HP Project and Portfolio Management Center

Software Version: 8.00

Web Services Programmer's Guide

Document Release Date: February 2010 (updated January 2011)
Software Release Date: August 2009

Legal Notices

Warranty

The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notices

Trademark Notices

Adobe® is a trademark of Adobe Systems Incorporated.

Intel®, Intel® Itanium®, Intel® Xeon™, and Pentium® are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Java™ is a U.S. trademark of Sun Microsystems, Inc.

Microsoft®, Windows®, and Windows® XP are U.S. registered trademarks of Microsoft Corporation.

Microsoft Vista® is either a registered trademark or trademark of Microsoft Corporation in the United States and/or other countries.

Oracle® is a registered U.S. trademark of Oracle Corporation, Redwood City, California.

UNIX® is a registered trademark of The Open Group.

Documentation Updates

This manual's title page contains the following identifying information:

- Software version number, which indicates the software version
- Document release date, which changes each time the document is updated
- Software release date, which indicates the release date of this version of the software

To check for recent updates, or to verify that you are using the most recent edition of a document, go to:

h20230.www2.hp.com/selfsolve/manuals

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your HP sales representative for details.

Support

You can visit the HP Software Support Web site at:

hp.com/go/hpsoftwaresupport

HP Software Support Online provides an efficient way to access interactive technical support tools. As a valued support customer, you can benefit by using the support site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HP support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and sign in. Many also require a support contract.

To find more information about access levels, go to:

h20230.www2.hp.com/new_access_levels.jsp

To register for an HP Passport ID, go to:

h20229.www2.hp.com/passport-registration.html

Contents

1 HP Demand Management Web Services

Overview

HP Demand Management Web services provides interfaces for reading, creating, updating, and deleting request entities in HP Project and Portfolio Management Center (PPM Center). In addition, you can execute workflow transitions on a specified request by using these Web service operations.

You can access all request fields, including user defined fields by using field token, except note, and remote reference.

References

Data types:
`webservice_toolkit\java\conf\xsd\`**`Demand.xsd`**

Operations:
`webservice_toolkit\java\conf\wsdl\`**`DemandService.wsdl`**

Java sample code:
`webservice_toolkit\java\client\src\examples\dm\`
`DemandServiceClient.java`

.NET sample code:
`webservice_toolkit\MicrosoftDotNet\DemandServiceTest`

Operations History

HP Demand Management Web services provides many operations starting with PPM Center version 7.1. *Table 1-1* lists the HP Demand Management Web service operations by version.

Table 1-1. HP Demand Management Web Service operations

Web Service operation	7.1	7.5	8.00
setRequestRemoteReferenceStatus	Yes	Yes	Yes
addRequestNotes	Yes	Yes	Yes
createRequest	Yes	Yes; Revised	Yes (Same with 7.5)
deleteRequests	Yes	Yes	Yes
setRequestFields	Yes	Yes	Yes
getRequests	Yes	Yes; Revised	Yes (Same with 7.5)
executeWFTransitions	No	Yes	Yes

Terms and Concepts

This section describes the following three HP Demand Management terms.

- *Field Token*
- *Remote Reference*
- *URL Reference*

For additional terms and concepts, refer to *HP Demand Management User's Guide*.

Field Token

The HP Demand Management Web services identifies request fields with field tokens. A field token is a string with the following components and attributes:

- Alphanumeric characters
- Underscores (_)
- Always upper case
- Unique among all fields of the request type and request header type

When field tokens are used in a Web service operation, a prefix is required to indicate the source or type of the field. Valid prefixes include the following:

Prefix	Description	Example
REQ	Indicates the field is from a request header type.	REQ.DESCRIPTION
REQD	Indicates the field is from a request type.	REQD.TABLE_TOKEN
UD	Indicates the field is a user data field.	UD.CUST_FIELD
T	Indicates the field is a column of a table component field.	T. TABLE_TOKEN. COLUMN_TOKEN

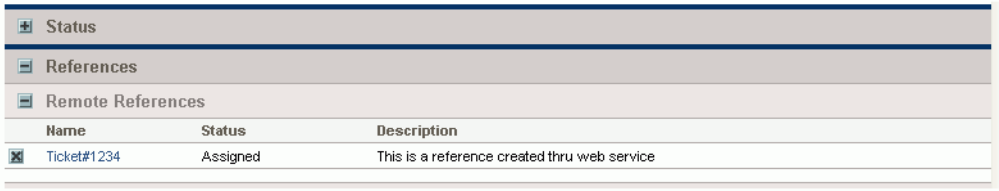
In some cases, you will see the prefixes “VP” or “P.” For example, REQD.VP.PLATFORM. These are legacy prefixes that were designed to differentiate the display value (with the prefix VP) and the internal code value (with the prefix P). The current Web service implementation will always use the display value instead. As a result, legacy prefixes are ignored.

