PTC Integrity™ Microsoft Project Integration Guide

PTC Inc., 140 Kendrick Street, Needham, MA 02494 USA

Contents

1 Introduction

The Integrity integration with Microsoft® Project combines the powerful project management features of Microsoft Project with the flexible workflow of Integrity. This allows project managers to track and monitor project status, cost, and readiness, as well as potential associated risks.

The `Integrity2009_MSProject2003_Integration.zip` and the server templates zip files according to Integrity releases are available for download from the Integrity Support Center https://support.ptc.com/appserver/cs/portal/.

2

Overview of the Integration

The Microsoft Project integration works through an add-in component that provides access to the available functionality through the Microsoft Project **Tools** menu. The integration supports:

- Creation of new Integrity items from Microsoft Project tasks

  Creating new items from a task adds more detail to a task, such as a history of state changes, attachments, related items, and change package information. Using the **Synchronize All Tasks** command, you can automatically create items based on existing tasks in Microsoft Project. A hyperlink to the Integrity item also displays next to the Microsoft Project task. Once an item is created, you can track and monitor the item's state changes by synchronizing the task in Microsoft Project.

- Creation of new Microsoft Project tasks from existing Integrity items returned in a query

  Synchronizing by query adds the individual Integrity items as task items in the Microsoft Project file. The **Synchronize Tasks By Query** command essentially populates the Microsoft Project file with items defined only in the target Integrity query. For each item found in the Integrity query, the integration creates a task in Microsoft Project.

- Synchronization of selected or linked tasks

  Maintaining synchronization between Integrity and Microsoft Project is carried out using either the **Synchronize Linked Tasks** or the**Synchronize Selected Tasks** commands. Using the **Synchronize Linked Tasks** command, Integrity publishes and retrieves updates for all tasks in the Microsoft Project file that have been linked between Integrity and Microsoft project. Linked tasks are those tasks that have been previously synchronized between Integrity and Microsoft Project. Using the **Synchronize Selected Tasks** command, selected linked tasks are updated with the information from Microsoft Project, and selected unlinked tasks are added and linked to Integrity.

- Conflict detection for resources or field contents

  Conflict detection operates in cases where a change is entered for the field contents or resources (assigned users) associated with a task in Microsoft Project. Different information is entered for the linked item in Integrity. The integration provides the **Conflict Detected** window to help you resolve the identified differences.

3

Before You Start Microsoft Project Using Integrity

This section provides details on basic system requirements, integration components, assumptions, and known items for the Microsoft Project integration as follows:

- Integrity Server Requirements in Microsoft Project on page 12
- Client Requirements on page 12
- Integration Components on page 13
- Assumptions About Integrating the Microsoft Project on page 13
- Key Considerations on page 13

Integrity Server Requirements in Microsoft Project

- Integrity Server 2009 or higher.
- General requirements for the Integrity Server are as described in the *PTC Integrity Server Administration Guide*. The Integrity Server must be configured to accept remote API connections (see Configuring the Integrity Server on page 18).
- Integrity database requirements are as described in the *PTC Integrity Server Administration Guide*
- The Integrity Server has an appropriate configuration of administration objects installed for the desired purpose.
- The required XML mapping template is installed.

Client Requirements

- Microsoft Windows Vista or 7
- Microsoft .NET Framework 2.0 runtime or higher
- For an automated enterprise-wide installation by an administration, ensure that you also have the following:
  - Windows Installer 2.0
  - Shared Add-in Support Update for Microsoft .NET Framework 2.0 (KB908002). This is included in the KB908002 directory of the ZIP file
- Microsoft Project (for the list of supported versions, see the Support section of http://www.ptc.com).
- Administrators require the Integrity Administration Client for configuring ACL permissions and setting up Integrity workflow.
- The integration user must have a valid user ID in the realm. Also, the integration user must be allowed the View Admin permissions under the `mks:im access control list (ACL)`. For more information on configuring ACL permissions, see the *PTC Integrity Server Administration Guide*.

> **Note**
>
> To use the Microsoft Project integration, users do not require the Integrity Client.

Integration Components

The Microsoft Project integration is available for download from the PTC - Integrity Support Center (http://www.ptc.com/support/integrity.htm). The Microsoft Project integration includes the following components (available as ZIP files):

- `Integrity2009_MSProject2003_Integration.zip` — the Integrity to Microsoft Project Integration install (with Setup Wizard), including the `MKSProjectTemplate.mpt` Project template.
- `Server-Templates` ZIP — the XML mapping templates that configure the integration.

Assumptions About Integrating the Microsoft Project

- You know how to use Microsoft Project. For more information about using the product, refer to the appropriate documentation from the product vendor.
- You are using one of the sample XML mapping templates and have installed an appropriate configuration of administration objects.
- If you are modifying the XML mapping template, you understand and can use XML.
- You have made any required modifications to the internal field values in the XML mapping template to reflect the solution installed field name. For example, if you installed the solution with a prefix, that same prefix is present in the internal field values contained in the XML mapping template. For general information on configuring the XML mapping template, see Configuring a Mapping Template on page 43 For more detailed information, contact PTC - Integrity Support.

Key Considerations

- For existing linked tasks, changes to the task/sub-task hierarchy cannot be made from within Integrity. Any changes to the task/sub-task hierarchy must be made in the project file from within Microsoft Project.
- The Microsoft Project integration does not support commas (,) in user names. Commas contained in user names are changed to spaces. Semi-colons (;), and open and closed brackets are supported in the integration.
- The Microsoft Project integration does not set the following fields:
  `Constraint Type`
  `Request/Demand`

```
Status
Work Contour
```

These fields are read-only and therefore Integrity cannot change the values directly. Other fields (such as `Start Date`, `Finish Date`, or `Status`) can indirectly drive the values for these fields.

Note

To avoid errors, read-only fields should be set to the direction value of in. If a read-only field is set to a bidirectional value (`both`) or the value of `out`, the resulting error is recorded in the log file.

- When working with two Microsoft Project files with shared items in Integrity, running synchronizations with both projects open does not synchronize information across the project files. Integration Independently handles all projects.
- Conflict detection may not function correctly when the Integrity Server and the database reside on separate machines, and clocks for the two machines are not in sync. In this situation, conflict detection errors can occur if an item modification timestamp pre-dates an item creation timestamp.

  To ensure that conflict detection works correctly, use a clock synchronization utility to keep the two machine clocks in sync.
- Microsoft Project includes a limitation of 255 characters for text fields, outline code fields, and values and characters in Enterprise Resource Multi-Value (ERMV) fields. For fields containing more than 255 characters, Microsoft Project truncates the 18 data and adds an ellipsis (…). A Microsoft Project Notes field can hold more than 255 characters; however, a Notes field only displays a maximum of 255 characters before truncation occurs.

  To avoid having truncated data exported to Integrity, consider the following:
  - If you routinely have data in excess of 255 characters, only synchronize from Integrity to a Notes field in Microsoft Project. Avoid performing bidirectional synchronizations.
  - If you need to synchronize data bidirectionally, map the Project Notes field to a 255-character text field in Integrity. If necessary, use an Integrity event trigger for editing operations to copy only the first 255 characters from the Integrity field to the Project Notes Field.

For more information on the 255 character limit in Microsoft Project, refer to Microsoft's product documentation.

- For purposes of running the Microsoft Project integration, only Integrity 2007 or greater can connect to Integrity Server 2009. Older versions of the Microsoft Project XML mapping templates (for example, `MSprojint.xml` for Integrity 2006) are not supported with Integrity 2009.

