Teamcenter Connector for Mendix 2.1
Developer’s Guide
# Contents

- Getting started with Teamcenter Connector ................................................................. 1
- Introduction .................................................................................................................. 1
- Prerequisites .................................................................................................................. 1
- Download Teamcenter Connector, the sample application, and dependencies .......... 1
- Configure to connect to Teamcenter ............................................................................. 2
- Configure to connect to Teamcenter in production mode ......................................... 5
- Understanding the Teamcenter Connector Domain Model ......................................... 9
- Using Teamcenter services through Teamcenter Connector: process workflow ........ 11
- Using Teamcenter services included with Teamcenter Connector ............................... 13
  - Introduction ............................................................................................................. 13
  - Supported services ................................................................................................. 13
  - Process to use Teamcenter services available in the Teamcenter Connector .......... 14
  - Example of using the available Teamcenter services .......................................... 14
  - How to handle errors .............................................................................................. 14
- Extending the Domain Model ...................................................................................... 17
  - Guidelines to extend the Teamcenter Connector Domain Model ............................ 17
  - Example: Extend the domain model to access additional information ................. 17
  - Process to extend the domain model while using available Teamcenter services ......... 21
    - Example: Extend the domain model to call a service to create a Change Notice Revision .............................................................................................................. 22
- Using Teamcenter services not included in Teamcenter Connector ............................ 29
  - Using the Java action ............................................................................................... 29
  - Using the Java Method ............................................................................................ 37
- Teamcenter Connector services ................................................................................... 39
  - Call Teamcenter Service .......................................................................................... 41
  - Create BOM Windows ............................................................................................. 42
  - Create BOM Windows2 ............................................................................................ 43
  - Close BOM Windows ............................................................................................... 44
  - CreateObject ............................................................................................................ 45
  - Create Relation ........................................................................................................ 46
Create Workflow ............................................................................................................. 47
Download File ............................................................................................................... 48
Expand All Levels ......................................................................................................... 49
Expand One Level .......................................................................................................... 50
Find Saved Queries ...................................................................................................... 51
Find User Tasks ............................................................................................................ 52
Get All Tasks ................................................................................................................ 53
Get Dataset File Types ................................................................................................. 54
Get Dataset Types ........................................................................................................ 55
Get Properties ............................................................................................................... 56
Get Revision Rules ........................................................................................................ 57
Get Teamcenter Session Information ............................................................................ 58
Get Workflow Templates ............................................................................................... 59
Get Variant Rules ......................................................................................................... 60
Login... ......................................................................................................................... 61
Logout ............................................................................................................................ 64
Perform Action .............................................................................................................. 65
Perform General Search ............................................................................................... 66
Perform Item Simple Search ........................................................................................ 67
Perform Search ............................................................................................................. 68
Revise Object ................................................................................................................ 69
Saved Query Search ..................................................................................................... 70
Update Properties ......................................................................................................... 71
Upload File .................................................................................................................... 72
Getting started with Teamcenter Connector

Introduction

Teamcenter is a virtual gateway to your company’s product information, connecting all who need to collaborate with product and process knowledge. Teamcenter enables you to digitally manage your product and manufacturing data in the context of the product life cycle.

Teamcenter Connector for Mendix enables Mendix developers to access product data from Teamcenter or create and modify product data in Teamcenter.

This documentation provides guidance on using Teamcenter Connector for Mendix. It assumes that you are familiar with Mendix concepts, processes, and terminology for application development.

Prerequisites

To use Teamcenter Connector for Mendix, you need the following:

- Mendix Business Modeler 8.5.0
- Appropriate Teamcenter licenses
- A running and accessible Teamcenter instance. The minimum Teamcenter version supported is version 10.1.6.

Download Teamcenter Connector, the sample application, and dependencies

Teamcenter Connector for Mendix is available on the Mendix App Store. To add the connector to your project:

1. Open your Mendix Modeler project.
2. Click the AppStore icon on the menu bar to access the Mendix App Store.
3. Search or navigate to the Teamcenter Mendix Connector and click the link.
4. On the Teamcenter Connector for Mendix page, click Download.
   In the Import Module dialog box, click Import.
The connector is imported in your project in the **App Store Modules** folder. Similarly, download the following applications from the Mendix App Store:

- (Optional) Siemens PLM Software UI Resources. This application should be in the project that contains the Teamcenter Connector.
- (Optional) Teamcenter Sample Application. The Sample Application contains sample microflows. You *do not* require to download the Sample Application in the same project that also contains the Teamcenter Connector.

**Configure to connect to Teamcenter**

To help administrators configure the connection to Teamcenter, the Teamcenter Connector contains configuration microflows and pages. To configure the connection to Teamcenter, the Mendix administrator must do the following:

1. In the navigation, update the Default home page and the Home menu item to connect to the AdminLogin microflow.

   ![Configuration screenshot]

   1. **Default home page**: TcConnector.AdminLogin
   2. **Home menu item**: Call microflow 'TcConnector.AdminLogin'

2. Run your project.
The browser displays the AdminHomePage.

3. Click the **TEAMCENTER CONFIGURATIONS** tile in the web browser.
4. Click **New** in the Teamcenter Environment Configuration page.
5. In the Add Teamcenter Configuration dialog box, specify the required fields and click **Save**.
6. If you have enabled SSO:
   a. Ensure that you add the **Add RegisterRequestHandlers** microflow to your startup microflow.
   b. Ensure that you update the SSO configuration information such as **SSO Login Server URL**, **SSO Identity Server URL**, **Teamcenter Application ID**, **Mendix Application ID**.
### Add Teamcenter Configuration

**Configuration Name:**
- MyConfig

**Teamcenter Host Address:**
- http://pnl6w2720:7001/tc

**Teamcenter FMS URL:**
- http://<HOSTNAME>:<FMSPORTNAME>

**SSO Enabled**
- Yes
- No

**SSO Login Server URL**
- http://<HOSTNAME>:<PORTNAME>/login_service/weblogin/login_redirect

**SSO Identity Server URL**
- http://<HOSTNAME>:<PORTNAME>/identity_service

**Teamcenter Application ID:**
- 

**Mendix Application ID:**
- 

**Active**
- Yes
- No

[Save] [Cancel]
7. Once you have updated the Teamcenter configuration information, log on to Teamcenter using the TEAMCENTER LOGIN icon.

If the previous steps, the configuration shows the Home page. If you want to customize the page that you finally see, create a new microflow that contains the ExecuteAdminLogin service. Update your Default home page and the Home menu item to connect to the microflow you created.

The Teamcenter Sample Application has a microflow called MyAdminLogin that uses the ExecuteAdminLogin service. Refer to that microflow for help.

Configure to connect to Teamcenter in production mode

You can use the AdminLogin and the UserLogin microflows to configure the security of the Teamcenter Connector when the security level of your Mendix project is set to Production. Configure the security as follows:
1. In navigation, update the **Role-based home pages** to connect to **AdminLogin** and **UserLogin** microflows based on the role.

![Role-based home pages](image)

2. In navigation, update the Home menu items to include the **AdminLogin** and **UserLogin** microflows.

![Home menu items](image)

3. In the security page of your project, add the respective administrator and user roles in the **User roles** tab.
4. Run your project, and ensure that administrators and users can view their configured login pages.

In the previous steps, the configuration shows the Home page. If you want to customize the page that you finally see, duplicate the AdminLogin and the UserLogin microflows.