Remote Reference

A remote reference refers to an item that exists on a remote system, which can be a PPM Server or another type of server.

Remote reference items appear in the **Reference** section of a request. A remote reference is only available for the Web service interface and cannot be created through the PPM Center standard interface. The purpose of the remote reference is to link a request or an item on an external system to a request in the current PPM Center system.

The following example illustrates a remote reference. The hyperlink attached to the name column is pointing to the remote system URL.



URL Reference

A URL reference refers to another type of item in the **References** section of a request. You can also create a URL reference by using the PPM Center standard interface.

The following example illustrates a URL reference. The hyperlink attached to the name column is pointing to the referred URL.

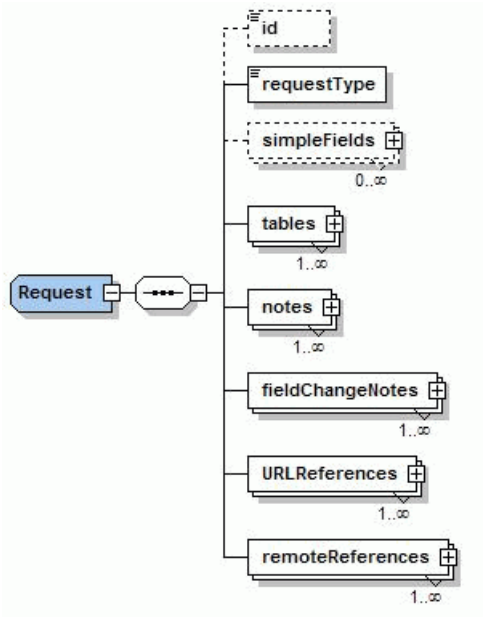
When you use the getRequests operation to obtain the content of a request, references of any type are converted to URL references. For more information, see the operation *getRequests*.

For more information about general HP Demand Management terms and concepts, see the *HP Demand Management User's Guide.*

Data Types

HP Demand Management Web services includes the following data types:

Request



For more information about how to create Request/id, see *ExampleGetRequests01*`.java`.

For more information about how to create Request/requestType, see *ExampleCreateRequest*`.java`

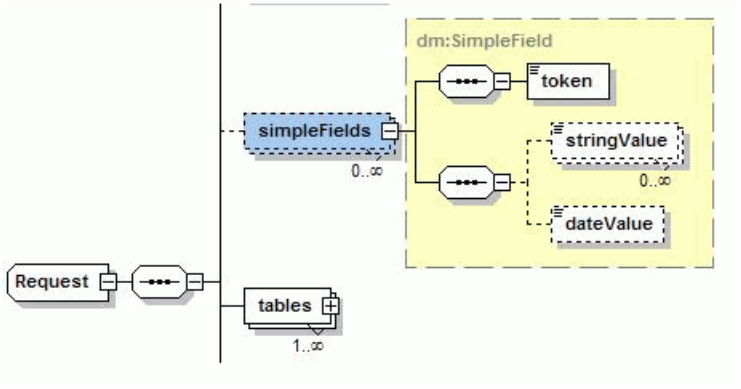
Property	Type	Description	Required
Id	string	Request ID. A numeric value, which is typically greater than 30,000.	No[a]
requestType	string	Request type name. This field is case-sensitive.	Yes
simpleFields	SimpleField	An array list of the following elements: • token • stringValue • dateValue	No
tables	Table	An array list of token and columns elements.	No

Property	Type	Description	Required
notes	Note	An array list of the following optional elements: • author • creationDate • content	No
fieldChangeNotes	FieldChangeNote	An array list of the following optional elements: • author • creationDate • content The following are required elements: • fieldPrompt • oldValue • newValue	No
URLReferences	URLReference	An array list of the following optional elements: • addedBy • creationDate • description • name The following is the required element: • refUR	No

Property	Type	Description	Required
remoteReferences	RemoteReference	An array list of the following optional elements: • addedBy • creationDate • description • name The following are required elements: • Identifier • displayURL • status	No

a. Property ID is required for certain operations such as setRequestFields.

simpleFields (type SimpleField)