4

Pre-installation Administration

Before users can install and use the Microsoft Project integration, an administration with access to the Integrity Server must carry out certain configuration tasks for both the Integrity Server and Integrity. Pre-installation administration tasks include configuring the Integrity Server to allow for remote API connections and defining an appropriate configuration of administration objects. Depending on the Microsoft Project fields used in your organization, the administration may also need to modify the XML mapping template that configures the integration.

This section provides the following information on the required pre-installation administration tasks:

- Configuring the Integrity Server on page 18
- Configuring Integrity on page 19
- Setting Permissions in Microsoft Project on page 19
- Modifying the Integration XML Mapping Template on page 19

Configuring the Integrity Server

Before users can work with the Microsoft Project integration, the administration must configure the Integrity Server to allow remote API connections. This configuration sets the connection policy for a server integration point. The two policies that control API access to the Integrity Server are the connection and authentication policies. Settings for these policies are specified in the following file on the Integrity Server:

`installdir/config/client/IntegrityClientSite.rc`

Where *installdir* is the path to the directory where you installed the Integrity Server. The required settings for these policies depend on your integration point; however, defaults are set for the server integration points. For the Microsoft Project integration, changes are required to the connection policy only. No changes are required for the authentication policy settings.

Connection Policy

Within the `IntegrityClientSite.rc` file, the default connection policy specifies the following property:

```
daemon.connectionPolicy=mks.ic.common.policy.
ICAllowSpecificConnectionPolicy
```

The default setting allows only a specific set of IP addresses to connect. The required setting allows remote API connections.

To Configure the API Connection Policy for the Integrity Server

1. Open the following file in a text editor:

   `installdir/config/client/IntegrityClientSite.rc`

   Where *installdir* is the path to the directory where you installed the Integrity Server.
2. Comment out the default policy (that is, insert the # symbol), and then uncomment the following property:

   ```
   daemon.connectionPolicy=mks.ic.common.policy.
   ICAllowAllConnectionPolicy
   ```

   Tip

   When you uncomment a property (by removing the # symbol), you enable the required property.

   This allows all clients to request an API connection; however, all connecting clients still require the proper authentication.

If there is a requirement to specify users who are allowed to connect, use a comma-delimited list of user IDs to define the property for `daemon.validUsersList`. Using a comma-delimited list does not change the connection policy setting.

3. Save and close the file.
4. To have the changes take effect, restart the Integrity Server.

Note

For more information on configuring the Integrity Server to allow remote API connections, see the *PTC Integrity Integrations Builder Guide*.

Configuring Integrity

This version of the Microsoft Project integration provides improved flexibility for configuring the integration. As an alternative, the basic configuration can be extended and modified to suit the workflows and types used for projects within your organization. To create a custom configuration for the Microsoft Project integration, contact PTC - Integrity Support.

Setting Permissions in Microsoft Project

The integration user must have a valid user ID and must be allowed the `ViewAdmin` permissions under the `mks:im` access control list (ACL). For more information on configuring ACL permissions, see the *PTC Integrity Server Administration Guide*.

Modifying the Integration XML Mapping Template

The Microsoft Project integration is based on a configurable XML mapping template and is available for download on the Integrity Support Center (http://www.ptc.com/support/integrity.htm). The following sample XML mapping templates are typical implementation scenarios and enable you to use all the supported features of the Microsoft Project integration:

```
MS_Project_Test_Objective.xml.sample
MS_Project_Work_Items.xml.sample
```

You can also create additional XML mapping template files to support multiple workflows for different Microsoft Project files. Each mapping template you create must have a unique ID. If you need to modify the existing mapping template or create additional templates, contact PTC - Integrity Support. For general information on modifying the XML template, see Configuring a Mapping Template on page 43.

> **Note**
>
> If you need to work with additional features, such as date fields, contact PTC - Integrity Support for assistance.

To modify the Project integration XML mapping template

1. From the `Server-Templates` ZIP on the Integrity Support Center, extract the sample template files to the following directory on the Integrity Server:

   `installdir\data\gateway\mappings\`

   Where *installdir* is the path to the directory where you installed the Integrity Server.
2. Rename the files to remove the `.sample` extension, and then modify the XML mapping fields as required to correspond to your existing workflow. Ensure that the internal field values used in the XML mapping template correspond to the Integrity field names as installed for your existing workflow.

5

Installing the Microsoft Project Integration

You install the Microsoft Project integration using the Setup Wizard provided with the `Project- Integration` ZIP file. The integration ZIP file is available for download from the Integrity Support Center (http://www.ptc.com/support/integrity.htm).

There are two integration installation options included in the ZIP file:

- `setup.exe`. This is recommended for individual users installing the integration.
- `ProjectIntegrationSetup.msi`. This is recommended for an automated enterprise-wide installation by administration.

Before you run an installer, ensure that you have the prerequisites listed in Client Requirements on page 12.

To install the Microsoft Project integration

1. Close Microsoft Project, if currently open.
2. From the `Project-Integration` ZIP file, extract all files to a common directory, and run the `setup.exe` or `ProjectIntegrationSetup.msi` file. The Setup Wizard displays.
3. To continue, follow the instructions provided in the Setup Wizard.
4. To exit the Setup Wizard, click **Close**.
5. From the common directory you unzipped to, copy the `MKSProjectTemplate.mpt` file to the Microsoft templates directory on the machine using the integration. On Windows, this directory is:

`C:/Documents and Settings/`*`username`*`/Application Data/Microsoft/Templates/`

You can now open Microsoft Project and configure the required Microsoft Project properties. For more information, see Microsoft Project Configuration on page 25.

6

Repairing or Removing the Integration

You can use the Setup Wizard to repair or remove the Microsoft Project integration.

Tip

You can also remove the Microsoft Project integration using **Windows Start ▸ Control Panel ▸ Add or Remove Programs**.

To Repair or Remove the Microsoft Project Integration Using the Setup Wizard

1. Ensure that you close Microsoft Project before attempting to repair or remove the integration.
2. From the `Project-Integration` ZIP file, extract and run the `setup.exe` or `ProjectIntegrationSetup.msi` file. The Setup Wizard displays.
3. Select one of the available options as required:
   - To repair the integration with Integrity, select the option for **Repair Integrity to Microsoft Project Integration**.
   - To remove the integration with Integrity, select the option for **Remove Integrity to Microsoft Project Integration**.
4. To continue the selected operation, click **Finish**. The integration with Integrity is repaired or removed, according to the option you selected.

7

Microsoft Project Configuration

To use the integration with Microsoft Project, certain custom properties and fields are created for each project file you are working on. You can also enable logging to assist with any diagnostic work. This section provides information on the following topics:

- Integrity Custom Project Properties on page 26
- Integrity Custom Project Fields on page 27
- Enabling SSL Communication on page 30
- Using Microsoft Project Server and Enterprise Resources on page 30
- Logging Configuration for the Microsoft Project on page 32

Integrity Custom Project Properties

To use the integration with Microsoft Project, certain custom properties are configured for each project file you are working on. The required custom properties are created automatically when you run the first synchronization operation. The required integration properties are configured under **File ▸ Properties**.

> **Note**
>
> The `Integration ID` property is automatically generated after the first synchronization. It includes a unique identification key and has the form: `project name.mpp: identification key` Do not modify this automatically generated property.