In the security page of Teamcenter Connector, select the appropriate permissions for the new microflows you created.
Update your Default home page and the Home menu item to connect to the microflow you created.
Understanding the Teamcenter Connector Domain Model

The Domain Model is a data model that describes the information in your application domain in an abstract way. It is central to the architecture of your application. The Domain Model consists of entities and their relations that are represented by associations.

The Teamcenter Connector for Mendix Domain Model represents Teamcenter business object types and their properties.

You can view the Teamcenter Connector Domain Model by navigating to TcConnector→Domain Model from the Project Explorer.

To export the Domain Model documentation, right-click the Project Explorer and choose Export documentation. The Domain Model documentation is exported as an HTML file.
Using Teamcenter services through Teamcenter Connector: process workflow
Using Teamcenter services included with Teamcenter Connector

Introduction

Teamcenter services are provided through Java actions. You can see the available services in the Teamcenter section of the Toolbox.

Supported services

For information about the supported Teamcenter services, see Teamcenter Connector services.
Process to use Teamcenter services available in the Teamcenter Connector

1. Design a microflow as per your business logic.
2. Drag a service from the Teamcenter category of the toolbox into the microflow.
3. Specify input parameters for the Teamcenter service.
4. Specify how data is retrieved.
5. Test your application.

Example of using the available Teamcenter services

The Sample Application has microflows that use the available services such as CreateItem and Search. Download the Sample Application from the Mendix App Store and import it into your project.

How to handle errors

It is a good practice to set up error handling on all your Teamcenter services in a microflow. Use the HandleServiceErrors microflow to handle errors. To handle errors:

1. In your microflow, right-click your Teamcenter service and choose Set error handling.
2. In the Error handling dialog box, choose the error handling component.

Error handling

<table>
<thead>
<tr>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Custom without rollback</td>
</tr>
<tr>
<td>Rollback</td>
</tr>
<tr>
<td>Custom with rollback</td>
</tr>
<tr>
<td>Custom without rollback</td>
</tr>
<tr>
<td>Continue</td>
</tr>
</tbody>
</table>

3. Drag the Microflow call service from the Toolbox to your microflow.
   a. Update the Action section of the microflow and select the TcConnector.HandleServiceErrors microflow.
b. Update the **ServiceException** parameter and specify the type of error.

![Call Microflow](image)

4. Connect the **Microflow call** service with the service from where the error will originate.

![Diagram](image)
5. Right-click the anchor point of the Teamcenter service and choose **Set as error handler**.

6. Specify an end event for your **Handle Service Errors** service.
Extending the Domain Model

Guidelines to extend the Teamcenter Connector Domain Model

The Domain Model in Mendix consists of entities, associations, and annotations. It is analogous to the Teamcenter data model. When you want to extend the Domain Model, ensure that:

- You extend the Domain Model in a separate module and not the Teamcenter Connector Domain Model.
- The entities and associations must match the corresponding Teamcenter object type names and their properties. You can find Teamcenter object types and properties in the Teamcenter Developer Documentation or in the Business Modeler IDE application in Teamcenter.

Example: Extend the domain model to access additional information

To access other object types or properties which are not already defined, you must add their definition to your app's Domain Model. This process is the same regardless of whether the object or property is OOTB or custom.

The Teamcenter BMIDE view of a custom business object is as follows. It is a child of the ItemRevision object and contains two new properties.

The Mendix Domain Model entity that you must create to retrieve these properties is as follows.
I. Examine the Teamcenter object

Examine the Teamcenter object and determine which properties you want to retrieve.

a. This object has **ItemRevision** as its parent.

<table>
<thead>
<tr>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project: c9x</td>
</tr>
<tr>
<td>Name: C9MyCustomItemRevision</td>
</tr>
<tr>
<td>Display Name: My Custom Item Revision</td>
</tr>
<tr>
<td>Storage Class: C9MyCustomItemRevision</td>
</tr>
<tr>
<td>Parent: ItemRevision</td>
</tr>
<tr>
<td>Item: C9MyCustomItem</td>
</tr>
<tr>
<td>Form: C9MyCustomItemRevisionMaster</td>
</tr>
<tr>
<td>Icon: Default</td>
</tr>
</tbody>
</table>

b. It defines two properties that you want to retrieve.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Storage T...</th>
</tr>
</thead>
<tbody>
<tr>
<td>c9MyCustomString</td>
<td>Attribute</td>
<td>String[128]</td>
</tr>
<tr>
<td>c9MyCustomBoolean</td>
<td>Attribute</td>
<td>Boolean</td>
</tr>
</tbody>
</table>
II. Create a new domain entity

1. In your module's Domain Model, create a new entity.

2. Change the entity's name to match the Teamcenter object type name.

3. Define the new entity's generalization to match the Teamcenter parent object type.
III. Define new attributes to match the Teamcenter properties

1. Match the Teamcenter property name\(^1\), attribute type\(^2\), and other parameters\(^3\) to create the Mendix attribute.
   
   - Teamcenter
     
     ![Teamcenter Property Definition](image)
     
   - Mendix
     
     ![Mendix Attribute Editor](image)
IV. Process complete
You can now use your new Mendix domain entity and its attributes.

Repeat this process for each object whose properties you want to work with.

Process to extend the domain model while using available Teamcenter services

- Create the required entities in a separate module as a subtype of an existing entity (specialized entity).
- For example, if you want to create an entity for the ChangeNoticeRevision business object, you can create it as a subtype of the ItemRevision entity. In this case, the ChangeNoticeRevision entity is a specialized entity based on the ItemRevision entity.
- If the required entity is available but the required properties are not available, create a subtype of the entity in a new module and then add the required properties to the new entity.
  Caution
  Siemens recommends that you always create new entities and not make any changes to the entities that come with Teamcenter Connector.
- Design a microflow as per your business logic.
- Drag a service from the Teamcenter section of the toolbox into the microflow.
- Specify inputs for the Teamcenter service.
- Instantiate and specify input parameters to the Teamcenter service.
- Specify business object mapping to the Teamcenter service.
- Specify how data is retrieved.
- Test your application.
Example: Extend the domain model to call a service to create a Change Notice Revision

1. Create a module that represents the Change management domain. Skip this step if the module exists.

2. Define the Domain Model.
   Create ChangeNotice and ChangeNoticeRevision entities based on the TcConnector.Item and TcConnector.ItemRevision entities. Skip this step if the entities exist. Ensure that the names and properties of the defined entity match the corresponding Teamcenter business object type name and properties. The Reference properties on Teamcenter object types must be represented as associations in the Mendix Domain Model.

   Tip: You can find Teamcenter business objects and their properties in the Teamcenter Developer Documentation.

3. Plan your microflow and decide what services you want to use. The Teamcenter services are available in the microflow Toolbox under the Teamcenter category. For creating a Change Notice Revision, the microflow typically consists of the following activities:

4. Select the appropriate Teamcenter service and drag it to your microflow. For example, you can use the Create Object service under the Teamcenter category to
create a change notice.

Teamcenter Connector for Mendix
Your microflow now appears as follows:

![Diagram of Create Object microflow]

5. In your microflow, specify the input parameters that the Teamcenter service will use. The CreateObject service requires two input parameters:
   - An input variable or entity that contains the information required to create the change notice.
   - The mapping between Teamcenter business object names and Mendix entities. In the following example, the input entity, is the default TcConnector.CreateInput parameter.

