PRIMAVERA

Gateway Developer's Guide
Release 15.1

March 2015

Contents

Overview

Primavera Gateway is an application that facilitates sharing and synchronizing project, resource, and other data between Primavera applications and enterprise applications.

Primavera Gateway uses and delivers *Provider* applications which channel data between a source application and a destination application, enabling you to combine management and scheduling functionality of Primavera applications with other enterprise software.

A Gateway *Event Provider* is an *optional* mechanism which works in conjunction with a Gateway provider to trigger synchronization jobs within Gateway based on the occurrence of specific events in a provider application, and keep two applications in sync.

Providers can reside on either side of a data flow connecting a source application with a destination application. Based on your organization's requirements for sharing data, you can create additional providers that can work with Primavera Gateway, enabling you to share and synchronize data between a source application and a destination application by using providers on either side of the data flow.

The *Gateway Developer's Guide* describes how to create additional providers and corresponding event providers for Primavera Gateway specific to your requirements to synchronize and share data with Primavera applications.

Primavera Providers

The following providers are delivered with Primavera Gateway to support integrations with Primavera applications:

- P6 provider

  The P6 provider enables you to share data with P6 EPPM. Primavera Gateway supports P6 EPPM integration with the Oracle Primavera Prime application and a Sample provider.
- Prime provider

  The Prime provider enables you to share data with the Oracle Primavera Prime application. Primavera Gateway supports Oracle Primavera Prime integrations with the P6 EPPM application.
- EnterpriseTrack provider

  The EnterpriseTrack provider enables you to share data with Oracle Instantis EnterpriseTrack application. Primavera Gateway supports Oracle Instantis EnterpriseTrack integration with a Sample provider.
- Unifier provider

  The Unifier provider enables you to share data with the Primavera Unifier application. Primavera Gateway supports Primavera Unifier integration with the P6 EPPM application.
- Sample provider

The Sample provider is a provider for demonstration purposes only. The purpose of the Sample provider is to illustrate how to use Primavera Gateway to synchronize data between a Primavera application and the Sample provider. Primavera Gateway supports a Sample provider integration with P6 EPPM and Oracle Instantis EnterpriseTrack applications.

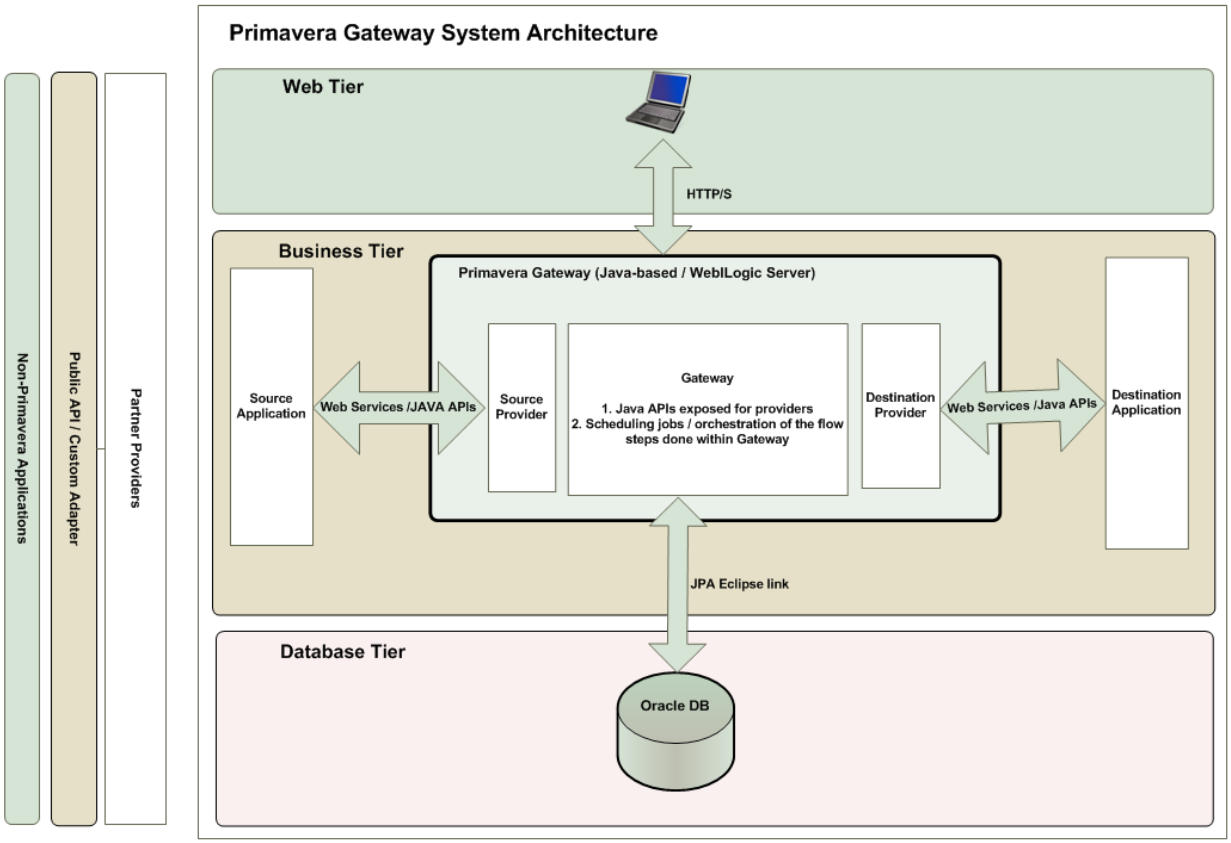
Sample Event Provider

The Sample event provider is an example of an event provider delivered with Primavera Gateway to demonstrate the use of an event listener with the Sample provider. This event provider triggers synchronization jobs in the Sample provider when an event is simulated by setting the event processed attribute to false.

In This Section

Primavera Gateway System Architecture

Primavera Gateway is a three-tier system that includes web, business, and database tiers. Each of these tiers provide specific functions to synchronize data between a source application and a destination application. The system architecture diagram below describes how these three tiers work together to synchronize data between applications.



The following table provides additional information about how the three tiers work together to synchronize data.

Tier	Description
1. Web Tier	This tier provides a browser-based user interface. You use this interface to create, view, schedule, and monitor business flows to synchronize data between a source application and a destination application.
2. Business Tier	This tier provides the provider logic that orchestrates the business flow steps and includes the Gateway Framework and the following providers: ▸ P6 ▸ Prime ▸ EnterpriseTrack ▸ Unifier ▸ Sample The Gateway framework is a web application that is deployed on a WebLogic application server and provides the following functions: ▸ Provides orchestration of the business flow steps within Primavera Gateway. ▸ Provides job scheduling services.
3. Database Tier	This tier stores the Primavera Gateway schema and data, which includes the following items: ▸ Cross references (Xref), data value mappings (DVM), flows, metadata (data dictionary), and customizations ▸ Audit information, logs, and intermediate artifacts ▸ Configuration settings ▸ Schedules

Minimum Requirements

To synchronize data using Primavera Gateway, the following requirements must be met:

- At least one application must be a Primavera application in the data flow.

- To develop a provider, you must be familiar with your application, the data that it supports, and how it relates to business objects in a Primavera application. You will need to analyze your business objects and then create or modify the XML files associated with the relevant provider application. Refer to the XML files of the Sample provider as an example to describe your business objects.

Getting Started

In This Section

Considerations for Creating Providers and Event Providers

Before creating a provider to work with Primavera Gateway, the following key decisions have to be made:

1) Identifying what data is to be shared and exchanged between the applications.
   - Use the Primavera Gateway user interface to review what data is available in a Primavera application that you want to integrate with. For a detailed listing of business objects supported by each Primavera provider, see the *Providers Reference Guide*.
   - Refer to your application's documentation to review what data is available for integration.
2) Identifying what business flows are supported in Primavera Gateway for each Primavera provider. For more details on supported flows, see ***Flow Types Supported for Primavera Providers*** (on page 16).
3) Identifying additional data elements that need to be supported and shared. This includes user-defined fields (UDFs), metadata, and field mappings.
4) For event providers, identify and define what type of events should trigger a corresponding synchronization job in Gateway for a provider application.

All of the above decisions help in determining your source and destination application for synchronizing data in Primavera Gateway.

Using the Sample Provider Code to Develop a Provider

Primavera Gateway includes a Sample provider that you can use to get started. The Sample provider contains the flows, XML files, and sample Java code that uses best practices to help you get started. When you install Primavera Gateway, you can choose to load the Sample provider containing seed data into the Gateway database to get started with your implementations. The data that comes with the Sample provider can be customized for your environment.

Components of a Provider

The following artifacts have to be created and packaged for a provider:

- Description XML Files

- Data definition file
- Provider description file
- Cross reference definition file
- Field mapping template definition file
- Flow definition template file
- Flow file
- Data value mapping (DVM) definition file
- Eventing definition file

> **Note:** The DVM definition file is optional and is only required if there are enumerated fields that require mapping.

- Java code, if any, that implements the provider is packaged in a .jar file.

Using the Sample Event Provider to Develop an Event Provider

Primavera Gateway includes a Sample event provider that you can use to get started. The Sample event provider contains the flows, and XML files, and sample Java code that uses best practices to help you get started.

The Sample event provider consists of a XML descriptor file and some Java code that implements Gateway event provider interface. Use the sample event provider as an example to write a Gateway event provider that satisfies your requirements. The sample event provider consists of

- XML descriptor file called SampleEventProvider.xml

  The XML descriptor file is located in the **SampleProvider/data/moredata** folder.
- Two Java classes
  - SampleEventProvider which implements EventProvider interface
  - SampleEventLoader class which handles reading events out of a event XML file and marking the events processed

  The two classes are included in the **sampleprovider.jar**.

For more details, see the chapter, ***Creating an Event Provider*** (on page 41).

Components of an Event Provider

A Gateway provider can be associated with of one or more event listeners, where each event listener can be configured separately and trigger a separate synchronization. Configuration parameters can be defined at the provider level or at the listener level. The configuration parameters are defined to either request user input or provide opportunities for users to influence how the system works.

For example, for a P6 event provider, you must define parameters at the provider level to determine how to communicate with the JMS queue. There can be one listener defined for project data and another listener defined for master data. For the project data listener, you can define a parameter for the objects that should be monitored by this listener.

> **Note:** Each event provider can be associated with only *one event provider instance* for each deployment of the same application. However, within the event provider instance, listener instances can be created for each listener.

The following artifacts have to be created and packaged for an event provider:

- Event Provider XML descriptor file
- Java class that implements EventProvider interface

Creating a Provider

Developing a provider for Primavera Gateway involves the following steps:

1) Determining what data should be exchanged between two applications
2) Determining what flows to support to integrate the data
3) Creating the Java provider code and a set of description XML files.
   The Java code supports the flows.
4) Packaging all of the artifacts together so that they can be installed
5) Installing and testing the provider

The following sections illustrate how to complete each task listed above to create a provider application in accordance to your organization's requirements that exchanges data using the Primavera Gateway framework.

In This Section

Determine What Data Should be Exchanged With a Primavera Application

Gather information about the business objects that the Primavera Gateway supports by following these steps to view the data dictionaries in the Primavera Gateway user interface.

To view the data dictionary for a Primavera application:

1) Log in to Primavera Gateway.
2) Select **Data Dictionary** and select any of the following options to view the corresponding business objects supported in Primavera Gateway:
   - EnterpriseTrack
   - Gateway
   - P6
   - Prime
   - Sample
   - Unifier

   For a detailed list of business objects supported by each Primavera provider, also see the *Providers Reference Guide*.

3) To view additional information about a specific business object, select the business object to view a list of fields that are associated with the selected business object.

Determine What Business Flows are Required for the Provider

Business flows define an end-to-end synchronization. Flows are uni-directional. A flow can define synchronization of data in one direction only. They copy data from a source application to a destination application. Determine what flows are required for your provider as follows:

1) Identify what business objects are supported in Gateway for a Primavera provider.
2) In your application, identify what business objects you wish to integrate with using Gateway.
3) Identify the flow types that are to be supported by your provider and specify these in the provider.xml file. The flow types supported are:
   - Import Master Data
   - Export Master Data
   - Import Project Data
   - Export Project Data
   - Master Data (for data integration between distinct P6 environments)
   - Project Data (for data integration between distinct P6 environments)
4) Define the role of the application with which data is to be integrated by specifying the `FlowSide` element as any of the following values:
   - `Source`
   - `Destination`

   For example, if the role of the P6 provider in the data flow is that of a Host, then set `FlowSide = Source` in the Provider description XML file.

Flow Types Supported for Primavera Providers

Primavera Gateway supports the following basic flow types:

- Import Master Data

  Define Master data flows to only import global data from a source application to a destination application.
- Export Master Data

  Define Master data flows to export a subset of global data from a source application to a destination application.
- Import Project Data

  Define project data flows to import project data from a source application to a destination application.
- Export Project Data

  Define project data flows to export project data from a source application to a destination application.
- Master Data

Use this flow type to export or import master data between distinct P6 environments only. For example, export master data from Testing environment to a Production environment of a P6 EPPM application.

- Project Data

  Use this flow type to export or import project data between distinct P6 environments only. For example, export project data from Testing environment to a Production environment of a P6 EPPM application.

The following table identifies the default flow types supported for each Primavera provider:

Flows	Etrack	P6	Prime	Sample	Unifier
Import Master Data		x	x	x	
Export Master Data		x	x	x	
Import Project Data	x	x	x	x	x
Export Project Data	x	x	x	x	x
Master Data		x			
Project Data		x			

Define the Flow Steps in a Flow

Flows contain one or more flow steps. Each flow is a distinct sequence of the types of flow steps listed below:

- Load

  This step loads the source data and passes it on to the next step.

- Convert

  This step converts the source data to the Gateway data structure and the destination data structure.

- Compare

  This step compares the source and destination data and identifies changed, deleted, and added objects so that the system can synchronize the data efficiently.

  To ensure that the data that follows the same structure when it is compared, both the source and the destination data must be converted to the Gateway format before it can be compared. When identifying objects that are deleted in the source data, the compare step uses the cross reference table to determine whether the data has been synchronized in the past, the presence of the data in the cross reference table indicates that the data has been synchronized in the past. During the Compare step, the system marks objects that have been deleted in the source data for deletion in the destination system only if the record is in the cross reference table and a delete parameter is associated with the flow.