The following properties are automatically configured for each Microsoft Project `.mpp` file when you run a synchronization with Integrity:

Name	Type	Value
`Integrity Server Host`	Text	Name of the Integrity Server.
`Integrity Server Port`	Number	Port number used by Integrity to connect to the Integrity Server. By default, the port number is `7001`. Check with the administration to confirm the correct port number.

Name	Type	Value
`Template Name`	Text	By default, references the integration XML mapping template installed on the Integrity Server. The mapping template is configured to use the supported features of the Microsoft Project integration. You can also modify the template to suit the workflow and types used for your projects. For more information on modifying the template, see Modifying the Integration XML Mapping Template on page 19. **Note** This property can also be set to reference another template file.
`Integration ID`	Number	Unique generated identifier key. **Note** Do not modify the generated value, even if you rename the project file.

Integrity Custom Project Fields

The integration also requires certain custom fields to work with the default template installed with Integrity. The required custom fields are pre-defined in the `MKSProjectTemplate.mpt` template file provided with the integration. To use the Integrity template, you must copy the `MKSProjectTemplate.mpt` file to the Microsoft templates directory on the machine using the integration. Once this step is completed, the required fields are created dynamically if they do not exist.

The custom project fields include:

Integrity Custom Project Field	Description
`MKS Issue`	Identifier number of the item in Integrity. Column displayed automatically when using the `MKSProjectTemplate.mpt` file as the template.
`MKS User ID`	User name of the user in Integrity. This is the login ID rather than the full name. Column displayed automatically on the **Resources Sheet** view when using the `MKSProjectTemplate.mpt` file as the template.
`MKS Issue Type`	Type of item in Integrity. In the default configuration, this is one of `Project`, `Requirement`, `Test Objective`, `Test Plan`, `Work Item`, `Feature`, or `Task`. Column displayed automatically when using the `MKSProjectTemplate.mpt` file as the template.
`MKS State`	Initial state for items of the defined type. Field is visible in Integrity, but no column displays in Microsoft Project. **Note** To view Integrity state information in Microsoft Project, you must create a new column for `MKS State`.

Integrity Custom Project Field	Description
	Note Once you create the MKS State column, all tasks must have a valid initial state for the associated Integrity items of that type. If no valid initial state exists for the type, an error occurs during all subsequent synchronizations.
MKS Effort	Pick list field with options of Small, Medium, Large, and Very Large. Information is published from Integrity to Microsoft Project only. Column not displayed by default. To display the column in Microsoft Project, select a task view and choose**Insert ▸ Column**. In the **Column Definition** window, select MKS Effort from the **Field Name** list.
MKS Risk	Pick list field with options of Low, Medium, and High. Information is published from Integrity to Microsoft Project only. Column not displayed by default. To display the column in Microsoft Project, select a task view and choose **Insert ▸ Column**. In the **Column Definition** window, select MKS Risk from the **Field Name** list.
MKS Requirement ID	Is the content of the External IDfield in Integrity. Information is published from Integrity to Microsoft Project only. Column not displayed by default. To display the column in Microsoft Project, select a task view and choose**Insert ▸ Column**. In the **Column Definition** window, select MKS Requirement IDfrom the **Field Name** list.

Enabling SSL Communication

When using the Integrity integration with Microsoft Project, you can choose a Secure Sockets Layer (SSL) connection. To configure SSL for the Microsoft Project integration:

1. Open the Microsoft Project (`.mpp`) file and select **File ▸ Info ▸ Project Information ▸ Advanced Properties**.
2. Click the **Custom** tab and in the **Name** field, add a property for `MKS SSL Connection`.
3. In the **Type** field, select **Yes or no**. In the **Value** field, choose the **Yes** option. To accept the changes click **Add** and then **OK**.
4. Save and reopen the `.mpp` file. The SSL connection to Microsoft Project is now enabled. For Microsoft Project, SSL communications are enabled on a per-session basis. Therefore, if you close Microsoft Project, you must reset the `MKS SSL Connection` property to use secure communications when you start a new session. To deactivate SSL during the same Project session, you can set `MKS SSL Connection` to **No**.

Note

To use an SSL connection with the Microsoft Project integration, you must also enable SSL communications for the server-side API. The API authenticates its session via a separate HTTP connection to the Integrity Server. To allow authentication to use the SSL connection, the Certificate Authority associated with SSL certificate must be registered with the JRE used by the Integrity Server.

Using Microsoft Project Server and Enterprise Resources

Microsoft Project Server stores a pool of Enterprise Resources. To use Enterprise Resources within a project, they must first be imported into the project file. In Microsoft Project Server, the menu action for **Insert ▸ New Resource From ▸ Project Server** and the menu action for **Tools ▸ Build Team from Enterprise**. Once imported into the Microsoft Project file, the Enterprise Resources are treated as normal resources for purposes of editing the project. The mechanism for linking Integrity users with resources in Microsoft Project is based on a required custom resource field (`MKS User ID`). The `MKS User ID` field provides the link between the two systems. It also allows the Microsoft Project resource to own the resource field for purposes of representation, reporting, and control. The Integrity user identifier (as based on the user's Integrity logon) is expected to exactly match the value in the `MKS User ID` custom resource field. This custom field exists in the `MKSProjectTemplate.mpt`, but can be added to any project file as a custom text field for resources. To use the Microsoft Project Server Enterprise Resources, you must edit and record the Integrity user ID information with those resources. When this information is imported into the Microsoft Project file, it is represented with the `MKS User ID` field. For more information on working with Enterprise Resources in the Integrity integration, contact PTC- Integrity Support.

Note

The Integrity **Synchronize Resources** operation pre-loads the Microsoft Project file with resources from the Integrity user base and also populates the custom `MKS User ID`field. To use Enterprise Resources in a synchronization operation, resources in the Microsoft Project file must have user information populated in the `MKS User ID` field. Whether the information is populated automatically or manually, is a matter of preference.

To use Microsoft Project Enterprise Resources in Integrity

1. From the Microsoft Project menu, select **Tools** ▸ **Enterprise Options** ▸ **Open Enterprise Resource Pool** and check out the resources you want associated with Integrity users
2. From the menu, select **Insert** ▸ **Column** and in the resource sheet, insert a custom text field. Ensure that this is the same custom text field that is referenced as the `MKS User ID` field. In the `MKSProjectTemplate.mpt` file, this custom text field is `Text20`.
3. In the created custom text field, enter the Integrity user ID for each Enterprise user. Save these resources back to the Microsoft Project Server resource pool.
4. Insert the Enterprise Resources into the Microsoft Project file.
5. If required, rename the created custom text field to `MKS User ID`. This step is not required if you are using the `MKSProjectTemplate.mpt` and you used `Text20` as the custom field in step 2.
6. You can now synchronize tasks between Microsoft Project and Integrity.

Logging Configuration for the Microsoft Project

To enable logging for the Microsoft Project integration, you must add the `MKS Enable Logging` custom property for each project file you are working on. By default, the `MKS Enable Logging` property is set to `No` and logging is not activated. The `MKS Enable Logging` custom property is configured in the `.mpp` file by selecting **File ▸ Properties**. Once enabled, log and error messages are sent to `<temp dir>/projint.log` on the machine running the integration.

To configure Logging for the Microsoft Project Integration

1. Open the Microsoft Project (`.mpp`) file and select **File ▸ Properties**.
2. Click the **Custom** tab and in the **Name** field, add the following property:

   ```
   MKS Enable Logging
   ```

3. In the **Type** field, select **Yes or no**.
4. In the **Value** field, choose the **Yes** option.
5. To accept the changes click, **OK**.
6. Save and reopen the `.mpp` file. Log and error messages are sent to `<temp dir>/projint.log` file and can be checked to assist with any diagnostic work.