![Create Object dialog]

6. In the previous step, an existing input entity served as the input parameter. If you need specific properties to be sent as an input, you must create new input entities or specialize the default input entity and instantiate it. For example, to send properties specific to Change Notice and Change Notice Revision, you must specialize the CreateInput entity as follows:
   - In your Change Management module, create the entities CNCreateInput and CNCompoundCreateInput deriving from the ItemCreateInput and ItemRevisionCompoundCreateInput entities that are available in the
Teamcenter connector.

- Provide input parameters to your microflow. When creating input parameters, ensure that you instantiate it if necessary. Your microflow appears as follows:

```
  CNBCreatelnput  TcConnecor.ItemRevisionCompoundCreateInput
  CNCreateInput  CNCompoundCreateInput

  The input parameter contains the instantiated entities CNBCreatelnput and CNCompoundCreateInput.

  7. You must also specify the **Business object mapping**, which is the mapping between an entity and the corresponding Teamcenter business object types that the microflow is dealing with.

  For example, the Teamcenter business object `ChangeNotice` is represented by the `ChangeManagement.ChangeNotice` entity and `ChangeNoticeRevision`, by the `ChangeManagement.ChangeNoticeRevision` entity. Thus, the mapping can be specified as:

  'ChangeNotice=ChangeManagement.ChangeNotice;ChangeNoticeRevision=ChangeManagement.ChangeNoticeRevision'.
8. Retrieve the objects by dragging the **Retrieve** action to your microflow.

- To retrieve response data use the `$createObjectResponse/ResponseData` association.

- To retrieve partial error data, drag the **Microflow call** action into your microflow and call the use the `TcConnector.ShowPartialErrors` microflow.
- To retrieve created objects use the $ServiceData/Created association.

Your microflow appears as follows:

9. Test your microflow.
Using Teamcenter services not included in Teamcenter Connector

The Teamcenter connector provides two APIs for calling any service operation that is part of Teamcenter Services.

- **CallTeamcenterService — Java action**
  This Java action can be used directly in any microflow and maps the Mendix domain entities to or from the JSON data structures that are used by Teamcenter Services operations.

- **TcConnection.callTeamcenterService — Java method**
  This Java method can only be used from within the Java code and requires that the caller create and parse JSONObjects directly for the call. The TcConnector class also exposes an entity version of the callTeamcenterService method that matches the Java action.

Both APIs have general error handling and take care of the Teamcenter server session management. The Login service operation must be called before calling any other service operation using these APIs. The ShowLoginPage and ExecuteLogin Microflows perform this function.

### Using the Java action

The CallTeamcenterService Java action is the entry point for calling Teamcenter service operations from a microflow. This Java action allows the developer to create a custom mapping between the Teamcenter service operation and the Mendix Domain Model entities. This Java action takes the following arguments:

- **Service Name**: The name of the Teamcenter service operation
- **Input Argument**: The entity containing the input data for the service operation.
- **Response Object**: The entity instance to which the service operation response will be written. This may be any entity type that extends from ServiceResponse.
- **Operation Mapping**: The mapping definition for this service operation.
- **Business Object Mapping**: The mapping of Teamcenter business object names to Mendix entity names.

#### Operation mapping
This maps a Teamcenter service operation request and a response each to the Mendix entities. The mapping is defined as a JSON document, either as a file or provided as an input string to the CallTeamcenterService Java action. The JSON schema for an operation mapping document is as follows:

```json
{
  ServiceOperation: The operation name (Core-2011-06-Session/login).
  InputType: The Entity type to map the input from.
  ResponseType: The Entity type to map the response to.
  ObjectMapping: The business object mappings.
  OperationInput: The template for the service operation input. Must represent the complete service operation input.
  OperationResponse: The template for the service operation response. May contain only the specific elements of the response that are mapped.
}
```

Within the **OperationInput** and **OperationResponse** templates, values are either hard coded or use the $Input or $Response substitution keywords. These substitution keys have the following syntax:

```
$Input[/Association]/Attribute[;Instruction]
$Response[/Association]/Attribute
```

Where:

- **Association** — Optional association name on the given entity type. Multiple associations can be sequenced, each separated by a ' / '.
- **Attribute** — Optional attribute name on the given entity type.
- **Instruction** — Optional instruction to be applied to the substitution. Multiple instructions can be used, each separated with a semicolon. The supported instructions are:
  - AttributeAsArray — single valued JSONArray for each attribute value.
  - DateFormat=Format — Use the custom date format for serializing date attributes.
  - ignoreNull — Use to omit the key if the value of that key is null.

Examples of which attributes that will be mapped:

- **$Input**
  The full entity
- **$Input/TcConnector.itemRev**
  The entity referenced by the TcConnector.itemRev association
- **$Input/TcConnector.user/person**
  The person attribute on the referenced entity (TcConnector.user association)
- **$Input;DateFormat=MM/dd/yyyy**
  The full entity, with any date attributes serialized in the format mm/dd/yyyy

Note: This is a JSON document, so the forward slash ‘ / ’ character in any quoted string must be escaped ‘ \' .
A sample mapping JSON document:

```json
{
    "ServiceOperation": "Cad-2007-01-StructureManagement\createBOMWindows",
    "InputType": "TcConnector.CreateBomWindowInput",
    "ResponseType": "TcConnector.CreateBomWindowResponse",
    "ObjectMapping": "BOMLine=TcConnector.BOMLine",
    "OperationInput": {
        "info": [
            {
                "clientId": "CreateBOMWindows",
                "item": "",
                "itemRev": "$Input\TcConnector.itemRev",
                "bomView": "",
                "objectForConfigure": "",
                "activeAssemblyArrangement": "",
                "revRuleConfigInfo": {
                    "clientId": "",
                    "revRule": "$Input\TcConnector.revRule",
                    "props": {
                        "unitNo": -1,
                        "date": "",
                        "today": true,
                        "endItem": "",
                        "endItemRevision": "",
                        "overrideFolders": [
                            {
                                "ruleEntry": "",
                                "folder": ""
                            }
                        ]
                    }
                }
            }
        ],
        "OperationResponse": {
            "output": [
                {
                    "bomLine": "$Response\TcConnector.createBomWindowResponseBOMLine"
                }
            ]
        }
    }
}
```

**ExpandGRMRelationsForPrimary Example:**

The SOA request and response structure for 'Core-2007-09-DataManagement/expandGRMRelationsForPrimary' are as follows:
To create the operation mapping, refer to the request and response entities and substitute the corresponding entity attributes and associations for the key or the structure. In this example we have created the entities structure as follows:

```
Library: Core
Service: DataManagement
Year: 2007-09
Url: Core-2007-09-DataManagement/expandGRMRelationsForPrimary
Soa Dependency Inclusion: "Teamcenter.Soa.Core_2007_09.DataManagement.expandGRMRelationsForPrimary"
Request:
{
    primaryObjects: "JModelObject[]",
    pref:
    {
        expItemRev: "bool",
        returnRelations: "bool",
        info:
        {
            relationTypeName: "String",
            otherSideObjectTypes: "String[]"
        }
    }
}
Response:
{
    output:
    {
        inputObject: "JModelObject",
        relationshipData:
        {
            relationshipObjects:
            {
                otherSideObject: "JModelObject",
                relation: "JModelObject"
            },
            relationName: "String"
        },
        serviceData: "JServiceData",
    }
}
```
Request Entities:

1. **ExpandGRMPrimaryInput** – Input entity.
2. **Relation** – Represents the info structure in the request.
3. **relations** – Many-to-one association between ExpandGRMPrimaryInput and Relation entities.
4. **primaryObjects** – One-to-one association with list of model objects represents the `primaryObjects` key in request

Response Entities:

1. **ExpandGRMPrimaryResponse** – Response Entity derived from ServiceResponse
3. **relationshipObjects** – many to One Association between ExpandGRMPrimaryResponse and relationshipObject entities.
4. **otherSideObject** – One to One association with ModelObject represents the `otherSideObject` key in the response.

Following is the operation mapping file for expandGRMRelationsForPrimary SOA call:

```json
{
  "ServiceOperation": "Core-2007-09-DataManagement/expandGRMRelationsForPrimary",
  "InputType": "ToConnectorSample.ExpandGRMPrimaryInput",
  "ResponseType": "ToConnectorSample.ExpandGRMPrimaryResponse",
  "ObjectMapping": "",
  "OperationInput": {
    "primaryObjects": [ "$Input\ToConnectorSample.primaryObjects/ToConnector.ListOfModelObjects" ],
    "pref": [ "true", "false" ],
    "returnRelations": [ "true", "false" ],
    "info": [ "$Input\ToConnectorSample.relations" ]
  },
  "OperationResponse": {
    "output": [ ]
  }
}
```

**CreateWorkflow Example:**

The SOA request and response structure for `Workflow-2014-10-Workflow/createWorkflow` are as follows:
To create the operation mapping, refer the request and response entities and substitute the corresponding entity attributes and associations for the key or the structure. In this example we have created the entities structure as follows:

Request Entities:
1. **CreateWorkflowInput** – Input entity.
2. **createWorkflowtRelationTypes** – Entity representing attachmentRelationTypes key in the request.
3. **relationTypes** – One to Many Association between CreateWorkflowInput and createWorkflowRelationTypes entities.

4. **attachments** - One to Many Association between CreateWorkflowInput and ModelObject entities representing attachments key in the request.

5. **assignedUser** - One to Many Association between CreateWorkflowInput and User entities representing assignedUserList key in the request.

6. **responsibleParty** - One to One Association between CreateWorkflowInput and User entities representing responsibleParty key in the request.

7. **workflowOwner** - One to One Association between CreateWorkflowInput and User entities representing workflowOwner key in the request.

8. **processName** – String attribute on CreateWorkflowInput entity representing processName key in request.

9. **processDescription** – String attribute on CreateWorkflowInput entity representing processDescription key in request.

10. **processTemplate** – String attribute on CreateWorkflowInput entity representing processTemplate key in request.

11. **dueDate** – Date and time attribute on CreateWorkflowInput entity representing dueDate key in request.

12. **attachmentRelationTypes** – String attribute on createWorkflowRelationTypes entity representing the attachmentRelationTypes key in request, which is array of strings, so we add the instruction AttributeAsArray after the path to attribute. This will take care to give the input to this key in form of array of strings.

**Response Entities:**

1. **CreateWorkflowResponse** – Response Entity derived from ServiceResponse.

2. **workflowTask** – One to Many association between CreateWorkflowResponse and EPMTask entities representing workflowTask structure of the response.

Following is the operation mapping file for createWorkflow SOA call:
Entity Mapping

Mendix Domain Model entities are mapped to Teamcenter service operation data structures based on naming conventions. The Teamcenter structure element names (keys in JSON document) map one-to-one to the entity member (attributes or associations) names, with the following caveats:

- The entity member name is prefixed with an underscore '_', for example 'type'. In this case the '_' is ignored, thus matching the Teamcenter name of 'type'.
- The entity member name is suffixed with '__XXX', for example, 'phone__Home'. In this case the '__Home' is ignored, thus matching the Teamcenter name of 'phone'.

When traversing entities across associations, the entity names are not used. Only the associations between entities must match or be mappable.

ModelObject Mapping

The ModelObject entity is the Mendix Domain Model representation of a Teamcenter business object. Any entity that extends from ModelObject is considered to be a ModelObject. The mapping of ModelObject entities follows the general mapping of entity mapping (see previous section), with the following additions:

- Member names defined directly on the ModelObject entity (UID, _Type...) are ignored.
- All other entity member names are mapped one-to-one with the business object type property name.
• Entity Attribute types must match the type of the business object property type. Attributes of type String are the display value or localized value of the business object property, while attributes of other types (Boolean, Decimal, and Integer, Localized Date and time), are the database value of the business object property. To map a database value of a business object String property, the attribute name must be suffixed with '__DB' that is, 'description__DB'.

**Business object mapping**

This refers to the mapping of Teamcenter business object type names to Mendix entity names. This mapping is applied to the business objects that are returned from the service operation. The syntax for this mapping is a semicolon-separated list of Teamcenter or Mendix names.

BOMLine=TcConnector.BOMLine;ItemRevision=TcConnector.ItemRevision

For any business object returned from the service operation that is not in this mapping (that is, EngChange_Revision), the nearest mapped parent (ItemRevision) is instantiated.

**Error Handling**

The CallTeamcenterService Java action handles all non-service errors that occur during the processing of the service request. This includes networking errors (HTTP errors connecting to the Teamcenter server), session time-out, and parsing errors. These errors are displayed as an exception, and the calling microflow must set an error handler on the Java action and then create a flow from the Java action that is set as the error handler. This flow must display the $latestError/Message and then exit the microflow.

**Using the Java Method**

The callTeamcenterService Java method (tcconnector.foundation.TcConnection.callTeamcenterService) is the entry point for calling Teamcenter service operations from the Java code. This Java method differs from the CallTeamcenterService Java action in that the input and output data are JSONObject versus Mendix entities. The developer is responsible to creating the JSONObject input that conforms to the Teamcenter service operation definition and parsing the returned JSONObject. This Java method takes 3 arguments:

• **Service Name:** The name of the Teamcenter service operation

• **Input Argument:** The JSONObject containing the input data for the service operation.

• **Policy:** The Object Property Policy defining which values should be returned.

A JSONObject is returned from this method with the contents of the service operation response. All business object references (UIDs) throughout the response structure are replaced with the full ModelObject instance from the ServiceData. JSONObject instances that represent a ModelObject or ServiceData can be cast to JModelObject or JServiceData:
JModelObject bomLine = (JModelObject)output.getJSONObject("bomline");
JServiceData sd = (JServiceData)response.getJSONObject("serviceData");

Both the JModelObject and JServiceData class have methods to conveniently access data on this structure and convert to Mendix entities.

**The JModelObject Class**

The JModelObject class (tcconector.foundation.JModelObject) extends the JSONObject class to represent a single Teamcenter business object (ModelObject). This class has convenience methods to access property values and other elements defined on the ModelObject type. Conversation of a JModelObject to an entity uses ModelObject Mapping.

**The JServiceData Class**

The JServiceData class (tcconector.foundation.JServiceData) extends the JSONObject class to represent the ServiceData structure common to most Teamcenter service operations. This class has convenience methods to access data in the ServiceData. Conversion of JServiceData to an entity uses ModelObject Mapping.

**The JPolicy Class**

The JPolicy class (tcconector.foundation.JPolicy) extends the JSONObject class to represent an Object Property Policy. The Object Property Policy defines what properties should be returned from the service operation call for the given set of business object types. The JPolicy is constructed from Business Object Mapping, which defines a policy that includes all the entity member names.
