Oracle Primavera Gateway

ORACLE
PRIMAVERA

Disclaimer

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Contents

What Is Primavera Gateway

Oracle Primavera Gateway allows data to be moved between two applications (at least one of them is a Primavera application) on schedule or on demand. It is a single integration hub where all the data integration happens with Primavera applications. As of right now, there are multiple ways how these integrations happen between Primavera applications and other applications, such as Enterprise Resource Planning (ERP) applications like E-Business Suite, PeopleSoft, JD Edwards and SAP. Having a single way of integrating with them makes it easier for customers to install just one integration application instead of several applications. It also reduces the overall cost for maintaining these integrations. Primavera Gateway is a light-weight integration web application that is deployed into WebLogic application server, with an Oracle database as its data store. When compared with AIA, it does not need SOA Suite, BPEL or everything above the black line (see Figure 1). With a lighter architecture, Primavera Gateway costs less and runs faster without sacrificing much of flexibility and scalability. With the support of mapping templates and other customization features, Primavera Gateway is more flexible and does a better job customizing an integration in accordance to a customer's need.

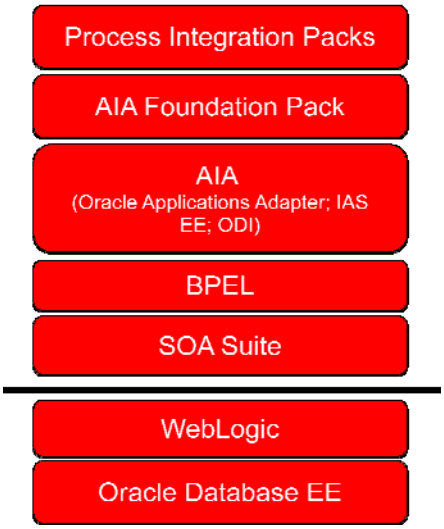

Figure 1

Primavera Gateway Architecture

Primavera Gateway consists of an Integration Broker in the middle, and at least two providers, one for each application that is involved in the integration (see Figure 2 below). Even though each flow only involves two providers at a time, Gateway can support multiple integrations using multiple providers simultaneously. For example, Gateway can support a P6 to SAP integration and a P6 to Unifier integration running at the same time.

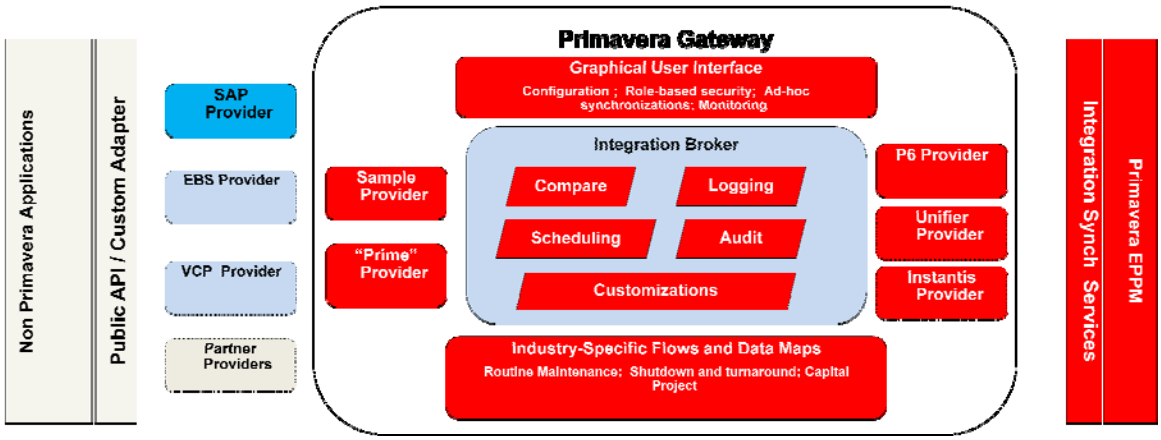

Figure 2

Primavera Gateway is designed to be easily extensible. All Primavera providers are built-in. These include the P6 Provider, Unifier Provider, Enterprise Track Provider and Prime Provider. As newer versions of the enterprise applications are developed, newer versions of the corresponding providers also will be made available. All Primavera data is represented in a canonical Gateway data format which is more stable than the individual application's data format. Therefore, a third party provider would need to only focus on supporting the mapping to this canonical Gateway data format as the applications evolve from one version to the next.