Tip

To deactivate logging, set `MKS Enable Logging` to **No**.

8

Using the Microsoft Project Integration

This section provides instructions on using the integration to synchronize information between the `.mpp` project file and items in the Integrity database. Once a synchronization operation is completed successfully, tasks are then linked between Integrity and Microsoft Project. Before performing any other synchronizations, it is important to first synchronize resources (users) for the project. If you add resources before synchronizing, those resources are included as entries in addition to the resources provided through the synchronization. You can delete the unwanted resources by selecting the associated resource number, right-clicking, and choosing **Delete Resource** from the shortcut menu. Linked tasks are those tasks that have been previously synchronized between Integrity and Microsoft Project. Unlinked tasks are tasks or items that exist either in Microsoft Project or Integrity and have not yet been synchronized through the integration. When a task is linked, the MKS Issue ID displays in the **MKS Issue** field of the task. When a synchronization is complete, the Microsoft Project status bar displays `Ready`.

Note

Ensure that the date format used in your Microsoft project file matches the default Microsoft Project date format—`month/day/year`. If you need to use another date format, contact PTC - Integrity Support for assistance.

Before a synchronization, you can link tasks manually by entering the `MKS Issue ID` number in the **MKS Issue** field. For any new tasks that are not linked in this way, Integrity creates a new Integrity item when the next synchronization runs. Integration functionality is accessible through the Microsoft Project **Tools** menu. This allows you to synchronize linked tasks, selected tasks, tasks in a single Integrity query, all tasks, and resources. The integration also provides conflict detection to help you resolve conflicts in the case of field content and assigned resources (users).

Tip

For task-related synchronization, you must be in a task-oriented view. Synchronizing resource is independent of the view.

Handling Of Empty Date Fields

If a synchronization includes an Integrity item with an empty date field, that date field information is removed from Microsoft Project. If a date field contains a value and you synchronize, that field information cannot be removed in future synchronizations.

The following scenarios show how the integration handles empty and populated date fields:

- Where there are no tasks in the Microsoft Project `.mpp` file, and you synchronize with one or more Integrity items containing an empty date field. The date information in the created Microsoft Project task depends on the date information of the original Integrity item. Wherever the Integrity item included a valid date, the created Microsoft Project task also has a date. Wherever the Integrity item contained an empty date field, the created Microsoft Project task has no date.
- Where Microsoft Project task contains a valid date and the corresponding Integrity item has an empty date field. A synchronization populates the Integrity item with the date information from the Microsoft Project task.

Handling Of Planned Date and Actual Date Fields

When you specify the actual start date in Microsoft Project, the scheduling engine updates the planned start date to the actual start date. The scheduling engine also updates the planned finish date to the actual finish date when you specify the actual finish date. Similarly, Integrity items containing the planned dates (start and finish) are updated to actual dates (start and finish) after a synchronization.

Key Tasks Of Microsoft Project Integration

The primary steps for using the integration are:

- Specifying Task Types on page 36 This task includes specifying tasks types using the MKS Issue Type custom project field.
- Setting Task Relationships on page 37 This task includes setting task relationships using the outline level in Microsoft Project.
- Synchronizing Resources on page 38. This task includes synchronizing project resources (or users) using the **Synchronize Resources** command.

- Synchronizing All Tasks on page 39 This task includes publishing task items from an existing Microsoft Project to Integrity using the **Synchronize All Tasks** command.
- Synchronizing Tasks By Query on page 39. This task includes publishing items from an existing Integrity query to Microsoft Project using the **Synchronize Tasks By Query** command. Each item from the query is linked to a task in Microsoft Project to create a new project.
- Synchronizing Linked Tasks on page 40 and Synchronizing Selected Tasks on page 41.These tasks include maintaining the status of items in Microsoft Project and Integrity using the **Synchronize Linked Tasks** and **Synchronize Selected Tasks** commands as required.
- Detecting Conflicts on page 41. This task includes resolving any conflicts detected during synchronizations.

Specifying Task Types

When creating a task in Microsoft project, you specify the type of Integrity item. For example `Project`, `Requirement`, `Test Objective`, `Test Plan`, `Work Item`, `Feature`, `or Task` using the `MKS Issue Type` custom project field. Once the task is assigned to a type, the next synchronization creates that task as an item of the specified type in Integrity. The mapped fields are then dependent on this type. The `MKS Issue Type` custom project field is displayed automatically in Microsoft Project when using the `MKSProjectTemplate.mpt` file as the template. For more information on custom project fields, see Integrity Custom Project Fields on page 27.

Note

Using the default template, the `MKS Issue Type` is a required field. If you are not using the pre-configured `MKSProjectTemplate.mpt` file, you must create this field manually.

To Specify the Type of Task Using Microsoft Project

In the Microsoft Project file, create the required line items for the tasks. In the `MKS Issue Type` column, enter the type that is used for the linked item in Integrity. For example, `Project`, `Requirement`, `Test Objective`, `Test Plan`, `Work Item`, `Feature`, `Task`. Run the required synchronization —**Synchronize All Tasks** or **Synchronize Selected Tasks**—to publish the information to Integrity. For more information, see Synchronizing All Tasks on page 39 or Synchronizing Selected Tasks on page 41. The tasks are created as items of the specified types in Integrity.

Setting Task Relationships

When manually adding tasks in Microsoft Project, you can create relationships between tasks using the outline level. To relate tasks, you set the main task at the top outline level and indent the related subtasks as required. Each task is indented one additional level to the task it is related to. In Microsoft Project, the outline level is controlled through **Project ▸ Outline ▸ Indent** or **Outdent**. For more information, refer to the documentation provided with Microsoft Project. Once the parent/child relationships are set using the outline level in Microsoft Project, the next synchronization creates the specified relationships between the items in Integrity. Only `OutlineChildren` must be used while mapping task relationships downward through the hierarchy in Integrity. The relationships are then shown when you view the linked item in Integrity.

Note

The relationship hierarchy cannot be changed from within Integrity. Once the relationship hierarchy is set in Microsoft Project, synchronization is run to link with items in Integrity. Custom configurations can be created to allow greater flexibility.

For example, if you set the following hierarchy of relationships, running a synchronization that links to items in Integrity:

```
Requirement 1
     Feature 1
     Feature 2
Requirement 2
```

and then from Integrity attempt to change the hierarchy to:

```
Requirement 1
     Feature 1
Requirement 2
     Feature 2
```

The relationship hierarchy in Microsoft Project is not updated. To change the relationship hierarchy, you must make the required changes through Microsoft Project.

Note

If the integration encounters an unknown type during a synchronization, the unknown type may be removed from existing relationships in Integrity.

To Set the Task Relationship Using Microsoft Project

In the Microsoft Project file, create the required line items for the tasks. Using **Project ▸ Outline ▸ Indent** or **Outdent**, set the outline level required to relate the tasks. Each task is indented one additional level to the task it is related to. Run the required synchronization—**Synchronize All Tasks** or **Synchronize Selected Tasks**—to publish the information to Integrity. For more information, see Synchronizing All Tasks on page 39 or Synchronizing Selected Tasks on page 41. The corresponding relationships are created for the linked items in Integrity.