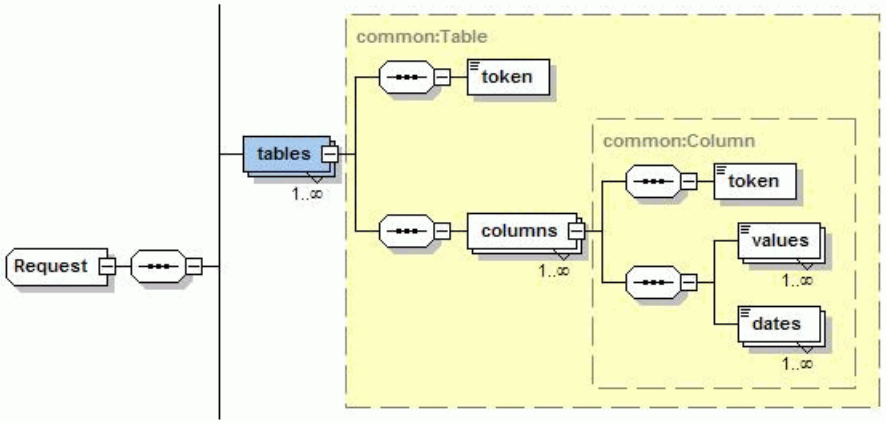


For more information about how to create Request/simpleFields, see *ExampleCreateRequest*`.java`.

Property	Type	Description	Required
token	String	A token value to identify the field. A proper prefix is required. See *Field Token* for more details.	Yes

Property	Type	Description	Required
stringValue	String	The value that represents the token.	No
dateValue	String	String value of the date field. The date field format is as defined in the XML Schema data type dateTime. For more information, refer to the *XML Schema Part 2: Datatypes Second Edition (section 3.2.7)* at the W3C Web site: http://www.w3.org	No

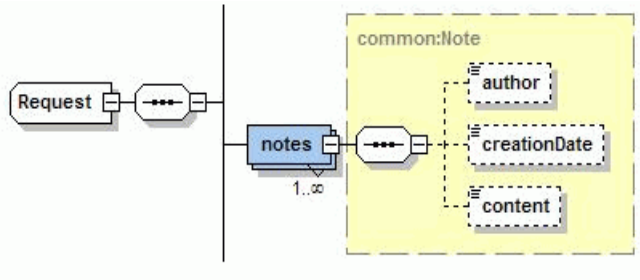
tables (type Table)



For more information about how to create Request/tables, see *ExampleCreateRequest*`.java`.

Property	Type	Description	Required
token	String	Token value to identify the field. A proper prefix is required. See *Field Token* for more details.	Yes
columns	Column	Array list of token, values, and dates. See the following three properties for details.	Yes
columns /token	String	Token value to identify the field. Proper prefix is required. See *Field Token* for more details.	Yes
columns /values	String	Array list of string values of the field. It allows single or multiple values for multi-select fields.	Yes
columns /dates	dateTime	Array list of string values of the field. It allows single or multiple values for multiple selected fields. This XML Schema type `dateTime` is equivalent to `java.util.Calendar`.	Yes

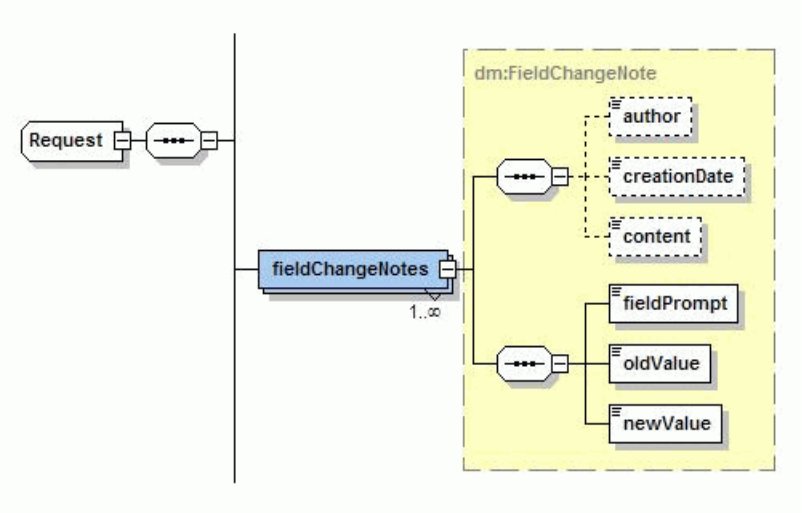
notes (type Note)



For more information about how to create Request/notes, see *ExampleCreateRequest*`.java`.