Primavera encourages third-party vendors to create additional Primavera Gateway providers to integrate non-Primavera applications with Primavera applications. Primavera Gateway also makes it easy to write a third-party provider. A Gateway provider consists of a set of Java code bundled in a jar file and a set of XML descriptor files. A very detailed sample provider is also built into the Primavera Gateway application that customers use as an example when writing their third-party provider. By opening up the integration system, and allowing third party vendors to participate, a lot more integration applications can be made available in the Primavera ecosystem to benefit Primavera customers, thereby providing more options for Primavera customers. The more flavors there are for the same provider, the greater the chances of these satisfying a customer's needs.

Integration applications built with Primavera Gateway are easily customizable. A customer can define new fields and new mapping templates directly within the Gateway user interface to map additional customer-specific fields. Customers can also write Groovy mapping templates right in the Gateway user interface to include simple logic which is more than directly copying a field.

Customers can also develop complex customization, the customers can still develop it themselves. This eliminates the need and costs of using integration vendors to develop a custom integration. Similar to a provider, a customization consists of a set of Java code bundled in a jar file and a single XML descriptor file. In many cases though, a single XML descriptor file alone will suffice. A sample customization project is also provided to serve as an example for developing a customization.

Primavera Gateway also provides a set of REST APIs for applications to invoke Gateway functionalities external to Gateway. For example, P6 EPPM 8.4 has developed dialogs that can initiate a Gateway synchronization right within P6 user interface, by leveraging Gateway API.

How Does Primavera Gateway Work

Providers and Flows

In Primavera Gateway, an integration is carried out using flows. A flow is made of multiple flow steps. In its simplest form, a flow is comprised of four flow steps as illustrated in Figure 3 below, where each arrow is a flow step.

Figure 3 ERP to P6 Flow

In the first step (a load step), the data is loaded from the source application. In the second step (a convert step), the data is converted from the source application format to this Primavera canonical data format called Gateway Data Format. In the third step (another convert step), the data is converted from Gateway format to the target application format. In the fourth and last step (a save step), the data is saved to the target application.

Among the four steps, the first two steps (the load step and the first convert step) are carried out by the provider of the source application, and the last two steps (the second convert step and the save step) are carried out by the provider of the target application. Each provider handles the communication with the application (load or save steps), and also the conversion between the application format and the Gateway format. The advantage is that the providers involved here are not required to be aware of the data format of the other application involved in the integration. This makes the whole system potentially pluggable.

For the two applications involved in the integration, one of them must be Primavera application (such as P6), and the other can be a Primavera application or a non-Primavera application (such as SAP). The providers for Primavera applications are provided out-of-the-box. The providers for non-Primavera applications can be developed mostly by third-party integration vendors. For Primavera Gateway 14.2, we already have the following providers: P6 Provider, Prime Provider, Unifier Provider, Enterprise Track Provider, EBS Provider, VCP Provider and SAP Provider. Additional providers are being developed.

In a more complex form involved with a compare step, as illustrated by Figure 4, the data is loaded from the source application and the target application simultaneously, converted to the Gateway format, compared to generate the delta (still in Gateway format), converted to the target data format, and eventually saved to the target application.

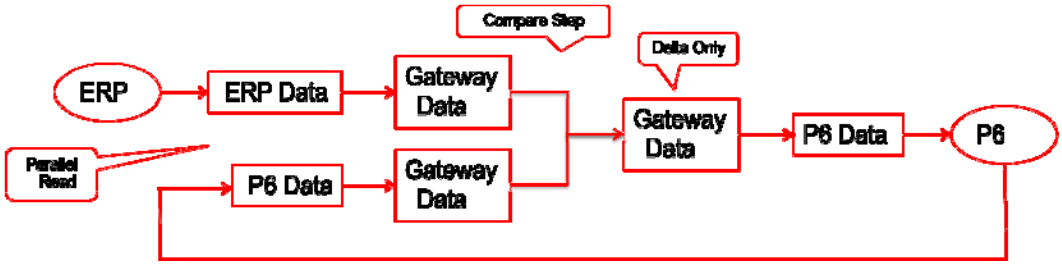

Figure 4 ERP to P6 Flow with Compare Step

The main benefit that a compare step brings is a better performance, as less data needs to be persisted to the target system. Saving data to an application takes much more time than reading data from the same application. For example, in a real world project, as the project progresses, each day brings just a little bit of change, even though the whole project can contain a lot of data. In this case, after using the compare step, only a small delta needs to be accounted for, just as much as the progress made since the last time the flow is run.