This step compares the source and destination data and identifies the delta or differences for the next step.

> **Note:** Project flows support the Compare step. Master data flows do not support the Compare step.

- Review

  (Optional) This step enables you to review the source data before updating the data in the destination application.
- Update

  This step saves the data into the destination system.

Flow Sequence When Importing Master Data

The following sequence describes the typical flow used to import master data:

- **Load from Source**: This step loads the source data into the Gateway framework to process.
- **Convert to Gateway Format**: This step converts the source provider data into the Gateway format.
- **Convert to Destination Format:** This step converts the master data in the Gateway format to the destination format so that it can be saved into the destination application's database.
- **Review Data:** (Optional) This step enables you to review the source data before updating the data in the destination application.
- **Update Destination**: This step saves the data into the destination application's database.

Flow Sequence When Importing Project Data

The following sequence describes the typical flow that is used to import project data from the source application:

- **Load from Source**: This step loads source project data from the source application into the Gateway framework so that it can be processed.
- **Convert from Source to Gateway Format**: This step converts source project Data to the Gateway format. The converted data is used as the source data in the **Compare** step.
- **Load from Destination**: This step loads project data from the destination application into the Gateway framework so that it can be processed.
- **Convert from Destination to Gateway Format:** This step converts destination's project data to the Gateway format. The converted data is used for the **Compare** step.
- **Compare Project Data**: This step compares the converted source Gateway data with the destination Gateway data. This step compares each XML object and uses the following rules to determine how the data is synchronized in the **Update Destination** step.
  - If the data is in the source object but not in the destination object, the data is created in the dstination object during the **Update Destination** step.
  - If the data is in the destination object but not in the source node, the system performs the following steps:

    * Inspects the cross reference tables to determine whether the data has ever been synchronized.

* Determines whether the Delete parameter has been set for the flow.

If the data is in the cross-reference tables and the delete parameter has been set for the flow, the data is deleted from the destination during the **Update Destination** step. Otherwise, the data is not deleted in the **Update Destination** step.

- Objects that contain updated data are marked for synchronization.
- Objects that contain the same data in both the source and the destination nodes are ignored.

- **Convert to Destination Format:** This step converts project data in Gateway format marked for creation or synchronization into the destination format so that it can be saved into the destination application's database.
- **Review Data:** (Optional) This step enables you to review the source data before updating the data in the destination application.
- **Update Destination**: This step saves the data into the destination application. Any data that is marked for deletion is deleted from the destination application's database.

Flow Sequence When Exporting Project Data

The following list describes a typical flow that is used to export data from the source application to the destination application:

- **Load from Source**: This step loads project data from the source application into the Gateway framework so that it can be processed.
- **Convert from Source to Gateway Format:** This step converts the source project data to the Gateway format. The converted data is used as the source data in the **Compare** step.
- **Load from Destination**: This step loads the project data from the destination application into the Gateway framework so that it can be processed.
- **Convert from Destination to Gateway Format**: This step converts destination project data to the Gateway format. The converted data is used as the destination data in the **Compare** step.
- **Compare Project Data**: This step compares the source data in Gateway format with the destination data in the Gateway format. This step compares each object and uses the following rules to determine how the data is synchronized in the **Update Destination** step.
  - If the data is in the source object but not in the destination object, data is created in the destination node during the **Update Destination** step.
  - If the data is in the destination object but not in the source node, the system performs the following steps:

    * Inspects the cross-reference tables to determine whether the data has ever been synchronized.

    * Determines whether the Delete parameter has been set for the flow.

    If the data is in the cross-reference tables and the Delete parameter has been set for the flow, data is deleted from the destination during the **Update Destination** step. Otherwise, data is not deleted in the **Update Destination** step.
  - Objects that contain updated data are marked for synchronization.
  - Objects that contain the same data in both the source and the destination nodes are ignored.

- **Convert to Destination Format:** This step converts project data that is marked for creation or synchronization from the Gateway format to the destination format so that it can be saved in the destination application.
- **Review Data**: (Optional) This step enables you to review the source data before updating the data in the destination application.
- **Update Destination**: This step saves project data in the destination application's database. Any data that is marked for deletion is deleted from the destination application.

Flow Sequence When Exporting Master Data

The following list describes how the steps are organized in a typical flow that is used to export Master data from a source application to the destination application:

- **Load from Source:** This step loads the master data from the source application into the Gateway Framework so that it can be processed.
- **Convert to Gateway Format:** This step converts the source master data from the source format to the Gateway format.
- **Convert to Destination Format:** This step converts the master data from Gateway format to the destination format.
- **Review data:** (Optional) This step enables you to review the source data before updating the data in the destination application.
- **Update Destination:** This step saves the master data into the destination application's database.

Flow Sequence When Using the Compare Step

A flow that supports a **Compare** step loads the project from both sides, determines the delta between each side, and uses only the difference to synchronize the data during the final update.

> **Note:** The Compare step is used in the Import Project Data flow and the Export Project Data flows. The Import Master Data and the Export Master Data flows do *not* support the Compare step.

Unlike the normal flow that consists of four steps (load, convert to Gateway, convert from Gateway, and Update Destination), a flow that supports the Compare step includes the following additional steps:

- Load data from the other application
- Convert the data to the Gateway format
- Compare

The Compare step is supported by the Gateway framework code; providers do not have to implement it. Providers will need to implement the extra load and convert steps as these must be implemented by the the provider of the destination application. The destination provider must ask for the key of the project that is being loaded to the source side of the implementation when supporting the compare functionality.

Source Provider

For the Import Project Data flow, the source provider needs to communicate to the destination side which project it is loading when the Primavera Gateway loads the initial project data from the source side. To do that, the source provider needs to implement the **getProjectKeyForCompare** method in the **FlowProvider** interface.

Normally, a provider will determine which project it is to load from the filter or the parameters that users set in the Gateway user interface. The implementation of the method needs to return a Gateway side value of this project key.

The following is a sample code snippet from the Import Project Data flow in **SampleProvider.java**:

```
	@Override
	public Map<String, String> getProjectKeyForCompare(String flowType, FlowContext context)
throws ProviderException {
		SampleFlowType type = getFlowType(flowType);
		switch (type) {
		case SyncProjectImport:
			String sampleProjectKey = (String) context.getParameter("ImportProjectId");
			if (StringUtils.isEmpty(sampleProjectKey)) {
				return null;
			} else {
				Map<String, String> keyMap = new HashMap<String, String>();
				keyMap.put("ObjectId", context.getXRefValueByGuest("Project",
sampleProjectKey));
				keyMap.put("Id", sampleProjectKey);
				return keyMap;
			}
		default:
			throw new UnsupportedOperationException("Compare not supported.");
		}
	}
```

Destination Provider

For the Export Project Data flow, Primavera Gateway loads the initial project data from the source side. The destination provider needs to ask for the project key so that it can load the same project.

The LoadStepContext interface has two methods for this use case:

- **isLoadStepForCompare** method can tell you whether this load step is invoked as a companion load step for the Compare mechanism.
- **getProjectKeyForCompare** method can tell you which project you should load. The project key returned by **getProjectKeyForCompare** is already a destination side value.

The following is a code snippet from the Export Project Data flow in **ProjectLoadStep** of the Sample provider:

```
if (context.isLoadStepForCompare()) {
        Map<String, String> projectKeys = context.getProjectKeyForCompare();
                String projectId = null;
                if (projectKeys != null) {
                        String objectId = projectKeys.get("ObjectId");
                        if (StringUtils.isEmpty(objectId)) {
                                projectId = projectKeys.get("Id");
                        } else {
                                projectId = objectId;
                        }
                }
                if ((projectId == null) || projectId.isEmpty()) {
                        return new PDIDocumentImpl();
                } else {
                        return getOneProject(projectId, context);
                }
        }
```

Flow Sequence When Using Master Data

The Master data flow type is used to specifically to export or import master data between *two distinct environments* of the P6 application. For example, you would use this flow to transfer data from a *P6 Testing environment to P6 production environment*. The steps are organized as follows:

- **Load from Source:** This step loads the master data from the source application into the Gateway Framework so that it can be processed.
- **Convert from Source to Gateway Format:** This step converts the source master data from the source format to the Gateway format.
- **Load from Destination**: This step loads project data from the destination application into the Gateway framework so that it can be processed.
- **Convert from Destination to Gateway Format:** This step converts destination's master data to the Gateway format. The converted data is used for the **Compare** step.
- **Compare Data**: This step compares the converted source Gateway data with the destination Gateway data. This step compares each XML object and uses the following rules to determine how the data is synchronized in the **Update Destination** step.
  - If the data is in the source object but not in the destination object, the data is created in the destination object during the **Update Destination** step.
  - If the data is in the destination object but not in the source node, the system performs the following steps:

    * Inspects the cross reference tables to determine whether the data has ever been synchronized.

    * Determines whether the Delete parameter has been set for the flow.

    If the data is in the cross-reference tables and the delete parameter has been set for the flow, the data is deleted from the destination during the **Update Destination** step. Otherwise, the data is not deleted in the **Update Destination** step.
  - Objects that contain updated data are marked for synchronization.
  - Objects that contain the same data in both the source and the destination nodes are ignored.

- **Convert to Destination Format:** This step converts the master data from Gateway format to the destination format.
- **Review data:** (Optional) This step enables you to review the source data before updating the data in the destination application.
- **Update Destination:** This step saves the master data into the destination application's database.

Flow Sequence When Using Project Data

The Project data flow type is used to specifically to export or import project data between *two distinct environments* of the P6 application. For example, you would use this flow to transfer project data from a *P6 Testing environment to P6 production environment*. The steps are organized as follows:

- **Load from Source:** This step loads the master data from the source application into the Gateway framework so that it can be processed.
- **Convert from Source to Gateway Format:** This step converts the source project data from the source format to the Gateway format.
- **Load from Destination:** This step loads the project data from the destination application into the Gateway framework so that it can be processed.
- **Convert from Destination to Gateway Format:** This step converts the destination project data to the Gateway format.
- **Compare Project Data:** This steps compares the source data with the destination data in the current Gateway format.
- **Convert to Destination Format:** This step converts the project data from Gateway format to the destination format.
- **Review data:** (Optional) This step enables you to review the source data before updating the data in the destination application.
- **Update Destination:** This step saves the project data into the destination application's database.

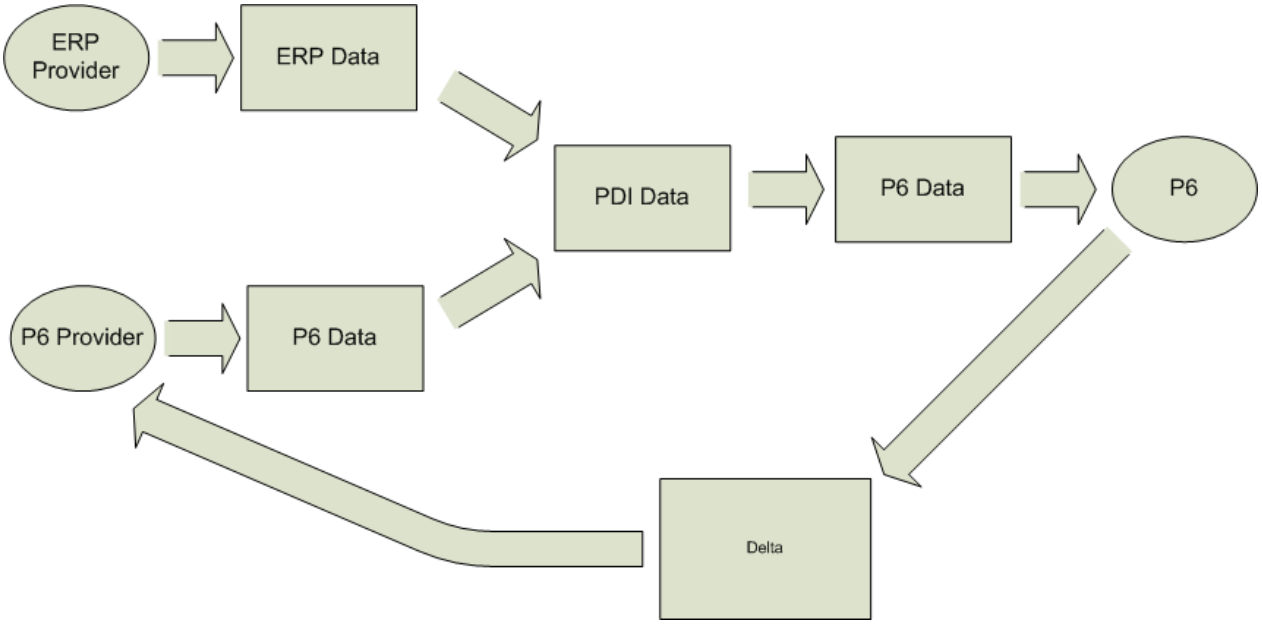
Examples of Sample Flows

There are two basic flow types. In the first type, all of the data is synchronized, regardless of whether the data has changed.

Flow Type 1: Synchronize all of the data in the flow

In the second type of flow, a compare step is introduced that synchronizes the delta.

Flow Type 2 – Synchronize Delta Information



Creating the Java Provider Code and the Description XML Files

To support the flows designed for your provider, create custom java code, custom java mapping and custom java steps.

Creating Java Provider Code

To create Java provider code, the following interfaces are supported:

- FlowProviderInterface
- FlowStepContextInterface

For additional information see the Java documentation with Primavera Gateway.