Teamcenter Connector services

Call Teamcenter Service
Create BOM Windows
Create BOM Windows2
Close BOM Windows
CreateObject
Create Relation
Create Workflow
Download File
Expand All Levels
Expand One Level
Find Saved Queries
Find User Tasks
Get All Tasks
Get Dataset File Types
Get Dataset Types
Get Properties
Get Revision Rules
Get Teamcenter Session Information
Get Workflow Templates
Get Variant Rules
Login
Logout
Perform Action
Perform General Search
Perform Item Simple Search
Perform Search
Revise Object
Saved Query Search
Update Properties
Upload File
Call Teamcenter Service

This service calls any Teamcenter service from a microflow. To call a Teamcenter service, you must create custom mapping between the Teamcenter Service Operation and the Mendix Domain Model Entities.

Teamcenter version
Teamcenter 10.1.5

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ServiceName</td>
<td>String</td>
<td>Fully qualified name of the service operation, for example, Core-2011-06-Session/login.</td>
</tr>
<tr>
<td>InputArgument</td>
<td>Object</td>
<td>The input entity for the service operation.</td>
</tr>
<tr>
<td>ResponseObject</td>
<td>Object</td>
<td>The returned object. If null an instance of ServiceResponse will be created for the return.</td>
</tr>
<tr>
<td>OperationMapping</td>
<td>String</td>
<td>Path for the operation mapping definition. This path is, relative to resources/OperationMappings. Ensure that you use the forward slash (/) as the path separator or you may encounter errors. The InputArgument and Response are mapped to the service operation input and output based on the definition in the mapping file.</td>
</tr>
<tr>
<td>BusinessObjectMappings</td>
<td>String</td>
<td>A semicolon-separated list of Teamcenter business object names and entity names, for example, BOMLine=TcConnector.BOMLine;ItemRevision=TcConnector.ItemRevision.</td>
</tr>
</tbody>
</table>

Returns

A service response of type TcConnector.ServiceResponse. The returned list of model objects can be retrieved using an appropriate association. Partial errors can be retrieved using TcConnector.ResponseData and TcConnector.PartialErrors.
Create BOM Windows

Creates a BOM Window and sets the input item revision as the top line.

SOA URL

Cad-2007-01-StructureManagement/createBOMWindows

Teamcenter version

Teamcenter 2007

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InputData</td>
<td>Object</td>
<td>Input for creating the BOM Window.</td>
</tr>
<tr>
<td>BusinessObjectMappings</td>
<td>String</td>
<td>A semicolon-separated list of Teamcenter business object names and entity names, for example, 'Item=TcConnector.Item;ItemRevision=TcConnector.ItemRevision'</td>
</tr>
</tbody>
</table>

Returns

An entity of type TcConnector. CreateBomWindowResponse. Top line can be retrieved using association TcConnector.bomLine_BOMWindow.

Partial errors can be retrieved using the association TcConnector.PartialErrors.
Create BOM Windows2

Creates a BOM Window and sets the input item revision as the top line. This action takes additional inputs such as variant rule, configuration context, and comWinPropRagMap.

SOA URL
Cad-2013-05-StructureManagement/createBOMWindows2

Teamcenter version
Teamcenter 10.1

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InputData</td>
<td>Object</td>
<td>Input for creating the BOM Window.</td>
</tr>
<tr>
<td>BusinessObjectMappings</td>
<td>String</td>
<td>A semicolon-separated list of Teamcenter business object names and entity names, for example, 'Item=TcConnector.Item;ItemRevision=TcConnector.ItemRevision'</td>
</tr>
</tbody>
</table>

Returns
An entity of type TcConnector. CreateBomWindowResponse. Top line can be retrieved using association TcConnector.bomLine_BOMWindow.
Partial errors can be retrieved using the association TcConnector.ResponseData or TcConnector.PartialErrors.
Close BOM Windows

Closes a BOM window. Use this service to close the BOM Window created with the Create BOM Windows and after calls to Expand All Levels and Expand One Level product structure services are complete.

SOA URL

Cad-2007-01-StructureManagement/closeBOMWindows

Teamcenter version

Teamcenter 2007

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InputData</td>
<td>Object</td>
<td>Input for creating the BOM Window.</td>
</tr>
<tr>
<td>BusinessObjectMappings</td>
<td>String</td>
<td>A semicolon-separated list of Teamcenter business object names and entity names, for example, 'Item=TcConnector.Item;ItemRevision=TcConnector.ItemRevision'</td>
</tr>
</tbody>
</table>

Returns

An entity of type TcConnector.CreateBomWindowResponse. Top line can be retrieved using association TcConnector.bomLine_BOMWindow.

Partial errors can be retrieved using the association TcConnector.ResponseData or TcConnector.PartialErrors.
**CreateObject**

Creates a Teamcenter business object of any type. This action also creates secondary (compound) objects if the CompoundCreateInput for the secondary object is represented in the recursive CompoundCreateInput object. For example, Item is primary object that also creates Item Master and ItemRevision. ItemRevision in turn creates ItemRevision Master. The input for all these levels is passed in through the recursive CompoundCreateInput object.

**SOA URL**

Core-2015-07-DataManagement/createRelateAndSubmitObjects2

**Teamcenter version**

Teamcenter 11.2

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InputEntity</td>
<td>Object</td>
<td>Input for closing BOM window.</td>
</tr>
<tr>
<td>BusinessObjectMapping</td>
<td>String</td>
<td>A semicolon-separated list of Teamcenter business object names and entity names, for example, 'Item=TcConnector.Item;ItemRevision=TcConnector.ItemRevision'</td>
</tr>
</tbody>
</table>

**Returns**

An object of type TcConnector.ServiceResponse.

Created objects can be retrieved using the associations TcConnector.ServiceResponse, TcConnector.ResponseData. Partial errors can be retrieved using TcConnector.ResponseData and TcConnector.PartialErrors.
Create Relation

Creates the specified relation between the input primary and secondary objects. If the primary object has a relation property with the specified relation name, then the secondary object is associated with the primary object through the relation property.
If the relation has mandatory properties, ensure that they are added to the entities. The Teamcenter Connector does not enforce the use of mandatory properties.

SOA URL
Core-2006-03-DataManagement/createRelations

Teamcenter version
Teamcenter Engineering 2005 SR1

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InputData</td>
<td>Object</td>
<td>Input argument for invoking the createRelations service.</td>
</tr>
<tr>
<td>BusinessObjectMappings</td>
<td>String</td>
<td>A semicolon-separated list of Teamcenter business object names and entity names, for example, ImanRelation=TcConnector.ImanRelation.</td>
</tr>
</tbody>
</table>

Returns
An entity of type TcConnector.CreateRelationResponse that contains the created relations.
Partial errors can be retrieved using the association TcConnector.PartialErrors.
Create Workflow

Creates a Teamcenter Workflow object.

**SOA URL**

Workflow-2014-10-Workflow/createWorkflow

**Teamcenter version**

Teamcenter 11.1

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InputData</td>
<td>Object</td>
<td>Input argument for creating a Teamcenter workflow object.</td>
</tr>
<tr>
<td>BusinessObjectMappings</td>
<td>String</td>
<td>A semicolon-separated list of Teamcenter business object names and Entity names, for example, ImanRelation=TcConnector.ImanRelation.</td>
</tr>
</tbody>
</table>

**Returns**

An entity of type TcConnector.CreateWorkflowResponse that contains the created workflow objects.

Partial errors can be retrieved using the association TcConnector.PartialErrors.
Download File

Downloads all the files associated with the dataset that is passed as an input parameter.

SOA URL

Core-2006-03-DataManagement/getProperties