Synchronizing Resources

The **Synchronize Resources** command publishes resources to both Microsoft Project and Integrity and ensures that resources (users) are created/updated in the project file, and updated in Integrity. If the Integrity user does not have a full name, the `MKS User ID` is used as the resource name. If a full name is later added in Integrity, the resource name is updated to the full name in the next synchronization.

> **Note**
>
> Before performing any other synchronizations, you must first synchronize resources (users) for the project. If you add resources before synchronizing, those resources are included as entries in addition to the resources provided through the synchronization. You can delete the unwanted resources by selecting the associated resource number, right-clicking, and choosing **Delete Resource** from the shortcut menu.

To Synchronize Resources

1. From Microsoft Project, select **Add in ▸ PTC Integrity ▸ Synchronize Resources**.
2. Enter the Integrity Server **Details** and **Connection Details** if prompted, and from the list, choose the XML mapping template you want to use. A connection to Integrity is opened and all resources are synchronized. The Microsoft Project status bar displays **Ready** to indicate that the synchronization is complete.

> **Tip**
>
> You can also link resources manually by entering the user’s login ID in the `MKS User ID field`.

Synchronizing All Tasks

The **Synchronize All Tasks** command essentially allows you to populate Integrity with task items defined only in the Microsoft Project file. For each task item in Microsoft Project, the integration creates an item in Integrity, or updates the item if the task is already linked.

To synchronize All Tasks

1. From Microsoft Project, select **Add-Ins ▸ PTC Integrity ▸ Synchronize All Tasks**.
2. Enter the **Integrity Server Details** and **Connection Details** if prompted and, from the list, choose the XML mapping template you want to use. A connection to Integrity is opened and all tasks are synchronized. The Microsoft Project status bar displays **Ready** to indicate that the synchronization is complete.

Synchronizing Tasks By Query

The **Synchronize Task By Query** command allows you to synchronize tasks based on a defined Integrity query. Synchronizing by query adds the individual Integrity items as task items in the Microsoft Project file. The **Synchronize Tasks By Query** command essentially populates the Microsoft Project file with items defined only in the target Integrity query. For each item found in the Integrity query, the integration creates a task in Microsoft Project. You can use this command to start a new `.mpp` file based on an existing query in Integrity. For example, for a query based on a specific type —`All Features`— you could create new project file that included all items of the type `Feature`. This is found by the Integrity query.

Note

When synchronizing Integrity items with Microsoft Project, new resources are created and assigned to the linked tasks. If the items have users (resources) that do not exist in Microsoft Project.

To Synchronize a Task by Integrity query

1. From Microsoft Project, select **Add-ins ▸ PTC Integrity ▸ Synchronize All Tasks.**
2. Enter the **Integrity Server Details** and **Connection Details** if prompted and, from

the list, choose the XML mapping template you want to use. The **Select an Integrity Query** window displays, retrieving the list of available Integrity queries.

3. From the list of Integrity queries, select a query to synchronize by. To accept your selection, click **OK**.

The synchronization with Integrity is performed and the Microsoft project file is populated with tasks based on items in the Integrity query. The Microsoft Project status bar displays `Ready` to indicate that the synchronization is complete.

Synchronizing Linked Tasks

When synchronize a linked task, Integrity publishes and retrieves updates for all tasks in the Microsoft Project file that have been linked between Integrity and Microsoft Project. Linked tasks are those tasks that have been previously synchronized between Integrity and Microsoft Project. Unlinked tasks are tasks or items that exist either in Microsoft Project or Integrity and have not yet been synchronized through the integration. When a task is linked, the `MKS Issue ID` displays in the **MKS Issue** field of the task.

When synchronizing linked tasks, you do not need to select every task in a set of relationships. If a relationship is mapped (as to predecessor, successor, outline parent, or outline children), Integrity automatically selects the related tasks, related tasks are in Integrity as required.

Note

Before a synchronization, you can link tasks manually by entering the `MKS Issue ID` number in the **MKS Issue** field. For any new tasks that are not linked in this way, Integrity creates a new Integrity item when the next synchronization runs.

To Synchronize a Linked Task

1. From Microsoft Project, select **Add-ins ▸ PTC Integrity ▸ Synchronize linked Tasks**.
2. Enter the **Integrity Server Details** and **Connection Details** if prompted and, from the list, choose the XML mapping template you want to use. A connection to Integrity is opened and all tasks are synchronized. The Microsoft Project status bar displays `Ready` to indicate that the synchronization is complete.

Synchronizing Selected Tasks

The Microsoft Project integration includes functionality that allows you to synchronize selected project tasks, whether those tasks are linked or unlinked. When synchronizing selected tasks, you highlight the project tasks you want to synchronize, run the **Synchronize Selected Tasks** operation, and the selected tasks are published to Integrity. Using the **Synchronize Selected Tasks** command, selected linked tasks are updated with the information from Microsoft Project, and selected unlinked tasks are added and linked to Integrity. When synchronizing selected tasks, you do not need to select every task in a set of relationships. If a relationship is mapped (predecessor, successor, outline parent, or outline children) Integrity automatically selects the related tasks, and new, related tasks are created in Integrity as required.

To Synchronize Selected Tasks

1. From Microsoft Project, select **Add-ins** ▸ **PTC Integrity** ▸ **Synchronize Selected Tasks**.
2. Enter the **Integrity Server Details** and **Connection Details** if prompted and, from the list, choose the XML mapping template you want to use. A connection to Integrity is opened and all tasks are synchronized. The Microsoft Project status bar displays `Ready` to indicate that the synchronization is complete.

Detecting Conflicts

Conflict detection operates in cases where a change is entered for the field contents or resources (assigned users) associated with a task in Microsoft Project. Different information is entered for the linked item in Integrity. The integration provides the **Conflict Detected** window to help you resolve the identified differences.

To Resolve an Identified Conflict

Review the content of the **Conflict Detected** window where the information is displayed in the **Microsoft Project** and Integrity text fields. In the **Resolution** field, enter the correct information that is used to populate both Microsoft Project and Integrity. To accept the change, click **OK**. The resolution information is published to both Integrity and Microsoft Project. For example, the following window shows the user `mchang` being set as the resource for the item or task.

9

Configuring a Mapping Template

This appendix provides information on the XML mapping template file that is used to configure the integrations with Microsoft Project. The mapping template file defines the necessary two-way relationships between internal Integrity items and external items. The two-way mappings allow each set of items to remain synchronized such that a change made in one set of items is reflected in the other.

To assist you in editing a mapping template, this appendix includes the following topics:

- Overview of Configuring a Mapping Template on page 43
- Mapping Template XML Elements on page 45
- Sub-Elements on page 56

Note

The mapping templates included with Integrity are functioning examples and should always be edited with care. If you are not familiar with XML, contact PTC - Integrity Support for further assistance.

Overview of Configuring a Mapping Template

While the configuration is an XML file like an IIF document or the Wizard Configuration file, it is more than a way of representing data. It is a small programming language that lets you define the process of importing and exporting items from an external data source and Integrity internal items.

The Gateway Configuration file processes items one at a time and determines the complete set of mapping rules to apply to the item before actually applying any of them. This allows later steps to change or negate previous mapping rules. It also means that regardless of what mappings are scheduled to be applied, it is always the original values of the internal and external fields.

There are two other important concepts for configuration files: conditional mappings and levels.

Conditional mappings are mapping instructions that only apply when they are activated by a `<map-conditional>` element (or the `<map-translation>` subelement). These conditional mappings are contained within `<map>` elements. Until mappings are activated, they are ignored. Once a `<map>` is activated, the mapping instructions within that element become active. This can include additional `<map-conditional>` elements and the corresponding `<map>` elements; although such `<map>` elements are also ignored until they are activated.