FlowProviderInterface

LoadStepExecutor

PDIDocument load(LoadStepContext context)

SaveStepExecutor

void save(SaveStepContext context, PDIDocument data)

ConvertStepExecutor

PDIDocument convert(ConvertStepContext context, PDIDocument data)

CompareStepExecutor

PDIDocument compare(CompareStepContext context, PDIDocument guestData, PDIDocument hostData)

FlowStepContext Interface

Accessing flow parameters

Map<String, Object> getFlowParameters()

Accessing data dictionary

```
String getFieldType(String objectName, String fieldName)
String getFieldCategory(String objectName, String fieldName)
String getTopic(String objectName, String fieldName)
```

Logging errors or payloads

```
void writeLog(Exception ex)
void writeMessage(String messageName, String message)
```

Accessing cross-reference

String getPDIXRefValue(String pdiObjectName, String hostValue)

Creating the Description XML files

To create the provider, add each object type to create a set of description XML files:

- **MetaData**:. To describe the business objects supported by the provider and the fields that these business objects contain.
- **MapTemplate**: To describe how your provider's business objects map to the Gateway and provider metadata files.
- **DVM**: To list the enumeration mappings for fields which have enumerated values.
- **Provider**: To define the following information for your provider:
  - Flows
  - Filters
  - Parameters
  - Class path to Java code that implements the provider

  For example, see the SampleProvider.xml file in the <Gateway_Home>\sample\sampleprovider\data folder. For additional information on content and format of the provider description XML file to create for your application, see Provider Files (*Provider.xml). The following example illustrates the code contained in the provider description XML file.
- **XRefData**: To describe the keys that are used to establish the links between the business objects.
- **ExtraMetaData**: To describe any UDF and Code fields that your application requires to be included in a Primavera provider, along with their corresponding fields in the Gateway format. A provider can contain more than one ExtraMetaData file.

> **Note**: Use the ExtraMetaData file to extend the Gateway data dictionary or the Primavera provider's data dictionary.

- **Flow**: This file contains custom flows and the flow steps in each flow that is supported by the provider.

The XML files use corresponding XSD files. For a detailed list of schema files and elements, see ***Appendix B - Primavera Gateway Schema Files*** (on page 55). You can also use the examples of data definition XML files created for the Sample provider as a starting point to develop your provider. The Sample provider data definition XML files are located in the **<Gateway_HOME>\sample\sampleprovider\data** folder.

Recommended Guidelines

When creating the provider, the following guidelines are recommended:

- Create the set of XML files for your provider in a separate **Data** folder.
- Never directly manipulate the XML files delivered in the **PDI Data** folder.

The XML files are used to load the seed data you create or customize into the Primavera Gateway database. The changes made in these XML files also directly impact the Gateway user interface elements. For example, if the MetaData.xml file now includes additional business objects, these business objects will be visible in the **Data Dictionary** menu of the relevant provider.

Supporting Filters in the Provider Description XML File

In Gateway, a filter is a mechanism for deciding what to load in a Load step. For example, in the Import Project Data flow and the Export Project Data, a filter would determine what projects to import or export. You can think of a filter as a special type of parameter. Filters are defined in a provider XML file and referenced in a load step via the LoadContext interface.

How Is Filter Defined

A filter is defined in a provider XML file much like normal parameters, except it has its own element tags for defining the structure of the filter. The following snippet from the SampleProvider.xml of the sample project illustrates how to define a filter:

```
<Parameter>
        <DefaultValue>Project:ImportProjectIds=E-1922,E-1833</DefaultValue>
        <Description>A comma separated list of project Ids</Description>
        <FilterOptions>
                <ObjectOptions>
                        <ObjectName>Project</ObjectName>
                        <Field>
                                <Name>ImportProjectIds</Name>
                                <DefaultValue>Comma Separated IDs</DefaultValue>
                        </Field>
                </ObjectOptions>
        </FilterOptions>
        <Name>ERPProjectFilter</Name>
        <Sequence>2</Sequence>
        <Title>ERP Project Filter</Title>
        <Type>Filter</Type>
</Parameter
```

As illustrated in the snippet above, the filter is defined by specifying a `Parameter` element that includes a `Type` element with a value of Filter. Additionally, the `FilterOptions` element defines the structure of the filter. There should be one `ObjectOptions` element for each object involved. Finally, the `Field` element includes elements that define the field name and a default value. There can be more than one `Field` element for each object.

Since a filter is just a parameter that has a data type of `Filter`, you can set its default value and attributes in much the same manner as these aspects are set for other parameters. When you create business flows in Primavera Gateway, you can set the default value of a filter and determine whether the filter is **hidden**, **optional**, **required**, or **read only** when the business flow is used in a synchronization. If you set the filter to **hidden** or **read only**, the value that you set for the filter can not be changed when the business flow is used in a synchronization. If you set the filter to **optional** or **required**, the value of the filter can be changed when the flow is used in a synchronization.

Setting Filters in Primavera Gateway

Follow the steps below to set the default value of a filter and determine whether the default value can be changed in the synchronization:

1) Log into Primavera Gateway with a user the has an Admin role.
2) Select **Flow Type** and choose a business flow type.
3) Select the **Business Flows** tab.
4) Select **Add** to create a new business flow or highlight an existing business flow and select **Edit**.
5) In the **General** tab, specify or change the information and click **Next**.
6) In the **Mappings** tab, click **Next**.
7) In the **Parameters** tab, choose a parameter that has a filter.

> **Note:** Filters are parameters that have a data type of Filter.

8) Build your criteria for the filter by adding and deleting rows.

9) In the **Attribute** column, depending on how you want the filter to be accessed, select **Hidden**, **Optional**, **Required**, or **Read only**.
10) Click **Save**.

Changing the Default Value of Filters

When you specify the attribute of a filter as **optional** or **required** in the business flow, you can override the default value when the business flow is used in a synchronization. Follow the steps below to override the default value of a filter.

1) Log into Primavera Gateway.
2) Select **Synchronizations.**
3) Select **Add** to create a new synchronization or highlight an existing synchronization and select **Edit**.
4) In the **Flow & Deployments** tab, specify or change the information and click **Next**.
5) In the **Parameters** tab, change your criteria for the filter by adding and deleting rows. and click **Next**.

> **Note:** Only filters that have been set as optional or required, or read only are displayed in this tab.

6) In the **Review & Submit** tab, click **Save**.

How to Refer to a Filter in the Provider Code

A filter can be referenced in a provider by calling the getParameter method from the FlowContext interface. Here is an example from SampleProvider.java of the sample provider project.

```
private static List<ChildFlowInfo> getChildFlowInfo(FlowContext context) throws ProviderException {
	@SuppressWarnings("unchecked")
	List<FilterValue> filters = (List<FilterValue>) context.getParameter("ERPProjectFilter");
	if ((filters == null) || (filters.size() == 0)) {
		return null;
	}
	String projectIds = null;
	for (FilterValue fv : filters) {
		String objectName = fv.getObjectName();
		// get Project specific filter information
		if ("Project".equals(objectName)) {
			if ("ImportProjectIds".equals(fv.getFieldName())) {
				projectIds = fv.getValue();
				break;
			}
		}
	}
	String[] projIds = projectIds.split(",");
	List<ChildFlowInfo> childInfoList = new ArrayList<ChildFlowInfo>();
	for (String id : projIds) {
		ChildFlowInfo childInfo1 = new ChildFlowInfo();
		childInfo1.setParameter("ImportProjectId", id);
		childInfoList.add(childInfo1);
	}
	return childInfoList;
}
```

You can see that the filter is retrieved by calling FlowContext.getParameter, and cast into a list of FilterValue. Each FilterValue is like a clause, which contains information like

object.field = some value

The net result of the filter is AND of all the clauses.

In this example, only Project object and ImportProjectIds field are defined. ImportProjectIds is simply a comma separated project ids. Each project id will be saved to the hidden ImportProjectId parameter, and a separate job will be spun off for synchronizing each project.

Example of a Provider Description XML File

The following is an example of a provider description XML file.

```
<Provider>
        <ApplicationName>P6</ApplicationName>
        <ClassPath>com.oracle.pgbu.pdi.p6provider.P6Provider</ClassPath>
        <FlowDefinition>
                <Name>Import Primavera Project Data</Name>
                <AppType>Host</AppType>
                <FlowBusinessObject>
                        <Name>Project</Name>
                </FlowBusinessObject>
                ...


...
                <Parameter>
                        <DefaultValue>Imported Projects</DefaultValue>
                        <Description>Destination location for synced projects</Description>
                        <Title>EPS Location</Title>
                        <Type>String</Type>
                </Parameter>
                ...
        </FlowDefinition>
        ...
</Provider>
```

EnterpriseTrack Provider XML Files

The following files are delivered for the EnterpriseTrack provider in the **etrackprovider\data** folder:

- EtrackDVM.xml: Lists the enumeration mappings for fields which have enumerated values in the EnterpriseTrack provider.
- EtrackMetaData.xml: Lists the business objects and the fields contained within each business object in the EnterpriseTrack provider.
- ETrackProvider.xml: Lists the flows and the flow steps contained in each flow that is supported in the EnterpriseTrack provider.

P6 Provider XML Files

The following files are delivered for the P6 provider in the **p6provider\data** folder:

- P6DVM.xml : Lists the enumeration mappings for fields which have enumerated values in the P6 provider.
- P6FieldMapTemplate.xml:
- P6MetaData.xml: Lists the business objects and the fields contained within each business object in the P6 provider.
- P6Provider.xml: Lists the flows and the flow steps contained in each flow that is supported by the P6 provider.
- P6XrefData.xml: Describes the cross-reference mappings supported in the P6 provider.

The following file is delivered in the **p6provider\data\moredata** folder:

- MasterDataBusinessFlow.xml: XML file used to demonstrate how to create business flow using the data loader.

PDI XML Files

The following files are delivered for Gateway objects in PDI format in the **pdi\data** folder:

- ExportGlobalData.xml: To export and synchronize global data from a source application to a destination application.
- ExportProjectData.xml: To export and synchronize project data from a source application to a destination application.
- ImportMasterData.xml: To import master data from a source application to a destination application.
- ImportProjectData.xml: To import project data from a source application to a destination application.
- MasterData.xml: To import master data from a source application to a destination application.
- PDIMetaData.xml: A list of Gateway business objects and the fields contained within each business object.
- ProjectData.xml: To import project data from a source application to a destination application.
- Settings.xml: Contains all the Gateway Settings.

Sample Provider XML Files

The **Sample** folder contains examples of XML files that have been created to demonstrate an integration between the Sample provider and P6. You can use these XML files as examples to build the necessary XML files for your provider.

The **\sample\sampleprovider\data** folder contains the following XML files:

- SampleDVM.xml: Describes a list of the enumeration mappings for fields which have enumerated values in the Sample provider.
- SampleMetaData.xml: Describes how the Sample provider's business objects maps to Gateway and the provider metadata files.
- SampleProvider.xml: Describes the flows and the flow steps that the Sample provider supports.

The **\sampleprovider\data\p6** folder contains the following XML files as examples that have been created to integrate the Sample provider with P6 data:

- P6ExtraMetaData.xml: Describes the P6 UDFs, Codes, etc. that have been added as metadata using the configuration utilities.
- PDIExtraMetaData.xml: Describes the Gateway UDFs, Codes, etc. in PDI format that have been added as metadata.
- SampleFieldMapTemplate.xml: Describes the field map templates supported in the Sample provider.
- SampleProviderCustomization.xml: Describes the customization mappings supported in the Sample provider.

- SampleXRefData.xml: Describes the cross-reference mappings supported in the Sample provider.

The **\sampleprovider\data\p6data** folder contains the SampleP6DataSetup.xml file, an example for how to configure P6 data. The P6 DataSetup utility uses this XML file to customize the data supported in your P6 application. The following data types can be specified in the SampleP6DataSetup.xml file:

- GlobalPreferences
- EPS
- UDFType
- NotebookTopic
- ProjectCodeType
- ResourceCodeType
- ActivityCodeType

The **\sampleprovider\data\etrack** folder contains the following XML files to demonstrate data integration between the Sample provider and Oracle Instantis EnterpriseTrack:

- ETrack_sample_FieldMap.xml: Describes the field mappings between the Sample provider and EnterpriseTrack provider.
- ETrack_Sample_XRefData.xml: Describes the cross-reference mappings between the Sample provider and EnterpriseTrack provider.

In the **\sampleprovider\data\moredata** folder, contains the following XML files:

- SampleFlows.xml: Describes the business flows and field mapping templates supported in the Sample provider.
- SampleEventProvider.xml: A example to illustrate how to develop an event provider for your provider application that works in Primavera Gateway.

Prime Provider XML Files

The following files are delivered for the Prime provider in the **prime\data** folder:

- P6ExtraDVM.xml: Describes additional data value mappings supported in the P6 provider.
- P6ExtraMetaData.xml: Describes the P6 UDFs, Codes, etc. that have been added as metadata using the configuration utilities.
- PDIExtraMetaData.xml: Describes the Gateway UDFs, Codes, etc. in PDI format that have been added as metadata.
- PrimeDVM.xml: Describes a list of the enumeration mappings for fields which have enumerated values in the Prime provider.
- PrimeFieldMapTemplate.xml: Describes the field map templates supported in the Prime provider.
- PrimeMetaData.xml: Lists the business objects and the fields contained within each business object in the Prime provider.
- PrimeProvider.xml: Lists the flows and the flow steps contained in each flow that is supported in the Prime provider.
- PrimeSampleFlows.xml: Describes the business flows and field mapping templates supported in the Prime provider.

- PrimeXRefData.xml: Describes the cross-reference mappings supported in the Prime provider.

The **\prime\data\p6data** folder contains the PrimeP6DataSetup.xml file. Use this XML file in the **Gateway-P6Setup.bat** or the **Gateway-P6Setup.sh** configuration utility to setup UDFs in the P6 EPPM database for an integration between Oracle Primavera Prime and P6 EPPM applications.

Unifier Provider XML Files

The following files are delivered for the Unifier provider in the **unifier\data** folder:

- PDIExtraMetaData.xml: Describes the Gateway UDFs, Codes, etc. in PDI format that have been added as metadata.
- UnifierDVM.xml: Describes a list of the enumeration mappings for fields which have enumerated values in the Unifier provider.
- UnifierFieldMapTemplate.xml: Describes the field map templates supported in the Unifier provider.
- UnifierMetaData.xml: Lists the business objects and the fields contained within each business object in the Unifier provider.
- UnifierProvider.xml: Lists the flows and the flow steps contained in each flow that is supported in the Unifier provider.
- UnifierSampleFlows.xml: Describes the business flows and field mapping templates supported in the Unifier provider.
- UnifierXRefData.xml: Describes the cross-reference mappings supported in the Unifier provider.