Core-2006-03-FileManagement/getFileReadTickets

Teamcenter version

Teamcenter Engineering 2005 SR1

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DatasetParameter</td>
<td>Object</td>
<td>A dataset object that has the UID of the Teamcenter dataset whose files are to be downloaded.</td>
</tr>
</tbody>
</table>

Returns

A Boolean type that returns the values True or False to represent the success or failure of the download, respectively.

The Dataset.Documents association of the input object is then updated to point to any downloaded files.
Expand All Levels

Expands the children at all levels of the relevant parent BOMLine.

SOA URL
Cad-2007-01-StructureManagement/expandPSAllLevels

Teamcenter version
Teamcenter 2007

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inputEntity</td>
<td>Object</td>
<td>The input for the BOM Line that is being expanded.</td>
</tr>
<tr>
<td>businessObjectMapping</td>
<td>String</td>
<td>A semicolon-separated list of Teamcenter business object names and entity names, for example, 'Item=TcConnector.Item;ItemRevision=TcConnector.ItemRevision'</td>
</tr>
</tbody>
</table>

Returns

An entity of type TcConnector. ExpandPSAllLevelsResponse.

Expanded child BOMLines with respective parents can be retrieved using the association TcConnector.ExpandPSAllLevelsOutput. Partial errors can be retrieved using the associations TcConnector.ResponseData and TcConnector.PartialErrors.
Expand One Level

Expands the first level children of the relevant parent BOMLine.

SOA URL

Cad-2007-01-StructureManagement/expandPSOneLevel

Teamcenter version

Teamcenter 2007

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inputEntity</td>
<td>Object</td>
<td>The input for the BOM Line that is being expanded one level.</td>
</tr>
<tr>
<td>businessObjectMapping</td>
<td>String</td>
<td>A semicolon-separated list of Teamcenter business object names and entity names, for example, 'Item=TcConnector.Item;ItemRevision=TcConnector.ItemRevision'</td>
</tr>
</tbody>
</table>

Returns

An entity of type TcConnector.ExpandPSOneLevelResponse.

Expanded child BOMLines can be retrieved using the association TcConnector.expandPSOneLevelResponseBOMLines. Partial errors can be retrieved using the associations TcConnector.ResponseData and TcConnector.PartialErrors.
Find Saved Queries

Sends a request to the data provider and receives the response. It routes the search request to a specific provider specified as providerName in the searchInput, assuming the searchCriteria for the provider is specified in the searchCriteriaInput object. For example, Awp0SavedQuerySearchProvider is the provider that is used for general search. The input criteria for GeneralSearch is passed through the searchCriteriaInput object.

SOA URL
Query-2010-04-SavedQuery/findSavedQueries

Teamcenter version
Teamcenter 8.2

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InputData</td>
<td>Object</td>
<td>This is a type of SearchInput entity that represents the information required to search for the business object.</td>
</tr>
<tr>
<td>BusinessObject Mappings</td>
<td>String</td>
<td>A semicolon-separated list of Teamcenter business object names and entity names, for example, Item=TcConnector.Item;ItemRevision=TcConnector.ItemRevision;WorkspaceObject=TcConnector.WorkspaceObject.</td>
</tr>
</tbody>
</table>

Returns
An object of type TcConnector.FindSavedQueryResponse.
**Find User Tasks**

Finds the Task inbox of the logged-on user. The Task inbox contains **Tasks to Perform** and **Tasks to Track**.

**SOA URL**

Core-2007-01-Session/getTCSessionInfo

Core-2006-03-DataManagement/getProperties

**Teamcenter version**

Teamcenter 2007

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InputData</td>
<td>Object</td>
<td>This is a type of SearchInput entity that represents the information required to search for the business object.</td>
</tr>
<tr>
<td>BusinessObject Mappings</td>
<td>String</td>
<td>A semicolon-separated list of Teamcenter business object names and entity names, for example, Item=TcConnector.Item;ItemRevision=TcConnector.ItemRevision;WorkspaceObject=TcConnector.WorkspaceObject.</td>
</tr>
</tbody>
</table>

**Returns**

Task Inboxes can be retrieved using the association TcConnector.TaskInbox_FindUsersTasksResponse. The tasks_to_perform and tasks_to_track entities are the associations between EPMTask and TaskInbox. Partial errors can be retrieved using the association TcConnector.ResponseData or TcConnector.PartialErrors.
Get All Tasks

Returns a list of workflow tasks for the specified Teamcenter business object type.

SOA URL

Workflow-2008-06-Workflow/getAllTasks

Teamcenter version

Teamcenter 8.2

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InputData</td>
<td>Object</td>
<td>Input argument for getting the workflow tasks.</td>
</tr>
<tr>
<td>BusinessObjectMappings</td>
<td>String</td>
<td>A semicolon-separated list of Teamcenter business object names and entity names, for example, ImanRelation=TcConnector.ImanRelation.</td>
</tr>
</tbody>
</table>

Returns

An entity of type TcConnector. GetAllTasksResponse that contains the workflow tasks.
**Get Dataset File Types**

Returns a list of named references applicable for the given dataset type. This named reference is required when creating a dataset.

**SOA URL**

Core-2007-06-DataManagement/getDatasetTypeInfo

**Teamcenter version**

Teamcenter 8.2

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dataset_type</td>
<td>String</td>
<td>The dataset type for which a list of named references must be returned.</td>
</tr>
</tbody>
</table>

**Returns**

List of type TcConnector.Pair. This is a list of valid named references for the input dataset type.
Get Dataset Types

Returns a list of the available dataset types. This dataset type is required while creating a dataset.

SOA URL

Core-2010-04-DataManagement/getAvailableTypesWithDisplayNames

Teamcenter version

Teamcenter 8.2

Parameters

None.

Returns

List of type TcConnector.Pair. This is a list of valid Teamcenter dataset types.
Get Properties

Receives properties of the specified model objects. The input entities will be updated with new property values.

SOA URL

Core-2006-03-DataManagement/getProperties

Teamcenter version

Teamcenter Engineering 2005 SR1

Parameters

| Name               | Type    | Description                                                                 |
|--------------------|---------|                                                                            |
| ListOfModelObjects | Object  | List of Mendix objects whose properties are to be fetched. All the properties available on the input object are retrieved. |
| BusinessMappings   | String  | A semicolon-separated list of Teamcenter business object names and entity names, for example, 'Item=TcConnector.Item;ItemRevision=TcConnector.ItemRevision'|

Returns

An object of type TcConnector.ServiceResponse.
Get Revision Rules

Gets all the persistent revision rules in the database.

SOA URL

Cad-2007-01-StructureManagement/getRevisionRules

Teamcenter version

Teamcenter 2007

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BusinessObjectMappings</td>
<td>String</td>
<td>A semicolon-separated list of Teamcenter business object names and entity names, for example, 'Item=TcConnector.Item;ItemRevision=TcConnector.ItemRevision'</td>
</tr>
</tbody>
</table>

Returns

The type GetRevisionRulesResponse, which contains RevisionRuleInfo. RevisionRuleInfo in turn contains the revision rule. Partial errors can be retrieved using the association TcConnector.ResponseData or TcConnector.PartialErrors.
Get Teamcenter Session Information

Retrieves information about the Teamcenter server session.

SOA URL
Core-2007-01-Session/getTCSessionInfo

Teamcenter version
Teamcenter 2007

Parameters
None

Returns
An object of type TcConnector.TcServerInfo.
**Get Workflow Templates**

Gets a list of workflow templates given in the list of target workspace objects and the **All** or **Assigned** criteria.

**SOA URL**

Workflow-2008-06-Workflow/getWorkflowTemplates

**Teamcenter version**

Teamcenter 8.0