The second benefit of the compare step is that it enables delete. By setting the allow deletion parameter, the compare step will report on data that is still in the target application, thereby identifying data that can be deleted in the target application.

The compare step is carried out entirely by the Integration Broker with the data converted into the generic Gateway format to compare data sets in a generic format. Therefore third party vendors do not have to address it in any manner. This is also one of the benefits of using the canonical Gateway Data Format. Without it, comparing source application data directly with target application data would be much more a tedious process.

Business Flows

Currently the following four flow types are as defined as out-of-the-box (OOTB):

- Import Master Data flow
- Export Master Data flow
- Import Project Data flow
- Export Project Data flow
- Master Data flow

The Import/Export Master Data flow supports global objects such as resource, role, calendar, currency and notebook topics etc.

The Import/Export Project Data flow supports project specific objects like project, WBS, activity, resource assignment etc.

The Master Data flow has been developed to have P6 on both sides of the flow to achieve P6 to P6 data integration. This is useful for configuration management use case, where data is pushed from a testing environment to a production environment.

Once a Primavera Gateway integration solution is installed at a customer site, administrators can create business flows based on these pre-defined flow types. When creating business flows, administrators need to make the following decisions:

- Identify the business objects that are to be included in the integration;
- For each business object, determine what mapping templates are to be used (see Fig. 5);

- Identify what parameters are to be visible for end users.

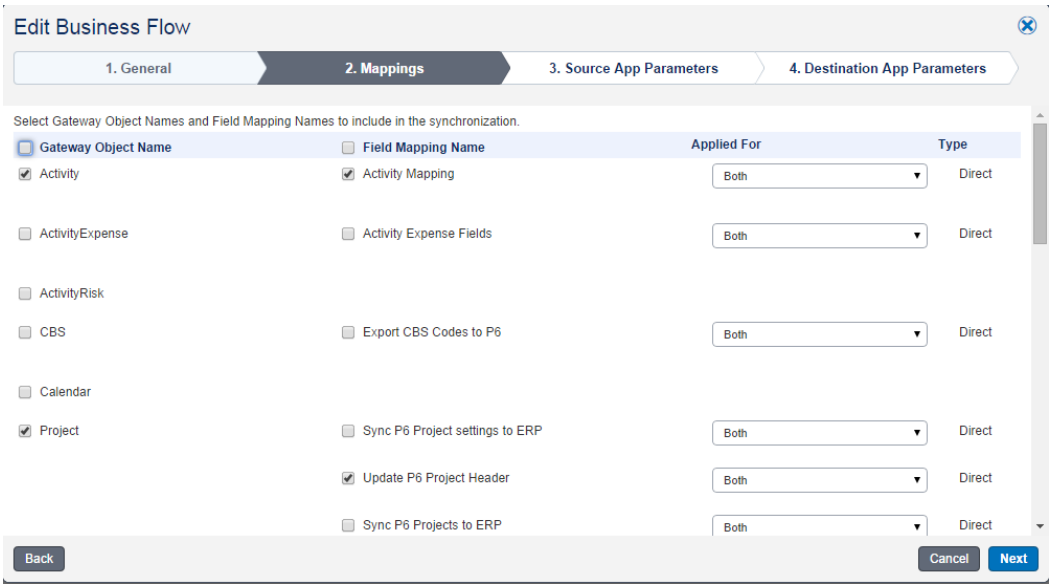

Fig. 5 Selection of objects and mapping templates

A mapping template handles the simplest form of field mapping. First, it is a one-to-one mapping. Each field in the source application is mapped to a field in Gateway side, and the Gateway field in turn is mapped to the target application side. Second, when the mapping template is applied at run time, the field values are directly copied over from the source side to the destination side, if the field type is not enum or foreign keys. For enum type fields, the values are mapped according to how they are defined in Data Value Mapping (DVM) XML files. For foreign key fields, the values are mapped by looking up cross reference. A parameter is usually defined by a provider in the provider descriptor XML file, specified by users in the user interface and used by the provider in Java code. When administrators are creating a business flow, they need to make the following decisions:

- Decide which parameters are to be made visible and hidden to end users,
- Define what default values must be set across the organization (see Fig. 6)

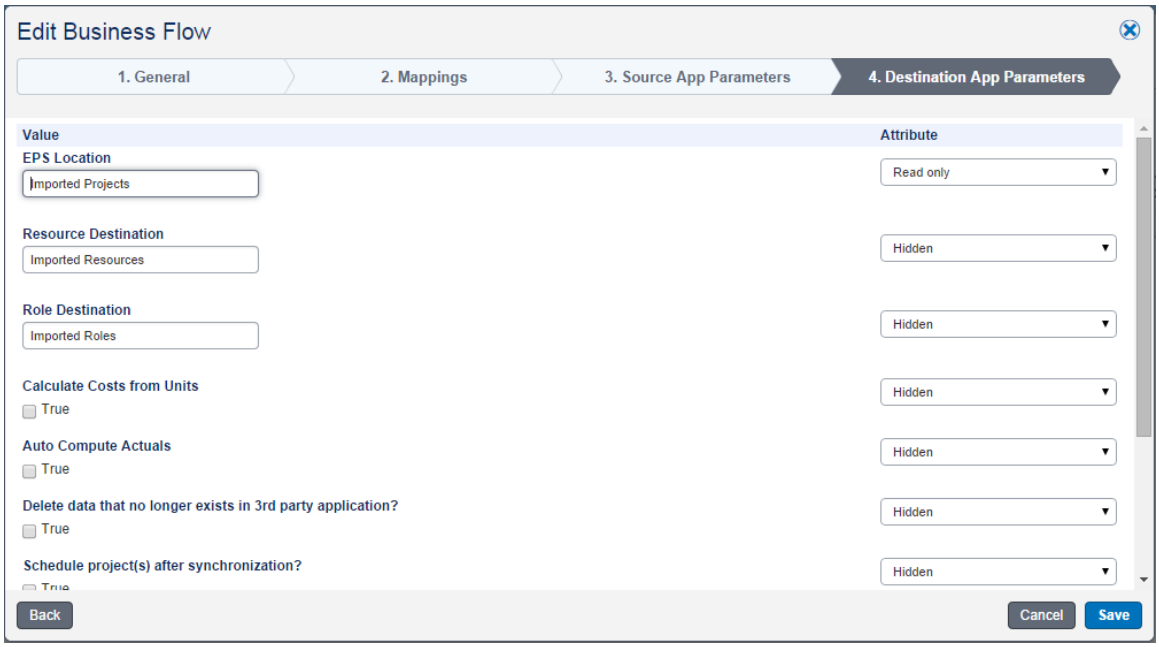

Fig. 6 Selection of parameters to show in Synchronization dialog

Synchronizations

Based on the business flows created by the administrators, the end users can create synchronizations. To create a synchronization, end users have to select application deployments and set parameter values.

Application deployments must be configured ahead of time, according to how many applications are involved in the integration and where they are deployed. For example, an organization can have a single SAP deployment and multiple P6 deployments (one for each region for example). An application deployment includes the configuration parameters that an application provider would need to communicate with the application itself. Again taking P6 as an example, the P6 deployment must include the URL to the P6 Adapter (a SOAP web service), and credentials for accessing the P6 Adapter.

