and characters. The result of this step provides access to any Teamcenter processes launched from this command prompt.

4. Start the pool manager from the same command prompt.

   Example: **CD C:\Siemens\Teamcenter12\pool_manager\confs\config1**

   START mgrstart.bat



5. Start the FSC service and the process manager by using the services tool. This starts Teamcenter.

6. In CAE Manager, use the **Server Launch** option and submit the job to the HPC on the UNIX system.

   This provides access to the Teamcenter server to enable Teamcenter to copy files to the UNC UNIX shared folder location.

Change Summary table does not show entry when revising CAE Revision

Issue

The **Change Summary** table does not show the entry when revising **CAE Revision**.

Issue details

When you use CAE objects as **Impacted Items** with change objects, then while revising the CAE impacted objects, the new revision does not get attached as the solution item.

1. Create the CAE object in Active Workspace.
2. Create a **Change Notice** revision.
   a. On the **HOME** page, click the **Changes** tile.
   b. Choose **More Commands** > **New** > **Create Change** > **Change Notice**.
   c. To create the change notice, specify information as appropriate and click **Create**.
3. Open the change notice, choose the **Affected Items** tab, and add the CAE object you created in the **Impacted Items** section.
4. To make the change notice active, submit the change notice to **ChangeNoticeDefaultWorkflowTemplate** workflow.
5. Open the change notice, choose the **Affected Items** tab, select the CAE object from the **Impacted Items** section, and click **Revise**.

   The new revision of the CAE object is not attached as a solution item to the same change notice revision.

Solution or Workaround

The `<inject type="dataset" src="Cm1ChangeContextProviderForSaveAsAndRevise"/>` was not present in the revised stylesheet of CAE objects.

As an administrator user, you modify the stylesheet to add the following line in the **Cae1CAEAnalysisRevRevise.xml** file:

```
<inject type="dataset" src="Cm1ChangeContextProviderForSaveAsAndRevise"/>
```

1. Log on to Teamcenter as a DBA user.

2. Search for the **Cae1CAEAnalysisRevRevise** style sheet dataset.

3. Add `inject type` as follows in the style sheet:

```
<rendering>
    <page>
        <section>
           <property name="item_revision_id" />
           <property name="object_name" />
           <property name="object_desc" />
           <inject type="dataset"
src="Cm1ChangeContextProviderForSaveAsAndRevise"/>
        </section>
    </page>
</rendering>
```

4. Save the style sheet.

Error while executing a StructureMap rule in CAE Manager

Issue

CAE Manager displays errors while executing a **StructureMap** rule.

Issue details

A customer reported that the structure map execution was causing Teamcenter to crash. However, with the same configuration, the data map was executing successfully.

The simulation development team also tried to execute the structure map with a simple Filter Rule configuration on the same environment and we got the same error.

Environment details

This issue was reported in the following environment:

- Teamcenter version: 12 (12.3.0.9(20210308.00)) 64-bit
- Active Workspace version: Not applicable
- Operating system platform: Windows
- Java version: Not applicable
- Reproducible in a default environment: No

Solution or Workaround

In the customer environment, the **CAE StructureMap** dataset was attached with the **NX_Simulation** relationship to the **CAE StructureMap Revision** business object. This was causing the issue.

Make sure the **CAE StructureMap** dataset is attached with the **Specification** relationship to the **CAE StructureMap Revision** business object.

TEM check fails if CAEItem instance is found in the database while patching from Teamcenter 13.2 release onwards

Issue

TEM check fails if **CAEItem** instance is found in the database while patching from Teamcenter 13.2 release onwards

Issue details

One of the data model object types **CAEItem** is marked as abstract in Teamcenter 13.2 release. As this object is marked as abstract, a check was introduced in TEM in 13.2 release to verify before patching if there are instances of **CAEItem** created in the database. If it finds any instances, the check fails and the user has to delete the instances of **CAEItem** from the database before proceeding with the patching. Ideally, there will not be any instances of **CAEItem** in the customer database and the check should always pass.

Environment details

This issue was reported in the following environment:

- Teamcenter version: 13.2 onwards
- Active Workspace version: Not applicable
- Operating system platform: Windows
- Java version: NA
- Reproducible in a default environment: Yes

Solution or Workaround

The administrator should patch the installer as per the standard process.

For more information, see *Patch Teamcenter Environment Manager* in *Teamcenter Upgrade Using TEM*.