Property	Type	Description	Required
author	string	Author of the note. This field can be any string value and it is not necessary to be a valid user name in PPM Center. This field is shown as “originally added by” in the **Note** section.	Yes
creationDate	dateTime	Date of the note. Notes are grouped by date when displayed. The XML Schema type `dateTime` is equivalent to `java.util.Calendar`.	Yes
content	string	Content of the note.	Yes

fieldChangeNotes (type FieldChangeNote)

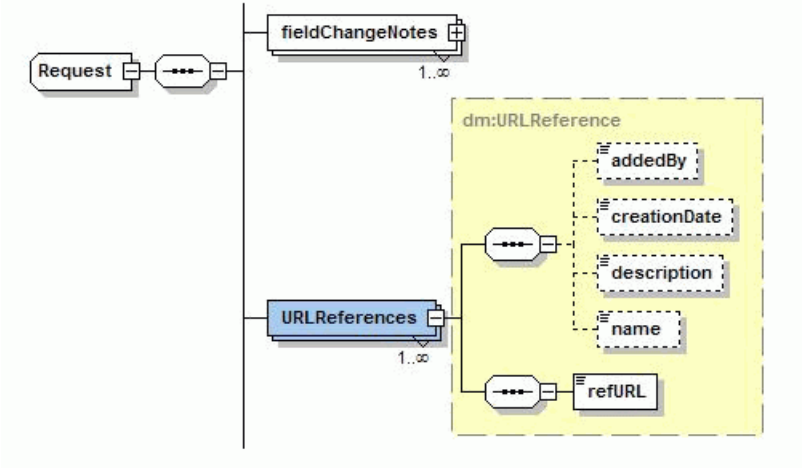


For more information about how to create Request/fieldChangeNotes, see *ExampleGetRequests01*`.java`.

Property	Type	Description	Required
author	string	Author of the note. This field can be any string value and it is not necessary to be a valid user name in PPM Center. This field is shown as “originally added by” in the **Note** section.	No
creationDate	dateTime	Date of the note. Notes are grouped by date when displayed. The XML Schema type `dateTime` is equivalent to `java.util.Calendar`.	No
content	string	Content of the note.	No
fieldPrompt	string	The filed prompt value to be changed.	Yes
oldValue	string	The old value of the field prompt.	Yes

Property	Type	Description	Required
newValue	string	The new value of the field prompt.	Yes

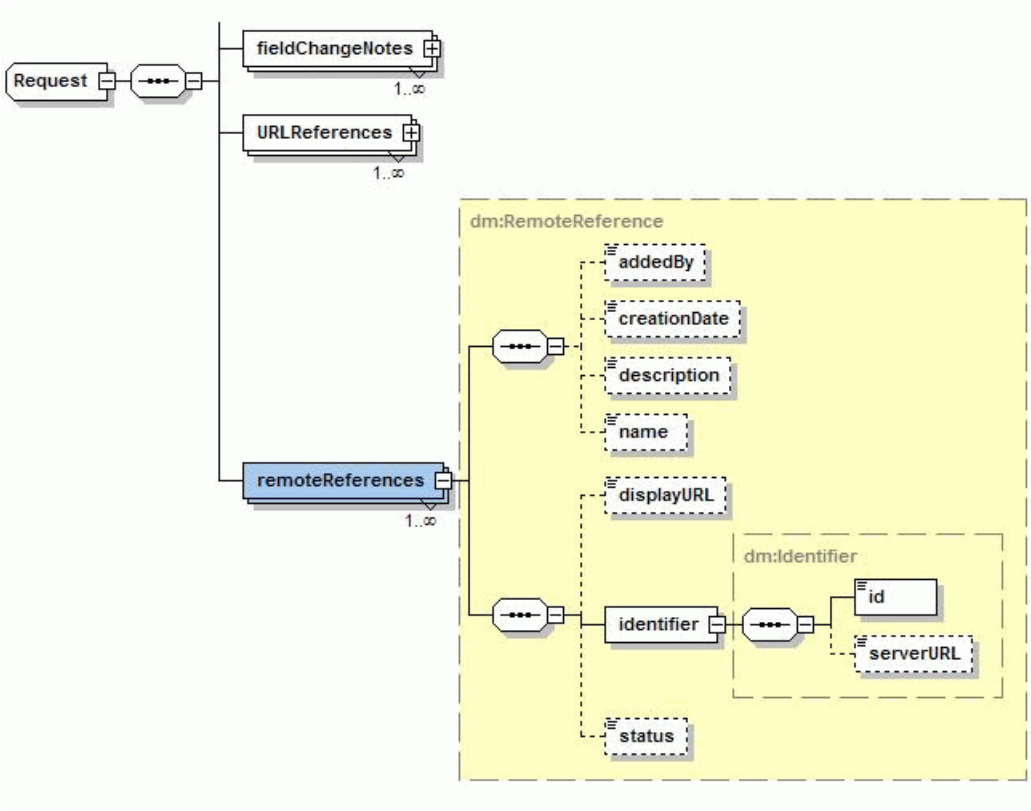
URLReferences (type URLReference)



For more information about how to create Request/URLReferences, see *ExampleCreateRequest*`.java`.