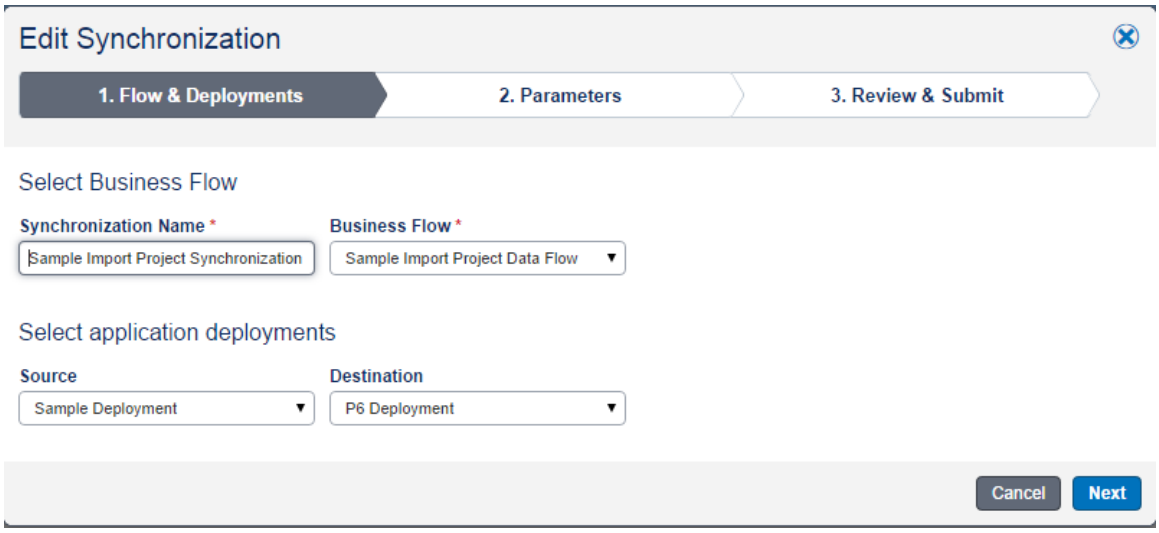

Fig. 7 Pick source and destination application deployments

In addition to picking application deployments for source/destination side of the flow (see Fig. 7 above), the end users must also specify the parameter values for the integration. For example, to export a project from P6, you must specify which project you want to export. These parameters are configured to be visible when the business flow is created by an administrator. Some parameters are configured to be visible but read only (solely for information purpose). Parameters can be left untouched when the default values are just fine whereas a few parameters have to be set, like the P6 project filter parameter mentioned above.

The synchronization can then be run right away or with a fixed schedule. You can schedule it to run daily, weekly or monthly (see Fig. 8).

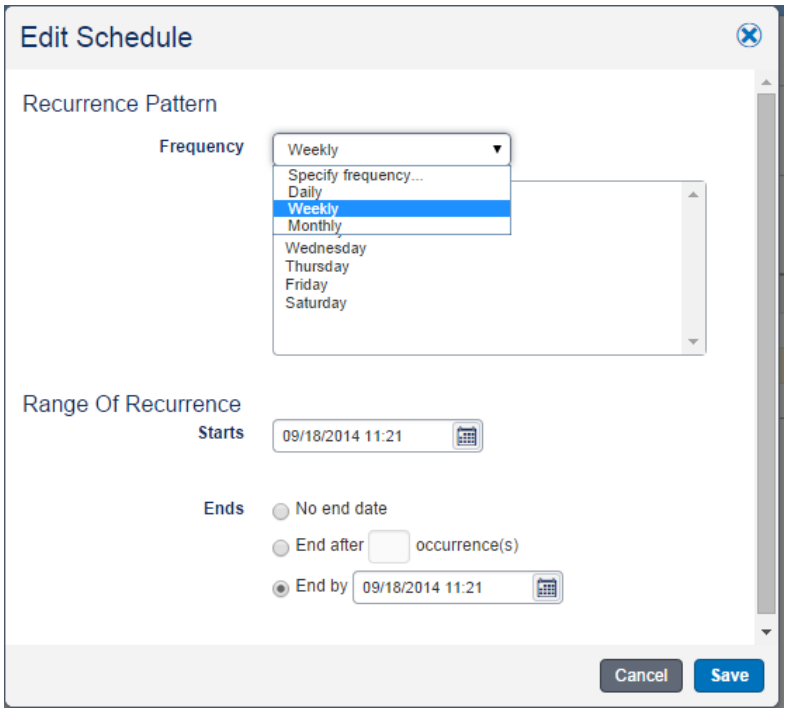

Fig. 8 Configure schedule for a synchronization

The synchronization can also be run right away with review enabled, by selecting the “Run with Review” button instead of “Run” button. It will run right away, but will transition to a Review state before the Save step. At that time, you can review all the changes. If you need to make any further changes, you can cancel the job without any changes taking effect (see Fig. 9).