Levels are a related concept. The process begins with the main instructions at level 1. Each time a `<map>` element is activated, the process moves up a level and remains at that new level until the `</map>` tag is reached. At that point, the process moves back to the previous level.

The importance of levels is in the way they interact with the `<map-conditional>` and `<map>` elements. When a `<map-conditional>` element is attempting to find a matching `<map>`, only `<map>` elements at the current level of processing are examined. This means that if you have a `<map-conditional>` element within a `<map>` element, the corresponding `<map>` elements that are activated with `<map-conditional>` must also appear within the same `<map>` element. Additionally, once a `<map-conditional>` on a given level has activated a `<map>` element and its contents, all other `<map-conditional>` elements at the current level are ignored.

Mapping Template XML Elements

This section describes the various XML elements of the mapping template, and provides information on how to configure those elements.

<mapping>

The <mapping> element is the highest level XML element. As such, it contains all other elements. There is only one <mapping> element in the file.

Attributes:

name

The `name` attribute specifies the name of the particular Gateway conversion process defined in the file. This name is the identifier through which a Gateway configuration is selected for use by the Gateway Wizard. It is expected to be unique across all known Gateway configurations.

template-version

The `template-version` attribute specifies the version of the XML template used for creating the Gateway Configuration file. This documentation describes version 2.1. This value is required for the proper operations of Gateway.

Parent Element:

none (highest level XML element)

Sub-Elements:

`<description>`, `<set-property>`, `<link-field>`, `<field>`, `<map-conditional>`, `<map>`

<description>

The `<description>` element specifies a description of the particular conversion process defined in the file. The Gateway Wizard displays this description when prompting for the selection of the Gateway configuration to use. There is one `<description>` element in a configuration element.

Parent Element:

<mapping>

Sub-Elements:

none

<set-property>

The `<set-property>` element sets an item control property to a given value. For example:

```
<set-property name="owner" value="external"/>
```

When importing items into Integrity, there are a few common properties worth mentioning:

prototype
This property helps Integrity identify the type of item it is importing. Valid values include `DOCUMENT` and `CONTENT`. Commonly, this `prototype` property is provided directly within the definition of the item data being imported. The Gateway configuration may be used to override and set the item's `prototype` property.

owner
This property helps Integrity determine whether the item is externally or internally owned. Valid values are `""external"` and `"internal"`. If a document, for example, externally owned data would mean that Integrity should delete any content that is no longer present within the data being reimported.

prior-data-field
This property, in the case where a document is being reimported, identifies to the attachment field within which an IIF document representing the previously imported data would be found. This property is only relevant to document items.

prior-data-name
In the case where a document is being reimported, identifies to the name of the attachment file (an IIF document) that represents the previously imported data. Previously imported data is used to help identify the set the changes to be performed on the data.

Attributes:

name
The `name` attribute specifies the name of the property to set.

value
The `value` attribute specifies the value to which the property indicated that by the name attribute is to be set.

Parent Element
`<mapping>, <map>`

Sub-Element:
none

<link-field>
The `<link-field>` element automatically creates a link between a unique ID field in the external data source and the internal data. By creating such a link, it is much easier to update existing fields when reimporting previously

imported data. The `<link-field>` element has four basic forms. The first form is:

```
<link-field field-type="id">
```

When using this form, each item in the external data source has a unique identifier (similar to the primary key of a database) which is the Integrity identifier. The second form is:

```
<link-field external="REQID" field-type="id">
```

In this form, a field in the external data named REQID contains the Integrity identifier for the corresponding field in the internal data. The third form is:

```
<link-field internal="DATA_KEY">
```

As with the first form, the external data source has a unique identifier but instead of using it as the Integrity identifier, it is stored in the internal field named `DATA_KEY`. Finally, the fourth form is:

```
<link-field external="REQ_ID" internal="DATA_KEY">
```

With this form, the external data field named `REQ_ID` is linked with the internal data field named `DATA_KEY`.

Attributes:

external

The `external` attribute specifies the name of a field in the external data source.

internal

The `internal` attribute specifies the name of a field in the internal data.

field-type

The `field-type="id"` attribute specifies that a field is an `id` field.

hash-code

Some external data sources (for example, Oracle databases) do not allow database look-ups using text strings, and only allow look-ups using integers. To accommodate such sources, you can specify the hash-code attribute which specifies a field name in which a `hash code` is stored.

Parent Element:

`<mapping>, <map>`

Sub-Elements:

none

<field>

The `<field>` element defines the mapping between fields in the external data source and fields in the internal data. The simplest version of this tag names only the external and internal fields. For example:

```
<field external="Type" internal="RQ_Share Category"/>
```

Attributes:

external

The `external` attribute specifies the name of a field in the external data source.

internal

The `internal` attribute specifies the name of a field in the internal data.

direction

The `direction` attribute specifies the direction of the relationship between the external data field and the internal data field.

Direction	Description
`in`	The internal data field is updated from the external data field.
`out`	The external data field is updated from the internal data field.
`both`	The external and internal data field can both be updated from the other, depending on context.
`none`	This is a special value that indicates that all mappings which have been declared so far are to be ignored and not applied.

When the direction attribute is not specified, the behavior is as though `direction=both` was specified.

field-type

The `field-type` attribute identifies what type of content the field holds. Possible values include:

field-type	Description
`type`	Integrity item type
`relationship`	Integrity relationship field
`attachment`	An attachment
`richcontent`	Rich content (that is, formatted text) possibly with links and graphics

`date`	Date with no time
`date-time`	Date and time

data-type

The `data-type` attribute determines how the data in the field is interpreted. Possible values include:

data-type	Description
`xhtml`	Specifies that rich content in a field is in XHTML as opposed to an Integrity Rich Content format.

attachment

The `attachment` attribute is used when mapping fields containing rich content to identify the Integrity attachment field that is used to store images embedded.

external-date-format

The `external-date-format` attribute specifies the date format used in the external data source. For a date field, the value for this attribute should be specified in the `yyyy-MM-dd HH:mm:ss` format. `HH` in the attribute value represents the 24-hour time format.

Note

If the value for the `external-date-format` attribute is not specified in the `yyyy-MM-dd HH:mm:ss` format, an error occurs during synchronization.

clobber

The `clobber` attribute defines how existing attachments associated with the field are handled. The clobber attribute is only meaningful when `field-type="attachment"` is also specified; otherwise, it is ignored. The `clobber` attribute can be set to `true` or `false`.

clobber	Description
`true`	All attachments for the field are replaced with the associated version of the attachment regardless of whether the attachment has change.
`false`	Existing attachments for the field are not removed and only updated attachments are modified.

When no `clobber` attribute is specified, the behavior is as though `clobber=false` was specified.

on-create-only

The `on-create-only` attribute can be set to `true` or `false`.

on-create-only	Description
`true`	The field is only populated on its initial creation. If the field exists, nothing is done.
`false`	The field is populated whether it already

value-translation-type

In some cases, data in the corresponding external and internal fields are stored in two different ways. For example, the external field may contain a range of integer values, while the internal field simply contains string values of Low, Medium, and High. The `value-translation-type` attribute specifies the relationship in such cases.

value-translation-type	Description
`string-string`	A string value in the external field corresponds to a different string value in the internal field.
`intrange-string`	A range of integers in the external field corresponds to a string value in the field.
`integer-integer`	An integer value in the external field corresponds to a different integer value in the internal field.

