

The Siemens logo is displayed in a white rectangular box in the upper left corner. The word "SIEMENS" is written in a bold, teal, sans-serif font. The background of the entire slide is a low-angle photograph of a modern glass skyscraper reaching towards a blue sky with scattered white clouds.

SIEMENS

Teamcenter MBSE Integration Gateway 4.3 Integration with Git

MBSEIG001 - 4.3

Contents

Overview of Git integration 1-1

An example of the flow of data between Git and Teamcenter 2-1

Setting up Git integration 3-1

Configure additional properties for import 4-1

Importing and exporting model data

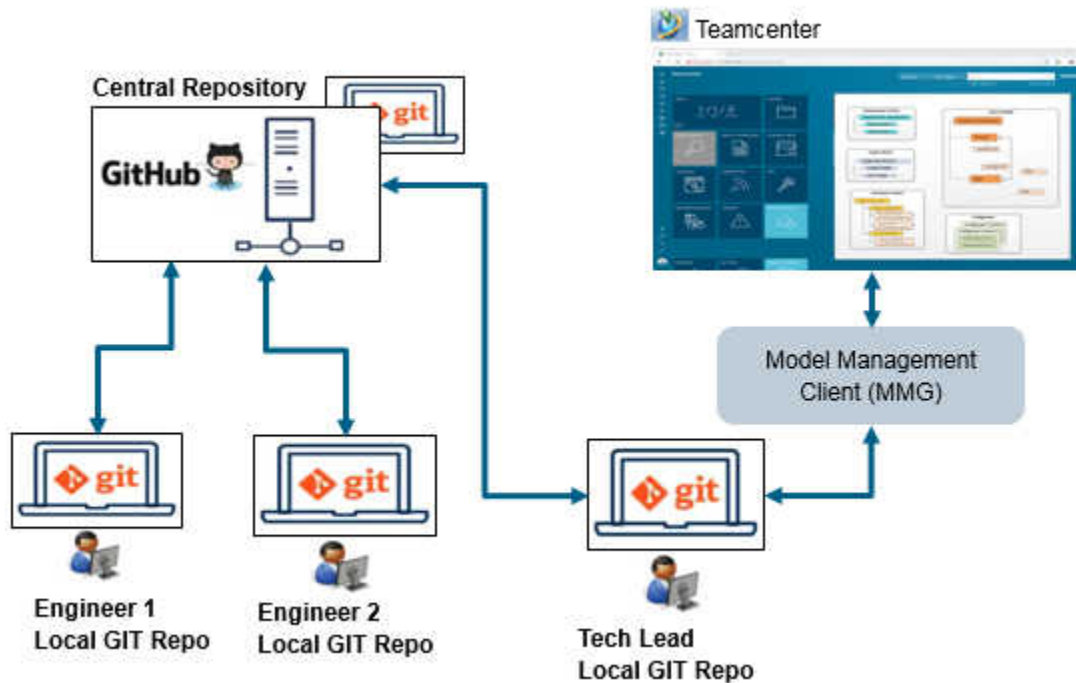
Import Git data into Teamcenter _____ 5-1

Update Git data in Teamcenter _____ 5-3

Compare the baselines of two model collection objects _____ 5-5

Download Git data from Teamcenter _____ 5-6

1. Overview of Git integration



The MBSE Integration Gateway Integration with Git allows modeling engineers and team leads to save the released models from Git to Teamcenter. The daily modeling work is performed and managed in the Git. When the model is ready for release *release* and the team is ready to share with the enterprise, the data is imported to Teamcenter. The data that is imported is the released data (tag) or the golden copy of the data.

The models in Git that are saved to Teamcenter have access to standard Teamcenter services and can be related to the system-of-interest such as system, function, and requirements elements for cross-product traceability. The models in Git can also be used in MBSE processes such as verification and system modeling. The folder structure of models that are imported is replicated in Teamcenter.

Using this, you can:

- Import Git data into Teamcenter.
Gold copies of the models are imported to Git.
- Update Git data in Teamcenter.
- Download Git data from Teamcenter.
Models from one or many systems can be downloaded for co-simulation.