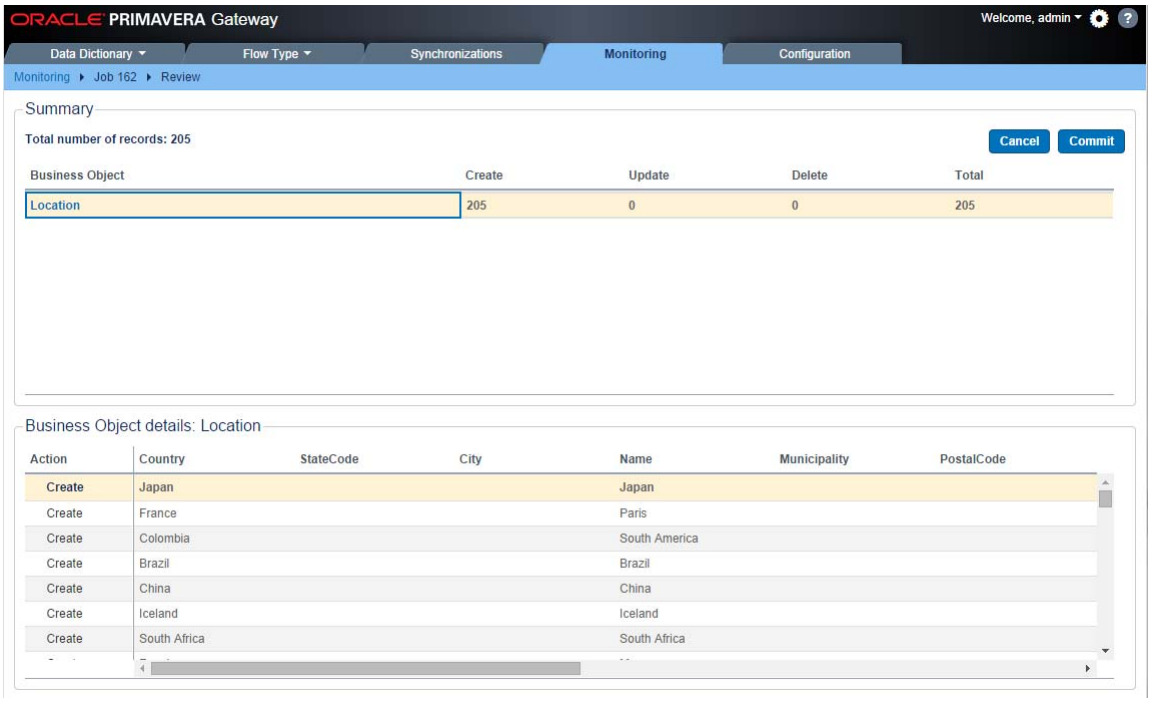

Fig. 9 Review before commit

Monitoring

After a synchronization is run, a job is created in the system. Jobs can be monitored at any time on the Monitoring page. A job is made up of steps. Once a step is completed, the status of the step is reported in a generated log. Users can review these logs to troublehshoot and fix errors (see Fig. 10).