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InputData</td>
<td>Object</td>
<td>The input data for the service operation.</td>
</tr>
<tr>
<td>BusinessObjectMappings</td>
<td>String</td>
<td>A semicolon-separated list of Teamcenter business object names and entity names, for example, 'Item=TcConnector.Item;ItemRevision=TcConnector.ItemRevision'</td>
</tr>
</tbody>
</table>

**Returns**

An entity of type GetWorkflowTemplateResponse. Workflow templates can be retrieved using the association TcConnector.Workflow templates. Partial errors can be retrieved using TcConnector.ResponseData and TcConnector.PartialErrors.
Get Variant Rules

Gets all the variant rules related with the given Item Revision.

SOA URL

Cad-2007-01-StructureManagement/getVariantRules

Teamcenter version

Teamcenter 2007

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ItemRevision</td>
<td>Object</td>
<td>References of the item revision to get the variant rules.</td>
</tr>
<tr>
<td>BusinessObjectMappings</td>
<td>String</td>
<td>A semicolon-separated list of Teamcenter business object names and entity names, for example, 'Item=TcConnector.Item;ItemRevision=TcConnector.ItemRevision'</td>
</tr>
</tbody>
</table>

Returns

The input Item Revision entity's TcConnector.variantRulesForItemRevision association will be updated with related variant rules. Partial errors can be retrieved using the association TcConnector.ResponseData or TcConnector.PartialErrors.
Login

Authenticates user credentials and initializes a Teamcenter session for the Mendix client. This operation throws an InvalidCredentialsException if the username, password, or group is not valid.

When the client application is deployed to a four-tier environment (communication through HTTP or TCCS), the login operation also contributes to the assigning of a Teamcenter server instance to the client session. The Teamcenter architecture varies from other client server architectures in that there is a dedicated instance of the Teamcenter server per client application. However, there are use cases where it is desirable for a single user to have multiple desktop applications running and each sharing a single instance of a Teamcenter server. This is controlled through the following elements:

<table>
<thead>
<tr>
<th>hostPath</th>
<th>From the Connection class constructor, this specifies the address (URI) the Teamcenter server is hosted on.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Username</td>
<td>From this login operation, this specifies the user’s Teamcenter user name.</td>
</tr>
<tr>
<td>sessionDiscriminator</td>
<td>From this login operation, this identifies the client session.</td>
</tr>
</tbody>
</table>

The hostPath argument determines the server machine that the client connects to. Once there, the pool manager on that host uses the username and sessionDiscriminator arguments of the login request to determine which Teamcenter server instance to assign the client to. If the pool manager has an existing Teamcenter server instance with the username/sessionDiscriminator key, the client is assigned to that existing instance of the Teamcenter server, therefore sharing the server with another client. Otherwise, a new instance of the Teamcenter server is used. There are a few general scenarios for the sessionDiscriminator argument:

<table>
<thead>
<tr>
<th>Blank</th>
<th>If the user jdoe logs on to Teamcenter using two or more client applications using a blank sessionDiscriminator argument (for example, jdoe/), all those clients are assigned to the same Teamcenter server instance. These client applications may be running on the same or on different client hosts.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constant</td>
<td>If the user jdoe logs on to Teamcenter using two or more client applications using a constant or fixed sessionDiscriminator argument (for example, jdoe/MyApp), those clients are assigned to the same Teamcenter server instance. This is similar to the blank sessionDiscriminator argument. The difference in this scenario is that only multiple instances of the client application using myApp started by jdoe share the same Teamcenter server instance.</td>
</tr>
</tbody>
</table>
If the user `jdoe` logs on using a unique random-generated string (for example, `jdoe/akdk938lakc`), the client application is assigned to a dedicated instance of the Teamcenter server.

The scenario you use depends on how your client application is used in the integrated environment. The most common case is the unique sessionDiscriminator value.

**SOA URL**
Core-2011-06-Session/login

**Teamcenter version**
Teamcenter 9.0

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserCredentials</td>
<td>Object</td>
<td>Credentials to use for logging on to Teamcenter</td>
</tr>
</tbody>
</table>

**Returns**

Basic information about the server and Partial Errors are returned when the authentication is successful, but the requested role or locale is not supported:

214102: The login is accepted, but the requested role is not valid. In this case, the default role is used.

214109: The login is accepted, but the login group was not specified. In this case, the default role is used.

128003: The logon is accepted. But data entry must be done using certain locales, as specified by the TC_language_data_entry preference. The details of the data entry are returned in the error message.

128004: The logon is accepted. But, data entry must only contain characters that belong to the encoding of the database instance. The information is in the error message.

Throws:

InvalidCredentialsException when the credentials supplied are invalid or the requested locale is not allowed.

515143: The logon was refused due to an invalid username or password.
515144: The logon was refused due to an invalid username or password.
515142: The logon was refused due to an invalid group.
128001: The logon was refused due to conflict with the encoding of the database instance.
128002: The logon was refused due to missing localization.
Logout

Retrieves the Teamcenter session for the user and attempts to log them out of Teamcenter. Once logged out, the cookies associated with the session are deleted, and the Teamcenter host address within the session is set to an empty string.

SOA URL
Core-2006-03-Session/logout

Teamcenter version
Teamcenter Engineering 2005 SR1

Parameters
None.

Returns
A Boolean type.
Perform Action

Performs the specified workflow action.

SOA URL
Workflow-2012-10-Workflow/performAction2

Teamcenter version
Teamcenter 9.1.2

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InputData</td>
<td>Object</td>
<td>Input for performing the workflow action.</td>
</tr>
</tbody>
</table>
| BusinessObjectMappings| String| A semicolon-separated list of Teamcenter business object names and entity names, for example, 'Item=TcConnector.Item;ItemRevision=TcConnector.ItemRevision'

Returns

An object of type TcConnector.ServiceResponse.
Created objects can be retrieved using the associations TcConnector.ServiceResponse, TcConnector.ResponseData, and TcConnector.Created. Partial errors can be retrieved using the associations TcConnector.ResponseData and TcConnector.PartialErrors.
Perform General Search

This service sends the request to and receives the response from the data provider Awp0SavedQuerySearchProvider for the query General... The input criteria for the GeneralSearch query is passed through the generalQuerySearchInput object that is extended from the .SearchInput object.

Note:
This service works only if the Teamcenter environment has an Active Workspace installation.

SOA URL
Query-2010-04-SavedQuery/findSavedQueries
Query-2014-11-Finder/performSearch

Teamcenter version
Teamcenter 8.2
Teamcenter 9.1.3

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>generalQuerySearchInput</td>
<td>Object</td>
<td>This is a type of SearchInput entity that represents the information required to search the business object.</td>
</tr>
<tr>
<td>businessObjectMapping</td>
<td>String</td>