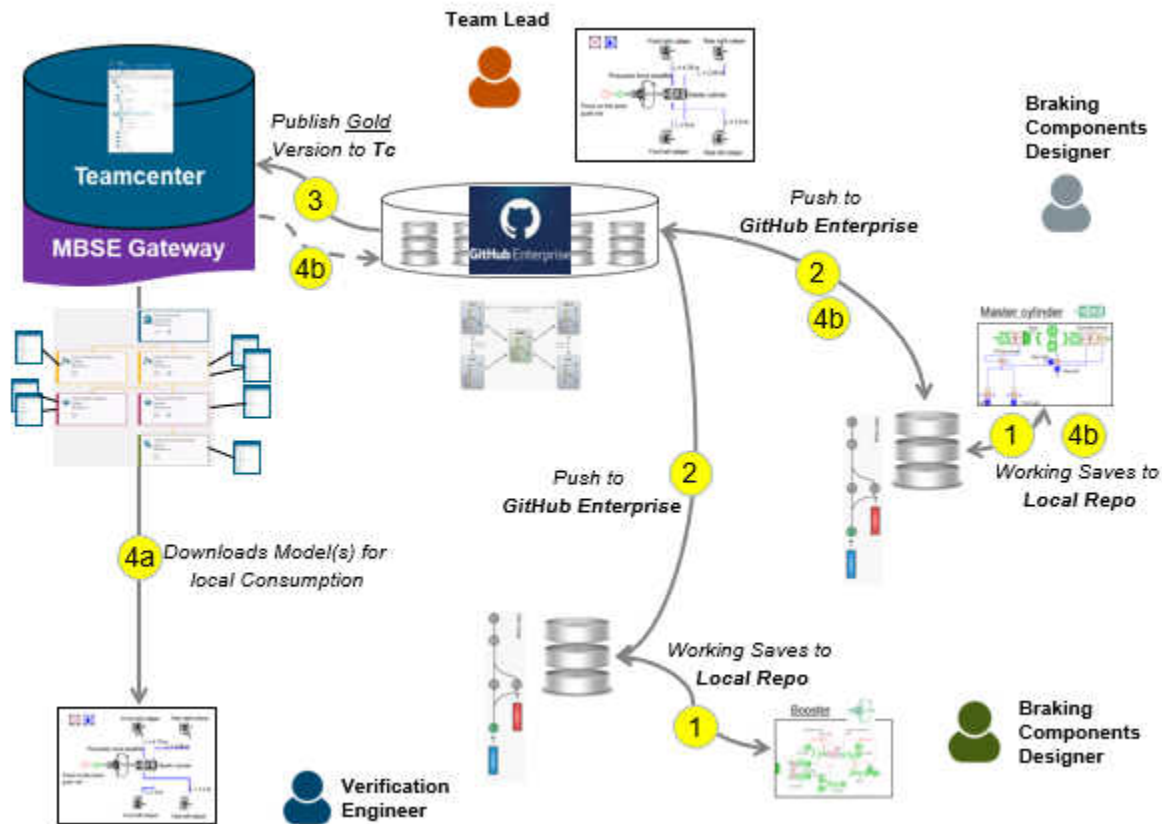
Note that:

- All the models from Git saved in Teamcenter are placed in the **Model Collection** object.
- References between models and model libraries depends on the connector configuration. For example, in the Simulink connector, references are preserved, while in the common connector, the references are not preserved.

The following SCM applications are supported:

- GitHub
- GitHub Enterprise
- GitLab

2. An example of the flow of data between Git and Teamcenter



This topic illustrates the flow of data between GitHub Enterprise and Teamcenter.

1. Modeling engineers author models in Git from their local GitHub repositories.
2. Engineers regularly push their work to the main GitHub Enterprise repository.
3. When the model has is ready for release and the team is ready to share the model with the enterprise, the team lead or the modeling engineer publishes or imports a *gold* version of the model to Teamcenter.
4.
 - a. A verification engineer downloads the model for local consumption (for example, performing a system simulation).
 - b. The modeling engineer downloads the latest model from GitHub Enterprise to the local GitHub repository and contiues with the authoring.