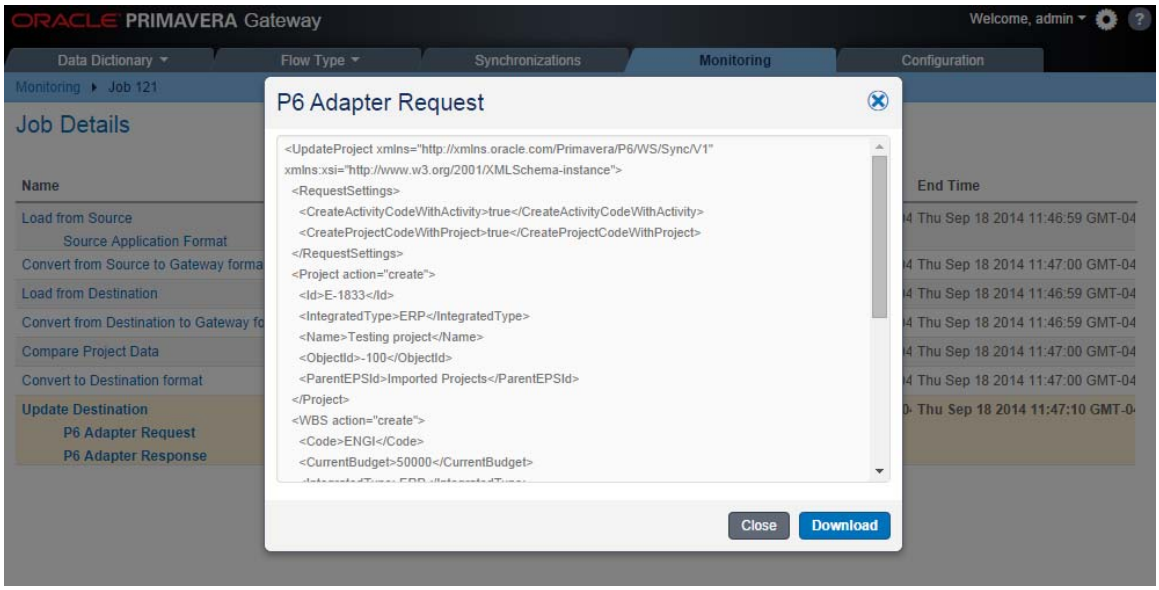

Fig. 10 Job details and logs

With P6 Provider, when an error occurs, by default the changes are fully rolled back in both P6 and Gateway, as if the flow is never run. However, you can also change this behavior by setting a parameter to ensure that only the failed objects are rolled back, while the changes to all other objects are still committed.

In addition to the logs on the Monitoring page, you can also configure Gateway to send email notification to notify you when a flow is run.

Customization

Every organization has its own process for accomplishing objectives. Most of applications are designed to allow themselves to be extended. For example, Primavera P6 has user defined fields and codes to allow customers to extend the system. For this reason, the integration needs to be extended as well for each organization.

While a custom-built integration is much more expensive than a generic integration application, a generic integration application can rarely be used as is. Some kind of modification would have to be made before the generic integration can be used in real production. This modification, here we called customization, can be expensive, since the customers mostly likely won't have the skills to do it themselves and they would need to hire some experts to do it.

Primavera Gateway is designed to be highly customizable, and makes the customization job easy. You can extend the data definition by adding additional fields to the existing objects, and then define additional field mappings for these fields. To extend the data definition, or to add additional mapping templates, it is as easy as writing a new XML file, and importing it from the Configuration tab (see Fig. 11).

```
<FieldMapTemplates>
    <App1Name>Sample</App1Name>
    <App2Name>P6</App2Name>
    <FieldMapTemplate>
        <Description>Cost code mapping for activity object</Description>
        <App1BusinessObjectName>Operation</App1BusinessObjectName>
        <Name>Activity Cost Code Mapping</Name>
        <PDIBusinessObjectName>Activity</PDIBusinessObjectName>
        <FieldMap>
            <App1>ERPCostCode</App1>
            <App2>CostCode</App2>
            <PDI>CostCode</PDI>
        </FieldMap>
    </FieldMapTemplate>
</FieldMapTemplates>
```

Fig. 11 Define a new mapping template in a customization XML file

With Gateway 14.2 release, you can now define a new mapping template right in the Gateway UI (see Fig. 12). If a field is not yet available, you can define it in this dialog as well. You can also export all these mapping templates to an external XML file for backup purpose, or import these back after a Gateway system upgrade.

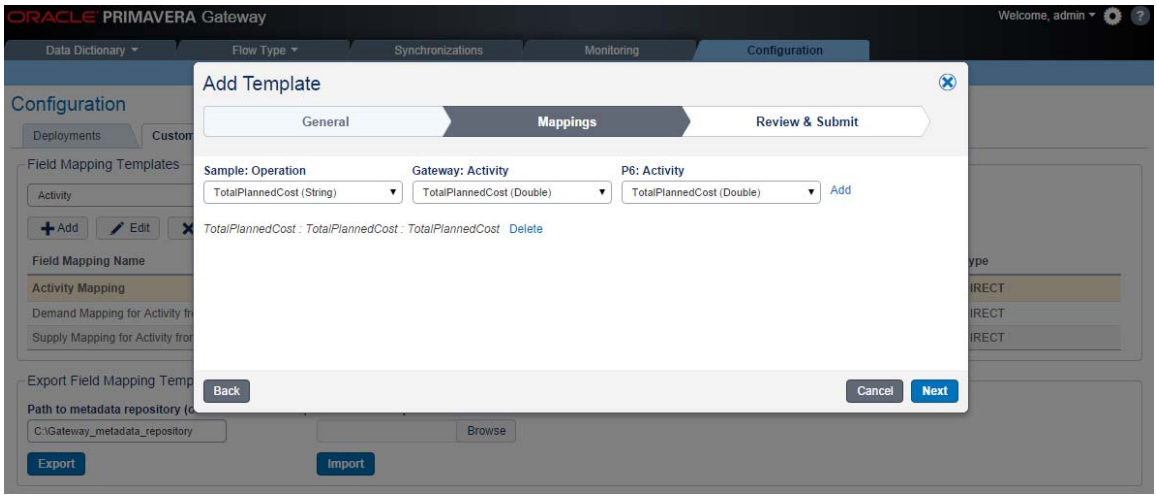

Fig. 12 Create a direct mapping template in UI

For more advanced field mappings, or mappings what are not as simple as one-to-one mappings, you can also use Java custom field mapping or Groovy custom field mapping. These two mechanisms give you the additional flexibility to define many-to-one mappings, and also introduce coding logics into the mapping.