When the `value-translation-type` attribute is specified, you must also specify a *value-translation* subelement which defines the exact method of translating from one to another.

comparable

The comparable attribute has no meaning to the mapping process itself. However, it is used by the Gateway Wizard to help determine the set of fields displayed in the differencing and changes preview before starting the import process. Valid values include:

Comparable	Description
`true`	The field is to be included in the differencing and changes preview.
`false`	The field is not to be included in the differencing and changes preview.

Parent Element:

`<mapping>, <map>`

Sub-Elements:

`<default>, <value-translation>`

<default>

The `<default>` element specifies a value that is assigned to a field when either only an external or internal field. This is named or the internal or external field from which it should be updated does not exist. The following example sets the `RQ_Categoryname` field in the internal data to a value of `Heading` when the internal data items are being first created.

```
<field internal="RQ_Category" direction="in" on-create-only="true">
       <default>Heading</default>
</field>
```

Attributes:

none

Parent Element:

`<field>`

Sub-Elements

none

<value-translation>

When a `<field>` element has a `value-translation-type` attribute attached, the `<value-translation>` element defines the external data source correspond to values in the internal data.

For example: define the correspondence between ranges of integers in the external field and a single string value in the internal field.

```
<field external="Priority" internal="RQ_Priority" direction="both"
value-translation-type="intrange-string">
      <value-translation external="0..250" internal="Low" />
      <value-translation external="251..500" internal="Medium" />
      <value-translation external="501..750" internal="High" />
      <value-translation external="751..1000" internal="Critical" />
</field>
```

Attributes:

external

The `external` attribute defines which value in the external field corresponds to the value in the internal field defined by the `internal` attribute. The type of value (or range) is specified by the `value-translation-type` attribute of the parent `<field>` element.

internal

The `internal` attribute defines which value in the internal field corresponds to the value or (range) in the internal field as defined by the `external` attribute. The type of value (or range) is specified by the `value-translation-type` attribute of the parent `<field>` element.

Parent Element:

`<field>`

Sub-Elements:

none

<map-conditional>

The `<map-conditional>` element specifies that the mapping of a field is conditional on the value of a field or its properties. Based on the value of the field or property, a set of specialized mapping rules (defined by a `<map>` element) is activated. In its simplest form, the value of the specified field or property is used as the name of the `<map>` to be activated. For example:

```
<map-conditional property="prototype" />
```

examines the value of the `prototype` property and looks for a `<map>` element with a name that matches the value of prototype. If found, that `<map>` is activated and the specialized mapping rules defined within that `<map>` are applied. Using this example, if the value of the prototype property is CONTENT and a `<map name="CONTENT">` element exists, that `<map>` is activated. Once a `<map-conditional>` element has activated a `<map>`, all other `<map-conditional>` elements at that level are ignored. For example:

```
<map-conditional external="Description" />
<map-conditional property="prototype" />
```

first looks for a `<map>` that has a name that matches the value of the external field name `Description`. If such a `<map>` is found, it is activated and the second `<map-conditional>` is ignored. However, if no such `<map>` is found, the second `<map-conditional>` is examined. It is examined to see if the field `<map>` matches the name of the `prototype` property. If the field or property whose value is being compared to <map> names does not exist in the current item, the <map-conditional> is ignored.

Attributes:

external

The `external` attribute specifies the name of a field in the external data source whose value is tested to find a matching <map> element.

internal

The `internal` attribute specifies the name of a field in the internal data whose value is tested to find a matching <map> element.

property

The property attribute specifies an item control property whose value is tested to find a matching <map> element.

direction

The `direction` attribute specifies that a matching <map> is only to be activated for one reason. If the direction of the relationship between the external data source field and the internal data field matches the specified value. Possible values include:

Direction	Description
`in`	The internal data field is updated from the external data field.
`out`	The external data field is updated from the internal data field.
`both`	The external and internal data field can both be updated from the other, depending on context.

For example:

```
<map-conditional property="prototype" direction="in">
```

only activates a matching <map> for the `prototype` value if the internal field is being updated

from the external field. That is, it is only activated if the mapping is an import operation.

default-map

The `default-map` attribute specifies the name (as defined by a <map> element) of a set of specialized mapping rules that are to be applied when the value being matched to <map> names. If it matches no <map> elements and has no other recognized value (it is a null value or its value is not represented in any <map-translation> subelement.

Parent Element:

<mapping>, <map>

Sub-Elements:

<map-translation>

<map-translation>

The <map-translation> element is a subelement of the <map-conditional> element. It lets you specify a list of values for the field or property being examined in the <map-conditional> and associate them with a <map> element. The specified <map> is activated if the examined field or property has the given value. For example:

```
<map-conditional external="Category">
     <map-translation external="Software" map-name="Application" />
     <map-translation external="Utility" map-name="Application" />
</map-conditional>
<map name="Application">
...
</map>
```

In this case, if the value of the external field named Category is Application, then <map name="Application"> is activated. If the value is not Application, the <map-translation> subelements come into play if the Category field is either Software or Utility, then the <map name="Application"> is also activated. When no external attribute is specified in a <map-translation> subelement, then any nonnull value for the field or property specified in the parent <map-conditional> element. Which has not already been matched activates the specified <map>. For example:

```
<map-conditional external="Category">
     <map-translation external="Software" map-name="Application" />
     <map-translation external="Utility" map-name="Application" />
     <map-translation map-name="Non-Application" />
</map-conditional>
<map name="Application">
...
</map>
<map name="Non-Application">
...
</map>
```

In this example, if the value of the Category field is either Application or Non-Application, the <map> element with that name is activated. If the field has another value, the <map-translation> subelements specify that a value of Software or Utility for that field activates <map name="Application"> and any other non-null value activates <map name="Non-Application">. You can only specify one <map-translation> subelement without an external attribute per <map-conditional> element.

Attributes:

external

In a <map-translation> element, the external attribute does not refer to the name of a field in the external data source. It specifies a possible value for the field or property in the parent <map-conditional> element that is being matched to <map> names. In the example, the external attributes in the <map-translation> elements specify possible values for the external field named Category.

map-name

The <map-name> attribute specifies the name (as defined by a <map> element) of a set of specialized mapping rules that should be activated when the field or property being matched in the parent <map-conditional> has a value that matches the value specified by the external attribute.

When no external attribute is specified and the field or property is being matched has a nonnull value, the specified <map-name> is activated regardless of the value of that field or property. There can only be one such <map-translation> sub-element per <map-conditional>.

Parent Element:

<map-conditional>

Sub-Elements:

none

<map>

The <map> element defines a specialized set of mappings (using <field> elements) that are only performed when activated by a <map-conditional> element. When a <map> element is not activated. You can nest <map> elements, allowing you to specialize the mappings to be performed even further.

For example:

```
<map-conditional property="prototype"/>
...
<map name="CONTENT">
     <map-conditional external="Category"/>
     ...
     <map name="Software Requirements">
     ...
     </map>
     <map name="Hardware Requirements">
     ...
</map>
```

In this example, the `prototype` property of the item is examined and the main level of the Gateway Configuration file. The level where `<map-conditional property="prototype"/>` resides is searched looking for a `<map>` with a name that matches the value of the `prototype` property. Thus, if `prototype` has a value of `CONTENT`, the `<map name="CONTENT">` element is activated. Within that `<map>`, the `<map-conditional external="Category">` element examines the value of the external data source field named `Category`. Which in this example, identifies a category of requirement and looks to activate a `<map>` with a name matching the value of `Category`. Such a matching `<map>` must exist at the same level as that `<map-conditional>` within the `<map>` element. The `<map name="Software Requirements">` or `<map name="Hardware Requirements">` is activated if the `Category` field has the value `"Software Requirements"` or `"Hardware Requirements"`, respectively.