<td>A semicolon-separated list of Teamcenter business object names and entity names, for example, Item=TcConnector.Item;ItemRevision=TcConnector.ItemRevision;WorkspaceObject=TcConnector.WorkspaceObject.</td>
</tr>
</tbody>
</table>

Returns
An entity of type TcConnector.SearchResponse. Search Results can be retrieved using the association TcConnector.searchResultsList. Partial errors can be retrieved using the association TcConnector.ResponseData or TcConnector.PartialErrors.
Perform Item Simple Search

This service sends the request to and receives the response from the data provider Awp0SavedQuerySearchProvider for the query Item - simple. The input criteria for the GeneralSearch query is passed through the generalQuerySearchInput object, which is extended from the .SearchInput object.

Note:
This service works only if the Teamcenter environment has an Active Workspace installation.

SOA URL
Query-2010-04-SavedQuery/findSavedQueries
Query-2014-11-Finder/performSearch

Teamcenter version
Teamcenter 8.2
Teamcenter 9.1.3

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>itemSimpleQuerySearchInput</td>
<td>Object</td>
<td>This is a type of SearchInput entity that represents the information required to search for the business object.</td>
</tr>
<tr>
<td>businessObjectMapping</td>
<td>String</td>
<td>A semicolon-separated list of Teamcenter business object names and entity names, for example, Item=TcConnector.Item;WorkspaceObject=TcConnector.WorkspaceObject.</td>
</tr>
</tbody>
</table>

Returns
An entity of type TcConnector.SearchResponse. Search Results can be retrieved using the association TcConnector.searchResultsList. Partial errors can be retrieved using the association TcConnector.ResponseData or TcConnector.PartialErrors.
Perform Search

Sends the request to and receives the response from the data provider. It routes search requests to a specific provider specified in providerName in the searchInput entity. This is assuming that the searchCriteria for the provider is represented in the searchCriteriaInput object. For example, Awp0SavedQuerySearchProvider is provider that is used for general search. The input criteria for GeneralSearch is passed through the searchCriteriaInput object.

SOA URL

Query-2014-11-Finder/performSearch

Teamcenter version

Teamcenter 9.1.3

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InputData</td>
<td>Object</td>
<td>This is a type of SearchInput entity that represents the information required to search for the business object.</td>
</tr>
<tr>
<td>BusinessObjectMappings</td>
<td>String</td>
<td>A semicolon-separated list of Teamcenter business object names and entity names, for example, Item=TcConnector.Item;WorkspaceObject=TcConnector.WorkspaceObject.</td>
</tr>
</tbody>
</table>

Returns

An entity of type TcConnector.SearchResponse. Search Results can be retrieved using the association TcConnector.searchResultsList. Partial errors can be retrieved using the association TcConnector.ResponseData or TcConnector.PartialErrors.
Revise Object

Revises business objects. This operation revises the given objects and copies or creates new objects using the data for the property values and deep copy data, assuming the reviseInput value for the object is provided. The input for the revise object is passed through the reviseInput entity.

SOA URL

Core-2015-07-DataManagement/getDeepCopyData
Core-2013-05-DataManagement/reviseObjects

Teamcenter version

Teamcenter 11.2
Teamcenter 10.1
Teamcenter 2007.1 MP1

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>objectToRevise</td>
<td>Object</td>
<td>The target business object being revised.</td>
</tr>
<tr>
<td>reviseInput</td>
<td>Object</td>
<td>This is a type of ReviseInput entity that represents the information required for revising the business object.</td>
</tr>
<tr>
<td>businessObjectMapping</td>
<td>String</td>
<td>A semicolon-separated list of Teamcenter business object names and entity names, for example, Item=TcConnector.Item;ItemRevision=TcConnector.ItemRevision;WorkspaceObject=TcConnector.WorkspaceObject.</td>
</tr>
</tbody>
</table>

Returns

An entity of type TcConnector.ReviseObjectsResponse. Revised objects can be retrieved using the association TcConnector.revise_output, and the revise tree can be retrieved using TcConnector.reviseTrees. Partial errors can be retrieved using the association TcConnector.ResponseData or TcConnector.PartialErrors.
Saved Query Search

Searches for saved queries.

**SOA URL**

Query-2010-04-SavedQuery/findSavedQueries
Query-2008-06-SavedQuery/executeSavedQueries

**Teamcenter version**

Teamcenter 8.0
Teamcenter 8.2

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>QueryName</td>
<td>String</td>
<td>The name of the query for which the search is to be performed.</td>
</tr>
<tr>
<td>InputData</td>
<td>Object</td>
<td>This is a type of SearchCriteria entity that represents the information required to search for the business object.</td>
</tr>
<tr>
<td>BusinessObject Mappings</td>
<td>String</td>
<td>A semicolon-separated list of Teamcenter business object names and entity names, for example, Item=TcConnector.Item;ItemRevision=TcConnector.ItemRevision;WorkspaceObject=TcConnector.WorkspaceObject.</td>
</tr>
</tbody>
</table>

**Returns**

An entity of type TcConnector.ServiceResponse. Search Results can be retrieved using the association TcConnector.ResponseData or TcConnector.plain. Partial errors can be retrieved using the association TcConnector.ResponseData or TcConnector.PartialErrors.
Update Properties

Updates Teamcenter objects corresponding to the input model object entities.

SOA URL

Core-2010-09-DataManagement/setProperties

Teamcenter version

Teamcenter 8.3

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>modelObjects</td>
<td>List</td>
<td>A list of model object entities with updated values whose corresponding Teamcenter objects are to be updated on the Teamcenter site.</td>
</tr>
<tr>
<td>businessObjectMapping</td>
<td>String</td>
<td>A semicolon-separated list of Teamcenter business object names and entity names, for example, Item=TcConnector.Item;ItemRevision=TcConnector.ItemRevision;WorkspaceObject=TcConnector.WorkspaceObject.</td>
</tr>
</tbody>
</table>

Returns

An entity of type TcConnector.ServiceResponse. Partial errors can be retrieved using the association TcConnector.ResponseData or TcConnector.PartialErrors. The list of modified model objects can be retrieved using the association TcConnector.ResponseData or TcConnector.Updated.
**Upload File**

Uploads one or more files to Teamcenter using the Teamcenter FMS service.

**SOA URL**

Core-2010-04-DataManagement/createDatasets
Core-2006-03-FileManagement/commitDatasetFiles

**Teamcenter version**

Teamcenter 8.2
Teamcenter Engineering 2005 SR1

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DatasetParameter</td>
<td>Object</td>
<td>Placeholder to maintain the files to be uploaded to Teamcenter. The Dataset.Documents association must hold the files to be uploaded.</td>
</tr>
<tr>
<td>NamedReference</td>
<td>String</td>
<td>Reference name to be used to associate the uploaded files to the dataset.</td>
</tr>
</tbody>
</table>

**Returns**

A Boolean type that returns True or False in case of success and failure, respectively. In addition to this, the UID property is updated on the dataset object, which is passed as an input to action.
Siemens Digital Industries Software

About Siemens Digital Industries Software

Siemens Digital Industries Software is driving transformation to enable a digital enterprise where engineering, manufacturing, and electronics design meet tomorrow. Our solutions help companies of all sizes create and leverage digital twins that provide organizations with new insights, opportunities, and levels of automation to drive innovation. For more information on Siemens Digital Industries Software products and services, visit siemens.com/software or follow us on LinkedIn, Twitter, Facebook, and Instagram.

Siemens Digital Industries Software — Where today meets tomorrow.

© 2020 Siemens. A list of relevant trademarks can be found here. All other trademarks belong to the irrespective owners.