3. Setting up Git integration

To ensure that the integration with Git works correctly, perform the following setup activities:

- **Install the following MBSE Integration Gateway server features:**
 - Gateway for Modeling
 - Support for Concurrent Modeling
- **Install the following MBSE Integration Gateway client feature:**
 - Support for Concurrent Modeling
- If you are using the common-connector based integration, update the *PROJECT_BHMIntegrationDefinition.xml* file. If you are using MATLAB, update the *MATLAB_BHMIntegrationDefinition.xml* file. These files are the named reference in the **PROJECT_BHM_INT_DEF_FILE** dataset.
Check if the XML file contains the following entry:

```
<!-- To specify SCM Tool Connector -->
<SCMToolConnector connectorType="GIT"
connectorClass="com.teamcenter.scmadapter.gitadapter.GitScmAdapter"
cloneInTC="true" toolName="gitlab" baseUrl="http://plgit/gitlab/"
localRepoPath="C:/temp/" />
```

If the entry is not present, add an entry.

Entry name	Description
cloneInTC	If this is set to true , the Git files are also copied along with the metadata. If set to false , only the metadata is copied.
toolName	Specifies the name of the Git tool such as GitHub or GitLab.
baseUrl	Specifies the base URL of the Git tool.
localRepoPath	Specifies the path where the Git files are temporarily downloaded before the files are imported into Teamcenter.

4. Configure additional properties for import

When you import models from Git to Teamcenter, you can import additional properties. On successful import, the properties are associated with the Teamcenter model collection. For the import to work, the following conditions must be met:

- The properties must be defined in the *PROJECT_BHMIntegrationDefinition.xml* file.
- The Teamcenter property that you want to associate with must exist.

To import properties:

1. Create a property file that contains the properties that you want to import. The properties in the property file must have a name-value pair.

Example:

Create a property file named *properties_input.txt*, and add the following property in the name-value format as follows:

```
//name = DESCRIPTION, value=This data was imported from GitHub.  
DESCRIPTION This data was imported from GitHub.
```

2. In the *PROJECT_BHMIntegrationDefinition.xml* file, add the property you defined in the property file, and specify the Teamcenter property it maps to.

Example:

```
<ObjectMapping type="Project" tctype="Fnd0Branch">
  <BHMElement type="RootModel" tctype="Fnd0Branch">
    <AttributeMappings >
      <AttributeMapping name="DESCRIPTION" tcattr="object_desc"/>
    </AttributeMappings >
  </BHMElement>
</ObjectMapping>
```

In this example, the element is **AttributeMapping**, and it has the **name** argument with the value **Description**. The value of the **name** argument must match the property in the property file created in the previous step.

The property, in this example, maps to a Teamcenter property named **object_desc**. Make sure that this Teamcenter property exists because if not, the import action ignores this entry.

3. Check if the additional properties are imported and associated to the model collection by **importing Git data into Teamcenter**.

5. Importing and exporting model data

Import Git data into Teamcenter

1. In Active Workspace, click **New** ⊕ > **Import Model Collection**.

Import Model Collection

Close

Folder/URL: *

http://gitlab/xyz/xyz/modelmanagement/tags/power

Model Collection:

Power Window

Baseline:

Property File(File Import only):

Import

2. In the **Import Model Collection** panel, provide the information for import as follows:

Name	Action to be performed
Folder/URL	Drop the Git tag or paste the URL of the git tag.
Model Collection	Specify the name of the model collection.

Name	Action to be performed
Baseline	Specify the name of the baseline.
Property File(File Import only)	Specify the path to the properties file. This file contains the definition for the additional properties that you want to import.