Packaging the Provider Artifacts

To load the provider files into the database and deploy in the WebLogic Primavera Gateway domain, create a top-level provider folder that contains the provider files. This folder must contain sub-folders with the following names:

- **data,** that contains the data definition and provider description XML files
- **lib,** that contains the provider jar files
- **lib/supplement**, that contains all other files required by the provider

Installing the Provider

After creating the provider folder, you will need to decide how to load the provider XML files into Primavera Gateway and deploy the jar files that contain the code that implements your provider application. Depending on the purpose select any of the following methods:

- Using the Primavera Gateway Installer

  Use this method if you are distributing the provider. Refer to the *Gateway Installation and Configuration Guide* for information on how to specify your provider folder during the install.
- Using the Primavera Gateway Configuration Utility

  Use this method if you are developing, testing, or upgrading the provider.

Loading Seed Data into the Database

After creating the necessary XML files for your provider, use Data Loader to load the XML files that you created for your provider with seed data into the Gateway database. When you install Primavera Gateway, the installer invokes the configuration utility that uses the XML files to load seed data into the Primavera Gateway database.

To customize or create your own provider, refer to the files in the P6, Gateway, and Schema folders. Use the Gateway Configuration Utility or upload your customization.xml from the **Configuration** page in the Gateway user interface. For more details, refer to the *Customization Guide*.

Configuring Primavera Gateway

The following configuration utilities are provided in Primavera Gateway. Run each utility for the purpose described below.

- **Gateway-P6Setup**: Run this utility to add User Defined Fields (UDFs) and codes to P6 EPPM.

  Access the Gateway-P6Setup utility from the **<Primavera_Gateway_Home>\pdi\snapshots\dbsetup-dist** folder.
- **Gateway.Configuration**: Run this utility to:
  - update, replace or customize Gateway metadata
  - add or remove providers, including Primavera providers

  Access the Gateway.Configuration utility from the **<Primavera_Gateway_Home>\pdi\snapshots\dbsetup-dist** folder.

> **Note:** Before executing the utilities on Windows or Linux ensure the following:
>
> - The supported Java JDK's bin folder is included in the PATH.
> - In the PATH it should be before the other JDK bin folders(if any).

Adding User-Defined Fields (UDFs) and Codes to P6 EPPM

You have the option to add User-Defined Fields (UDFs) and codes to P6 EPPM and to prepare the P6 EPPM database for synchronizing the data. Use this procedure to also add UDFs and codes from Oracle Primavera Prime, Primavera Unifier, to P6 EPPM.

For example, if your application has UDFs or codes that P6 EPPM does not have, you can add these UDFs or codes using an XML file that you edit and then point to it in the Gateway Configuration utility. The XML file needs to conform to the DataConfiguration.xsd schema. For more information on the DataConfiguration.xsd schema, see the *Primavera Gateway Developer's Guide*.

To add your application UDFs or codes to P6 EPPM:

1) Ensure that your **JAVA_HOME** variable is pointing to a supported JDK folder.

> **Note:** Ensure the following:

- The **bin** folder of the supported Java JDK is included in the PATH.
- If there are other JDK bin folders in the PATH, it should be listed first.

2) Navigate to the **<Primavera_Gateway_Home>/pdi/snapshots/dbsetup-dist folder**.
3) If you are installing on a non-Microsoft Windows system, type the following command to give execute permission for the utility:

   chmod 755 Gateway-P6Setup.sh
4) Run the following command:
   - For Windows installation, run **Gateway-P6Setup.bat**
   - For Linux and Solaris installations, run **./Gateway-P6Setup.sh**
5) Enter the following information in the **Primavera P6 Data Setup Utility** dialog box:
   a. In the **P6 Admin User Name** field, enter the user name of a user who has admin privileges on the P6 deployment.
   b. In the **P6 Admin Password** field, enter the password of the admin user.
   c. In the **P6 Adapter Endpoint URL** field, enter the address of the **SyncServiceV1 wsdl** file that is served by the P6 adapter you deployed as a prerequisite. This address would follow the format:

      <protocol>://<hostname>:<port number>/p6adapter/services/SyncServiceV1?wsdl
   d. In the **P6 database instance ID**, enter the system ID (SID) for the P6 database instance.
   e. In the **P6 Data XML File Path**, enter or select **Browse** to specify the path to the XML file that you modified with the UDFs or codes which you want to add to P6 EPPM.

      For a P6 EPPM with Sample provider integration, ensure you have installed the Sample provider, and now select the **SampleP6DataSetup.xml** in the **<Primavera_Gateway_Home>\sample\sampleprovider\data\p6data** folder. The path you enter must be the absolute path to the file.

      For a P6 EPPM with Oracle Primavera Prime integration, select the **PrimeP6DataSetup.xml** file in the **<Primavera_Gateway_Home>\primeprovider\data\p6data** folder. The path you enter must be the absolute path to the file.

      For a P6 EPPM with Primavera Unifier integration, locate the **UnifierP6DataSetup.xml** file for the Unifier provider. The path you enter must be the absolute path to the file.

      For a P6 EPPM with any third-party enterprise application integration, select the relevant **<third-party provider P6datasetup>XML** file from the **data** folder. The path you enter must be the absolute path to the file.
   f. Select **Run**. The UDFs or codes will be imported into the P6 EPPM deployment you selected.
   g. Select **Finish** to close the utility.

Adding, Updating, or Customizing Gateway Metadata

After modifying the P6ExtraMetaData and PDIExtraMetaData files, run the **Gateway-Configuration** utility to add your application's UDFs or Codes in the Gateway metadata. The utility is located in the following location:

- On Windows, go to **C:\<Primavera_Gateway_Home>\pdi\snapshots\dbsetup-dis**t

- On Linux and Solaris, go to **C:/<Primavera_Gateway_Home>/pdi/snapshots/dbsetup-dist**

1) Navigate to the **<Primavera_Gateway_Home>/pdi/snapshots/dbsetup-dist** folder.
2) If you are installing on a non-Microsoft Windows system, type the following command to give execute permission for the utility:

   chmod 755 Gateway-Configuration.sh
3) Run the following command:
   - For Windows installations, run **Gateway-Configuration.bat**
   - For Linux and Solaris installations, run **./Gateway-Configuration.sh**
4) In the **Primavera Gateway Configuration Utility** dialog box, enter the following information:
   a. Select **Manage Metadata**, and select **Next**.
   b. Select any of the following options to manage Gateway metadata and select **Next**.
      - **Replace Gateway Metadata**
      - **Update Gateway Metadata**
      - **Customize Gateway Metadata**

      Note: The **Customize Gateway Metadata** option only updates the Gateway database with metadata.
5) Based on the **Manage Metadata** option selected in the previous step, enter the following database connection details:
   a. In the **DBA User Name** field, enter the user name of the oracle database administrator.
   b. In the **DBA Password** field, enter the password of the oracle database administrator.
   c. In the **Database Host** field, enter the host name of the Oracle database on which you will be updating the Primavera Gateway database.
   d. In the **Database Host Port** field, enter the port number of the Oracle database.
   e. In the **Database Name** field, enter the Gateway database name and select any of the following methods to connect to the database.
      - In the **SID** field, enter the SID of the Oracle database.
      - In the **Service** field, enter the service name of the Oracle database.
   f. In the **Schema Owner** field, enter the schema owner name.
   g. In the **Schema Password** field, enter the password for the schema owner.
   h. In the **Provider data folder** field, verify the path name for the following, as applicable:
      - If you are using the Sample provider, then verify that this field contains the default Sample directory that contains all the Sample XML metadata/mapping templates.
      - If you chose to install additional providers, then verify that this field contains the home directory for each provider. The XML files in the associated **data** subdirectory must include all the required data to load metadata and mapping templates.

      For more information about these files, see the *Gateway Developer's Guide*.

      To add providers, select **Add** and locate the **data** folder for each provider you wish to add.

      To remove providers, select a provider and select **Remove**.

i. In the **Gateway Data Folder** field, verify the path name to the PDI data folder. For example, C:/PrimaveraGateway/pdi/data.

j. If you chose to **Customize Gateway Metadata**, then specify the XML file in the **Customization XML** field. Enter the path name or select **Browse** and locate the Customization.xml file for the specific provider.

> **Note:** This field displays only when you select the Customize Gateway Metadata option.

k. Select **Test Connection**. If the connection fails, modify the applicable fields and repeat as necessary.

l. Select **Run** to run the configuration utility.

Adding or Removing Providers

To add or remove providers, including Primavera providers, to Primavera Gateway:

1) Stop the Gateway domain before adding or removing providers or customizations.
2) Ensure the following:
   - The **bin** folder of the supported Java JDK is included in the PATH.
   - If there are other JDK bin folders in the PATH, they should be listed first.
3) Navigate to the **<Primavera_Gateway_Home>/pdi/snapshots/dbsetup-dist** folder.
4) If you are installing on a non-Microsoft Windows system, type the following command:
   chmod 755 Gateway-Configuration.sh
5) Run the following command:
   - For Windows installations, run **Gateway-Configuration.bat**
   - For Linux and Solaris installations, run **./Gateway-Configuration.sh**.
6) In the **Primavera Gateway Configuration Utility** dialog box, enter the following information:
   a. Select **Manage Providers**, and click **Next**.
   Selecting this option updates the pdi.ear file and the Gateway database with custom metadata from the XML files.
   b. In the **Select Gateway ear file (pdi.ear) location**, enter or click **Browse** to locate the .ear file.
   c. In the **Gateway domain location** field, enter or click **Browse** to specify the Gateway domain.
   d. In the **Installed Gateway Providers** field, review the list of providers displayed and perform any of the following actions:
      - To add a provider, click **Add Provider Location**, and navigate to the location of the provider.
      - To remove a provider listed in the Gateway user interface, select the provider and click **Remove**.
      - To delete a provider from the database and the pdi.ear file, select the **Delete** option and click **Remove**.
7) Enter the following database connection details:

a. In the **DBA User Name** field, enter the name of the database administrator.
b. In the **DBA Password** field, enter the password for the database administrator.
c. In the **Database Host** field, enter the host name of the Oracle database on which you will be updating the Primavera Gateway database.
d. In the **Database Host Port** field, enter or verify the port number of the Oracle database.
e. In the **Database Name** field, enter the Gateway database name and select any of the following methods to connect to the database.
    - In the **SID** field, enter the SID of the Oracle database.
    - In the **Service** field, enter the service name of the Oracle database.
f. In the **Schema Owner** field, enter the database user name to be used for the Primavera Gateway database. (This name should match the **Schema Owner** name when you installed Primavera Gateway.)
g. In the **Schema Password** field, enter the database password to be used for the Primavera Gateway database.
h. Select **Test Connection**. Modify the applicable fields if the connection fails and repeat as necessary.
i. Click **Update**. The status field displays a success message.
j. Click **Finish** to exit the configuration utility.

> **Note:** Redeploy pdi.ear to ensure the changes are reflected in Primavera Gateway.

Testing the Provider

To test your provider, set up an environment that includes the following components:

- The Primavera application that your provider is to connect to.

> **Note:** If you install P6 EPPM, then ensure that P6 EPPM, P6 Adapter and Primavera Gateway are installed and deployed on separate WebLogic domains.

- A provider folder that contains the following items:
    - **data** folder that contains the provider XML files
    - **lib** folder that contains the provider jar file
    - **lib/supplement** folder that contains all other files

Test your provider as follows:

1) Use the Primavera Gateway Configuration Utility to update or swap your provider with the files in your provider folder.
2) Log in to Primavera Gateway.
3) From the Primavera Gateway application, do the following:
    a. Create business flows.
    b. Create synchronizations

c. Run the flows
d. Monitor the flows

4) Check the results in the source and destination applications.

Creating an Event Provider

A Gateway event provider is an *optional* mechanism to enable external events to trigger Gateway synchronizations to run and keep two applications in sync.

For example, consider a project in P6 modified by a user. Having a P6 event provider would generate an event for the change in the P6 project and automatically run a Gateway synchronization job to push these changes to another application that is integrated with P6.

> **Note:** At run time, the project filter of the synchronization will be overwritten by the project information carried by the event, so that the right project would be pushed to the other side.

Developing an event provider for Primavera Gateway involves the following steps:

- Determining the type of event provider to create
- Creating the event provider java code and the event descriptor XML file
- Packaging the event provider

In This Section

Determining What Type of Event Provider to Create

There are two types of Gateway event providers.

- Polling type event provider

  A polling type event provider communicates with the event source to obtain the events when they happen. For example, a P6 event mechanism, where events are generated and sent to a JMS queue. To work with this, a P6 polling type event provider would have to implement the poll method. When the poll method is called at regular intervals by the Gateway integration broker, the JMS queue is polled to retrieve any available events.
- Receiving type event provider

The other Gateway event provider type is receiving type. In case of receiving type event provider, code must be written on the application side to send the event to Primavera Gateway through Gateway restful APIs. This type of event provider will *not* go and retrieve the events from the event source. It just receives the events that are passed by the Gateway integration broker, and processes them accordingly.

Creating the Event Provider Java Code and the Event Descriptor XML File

This section provides guidelines for using the Sample Event Provider source files delivered with Primavera Gateway.

Creating the Event Provider Java Code

A Gateway event provider must have a Java class that implements the EventProvider interface. Several interfaces and classes are related to an event provider. These include:

- EventProvider: The event provider interface that must be implemented by a Gateway event provider
- EventContext: This can be used to retrieve useful information about the event listener instance and the event provider instance, including values of configuration parameters at both the provider instance level and the listener instance level.
- EventBroker: The reference to the singleton event broker instance. This can be retrieved from the EventContext. It schedules the synchronization to run in the future according to EventTrigger passed in from the event provider, and handles the merging of the EventTrigger when necessary.
- EventTrigger - This contains information to identify a Gateway synchronization and the parameter values to overwrite. The event provider needs to translate an event to an EventTrigger, and pass to EventBroker to trigger synchronization to run.
- Event - This contains information about an event

If the event provider is a polling type, it should have real implementation for the "poll" method. If the event provider is a receiving type, it should implement the "receive" method instead.The sample event provider implements both poll and receive methods.

Using the Poll Method

The event provider poll method is called regularly by the event broker to check if there are new events. The poll method of the SampleEventProvider works as follows:

1) It first retrieves the EventFilePath parameter.
2) It uses the SampleEventLoader class to retrieve the next event if available.
3) It calls the fireEvent method to process the event.
   a. It first checks where there is a matching project field that is changed.
   b. It creates an EventTrigger with the proper project filter, and pass on to the event broker to process.