Property	Type	Description	Required
addedBy	string	The user that added the URL reference. It can be any string value and it is not necessary to be a valid user name in PPM Center.	No
creationDate	dateTime	Date on which the note was created. The XML Schema type `dateTime` is equivalent to `java.util.Calendar`.	No
description	string	The description of the URL reference.	No
name	string	The name of the URL reference.	No
refURL	string	The old value of the field prompt.	Yes

remoteReference



For more information about how to create Request/remoteReferences, see *ExampleCreateRequest*`.java`.

Property	Type	Description	Required
addedBy	string	The name of the user who added the remote reference	No
creationDate	dateTime	The date on which the remote reference was added. The XML Schema type `dateTime` is equivalent to `java.util.Calendar`.	No
description	string	The description of the remote reference.	No
name	string	The name of the remote reference	No
displayURL	string	The URL of the remote reference	No
identifier	Identifier	The ID and optional serverURL elements	Yes
identifier / id	string	Request ID. A numeric value, which is normally greater than 30,000.	Yes
identifier / serverURL	string	URL of the PPM Server. It is mainly used in request remote reference. This value defaults to an empty string.	Yes
status	string	Status of the reference	Yes

Operations

The following operations are included in HP Demand Management Web services:

- *getRequests* on page 40
- *createRequest* on page 44

- *setRequestFields* on page 50
- *setRequestRemoteReferenceStatus* on page 55
- *addRequestNotes* on page 60
- *executeWFTransitions* on page 63
- *deleteRequests* on page 66

getRequests

Purpose

This operation reads the information of a request in PPM Center.

Function

This operation reads the content of a request or a list of requests, identified by request IDs in PPM Center.

The following content of a request can be retrieved through this operation:

- Simple Fields, defined in request header type, request type, or field group if applicable
- Field of Table Component
- Note
- FieldChangeNotes
- References, except remote references

All reference types are converted to URL references. See the following list for details:

- Attachment

  URL name is the attachment name, and the URL hyperlink points to the URL to access the attachment download.

- Package Reference

URL name is in the format of “Package <package ID> on localhost,” and the URL hyperlink points to the URL to access the package.

- Program Reference

  URL name is in the format of “Program <program name> on localhost,” and the URL hyperlink points to the URL to access the program.

- Project Reference

  URL name is in the format of “Project <project name> on localhost,” and the URL hyperlink points to the URL to access the project.

- Task Reference

  URL name is in the format of “Task <task name> on localhost,” and the URL hyperlink points to the URL to access the task.

- Request Reference

  URL name is in the format of “Request <request ID> on localhost,” and the URL hyperlink points to the URL to access the request.

- URL Reference

Limitations

This operation has the following limitations:

- Does not read remote references, which can be added to a request when creating the request by using the *createRequest* operation.
- For attachments, this operation does not return the attached document, but rather the URL pointing to the document on the PPM server.
- No additional security check. After the user passes authentication, the user can read any request even if the user does not have access to that request.

Input

A collection of request IDs

Return

A collection of request objects. If the specified request ID does not exist in PPM Center, the operation is ignored and no error is generated.

The getRequest operation was introduced in PPM Center version 7.1. The return type of this operation has been revised since version 7.5. The following table illustrates this revision:

Web Service Operation	PPM Center Version
Request[] getRequests (Identifier[] ids)	7.1
WSRequest[] getRequests (Identifier[] ids)	7.5 and 8.00

Java Interface

```
GetRequestsResponseDocument getRequests(GetRequestsDocument in)
```

Parameters	Description
GetRequestsDocument	Wrapper of the collection of request identifiers, or Identifier[].
GetRequestsResponseDocument	Wrapper of the collection of request objects, or Request[].

Java Examples

Example: get data for a single request.

ExampleGetRequests01

```
public class ExampleGetRequests01 {
. . .
    private void getRequests(String serviceURL, String
requestId) throws Exception {

        // construct the Identifier array
        Identifier[] ids = new Identifier[1];
        ids[0] = Identifier.Factory.newInstance();
        ids[0].setId(requestId);
        // get Webservice handler
        DemandServiceStub stub = new DemandServiceStub(ctx,
serviceURL);
        // Construct message to send
```

```
        GetRequestsDocument inDoc =
GetRequestsDocument.Factory.newInstance();
        GetRequestsDocument.GetRequests getRequests =
inDoc.addNewGetRequests();
        getRequests.setRequestIdsArray(ids);
. . .
    }
}
```