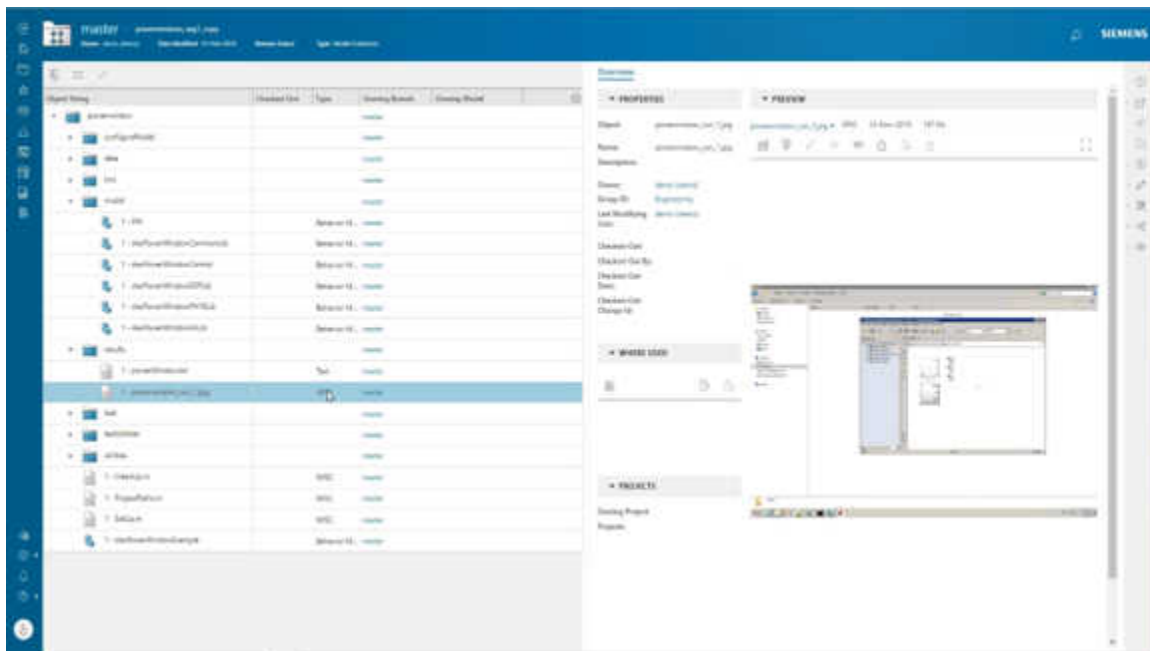
- Click **Import**.

A MIME type file is downloaded in your browser. Depending on how your browser is configured, this file may automatically launch the **Model Management** client.


- Enter your Git **User ID** and **Password** or the **Personal access token** (generated using the Git tool) in the **Model Management** client and click **Login**.

The **Model Management** client displays the status of the import. Once the import is complete, close the client and refresh your browser.

The Git files are imported to your **Home** folder and organized under a **Model Collection** type object.



Update Git data in Teamcenter

1. In Active Workspace, open a model collection that you want to update, and click **Share**  > **Update Model Collection**.

Update Model Collection ✕ Close

Folder/URL: *

/gitlab//modelmanagement/tags/powerwindow_v3

Baseline:

Update

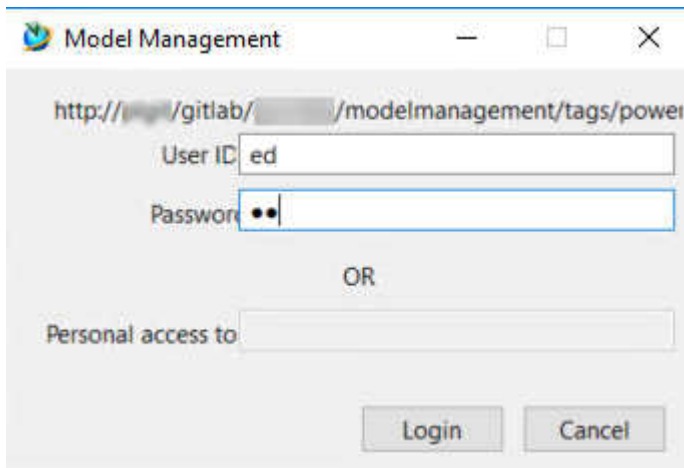
- In the **Update Model Collection** panel, provide information for the import as follows:

Name	Action to be performed
Folder/URL	Drop the Git tag or paste the URL of the git tag.
Baseline	Specify a name for the baseline.