Using the Receive Method

The event provider Receive method is called by the event broker when an event arrives through the Gateway event API call. Since the event is already passed in, there is no need to retrieve the event any more. In the receive method of the sample event provider, it directly calls the fireEvent method.

Creating the Event Provider Descriptor XML File

Use the Sample Event Provider descriptor file as a template, and modify this file to suit your requirements. This XML descriptor file contains four sections:

- The **General** properties section contains several properties, such as the name, description, version, and application name to refer to the application, type and class path.
  - The type of event provider can be Polling or Receiving.
  - ClassPath is the package path of the event provider. The Gateway integration broker uses this information to load the event provider at run time using Java reflection mechanism.
- The **EventProviderConfigs** section defines event provider level configuration parameters.
- The **EventListeners** section defines event listeners for the event provider.
- The **EventProviderInstance** section describes information for an event provider instance.

> **Note:** Only *one* event provider instance must be specified for each application deployment.

  - The **AppInstanceName** points to the application deployment.
  - The **EventListenerInstance** section contains properties for an event listener instance.

    ListenerName points to the listener that this instance belongs to.

    FlowName and SynchronizationName in combination determines the synchronization to be run by the listener.

    DelayInMinutes describes the interval that the event broker will wait between when it receives the event and when the corresponding synchronization starts to run. During this time, if another event comes that would trigger the same synchronization with the same parameters (such as project filter), this event is merged together without scheduling another synchronization to run later. A synchronization in Gateway often loads the full project, so it could be expensive to run. This mechanism reduces the number of times that a synchronization would run with the cost of longer wait before a synchronization starts to run.

    Tip: If you notice that the Gateway is under heavy load, increment the value of this setting to make synchronizations wait longer and run less.

    ListerConfigs contains default values for the listener level configuration parameters.

Packaging the Event Provider

Ensure that the event provider is packaged within the Provider's jar file.

Appendix A - Sample Provider

In This Section

A Sample Provider

Primavera Gateway ships with a Sample provider to illustrate how to use Primavera Gateway to synchronize data between P6 EPPM and the Sample provider. In this case, the Sample provider plays the role of a Guest in all the flow types. The Sample provider demonstrates how to share data from the following objects:

Import Master Data: Demonstrates how to import the following master data from the Sample provider to P6:

- Role
- Role Rate
- Resource
- Resource Rate
- Expense Category
- Work Order Category
- Calendar
- Location
- Cost Account
- Financial Period
- UDF Type
- Project Code Type
- Project Code

Export Master Data: Demonstrates how to export the following master data from P6 to the Sample Provider:

- EPS
- Project Code
- Project Code Type
- Location
- UDF Type

Import Project Data: Demonstrates how to import the following project data from the Sample provider to P6:

- Calendar
- Project
- Operation
- Operation Expense
- Project Resource
- Relationship
- Resource
- Resource Assignment
- Work Order

Export Project Data: Demonstrates how to export the following project data from P6 to the Sample provider:

- Operations
- Operation Expense
- Project Resource
- Project
- Relationship
- Resource Assignment
- Work Order Spread
- Work Order Expense Spread
- Work Order Resource Spread
- Work Order

Example: SampleProvider XML Document

```
<?xml version="1.1" encoding="UTF-8"?>
<Provider xmlns="http://xmlns.oracle.com/Primavera/PDI/Provider/V1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.oracle.com/Primavera/PDI/Provider/V1    Provider.xsd">
        <ApplicationName>Sample</ApplicationName>
        <ClassPath>com.oracle.pgbu.pdi.sampleprovider.SampleProvider</ClassPath>
        <FlowDefinition>
                <Name>Import Project Data</Name>
                <AppType>Guest</AppType>
                <FlowBusinessObject>
                        <Name>Project</Name>
                </FlowBusinessObject>
        ...
                        <Parameter>
                        <DefaultValue/>
                        <Description>Place holder for project Id</Description>
                        <Name>ImportProjectId</Name>
                        <Sequence>1</Sequence>
                        <Title>Import Project Id</Title>
                        <Type>HiddenString</Type>
                </Parameter>
                <Parameter>
                        <DefaultValue>Project:ImportProjectIds=E-1922,E-1833</DefaultValue>
                        <Description>A comma separated list of project Ids</Description>
                        <FilterOptions>
                                <ObjectOptions>
                                        <ObjectName>Project</ObjectName>
                                        <Field>
                                                <Name>ImportProjectIds</Name>
                                                <DefaultValue>Comma Separated IDs</DefaultValue>
                                        </Field>
                                </ObjectOptions>
                        </FilterOptions>
                        <Name>ERPProjectFilter</Name>
                        <Sequence>2</Sequence>
                        <Title>ERP Project Filter</Title>
                        <Type>Filter</Type>
                </Parameter>
        </FlowDefinition>
        ....
</Provider>
```

Example: Assigning A Notebook Topic to an Activity

This topic presents an example that illustrates a technique for adding codes or user defined fields that have spaces in the field names. This technique involves adding a topic attribute to the Field element when you add the field to the ExtraMetaData.xml file.

> **Note**: The NotebookTopic must exist in P6 database. End user can use the data setup utility to create a NotebookTopic.

2. Add a FieldCategory for "Note" in P6MetaData.xml.

```
<FieldCategory>
		<Name>Note</Name>
		<Description>Maps to Project Note or Activity Note in P6 side</Description>
</FieldCategory>
```

3. Add a field under Activity in P6ExtraMetaData.xml, PDIExtraMetaData.xml and ERPMetaData.xml. For example, if I created or have a NotebookTopic with "Lessons Learned" as the name in P6 database:

The field element in P6ExtraMetaData.xml should be defined as follows:

> **Note**: The value of the "topic" attribute is the NotebookTopic name.

```
<Field category="Note" topic="Lessons Learned">
		<Description>An activity note field to store notes.</Description>
		<Name>LessonsLearned</Name>
		<Type>String</Type>
</Field>
```

The field element in PDIExtraMetaData.xml should be as follows:

```
<Field>
		<Description>An activity note field to store notes.</Description>
		<Name>LessonsLearned</Name>
		<Type>String</Type>
</Field>
```

The field element in ERPMetaData.xml should be as follows:

```
<Field>
	<Description>The long text value.</Description>
		<Name>LessonsLearned</Name>
		<Type>String</Type>
</Field>
```

Add a FieldMap element for Actiivty, in ERPFieldMapTemplate.xml file as follows:

```
<FieldMap>
		<Guest>LessonsLearned</Guest>
		<Host>LessonsLearned</Host>
		<PDI>LessonsLearned</PDI>
</FieldMap>
```

To assign a note to an Activity, add a "LessonsLearned" element as follow under the Task(Activity) element(refer to E-1922.xml file which is a sample input data for Import Project flow).

```
<Task>
        <ElementId>2056</ElementId>
        <Id>T-2056</Id>
        <ActivityShortText>Planning</ActivityShortText>
        <WBSElementId>E-1922-1</WBSElementId>
        <LessonsLearned>Add lessons learned notes to current activity.</LessonsLearned>
</Task>
```

The "<LessonsLearned>Add lessons learned notes to current activity.</LessonsLearned>" node will be converted to an ActivityNote object.

Example: Creating a Calendar Object

To import a Calendar object from your application into P6, you will need to create a calendar object along these lines:

```
<Calendar>
    <Name>IG Calendar</Name>
    <ElementId>1111</ElementId>
    <Type>Global</Type>
</Calendar>
```

```
<Calendar>
    <Name>IG CalendarPer1</Name>
    <ElementId>11</ElementId>
    <Type>Resource</Type>
    <IsPersonal>true</IsPersonal>
    <StandardWorkWeek>
        <StandardWorkHours>
            <DayOfWeek>Sunday</DayOfWeek>
            <WorkTime>
                <Finish>16:29:00</Finish>
                <Start>13:30:00</Start>
            </WorkTime>
            <WorkTime>
                <Finish>12:29:00</Finish>
                <Start>08:30:00</Start>
            </WorkTime>
        </StandardWorkHours>
        <StandardWorkHours>
            <DayOfWeek>Monday</DayOfWeek>
            <WorkTime>
                <Finish>16:29:00</Finish>
                <Start>08:30:00</Start>
            </WorkTime>
        </StandardWorkHours>
        <StandardWorkHours>
            <DayOfWeek>Tuesday</DayOfWeek>
            <WorkTime>
                <Finish>23:29:00</Finish>
                <Start>06:30:00</Start>
            </WorkTime>
        </StandardWorkHours>
        <StandardWorkHours>
            <DayOfWeek>Thursday</DayOfWeek>
            <WorkTime>
                <Finish>16:59:00</Finish>
                <Start>08:00:00</Start>
            </WorkTime>
        </StandardWorkHours>
        <StandardWorkHours>
            <DayOfWeek>Friday</DayOfWeek>
            <WorkTime>
                <Finish>16:59:00</Finish>
                <Start>00:30:00</Start>
            </WorkTime>
            <WorkTime>
                <Finish>19:29:00</Finish>
                <Start>17:30:00</Start>
            </WorkTime>
            <WorkTime>
```

```
            <Finish>23:29:00</Finish>
            <Start>20:30:00</Start>
         </WorkTime>
      </StandardWorkHours>
      <StandardWorkHours>
         <DayOfWeek>Saturday</DayOfWeek>
      </StandardWorkHours>
   </StandardWorkWeek>
   <HolidayOrExceptions>
      <HolidayOrException>
         <Date>2013-04-10T00:00:00</Date>
      </HolidayOrException>
      <HolidayOrException>
         <Date>2012-12-12T00:00:00</Date>
         <WorkTime>
            <Finish>04:29:00</Finish>
            <Start>02:30:00</Start>
         </WorkTime>
      </HolidayOrException>
   </HolidayOrExceptions>
</Calendar>
```

Appendix B - Sample Event Provider Descriptor File Notes

The **EventProviderInstance** section describes information for an event provider instance.

In the Sample event provider, the EventFilePath parameter is defined to point to the event source XML file. For demonstration purposes and for convenience, the Sample event provider retrieves events from an event source XML file, **SampleEvents.xml**, located in the ..\<Gateway_Home> sample\sampleprovider\src\com\oracle\pgbu\pdi\sampleevent folder.

To run the Sample event provider:

1. Copy the **SampleEvents.xml** file to your hard drive.
2. Modify the value of this configuration parameter in the user interface to point to it.
3. Change the **Processed** attribute to "**false**".

The Sample event provider will pick up this event for processing and modify the **Processed** attribute to "**true**" for this event in the file.

Also refer to the *Gateway User Help* for detailed instructions on using the Sample Event Listener.

The **EventListeners** section defines event listeners for the event provider.

The Sample provider has one listener called **ProjectEventListener**, which monitors for project changes.

This listener contains one configuration parameter called **ProjectFields**. Enter a comma-separated list of project field names for this parameter. This indicates the project fields which this listener is interested in. When the **ProjectFieldsChanged** element of an event in **SampleEvents.xml**, contains a field from the comma-separated list, this event will be processed. Otherwise, this event will be filtered out.

Appendix B - Primavera Gateway Schema Files

The **Schema** folder contains a set of XSD files delivered with Primavera Gateway. These XSD files are used by the XML files of each Primavera provider application. The following schema files are included:

- Dvm.xsd
- EventProvider.xsd
- FieldMapTemplate.xsd
- FlowDefinition.xsd
- Flow.xsd
- MetaData.xsd
- Provider.xsd
- XRef.xsd
- DataConfiguration.xsd

> **Note:** In the above XSD files, a few schema elements can be marked as deprecated. A deprecated field refers to those schema elements that are currently retained to ensure backward compatibility, but will be removed in next product release.

In This Section

Data Value Mapping Files (*DVM.xml)

The data value mapping files list the enumeration mappings for fields that have enumerated values.

The P6DVM.xml file located in the data folder, which can be used as a reference, maps the P6 field enumerations to the Gateway field enumerations. The sampleprovider/data folder contains a SampleDVM.xml file that provides an example use of this file to map the Sample provider field enumerations to the corresponding Gateway field enumerations.

Schema file

Dvm.xsd

Contents

A **ValueMaps** element that contains zero to many **ValueMap** elements. Each **ValueMap** element can contain the following elements:

Element	Type	Parents	Description
AppName	string restricted to maxLength(60)	ValueMap	The element that indicates the name of the application for which this mapping applies. This name must match the name of an App that is defined in a *MetaData.xml file.. For example, specify P6 as the content of the AppName element to indicate that the mapping applies to P6.
PDIObjectName	string restricted to maxLength(60)	ValueMap	The element that indicates the name of the PDI object for which this mapping applies.
PDIFieldName	string restricted to maxLength(60)	ValueMap	The element that indicates the name of the PDI field for which this mapping applies.
AppObjectName	string restricted to maxLength(60)	ValueMap	The element that indicates the name of the object for which this mapping applies. The object that is specified in this element must reside in the application that is specified by the AppName.
AppFieldName	string restricted to maxLength(60)	ValueMap	The element that indicates the name of the field for which this mapping applies. The element that is specified in this element must reside in the application that is specified by the AppName.

DirectCopy	boolean	ValueMap	The indicator that specifies whether to copy the field directly. Setting this element to true causes the system to copy the field without referring to the enumeration mappings. This element should only be set to true when the values are exactly the same in both side
Item	ItemType. See the **ItemType** table below.	ValueMap	The element that maps each AppFieldName field enumeration to its corresponding PDI enumeration. There can be zero to many item elements.

ItemType Table

Element	Type	Parent	Description
PDIValue	string restricted to maxLength(60)	Item	The element that provides the enumerated value in the PDI. This element is required if the parent element is present.
AppValue	string restricted to maxLength(60)	Item	The element that provides the enumerated value in the application that is specified by the AppName. This element is required if the parent element is present.