By using nested `<map>` elements, you can create complex AND/OR structures for your conditionals. For example, the above example represents the following logic:

- If the `prototype` is `"CONTENT"` AND the `Category` field is `"Software Requirements"`, use the `Software Requirements` mappings.
- If the `prototype` is `"CONTENT"` AND the `Category` field is `"Hardware Requirements"`, use the `Hardware Requirements` mappings.

Attributes:

name

The `name` attribute specifies a name for this particular set of mapping rules. This is the name specified with the `default-map` attribute of the `<map-conditional>` element or the `map-name` attribute of the `<map-translation>` element.

Parent Element:

`<mapping>, <map>`

Sub-Elements:

`<set-property>, <link-field>, <field>, <map-conditional>, <map>`

Mapping Template Example

The following mapping template shows how the various XML elements work together to configure the Microsoft Project integration.

The sample template includes mapping components for Work Items:

```
<?xml version="1.0" encoding="UTF-8"?>
<mapping name="Microsoft Project ALM Work Items"
  template-version="3.0">
   <!--
       Sample Mapping configuration for the ALM 2009 Solution.

       The ALM_ prefix on the internal field names will need to be changed to
       match the prefix used on the ALM 2009 solution installation.
   -->

   <!-- Identify mapping config based on the item's prototype property -->
    <!-- The MS Project Integration maps both ISSUEs and USERs -->
    <map-conditional property="prototype" />

    <!--Issue Mapping configuration -->
    <map-name="ISSUE">
      <!--
        For any integration with MS Project, there must be a
        (custom) field that can be used to store the associated
        MKS Issue ID for a task.  Using this integration, that
        field is expected to be called "MKS Issue".

        NOTE: Whereas this field must exist, it need not be visible.
      -->
      <link-field external="Task.MKS Issue" field-type="id" />

      <!--
       Mapping sub-configurations are based on MKS Integrity issue types.

       The "MKS Issue Type" is a required (custom) field that
       must exist in any MS Project file that is to participate
       in this integration as defined by this template.

       When creating new issues, the issue type (and hence the
       sub-configuration) is based on a custom Microsoft Project
       field.  This field is required and the mappings are limited
       to only the values specified below.

       If the field value is not one of the values specified below
       then by default we use the "Ignored Issue" subconfiguration.

       NOTE: Type cannot be changed in MKS Integrity after an issue
           is created. Changes in this field in MS Project are not
           recommended and will effectively be ignored.
       -->
      <map-field external="Task.MKS Issue Type"
                 internal="Type"
                 direction="both"
                 required="true"
                 field-type="type"
                 default-map="Ignored Issues">
```

```
  <value-translation external="Project" internal="ALM_Project" />
  <value-translation external="Work Item" internal="ALM_Work Item" />
</map-field>

<!-- Independent of the type, the following fields are mapped -->

<field external="Task.Name"
       internal="Summary" />

<field external="Task.Notes"
       internal="ALM_Description"
       direction="out" />

<field external="Task.Resource Names"
       internal="Assigned User" />

<!--
  Create (sub) configuration where we can effectively
  disable all mappings as defined in the common map above.
  This is a mechanism for recognizing irrelevant issues/tasks
  that should be ignored by the integration.
-->
<map name="Ignored Issues">
  <field external="Task.MKS Issue Type" direction="none" />
  <field external="Task.Name" direction="none" />
  <field external="Task.Notes" direction="none" />
  <field external="Task.Resource Names" direction="none" />
</map>

<!-- Mapping specific to MKS Integrity Project -->
<map name="Project">
   <field external="Task.MKS State"
          internal="State"
          on-create-only="true"
          required="true">
     <default>ALM_Proposed</default>
   </field>

   <field external="Task.Start"
          internal="ALM_Project Start Current Estimate"
          field-type="date-time"
          data-type="date"
          external-date-format="yyyy-MM-dd HH:mm:ss"
          direction="both" />

   <field external="Task.Finish"
          internal="ALM_Development Complete Current Estimate"
          field-type="date-time"
          data-type="date"
          external-date-format="yyyy-MM-dd HH:mm:ss"
          direction="both" />
```

```
      <field external="Task.OutlineChildren"
             internal="ALM_Work Items"
             field-type="relationship"
             include-related="true" />
   </map>

   <!-- Mapping specific to MKS Integrity Work Items  -->
   <map name="Work Item">
      <field external="Task.MKS State"
             internal="State"
             on-create-only="true"
             required="true">
         <default>ALM_Proposed</default>
      </field>

      <field external="Task.Start"
             internal="ALM_Planned Start Date"
             field-type="date-time"
             data-type="date"
             external-date-format="yyyy-MM-dd HH:mm:ss"
             direction="both" />

      <field external="Task.Finish"
             internal="ALM_Planned End Date"
             field-type="date-time"
             data-type="date"
             external-date-format="yyyy-MM-dd HH:mm:ss"
             direction="both" />

      <field external="Task.OutlineChildren"
             internal="ALM_Work Items"
             field-type="relationship"
             include-related="true" />

   </map>
</map>

<!-- User mapping configuration -->
<map name="USER">
   <!--
     For any integration with MS Project, there must be a
     (custom) field that can be used to store the associated
     MKS User names for a resource.  Using this integration, that
     field is expected to be called "MKS User ID".

     NOTE: Whereas this field must exist, it need not be visible.

     NOTE: As can be seen from the direction attributes in this
     sections, the integration is only used to pull information
     from MKS Integrity and place it in the MS Project file.
```

```
          There are no updates of resources from MS Project.
        -->
        <link-field external="Resource.MKS User ID" />

        <field external="Resource.Name"
               internal="fullname"
               direction="out" />

        <field external="Resource.Notes"
               internal="description"
               direction="out" />

        <field external="Resource.EMailAddress"
               internal="email"
               direction="out" />

    </map>
</mapping>
```

Key Consideration About the Mapping Template

Specify the value for the `yyyy-MM-dd HH:mm:ss` format in the mapping template file. The configuration of your mapping template can affect the time information update in Integrity and Microsoft Project after a synchronization as per the following scenarios:

- Scenario 1: The Integrity item is configured to include time information in the date field. However, the Integrity item does not contain time information in the date field. The XML elements `field-type="date-time"` and `date-type="date"` are specified in the mapping template file.

  In this scenario, the default time of Microsoft Project is appended to the dates in Microsoft Project after a synchronization. The default start time of Microsoft Project is appended to the planned start date and the actual start date. The default finish time of Microsoft Project is appended to the planned finish date and the actual finish date.

  When using the integration for creating or synchronizing Integrity items, the time information available in the Microsoft Project date fields for the planned dates (start and finish) and the actual dates (start and finish) is appended to the planned dates (start and finish) and the actual dates (start and finish) of the Integrity items, respectively, after a synchronization.
- Scenario 2: The Integrity item is not configured to include time information in the date field. The XML elements `field-type="date"` and `date-type="date"` are specified in the mapping template file.

  In this scenario, when you use the integration for creating or synchronizing Integrity items based on existing tasks in Microsoft Project, the time information included in the Microsoft Project date fields is not considered for the Integrity items after a synchronization.