- Click **Update**.

A MIME type file is downloaded in your browser. Depending on how your browser is configured, this file may automatically launch the **Model Management** client.

- Enter your Git **User ID** and **Password** or the **Personal access token** in the **Model Management** client and click **Login**.



The **Model Management** client displays the status of the update. Once the update is complete, close the client and refresh your browser.

After updating, you can compare the metadata of the two different baselines by performing a **model collection compare**.

Compare the baselines of two model collection objects

You can compare the metadata of two **Model Collection** objects.

1. Choose **General** from the **Advanced Search** panel and from the **Type** list, choose **Model Collection**. In the **Name** box, enter the name of the **Model Collection** object.

Click **Search**.

2. From the search results, select two **Model Collection** objects, and click **Manage** > **Compare Model Collection**.

The **Branch Content Comparison** page shows the result of the comparison of the two **Model Collection** objects. The colored markers against the names indicate the comparison results:

- Yellow: The contents of the **Model Collection** objects are different.
- Blue: The contents of the **Model Collection** object on the right are different.
- Red: The contents of the **Model Collection** object on the left are different.

Branch Content Comparison



K1

 Owner:
 Date Modified: 25-Nov-2019 15:52


K2

 Owner:
 Date Modified: 25-Nov-2019 15:54

Name	Result	Left Object	Right Object	Left State
testModelCollection	Different	testModelCollection	testModelCollection	
Sample	Left Only	1 - Sample		Published
Sample	Right Only		2 - Sample	
SingleModel	Right Only		1 - SingleModel	
SingleModel	Left Only	1 - SingleModel		Published
test1.txt	Right Only		1 - test1.txt	
test1.txt	Left Only	1 - test1.txt		Published
test2.txt	Right Only		2 - test2.txt	
test2.txt	Left Only	1 - test2.txt		Published

Download Git data from Teamcenter

1. In Active Workspace, open a model collection that you want to download and click **Share** > **Download Model Collection**.

A MIME type file is downloaded in your browser. Depending on how your browser is configured, this file may automatically launch the download using the **Model Management** client.

The model collection is downloaded in the staging directory.

Siemens Industry Software

Headquarters

Granite Park One
5800 Granite Parkway
Suite 600
Plano, TX 75024
USA
+1 972 987 3000

Americas

Granite Park One
5800 Granite Parkway
Suite 600
Plano, TX 75024
USA
+1 314 264 8499

Europe

Stephenson House
Sir William Siemens Square
Frimley, Camberley
Surrey, GU16 8QD
+44 (0) 1276 413200

Asia-Pacific

Suites 4301-4302, 43/F
AIA Kowloon Tower, Landmark East
100 How Ming Street
Kwun Tong, Kowloon
Hong Kong
+852 2230 3308

About Siemens PLM Software

Siemens PLM Software is a leading global provider of product lifecycle management (PLM) software and services with 7 million licensed seats and 71,000 customers worldwide. Headquartered in Plano, Texas, Siemens PLM Software works collaboratively with companies to deliver open solutions that help them turn more ideas into successful products. For more information on Siemens PLM Software products and services, visit www.siemens.com/plm.

© 2020 Siemens. Siemens, the Siemens logo and SIMATIC IT are registered trademarks of Siemens AG. Camstar, D-Cubed, Femap, Fibersim, Geolus, I-deas, JT, NX, Omneo, Parasolid, Solid Edge, Syncrofit, Teamcenter and Tecnomatix are trademarks or registered trademarks of Siemens Industry Software Inc. or its subsidiaries in the United States and in other countries. All other trademarks, registered trademarks or service marks belong to their respective holders.