Example: get data for a list of request IDs

```
public class ExampleGetRequests02 {
. . .
    private void getRequests(String serviceURL, String[]
requestIds) throws Exception {
        // construct the Identifier array
        Identifier[] ids = new Identifier[requestIds.length];
        for (int i = 0; i < requestIds.length; i++) {
            ids[i] = Identifier.Factory.newInstance();
            ids[i].setId(requestIds[i]);
        }
        // get Webservice handle
        DemandServiceStub stub = new DemandServiceStub(ctx,
serviceURL);
        // Construct message to send
        GetRequestsDocument inDoc =
GetRequestsDocument.Factory.newInstance();
        GetRequestsDocument.GetRequests getRequests =
inDoc.addNewGetRequests();
        getRequests.setRequestIdsArray(ids);
. . .
    }
}
```

Errors and Exceptions

Message Code	Message	Cause(s)	Possible Corrective Action
KNTA-11186	Internal error has occurred while calling PPM Web Service. Contact PPM support with the detail information if the problem persists. (KNTA-11186) Details: Missing required element.	You have not set the request ID.	Provide a request ID for the request.

Message Code	Message	Cause(s)	Possible Corrective Action
N/A	An error occurred when reading requests Open API Exception 9: Request with ID 30392 was not found in the system. Import was not performed.	The request ID does not exist on the PPM server.	Provide an existing request ID for the request.
KNTA-11186	Internal error has occurred while calling PPM Web Service. Contact PPM support with the detail information if the problem persists. (KNTA-11186) Details: Non digit data was set for number field. Original exception message: For input string: “30362A”.	The request ID value you provide is invalid.	The request ID value has to be a numeric value. Correct the value and try again.

createRequest

Purpose

This operation creates a new request in PPM Center.

Function

This operation creates a new request in PPM Center. You can specify the following request contents by using this operation:

- Simple Fields, defined in request header type, request type, or field group, if applicable
- Field of Table Component
- Note
- URL References
- Remote References

During this operation, the system applies the rules defined on the request. The field level rules are applied according to the sequence of the fields in the Web service request. Any field value that you enter will override a rule-derived value.

For simple fields defined as budget, benefit, or staffing profile, the field value is the exact name of the budget, benefit, or staffing profile.

Limitations

This operation has the following limitations:

- Does not support attachment, password field, and references other than URL reference.
- For chained field change rules, this operation does not perform cascading. For example, a rule defined on a field-A change leads to a field-B change, and another rule is defined to change field-C based on a field-B change. In this example, when you set the value for field-A using this operation, field-B will be changed as result of applying the rule, but field-C will not be affected.
- For required fields, the request will be created even the value for required field is not specified.
- There is no security check on the permission. After a user passes authentication, the user can create requests through Web service requests even if the user does not have the 'create request' permission.
- If the specified token of the field does not exist, there will be no error reported. The field will be simply ignored.

Input

A Request object, with all the desired content filled, and with no request ID.

The createRequest operation was introduced in PPM Center version 7.1. The argument of this operation has been revised since version 7.5. The following table illustrates this revision:

Web Service Operation	PPM Center Version
RemoteReference createRequest (**Request** req)	7.1
RemoteReference createRequest (**WSRequest** req)	7.5 and 8.00

Return

A remote reference to the request.

Java Interface

```
CreateRequestResponseDocument
createRequest(CreateRequestDocument in)
```

Parameters	Description
CreateRequestDocument	Wrapper of the request object.
CreateRequestResponseDocument	Wrapper of the remote reference to the newly created request.

Java Examples

Example: create a request of type `Mybug`, which is a copy of the out-of-box request type `Bug`, with the following changes:

- Adding a table component field on the request type:

  Field TOKEN: STAKE_HOLDER

  Number of columns: 2

  Column 1 TOKEN: ID

  Column 2 TOKEN: NAME

- Adding a auto-complete field on the request type, which allows multi-value

  Field TOKEN: REVIEWER

  Validation: PPM - User Id - Enabled

ExampleCreateRequest

```
public class ExampleCreateRequest {
. . .
    private void createRequest(String serviceURL, String
requestType) throws Exception {

        // Request component
        Request req = Request.Factory.newInstance();
        req.setRequestType(requestType);
        SimpleField[] fields = new SimpleField[6];

        // simpleFields component

        // Set field 'Description'
       SimpleField field_A = SimpleField.Factory.newInstance();
        field_A.setToken("REQ.DESCRIPTION");
       field_A.setStringValue1Array(new String[] { "WebService
Test" });
        fields[0] = field_A;

        // Set field 'Department'
       SimpleField field_B = SimpleField.Factory.newInstance();
        field_B.setToken("REQ.DEPARTMENT_NAME");
        field_B.setStringValue1Array(new String[] { "Finance"
});
        fields[1] = field_B;

        // Set field 'Module'
       SimpleField field_C = SimpleField.Factory.newInstance();
        field_C.setToken("REQD.VP.MODULE");
        field_C.setStringValue1Array(new String[] { "Module A"
});
        fields[2] = field_C;

        // Set field 'Platform'
       SimpleField field_D = SimpleField.Factory.newInstance();
        field_D.setToken("REQD.VP.PLATFORM");
        field_D.setStringValue1Array(new String[] { "Unix" });
        fields[3] = field_D;

        // Set field 'Impact'
       SimpleField field_E = SimpleField.Factory.newInstance();
        field_E.setToken("REQD.VP.IMPACT");
        field_E.setStringValue1Array(new String[] { "Warning"
});
        fields[4] = field_E;

        // Set field 'Reviewer', which allow multi-values
       SimpleField field_F = SimpleField.Factory.newInstance();
        field_F.setToken("REQD.REVIEWER");
        field_F.setStringValue1Array(new String[] { "admin",
"userx" });
        fields[5] = field_F;

        // Add all the fields to request object
```

```
        req.setSimpleFieldsArray(fields);

        // tables component
        Table t = req.addNewTables();
        t.setToken("REQD.STAKE_HOLDER"); // token of the field
        Column c = t.addNewColumns();
        c.setToken("T.STAKE_HOLDER.ID"); // token of the column
        c.setValuesArray(new String[] { "311", "312" }); //
value array
        c = t.addNewColumns();
        c.setToken("T.STAKE_HOLDER.NAME");
        c.setValuesArray(new String[] { "User1", "User2" });

        // notes component
        Note note = req.addNewNotes();
        note.setAuthor("admin");
        note.setContent("WebService Test Note");
        note.setCreationDate(Calendar.getInstance());

        // URLReferences component
        URLReference refURL = req.addNewURLReferences();
        refURL.setAddedBy("admin");
        refURL.setCreationDate(Calendar.getInstance());
       refURL.setDescription("This is a reference created thru
web service");
        refURL.setName("Reference URL");
        refURL.setRefURL("http://www.ref.com");

        // remoteReference component
        RemoteReference ref = req.addNewRemoteReferences();
        ref.setAddedBy("admin");
        ref.setCreationDate(Calendar.getInstance());
       ref.setDescription("This is a reference created thru web
service");
        ref.setName("Ticket#1234");
        ref.setDisplayURL("http://www.display.com");
        Identifier refId = Identifier.Factory.newInstance();
        refId.setId("31234");
        refId.setServerURL("http://server:port");
        ref.setIdentifier(refId);
        ref.setStatus("Assigned");

        // Get web service
        DemandServiceStub stub = new DemandServiceStub(ctx,
serviceURL);
        // Construct message to send
        CreateRequestDocument inDoc =
CreateRequestDocument.Factory.newInstance();
        CreateRequestDocument.CreateRequest createRequest =
inDoc.addNewCreateRequest();
        createRequest.setRequestObj(req);
. . .
    }
}
```

Errors and Exceptions

Message Code	Message	Cause(s)	Possible Corrective Action
N/A	Missing required element "{http://mercury.com/ppm/dm/1.0}requestType"(line -1, col -1, in SOAP-message)	Missing Request type.	Check and provide the right request type.

setRequestFields

Purpose

This operation updates one or more simple fields in a request in PPM Center.

Function

This operation updates the content of a request, identified by the request ID in PPM Center. You can update the following fields of a request through this operation:

- Simple Field from request header type
- Simple Field from request type
- Simple Field from Field Group

Exception: The following field is not updatable once the request is created. Therefore, you cannot update them using this operation:

- REQ.CREATED_BY_NAME
- REQ.CREATED_BY_USERNAME
- REQ.CREATED_BY_EMAIL
- REQ.CREATION_DATE
- REQ.CREATED_BY
- REQ.LAST_UPDATED_BY_USERNAME
- REQ.LAST_UPDATED_BY_EMAIL
- REQ.LAST_UPDATE_DATE
- REQ.LAST_UPDATED_BY
- REQ.REQUEST_TYPE_NAME
- REQ.REQUEST_TYPE_ID
- REQ.REQUEST_ID

- REQ.WORKFLOW_NAME
- REQ.WORKFLOW_ID
- REQ.STATUS_NAME
- REQ.STATUS_ID
- REQ.CONTACT_PHONE_NUMBER
- REQ.CONTACT_EMAIL
- REQD.CREATION_DATE
- REQD.LAST_UPDATE_DATE
- REQD.WORKFLOW_ID
- REQD.CREATED_BY
- REQD.REQUEST_ID
- REQD.LAST_UPDATED_BY
- SYS.USERNAME
- SYS.USER_ID

During this operation, the rules defined on the fields are applied, in the order of the fields' appearance in the Web service request. Any field in which you enter a value will override the rule-derived value.