With Java field mapping, you would need to write Java code in addition to the XML descriptor file. The Java code must be built into a Jar file and bundled into the integration application. With Groovy field mapping, the script is embedded right within the XML descriptor file, and you do not need to provide additional Jar file. With Gateway 14.2 release, you can now create a Groovy field mapping templates in the UI as well (see Fig. 13).

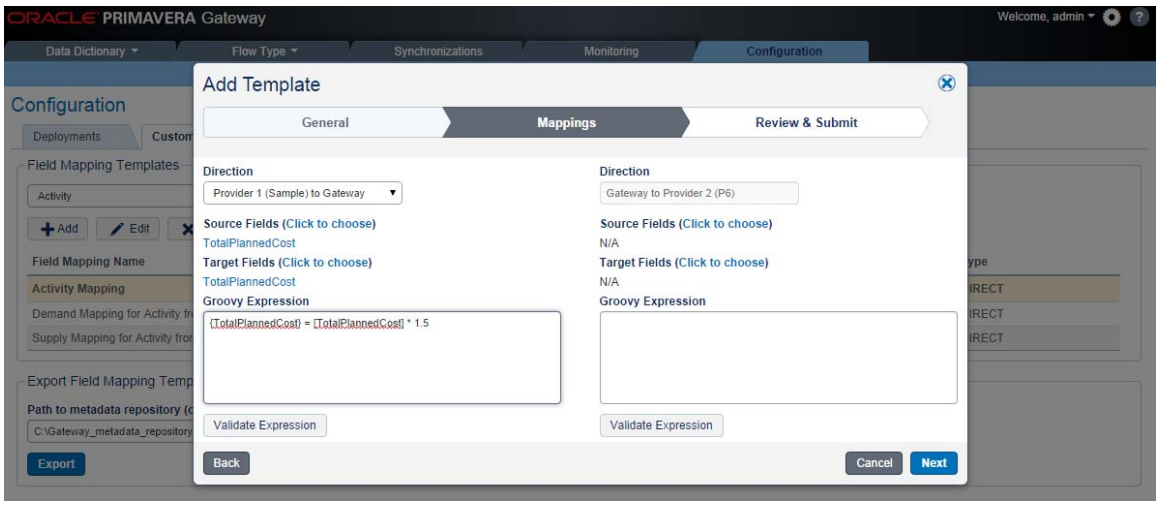

Fig. 13 Create a new Groovy field mapping template

Another feature for customization is the capability to define a custom step using Java code for a flow type, in addition to the existing load/convert/compare/save steps. The custom step can be inserted

anywhere after the load step and before the save step. A custom step Java code allows you control to the whole document (not just one object like custom field mapping), manipulate fields of existing objects, and add new objects or delete existing objects. Custom steps are complex, yet very powerful mechanisms with significant benefits than custom field mappings.

Sample Customization Project

Primavera Gateway delivers a sample customization project to illustrate how the customization should be done. The sample customization project contains examples for the following:

- How to extend P6 data definition with additional user defined fields and codes (resource codes/project codes/activity codes)
- How to define additional mapping template to map the newly added fields.
- Custom field mappings, including custom Java field mapping and Groovy field mapping,
- Custom step

Java source code for the custom Java field mapping and the custom step is also included.

Summary

Oracle Primavera Gateway is a light-weight integration framework, using Java, WebLogic application server and Oracle Database, for integrating with Oracle Primavera products. While it is a generic integration framework, it is built to be highly customizable. It is easy and economical to build customizations for Gateway integrations. Oracle Primavera Gateway should be your first choice for any integration needs with Oracle Primavera products.

]Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200

oracle.com

Oracle is committed to developing practices and products that help protect the environment

Hardware and Software, Engineered to Work Together