The following is an example of a ValueMap:

```
<ValueMap>
        <AppName>Sample</AppName>
        <PDIObjectName>Project</PDIObjectName>
        <PDIFieldName>Status</PDIFieldName>
        <AppObjectName>Project</AppObjectName>
        <AppFieldName>Status</AppFieldName>
        <Item>
                <PDIValue>Planned</PDIValue>
                <AppValue>SamplePlanned</AppValue>
        </Item>
        <Item>
                <PDIValue>Active</PDIValue>
                <AppValue>SampleActive</AppValue>
        </Item>
        <Item default="true">
                <PDIValue>Inactive</PDIValue>
                <AppValue>SampleInactive</AppValue>
        </Item>
        <Item>
                <PDIValue>What-If</PDIValue>
                <AppValue>SampleInactive</AppValue>
        </Item>
        <Item>
                <PDIValue>Requested</PDIValue>
                <AppValue>SampleInactive</AppValue>
        </Item>
</ValueMap>
```

Default Attribute

The Item element can include an optional default attribute. In the example above, Inactive, What-If, and Requested are all mapped to SampleInactive. In this case, when going from the PDI application to the Sample application, there is no ambiguity since there is a many to one relationship between the mappings. However, when going from the Sample application to the PDI, there is a one to many relationship between the mappings. In this case, the system resolves the ambiguity by mapping all of the Sample application SampleInactive enumerations for the Status field to the PDI Inactive enumeration.

Event Provider Files (*EventProvider.xml)

Purpose

The Event Provider file provide the format for designing an event provider for a provider application to generate event-based synchronizations in Primavera Gateway.

Schema file

EventProvider.xsd

Contents

An **EventProvider** element that contains that contains the following elements:

Element	Type	Parents	Description
Description	string restricted to maxLength(255)	EventProvider	A description of the event provider.
Name	string restricted to maxLength(60)	EventProvider	The name of the event provider.
ApplicationName	string restricted to maxLength(60)	EventProvider	The name of the application associated with the event provider.
Version	string restricted to maxLength(60)	EventProvider	The version number of the event provider.
Type	string restricted to maxLength(10)	EventProvider	The type of event provider. Choices include: Polling, Receiving.
ClassPath	string restricted to maxLength(255)	EventProvider	The class path for classes and other resource files used the Java interfaces for the event provider.
EventProviderConfigs	EventProviderConfigsType	EventProvider	The element that contains one or more event provider configurations for the event provider.
EventListeners	EventListenersType	EventProvider	The element that contains one or more event listeners associated with the event provider.
EventProviderInstance	EventProviderInstanceType	EventProvider	The element that contains one or more event provider instances associated with the event provider.

EventProviderConfigsType Table

Element	Type	Parents	Description
Config	EventConfigType	EventProviderConfigs	The element that contains one or more event provider configurations of the event provider.

EventConfigType Table

Element	Type	Parents	Description
DefaultValue	string restricted to maxLength(255)	Config	The default value of the event provider configuration.
Description	string restricted to maxLength(255)	Config	A description of the event provider configuration.
Name	string restricted to maxLength(60)	Config	The name of the event provider configuration.
Sequence	Integer	Config	The sequence number of the configuration. The sequence number determines the order in which the configuration parameters are displayed.
Title	string restricted to maxLength(60)	Config	The title of the event provider configuration.
Type	string restricted to maxLength(10)	Config	The type of configuration parameter.

EventListenersType Table

Element	Type	Parents	Description
EventListener	EventListenerType	EventProvider	The element that contains one or more event listeners associated with the event provider.

EventListenerType Table

Element	Type	Parents	Description
Description	string restricted to maxLength(255)	EventListener	A description of the event provider.

Element	Type	Parents	Description
Name	string restricted to maxLength(60)	EventListener	The name of the event provider.
EventListenerConfigs	EventListenerConfigsType	EventListener	The element that contains one or more event listener configurations associated with the event provider.

EventListenerConfigsType Table

Element	Type	Parents	Description
Config	EventConfigType	EventListenerConfigs	The element that contains one or more configuration parameters for each event listener configuration.

EventProviderInstanceType Table

Element	Type	Parents	Description
AppInstance	string restricted to maxLength(60)	EventProviderInstance	The application instance associated with the event provider.
Description	string restricted to maxLength(255)	EventProviderInstance	A description of the event provider instance.
ProviderConfigs	ProviderConfigsType	EventProviderInstance	The element that contains one or more event provider configurations associated with an event provider instance.
EventListenerInstances	EventListenerInstancesType	EventProviderInstance	The element that contains one or more event listener instances associated with an event provider instance.

ProviderConfigsType Table

Element	Type	Parents	Description
Config	EventInstanceConfigType	ProviderConfigs	The element that contains one or more configuration parameters associated with a provider configuration.

EventInstanceConfigType Table

Element	Type	Parents	Description
Name	string restricted to maxLength(60)	Config	The name of the configuration parameter.
Value	string restricted to maxLength(255)	Config	The value of the configuration parameter.

EventListenerInstancesType Table

Element	Type	Parents	Description
EventListenerInstance	EventListenerInstanceType	EventListenerInstances	The event listener instance associated with an event provider.

EventListenerInstanceType Table

Element	Type	Parents	Description
DelayInMinutes	integer	EventListenerInstance	The time duration between when an event is received by event provider and a synchronization job is triggered in the Gateway user interface.
Name	string restricted to maxLength(60)	EventListenerInstance	The name of the event listener instance.

Element	Type	Parents	Description
ListenerConfigs	ListenerConfigsType	EventListenerInstance	The element that contains one or more event listener configuration parameters associated with an event listener instance.
FlowName	string restricted to maxLength(60)	EventListenerInstance	The flow name associated with the event listener instance.
SynchronizationName	string restricted to maxLength(60)	EventListenerInstance	The synchronization name associated with the event listener instance.

ListenerConfigsType Table

Element	Type	Parents	Description
Config	EventInstanceConfigType	ListenerConfigs	The element that contains one or more event listener configuration parameters associated with an event listener configuration.

FieldMapTemplate Files (*FieldMapTemplate.XML)

Purpose

The FieldMapTemplate files defines the field mapping templates used by two provider applications.

Schema File

FieldMapTemplate.xsd

Contents

A **FieldMapTemplates** element that contains the following elements:

Element	Type	Parents	Description
GuestAppName	string restricted to maxLength(60)	FieldMapTemplates	The element that specifies the name of the guest application. The name should match the appropriate /MetaData/App/Name that is specified in the *MetaData.xml file
HostAppName	string restricted to maxLength(60)	FieldMapTemplates	The name of the host application. The name should match the appropriate /MetaData/App/Name that is specified in the *MetaData.xml file.
App1Name	string restricted to maxLength(60)	FieldMapTemplates	The name of the application that can be used as a host or a guest application. The name should match the appropriate /MetaData/App/Name that is specified in the *MetaData.xml file.
App2Name	string restricted to maxLength(60)	FieldMapTemplates	The name of the application that can be used as a host or a guest application. The name should match the appropriate /MetaData/App/Name that is specified in the *MetaData.xml file.
FieldMapTemplate	FieldMapTemplateType See the FieldMapTemplate table	FieldMapTemplates	The name of the field map template to be used for the two applications.

FieldMapTemplateType Table

Element	Type	Parents	Description
Description	string restricted to maxLength(255)	FieldMapTemplate	A description of the field map template.

Element	Type	Parents	Description
App1BusinessObjectName	string restricted to maxLength(60)	FieldMapTemplate	The business object name in the guest application if different from the Gateway business object name.
App2BusinessObjectName	string restricted to maxLength(60)	FieldMapTemplate	The business object name in the host application if different from the Gateway business object name.
GuestBusinessObjectName	string restricted to maxLength(60)	FieldMapTemplate	The business object name in the guest application if different from the Gateway business object name.
HostBusinessObjectName	string restricted to maxLength(60)	FieldMapTemplate	The business object name in the host application if different from the Gateway business object name.
Name	string restricted to maxLength(60) Minimum of 1.	FieldMapTemplate	The name of the field map template.
PDIBusinessObjectName	string restricted to maxLength(60) Minimum of 1.	FieldMapTemplate	The name of the business object in the Gateway format.
FieldMap	FieldMapType	FieldMapTemplate	The name of the field map defined in the field map template. A field map template can contain multiple field maps.

FieldMapType Table

Element	Type	Parents	Description
App1	string restricted to maxLength(60)	FieldMap	The name of the application designated as the guest.
App2	string restricted to maxLength(60)	FieldMap	The name of the application designated as the host.

Element	Type	Parents	Description
Guest	string restricted to maxLength(60)	FieldMap	The name of the field map in the guest application.
Host	string restricted to maxLength(60)	FieldMap	The name of the field map in the host application.
PDI	string restricted to maxLength(60)	FieldMap	The name of the field map in the Gateway format.

Flow Files (*Flow.XML)

The flow file defines the flows applicable to a provider application. Each flow can contain one or more business flows and synchronizations.

Schema File

Flow.xsd

Contents

Contains one or more **BusinessFlow** and **Synchronization** elements.

Element	Type	Parents	Description
BusinessFlow	BusinessFlowType (See BusinessFlowType table)	Flows	Zero to many elements containing child elements that describe the structure of a BusinessFlow.
Synchronization	SynchronizationType (See SynchronizationType table)	Flows	Zero to many elements containing child elements that describe the structure of a Synchronization.

BusinessFlowType Table

Element	Type	Parents	Description
Name	string	BusinessFlow	Element that contains the name of the business flow.
Description	string restricted to Maxlength (255)	BusinessFlow	Element that contains a description of the business flow
FlowDefinitionName	string restricted to MaxLength(60)	BusinessFlow	Element that contains the name of the Flow Definition.
SourceAppName	string restricted to MaxLength (60)	BusinessFlow	Element that contains the name of the source application.
DestinationAppName	string restricted to MaxLength (60)	BusinessFlow	Element that contains the name of the destination application.
PDIBusinessObjects	PDIBusinessObjectType (See the PDIBusinessObjectType table)	BusinessFlow	Container label for one or more PDI business objects.
FieldMapTemplates	FieldMapTemplatesType (See the FieldMapTemplatesType table)	BusinessFlow	Container label for one or more field mapping templates.
JavaCustomizationMappings	JavaCustomizationMappingsType (See the JavaCustomizationMappingsType table)	BusinessFlow	Container label for one or more java custom mappings.
Parameters	FlowParametersType (See the FlowParametersType table)	BusinessFlow	Container label for one or more parameters associated with a business flow.

PDIBusinessObjectsType Table

Element	Type	Parents	Description
Object	ObjectType (See ObjectType table)	PDIBusinessObjects	Container label for the names of the business object.

ObjectType Table

Element	Type	Parents	Description
Object	string restricted to maxLength(60)	Objects	Name of the business object.

FieldMapTemplatesType Table

Element	Type	Parents	Description
FieldMapTemplate	FieldMapTemplateType max unbounded. (See the FieldMapTemplateType table)	FieldMapTemplates	Container label for one or more names of FieldMapTemplate.

FieldMapTemplateType Table

Element	Type	Parents	Description
Name	String restricted to maxLength(60)	FieldMapTemplate	Name of the FieldMapTemplate.

JavaCustomizationMappingsType Table

Element	Type	Parents	Description
FieldMapping	JavaCustomizationMappingType (See the JavaCustomizationMappingType table)	JavaCustomizationMappings	Container label for one or more FieldMappings

JavaCustomizationMappingType Table

Element	Type	Parents	Description
Name	String restricted to maxLength(60)	FieldMapping	Name of the FieldMapping.

FlowParametersType Table

Element	Type	Parents	Description
Parameter	FlowParameterType (See the FlowParameterType table)	Parameters	Container label for one or more parameters.

FlowParameterType Table

Element	Type	Parents	Description
Parameter	FlowParameterType (See the FlowParameterType table) max unbounded	Parameters	Container label for one or more parameters.

FlowParameterType Table

Element	Type	Parents	Description
Name	string restricted to maxLength(60) max unbounded	Parameter	The name of the parameter.
Attribute	string enumeration values include: "Optional" Hidden Read only Required	Parameter	Attributes of each parameter.
DefaultValue	string restricted to maxLength(4000)	Parameter	The default value of the parameter.

SynchronizationType Table

Element	Type	Parents	Description
Name	string restricted to maxLength(60) max unbounded	Synchronization	The name of the synchronization.
BussinessFlowName	string restricted to maxLength(60)	Synchronization	The name of the Business Flow associated with the synchronization.
SourceAppInstance	string restricted to maxLength(60)	Synchronization	The instance of the source application.
DestinationAppInstance	string restricted to maxLength(60)	Synchronization	The instance of the destination application.
Parameters	SynchronizationParametersType (See the SynchronizationParametersType table)	Synchronization	Container label for one or more synchronization parameters.

SynchronizationParametersType Table

Element	Type	Parents	Description
Parameter	SynchronizationParameterType (See the SynchronizationParameterType table) max unbounded	Parameters	Container label for one or more more parameters associated with a specific synchronization.

SynchronizationParameterType Table

Element	Type	Parents	Description
Name	string restricted to maxLength(60)	Parameter	The name of the parameter associated with a specific synchronization.
Value	string restricted to maxLength(4000)	Parameter	The value of each parameter.

FlowDefinition Files (*FlowDefinition.XML)

The FlowDefinition file defines the direction of the data flow, the business objects in the data flow, and the parameters to be specified when exchanging data between two provider applications.

Schema File

FlowDefinition.xsd

Contents

One or more **FlowDefinitions** element of the type, FlowDefinitionType, which contain the following elements:

FlowDefinitionType Table

Element	Type	Parents	Description
Description	FlowDefinitionType	FlowDefinition	A short description of the flow definition defined in the xml file.
Direction	string GuestToHost HostToGuest	FlowDefinition	A container for the element describing the direction of the business flow.
Name	String restricted to MaxLength(60)	FlowDefinition	The name of the flow definition.
Priority	int	FlowDefinition	A container for the element to identify the priority of the business flow.
Type	String restricted to MaxLength(20)	FlowDefinition	A container for the element that identifies the type of business flow.
AppType	String restricted to MaxLength(10) PDI	FlowDefinition	A container for the element that identifies the AppType associated with the business flow.
FlowBusiness Object	FlowBusinessObjectType See the FlowBusinessObjectType table.	FlowDefinition	One or more container for the business objects associated with each business flows.
FlowStep	FlowStepType See the FlowStep table.	FlowDefinition	One or more flowsteps associated with the business flow.
Parameter	ParameterType See the ParameterType table.	FlowDefinition	One or more parameters associated with the business flow.

FlowBusinessObjectType Table

Element	Type	Parents	Description
Name	String restricted to MaxLength(60)	FlowBusinessObject	A container for the name of the business object included in the flow.

FlowStepType Table

Element	Type	Parents	Description
Description	String restricted to MaxLength(255)	FlowStep	A container for the element that describes the flow step.
Name	String restricted to MaxLength(60)	FlowStep	A container for the element for the name of the flow step.
Sequence	int	FlowStep	A container for the element for the sequence number of the flow step.
Type	string restricted to MaxLength(20) Compare Custom ConvertToPDI Review ConvertFromPDI Load Save	FlowStep	A container for the element the identifies the type of flow step.
OwnerAppType	string restricted to MaxLength(10) Guest Host PDI	FlowStep	The element that the system uses to determine which document format to use when parsing XML document data. For example, specifying the OwnerAppType as Host and P6 is the host of the flow causes the system to ask the P6 provider to provide the implementation of this step.

ParameterType Table

Element	Type	Parents	Description
DefaultValue	string restricted to MaxLength(4000)	Parameter	A container for the default value of the parameter.
Description	String restricted to MaxLength(255)	Parameter	A container for the description of the parameter.
EnumerationOptions	EnumerationOptionsType	Parameter	A container for the element that contains the list of enumerated values when the Parameter element is specified as an enum.
Name	String restricted to MaxLength(60)	Parameter	A container for the name of the parameter.
Sequence	int	Parameter	A container to list the order of the parameters in a sequence.
Title	String restricted to MaxLength(255)	Parameter	A container for the title of the parameter.
Type	String restricted to MaxLength(15) Boolean DateTime Double Int String Enum Custom Filter HiddenString	Parameter	A container for the data type of the parameter.

EnumerationOptionsType Table

Element	Type	Parents	Description
Enumeration	EnumerationType	Enumeration Options	A container for the element that specifies the name of the enumeration.

EnumerationType Table

Element	Type	Parents	Description
Name	String	Enumeration	A container for the element that specifies the name of the enumeration.

Data Dictionary Files (*MetaData.xml)

Purpose

The data dictionary files provide the format for the data that can be synchronized by Primavera Gateway.

Schema file

MetaData.xsd

Contents

A **MetaData** or an **ExtraMetaData** element that contains zero to many **App** elements each of the type AppType. For more details see the AppType table below:

AppType Table

Each AppType element contains the following child elements:

Element	Type	Parent	Description
Description	string restricted to maxLength(255)	App	The optional element that describes the data dictionary. This description is displayed in the Summary area of the Data Dictionary tab in the Primavera Gateway user interface.

Name	string restricted to maxLength(60)	App	The optional element that provides the name of the data dictionary. The content of this element is listed in the Summary area of the Data Dictionary tab in the Primavera Gateway user interface.
Type	string restricted to 'Host' 'Guest' 'PDI'	App	The optional element that determines the type of data dictionary. The content of this element is listed in the Summary area of the Data Dictionary tab in the Primavera Gateway user interface. This field is deprecated to currently support backward compatibility, and will be removed in a future release.
Version	string restricted to maxLength(60)	App	The optional element that provides the version of the data dictionary. The content of this element is listed in the Summary area of the Data Dictionary tab in the Primavera Gateway user interface.
ObjectCategory	ObjectCategoryType See the ObjectCategoryType table below.	App	The element that determines the type of object.
FieldCategory	FieldCategoryType See the FieldCategoryType table below.	App	Zero to many elements containing child elements that define the structure of the fields. For example, a Project business object might contain a field that is named ID that has a data type of String.
AppConfigs	AppConfigsType See the AppConfigType table below.	App	Zero to many elements containing child elements that describe the structure of the deployments.
AppInstance	AppInstanceType See the AppInstanceType table below.	App	Zero to many elements containing child elements that describe the instantiation of a deployment.

BusinessObject	BusinessObjectType See the BusinessObjectType table below.	App	Zero to many elements containing child elements that define the business objects. Examples of business objects include Projects, Resources, and Activities. Business object elements can contain zero or more Field elements. Elements in this node are included in the Data Dictionary tab of the Primavera Gateway user interface.

FieldCategoryType Table

Element	Type	Parent	Description
Name	string restricted to maxLength(60)	FieldCategory	The element that provides the name of the field category.
Description	string restricted to maxLength(255)	FieldCategory	The element that describes the purpose of the field category.

ObjectCategoryType Table

Element	Type	Parent	Description
Name	string restricted to maxLength(60)	ObjectCategory	The element that provides the name of the object category.
Description	string restricted to maxLength(255)	ObjectCategory	The element that describes the purpose of the object category.

AppConfigType Table

Each AppConfigType element contains the following child elements:

Element	Type	Parent	Description
Name	string restricted to maxLength(60)	AppConfigs	The element that defines the name that is used to look up the deployment in deployment code.

Title	string restricted to maxLength(60)	AppConfigs	The element that provides the title text that is used in the Primavera Gateway user interface.
SequenceNo	int	AppConfigs	The element that controls the ordering of the item in the Primavera Gateway user interface.
Type	string restricted to maxLength(10)	AppConfigs	The element that defines the data type of the configuration item.
Description	string restricted to maxLength(255)	AppConfigs	The element that describes the configuration item.
DefaultValue	string restricted to maxLength(255)	AppConfigs	The value that appears in the Primavera Gateway user interface and is used as the default value configuration item if no other input is provided.

AppInstanceType Table

Element	Type	Parent	Description
Description	string restricted to maxLength(255)	AppInstance	The element that describes the instantiated deployment.
Name	string restricted to maxLength(60)	AppInstance	The name of the instantiated deployment.
Configs	ConfigsType	AppInstance	Zero to many elements containing child elements that describe the structure of instantiated deployments.

ConfigsType Table

Element	Type	Parent	Description
Name	string	MetaData	The element that provides the name of the configuration item.

ConfigType Table

Element	Type	Parent	Description
Name	string	Configs	The element that provides the name of the configuration item.
Value	string	Configs	The element that provides the value of the configuration item.

BusinessObjectType Table

Element	Type	Parent	Description
Description	string restricted to maxLength(255)	BusinessObject	The optional element that describes the business object. This description is presented in the Data Dictionary tab in the Primavera Gateway user interface.
Name	string restricted to maxLength(60)	BusinessObject	The optional element that determines the name of the Business object. This name is presented in the Data Dictionary tab in the Primavera Gateway user interface.
Field	FieldType See the FieldType table below.	BusinessObject	The zero to many elements that contain the fields that are related to the business object.

FieldType Tables

FieldType Attribute Table

Attribute	Type	Element	Description
category	string	Field	The optional attribute that relates the field to a particular FieldCategory element.

topic	string	Field	The optional attribute used to specify an alternate name for the field. An example use is for specifying field name that contains spaces to be used as an alternate for the same field name without spaces.
required	boolean	Field	The optional attribute that determines whether the field must be synchronized.

FieldType Table

Element	Type	Parent	Description
Description	string restricted to maxLength(255)	Field	The optional element that describes the field. This description is presented in the Fields table on the Data Dictionary tab in the Primavera Gateway user interface.
Name	string restricted to maxLength(60)	Field	The optional element that determines the name of the field. This name is presented in the Fields table of the Data Dictionary tab in the Primavera Gateway user interface.
JoinTo	string restricted to maxLength(60)	Field	The element that contains the object that this field is joining to. This field is used when the field type is ForeignKey. For example, in P6 Data Dictionary, the WBSObjectId field in Activity joins to WBS object.

Type	string restricted to maxLength(32) Boolean DateTime Double Int String ForeignKey Password Enum StandardWorkWeek HolidayOrExceptions	Field	The optional element that defines the data type of the value of the field's data when it is synchronized.
MaxLength	positiveInteger	Field	The optional element that defines the maximum length of the value of the field's data when it is synchronized. This element can be used with the **Type** element whose content is String to limit the field's data values.
MaxValue	double	Field	The optional element that defines the maximum value of the field's data when it is synchronized. This element can be used with the **Type** element whose content is Int to limit the field's data values.
MinValue	double	Field	The optional element that defines the minimum value of the field's data when it is synchronized. This element can be used with the **Type** element whose content is Int to limit the field's data values.
FieldValue	FieldValueType For more details, see the FieldValueType table below.	Field	An optional element that defines the value of an enum field type. Field elements can contain zero or more FieldValue elements if the Field element contains a **Type** element whose content is Enum.

FieldValueType Table

Element	Type	Parent	Description
Description	string restricted to maxLength(255)	FieldValue	An optional element that describes the fieldValue.
Value	string restricted to maxLength(60)	FieldValue	An optional element that defines an enumerated value.

Provider Files (*Provider.xml)

Purpose

The provider files defines how the data will be exchanged between a source application and a destination application by defining the following information for each application:

- Flows
- Parameters
- Class path to Java code that implements the provider

Schema file

Provider.xsd

Contents

A Provider element containing the following elements:

Element	Type	Parent	Description
Name	string restricted to maxLength(60)	Provider	The name of the provider application.
ApplicationName	string restricted to maxLength(60)	Provider	The name of the application that is associated with this file.
Version	string restricted to maxLength(60)	Provider	The application version.
ClassPath	string restricted to maxLength(255)	Provider	The path to the Java class that implements the flow provider interfaces.

FlowDefinition	FlowDefinitionType	Provider	The container for the elements that describe the business objects that can be synchronized, the flows that the provider supports and the parameters that are associated with the flow. The information in the FlowDefinition element provides the ability to filter and limit what data is exchanged by the Primavera Gateway and corresponds to the flows in the Gateway user interface.
Settings	SettingsType	Provider	The container for the elements that define global settings that will show up in the Settings dialog box. When a flow runs, these settings will show up just as parameter of a flow type would.

SettingsType Table

Element	Type	Parent	Description
Parameter	ParameterType	SettingsType	The name of the parameter.

FlowDefinitionType Table

Element	Type	Parent	Description
Name	string restricted to maxLength(60)	FlowDefinition	The name of the business flow.
AppType	string restricted to maxLength(10)	FlowDefinition	The container for the element that determines whether a provider will show up in the Source side or Destination side. To maintain compatibility with version 1.0, defaults to Guest for the 1.0 provider files.

DisableCompare	boolean	FlowDefinition	The container for the element that determines whether the Compare step is to be applied or skipped.
FlowBusinessObject	FlowBusinessObjectType	FlowDefinition	The container for the element that defines the name of one of the business objects that can be included in the flow.
FlowStep	FlowStepType	FlowDefinition	The container for the element that defines the name of one of the individual steps in the flow which this provider supports or for which it provides an implementation.
Parameter	ParameterType	FlowDefinition	The container for the elements that defines the parameters that are associated with the business objects and flows.

FlowBusinessObjectType Table

Element	Type	Parent	Description
Name	string restricted to maxLength(60)	FlowBusinessObject	The name of the business flow

FlowStepType Table

Element	Type	Parent	Description
Name	string restricted to maxLength(60)	FlowStep	The name of the flow step in the business flow.

ParameterType Table

Element	Type	Parent	Description
DefaultValue	string restricted to maxLength(255)	Parameter	The default value for this custom parameter that is used when the parameter is included in the flow and the value is not changed.

Description	string restricted to maxLength(255)	Parameter	The element that provides a description of the parameter.
EnumerationOptions	EnumerationOptionsType	Parameter	The element that contains the enumerated options.
FilterOptions	FilterOptionsType	Parameter	The element that contains the filter options for the parameter.
Name	string restricted to maxLength(60)	Parameter	The element that provides the name of the parameter. The name is used to lookup the parameter in the provider Java code.
Sequence	int	Parameter	The element that provides the mechanism for ordering the parameters.
Title	string restricted to maxLength(255)	Parameter	The element that provides the display title of the parameter.
Type	string restricted to maxLength(15) Boolean DateTime Double Int String Password Enum Custom Filter HiddenString Group	Parameter	The element that defines the data type of the value of the field's data when it is synchronized.

EnumerationOptionsType Table

Element	Type	Parent	Description
Enumeration	EnumerationType	EnumerationOptions	The element that specifies the name of the enumeration.

EnumerationType Table

Element	Type	Parent	Description
Name	String	EnumerationType	The element that specifies the name of the enumeration.

FilterOptionsType Table

Element	Type	Parent	Description
ObjectOptions	ObjectOptionsType	FilterOptions	The element that contains the object options.

ObjectOptionsType Table

Element	Type	Parent	Description
ObjectName	string	ObjectOptions	The element that specifies the name of the object.
Field	FilterFieldType	ObjectOptions	The element that specifies the database field corresponding to the object.

FilterFieldType Table

Element	Type	Parent	Description
Name	string	Field	The element that specifies the name of the filter.
DefaultValue	string	Field	The element that specifies the default value of the filter.

XRef Files (*xref.xml)

Purpose

This file is used to describe the keys that are used to establish the links between the business objects.

Schema file

XRef.xsd

Contents

An XRefs element containing the following elements:

Element	Type	Parents	Description
XRefMap	XRefMapType	XRefs	The container element for the map.

XRefMapType Table

Element	Type	Parents	Description
GuestAppName	string	XRefMap	The element that specifies the guest application name. The name should match the appropriate /MetaData/App/Name that is specified in the *MetaData.xml file
HostAppName	string	XRefMap	The element that specifies the host application name. The name should match the appropriate /MetaData/App/Name that is specified in the *MetaData.xml file
App1Name	string	XRefMap	The element that specifies the application name which can be used as a source or a destination application.
App2Name	string	XRefMap	The element that specifies the application name which can be used as a source or a destination application.
XRefObject	XRefObjectType	XRefMap	The container element for elements that identify the object and the identifiers.

XRefMapType Table

Element	Type	Parents	Description
Name	string	XRefObject	The element that identifies the default name that is used when the GuestName, PDIName, or

			HostName is not specified.
App1Name	string	XRefObject	The element that specifies the application name which can be used as a source or a destination application.
GuestName	string	XRefObject	The Guest business object name if different from the value specified by the Name element.
PDIName	string	XRefObject	The PDI business object name if different from the value specified by the Name element.
App2Name	string	XRefObject	The element that specifies the application name which can be used as a source or a destination application.
HostName	string	XRefObject	The Host business object name if different from the value specified by the Name element.
App1PrimaryKeyFieldName	string	XRefObject	The Primary Key field name in the application designated in the App1 element.
App2PrimaryKeyFieldName	string	XRefObject	The Primary Key field name in the application designated in the App2 element.
GuestPrimaryKeyFieldName	string	XRefObject	The element that specifies the name of the Guest field to be used to find the matching field.
HostPrimaryKeyFieldName	string	XRefObject	The element that specifies the name of the Host field to be used to find the matching field.
PDIPrimaryKeyFieldName	string	XRefObject	The element that specifies the name of the PDI field to be used to find the matching field.

DataConfiguration File (*dataConfiguration.XML)

The data configuration file defines and identifies the data that is to be configured in the P6 application.

Schema File

DataConfiguration.xsd

Contents

Contains one DataConfiguration element which can contain one or more of the following elements:

Element	Type	Parents	Description
GlobalPreferences	GlobalPreferencesType (See GlobalPreferencesType table below)	DataConfiguration	Zero to many elements containing child elements that describe the structure of the GlobalPreference.
UDFType	UDFTypeType (See UDFTypeType table below)	DataConfiguration	Zero to many elements containing child elements that describe the structure of the UDFType.
NotebookTopic	NotebookTopicType (See NotebookTopicType table below)	DataConfiguration	Zero to many elements containing child elements that describe the structure of the NotebookTopic.
ProjectCodeType	ProjectCodeTypeType (See ProjectCodeTypeType table below)	DataConfiguration	Zero to many elements containing child elements that describe the structure of the ProjectCodeType.
ResourceCodeType	ResourceCodeTypeType (See ResourceCodeTypeType table below)	DataConfiguration	Zero to many elements containing child elements that describe the structure of the ResourceCodeType.
ActivityCodeType	ActivityCodeTypeType (See ActivityCodeTypeType table below)	DataConfiguration	Zero to many elements containing child elements that describe the structure of the ActivityCodeType.
EPS	EPSType (See EPSType table below)	DataConfiguration	Zero to many elements containing child elements that describe the structure of the EPSe.

ActivityCodeTypeType Table

Element	Type	Parents	Description
EPSObjectId	int	ActivityCodeType	The element that describes the unique

Element	Type	Parents	Description
			identifier for the EPS.
Length	int restricted to maxLength(60)	ActivityCodeType	The maximum number of characters allowed for values of this activity code.
Name	string restricted to maxLength(40)	ActivityCodeType	The name of the Activity Code.
ProjectObjectId	int	ActivityCodeType	The element that describes the unique identifier for the project.
Scope	String restricted to "Global" "EPS" "Project"	ActivityCodeType	The scope of the code type: Global, EPS, or Project. An activity code with Global scope can be assigned to any activity. An activity code with EPS scope can be assigned only to an activity within a project under that particular EPS. Similarly, an activity code with Project scope can be assigned only to an activity within that particular project.

GlobalPreferencesType Table

Element	Type	Parents	Description
MaxActivityCodeTreeLevells	int restricted to maxInclusive(25)	GlobalPreferences	The maximum number of levels that can be created in activity code hierarchies in the Project Management application.

Element	Type	Parents	Description
MaxActivityCodesPerProject	int	GlobalPreferences	The maximum number of project-level activity user codes that can be created per project.
MaxActivityIdLength	int restricted to maxInclusive(40)	GlobalPreferences	The maximum number of characters allowed for activity IDs
MaxBaselinesPerProject	int restricted to minInclusive(1)	GlobalPreferences	The element that describes the maximum number of baselines that can be created per project.
MaxCostAccountLength	int restricted to maxInclusive(40)	GlobalPreferences	The maximum number of characters allowed for cost account IDs (at each level in the cost account tree).
MaxCostAccountTreeLevels	int restricted to maxInclusive25)	GlobalPreferences	The maximum number of levels that can be created in the cost account hierarchy in the Project Management application.
MaxOBSTreeLevels	int restricted to maxInclusive(25)	GlobalPreferences	The element that describes the maximum number of levels that can be created in OBS hierarchies in the Project Management application.
MaxProjectCodeTreeLevels	int restricted to maxInclusive(25)	GlobalPreferences	The maximum number of levels in the project category hierarchy in the Project Management application.
MaxProjectIdLength	int restricted to maxInclusive(40)	GlobalPreferences	The element that describes the maximum number characters allowed for project IDs.
MaxResourceCodeTreeLevels	int restricted to maxInclusive(25)	GlobalPreferences	The maximum number of levels in the resource code hierarchy in the Project Management application.

Element	Type	Parents	Description
MaxResourceIdLength	int restricted to maxInclusive(255)	GlobalPreferences	The maximum number of characters allowed for resource IDs (at each level in the resource tree).
MaxResourceTreeLevels	int restricted to maxInclusive(25)	GlobalPreferences	The maximum number of levels that can be created in the resource hierarchy.
MaxRoleIdLength	int restricted to maxInclusive(40)	GlobalPreferences	The maximum number characters allowed for role IDs.
MaxRoleTreeLevels	int restricted to maxInclusive(25)	GlobalPreferences	The maximum number of levels in the role hierarchy in the Project Management application.
MaxWBSCodeLength	int restricted to maxInclusive(40)	GlobalPreferences	The maximum number of characters allowed for WBS codes (at each level in the WBS tree).
MaxWBSTreeLevels	int restricted to maxInclusive(50)	GlobalPreferences	The maximum number of levels that can be created in WBS hierarchies.

NotebookTopicType Table

Element	Type	Parents	Description
AvailableForActivity	boolean	NotebookTopic	The flag indicating that the topic will be available to assign to activities.
AvailableForEPS	boolean	NotebookTopic	The flag indicating that the topic will be available to assign to EPS.
AvailableForProject	boolean	NotebookTopic	The flag indicating that the topic will be available to assign to projects.
AvailableForWBS	boolean	NotebookTopic	The flag indicating that the topic will be available to assign to WBS.

Element	Type	Parents	Description
Name	string restricted to maxLength(40)	NotebookTopic	The name of the notebook topic.

ProjectCodeTypeType Table

Element	Type	Parents	Description
Length	int max inclusive (32)	ProjectCodeType	The maximum number of characters allowed for values of this project code
Name	string restricted to maxLength(40)	ProjectCodeType	The name of the project code.

ResourceCodeTypeType Table

Element	Type	Parents	Description
Length	int restricted to maxIinclusive (32)	ResourceCodeType	The maximum number of characters allowed for values of this resource code
Name	string restricted to maxLength(40)	ResourceCodeType	The name of the resource code.

UDFTypeType Table

Element	Type	Parents	Description
DataType	string restricted to "Text" "Start Date" "Finish Date" "Cost" "Double" "Integer" "Indicator" "Code"	UDFType	The data type of the user-defined field.
SiubjectArea	string restricted to "Activity" "Activity Expense" "Activity Step" "Project" "Project Issue" "Project Risk" "Resource" "Resource Assignment" "WBS" "Work Products and Documents" "Activity Step Template Item"	UDFType	The subject area of the user-defined field.
Title	string restricted to maxLength(40)	UDFType	The name/title of the user-defined field.

EPSType Table

Element	Type	Parents	Description
Id	string restricted to maxLength(40)	EPS	The unique identifier for the EPS.
Name	string restricted to maxLength(100)	EPS	The name of the EPS.

For More Information

In This Section

Where to Get Documentation

Complete documentation libraries for Primavera Gateway releases are available on the Oracle Technology Network (OTN) at:

http://www.oracle.com/technetwork/documentation/default-1923957.html

From this location you can either view libraries online or download them to have local copies. We recommend viewing them from OTN to ensure you always access the latest versions, including critical corrections and enhancements.

Primavera Gateway is configured to access its help system on OTN. However, you can also install a local version when you install the software.

The documentation assumes a standard setup of the product, with full access rights to all features and functions.

The following table describes the core documents available for Primavera Gateway and lists the recommended readers by role.

Title	Description
Gateway Help	Describes how to work with Primavera Gateway and provides information to help users accomplish tasks. All users should read the Help.
Gateway Developer's Guide	Provides information on how third-party systems such as enterprise resource management (ERP) and enterprise asset management (EAM) systems can create their own providers in order to integrate with Primavera products. Developers of third-party providers that integrate with Primavera products via Primavera Gateway should read this book.

Title	Description
Gateway Customization Guide	Provides information on how to customize an existing third-party integration. Developers interested in customizing existing third-party providers that integrate with Primavera products via Primavera Gateway should read this book.
Gateway Provider Reference Guide	Provides a list of the business objects available for each supported provider. Developers of third-party providers that integrate with Primavera products via Primavera Gateway should read this book.
EBS Provider Reference Guide	Provides a list of the business objects available for the EBS provider. Developers of third-party providers that integrate with Primavera products via Primavera Gateway should read this book.
VCP Provider Reference Guide	Provides a list of the business objects available for the VCP provider. Developers of third-party providers that integrate with Primavera products via Primavera Gateway should read this book.
Manual Deployment Guide	Provides information on how to manually install and configure Primavera Gateway. The Primavera Gateway network administrator/database administrator and the administrator for the third-party or ERP system should read this guide.
Gateway Installation and Configuration Guide	Provides information on how to install and configure Primavera Gateway. Primavera Gateway is a product that facilitates integrations with Primavera products and third-party systems such as enterprise resource management (ERP) and enterprise asset management (EAM) systems. The Primavera Gateway network administrator/database administrator and the administrator for the third-party or ERP system should read this guide.
Gateway Upgrade Guide	Provides a sequence of procedures that must be completed to upgrade to a new version of Primavera Gateway. The Primavera Gateway network administrator/database administrator and the administrator for the third-party or ERP system should read this guide.

Title	Description
Gateway Performance and Sizing Guide	Provides hardware and software requirements for deploying Primavera Gateway. The Primavera Gateway network administrator/database administrator and the administrator for the third-party or ERP system should read this guide.
Gateway Security Guide	Provides guidelines on establishing a highly secure environment for all Primavera Gateway environments. The Primavera Gateway network administrator/database administrator and the administrator for the third-party or ERP system should read this guide.
Gateway API Programmer's Guide	Provides instructions on how to access and use Primavera Gateway REST APIs. The Primavera Gateway network administrator/database administrator and Primavera Gateway users having the Gateway Developer role should read this guide.
Connecting with Instantis EnterpriseTrack	Provides instructions on how to setup the integration environment for Oracle Instantis EnterpriseTrack in Primavera Gateway. The Primavera Gateway network administrator/database administrator and the administrator for the third-party system should read this guide.
Connecting Prime and P6 EPPM	Provides instructions on how to setup the integration environment between Oracle Primavera Prime and P6 Enterprise Project Portfolio Management in Primavera Gateway. The Primavera Gateway network administrator/database administrator and the administrator for the third-party system should read this guide.
Connecting Unifier and P6 EPPM	Provides instructions on how to setup the integration environment between Oracle Primavera Unifier and P6 Enterprise Project Portfolio Management in Primavera Gateway. The Primavera Gateway network administrator/database administrator and the administrator for the third-party system should read this guide.
Migrating P6 Data Between Distinct Environments	Provides instructions on how to setup the integration environment between distinct P6 deployments to transfer P6 data in Primavera Gateway. The Primavera Gateway network administrator/database administrator and the administrator should read this guide.

Title	Description
Connecting E-Business Suite	Provides instructions on how to setup the integration environments for Oracle E-Business Suite with P6 Enterprise Project Portfolio Management and Instantis EnterpriseTrack in Primavera Gateway. The Primavera Gateway network administrator/database administrator and the administrator for the third-party system should read this guide.
Connecting Value Chain Planning and P6 EPPM	Provides instructions on how to setup the integration environment between Oracle Value Chain Planning and P6 Enterprise Project Portfolio Management in Primavera Gateway. The Primavera Gateway network administrator/database administrator and the administrator for the third-party system should read this guide.
Configuring Gateway for Single Sign On	Provides instructions on how to configure Oracle Access Manager (OAM) and then enable Single Sign On for Primavera Gateway. The Primavera Gateway network administrator/database administrator should read this guide.
Gateway Licensing Information User Manual	Lists licensing information of all third-party software that is used or associated with the Oracle software program.
Tested Configurations	Lists the configurations that have been tested and verified to work with Primavera Gateway. The Primavera Gateway network administrator/database administrator and the administrator for the third-party or ERP system should read this guide.

Distributing Information to the Team

You can copy the online documentation to a network drive for access by project participants. Each team member can then view or print those portions that specifically relate to his or her role in the organization.

Throughout this documentation, the Security Guidance icon helps you to quickly identify security-related content to consider during the installation and configuration process.

Where to Get Training

To access comprehensive training for all Primavera products, go to:

http://education.oracle.com

Oracle Learning Library

The Oracle Learning Library (OLL) provides online learning content covering Primavera products. Content includes videos, tutorials, articles, demos, step-by-step instructions to accomplish specific tasks, and self-paced interactive learning modules.

To access the learning library's Primavera content, go to:

http://www.oracle.com/goto/oll

Where to Get Support

If you have a question about using Oracle products that you or your network administrator cannot resolve with information in the documentation or help, click http://support.oracle.com/. This page provides the latest information on contacting Oracle Global Customer Support, knowledge articles, and the support renewals process. For more information about working with Support, visit https://support.oracle.com/epmos/faces/DocumentDisplay?id=888813.2 to view Support Tools & Tips.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/us/support/contact-068555.html or visit http://www.oracle.com/us/corporate/accessibility/support/index.html if you are hearing impaired.

Using Primavera's Support Resource Centers

Primavera's Support Resource Center provides links to important support and product information. Primavera's Product Information Centers (PICs) organize documents found on My Oracle Support (MOS), providing quick access to product and version specific information such as important knowledge documents, Release Value Propositions, and Oracle University training. PICs also offer documentation on Lifetime Management, from planning to installs, upgrades, and maintenance.

Visit https://support.oracle.com/epmos/faces/DocumentDisplay?id=1486951.1 to access links to all of the current PICs.

PICs also provide access to:

- **Communities** are moderated by Oracle providing a place for collaboration among industry peers to share best practices.
- **News** from our development and strategy groups.
- **Education** contains a list of available Primavera product trainings through Oracle University. The Oracle Advisor Webcast program brings interactive expertise straight to the desktop using Oracle Web Conferencing technology. This capability brings you and Oracle experts together to access information about support services, products, technologies, best practices, and more.

For more information about working with Support, visit https://support.oracle.com/epmos/faces/DocumentDisplay?id=888813.2.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Legal Notices

Oracle Primavera Prime Gateway Developer's Guide