To update field with type of budget, benefit, or staffing profile, the name of the entity is the field value.

Limitations

This operation has the following limitations:

- Does not support updates of fields with attachments, password fields, and table component fields. Table component fields can be set only when you create a request. Once the request has been created, the Web service interface does not support update of table component fields.

- No additional security check including participant or field level access check on this operation. After the request passes the Web service authentication, the operation will try the update with no further security check.

For more information, see:

- addRequestNotes - add new note to existing request
- setRequestRemoteReferenceStatus - update the status in remote reference
- executeWFTransitions - execute work flow transition for the request

Input

A collection of changed fields, and the ID of the request.

Return

Upon success, the operation returns the ID of the request.

Java Interface

```
SetRequestFieldsResponseDocument
setRequestFields(SetRequestFieldsDocument in)
```

Parameters	Description
SetRequestFieldsDocument	Wrapper of the collection of request fields, or SimpleField[].
SetRequestFieldsResponseDocument	Wrapper of the return code of this operation.

Java Examples

Example: update several simple fields of an existing request.

```
public class ExampleSetRequestFields {
. . .
    private void setRequestFields(String serviceURL, String
requestId) throws Exception {
. . .
        // create the array of changed fields
        SimpleField[] fields = new SimpleField[2];

        // Set field 'Description'
       SimpleField field_A = SimpleField.Factory.newInstance();
        field_A.setToken("REQ.DESCRIPTION");
       field_A.setStringValue1Array(new String[] { "WebService
Test (Update)" });
        fields[0] = field_A;

        // Set field 'Department'
       SimpleField field_B = SimpleField.Factory.newInstance();
        field_B.setToken("REQ.DEPARTMENT_NAME");
        field_B.setStringValue1Array(new String[] { "IS" });
        fields[1] = field_B;

        // Get web service
        DemandServiceStub stub = new DemandServiceStub(ctx,
serviceURL);
        // Construct message to send
        SetRequestFieldsDocument inDoc =
SetRequestFieldsDocument.Factory.newInstance();
        SetRequestFieldsDocument.SetRequestFields
setRequestFields = inDoc
                .addNewSetRequestFields();
        setRequestFields.setRequestId(reqId);
        setRequestFields.setFieldsArray(fields);
. . .
    }
}
```

Errors and Exceptions

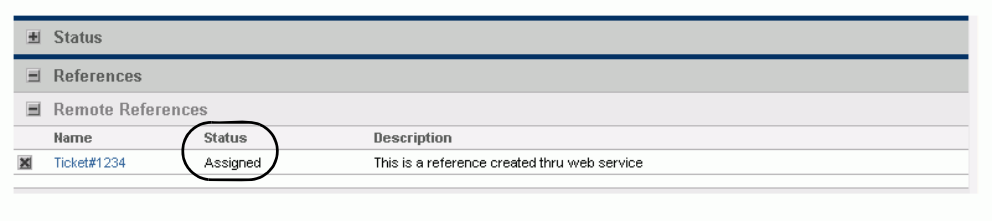
Message Code	Message	Cause(s)	Possible Corrective Action
KNTA-11186	ry.com/ppm/common/ 1.0}token"(line -1, col -1, in SOAP-message)	You have not provided the token.	Provide a token name to update.

Message Code	Message	Cause(s)	Possible Corrective Action
N/A	An error occurred when updating request fields-java.lang.Exception: Error 2: Error validating value "ISSP" for field REQ.DEPARTMENT_ NAME: no valid data was found for drop-down list.Error 2: Error validating value "ISSP" for field REQ.DEPARTMENT_ NAME: no valid data was found for drop-down list.	You have provided an invalid value for the token.	Provide a valid token value for update.
N/A	Open API Exception 4: The Request could not be locked because another party currently holds the lock.	Update collision.	Retry the operation.
N/A	An error occurred when updating request fields-java.lang.NumberFormatException: For input string: "30361A	You have provided an invalid request ID.	Provide a valid request ID.
KNTA-11186	y.com/ppm/dm/service/ 1.0}requestId	You have not provided the request ID.	Provide the request ID.
N/A	Error in loadAndLockRequest Open API Exception 9: Request with ID 30391 was not found in the system. Import was not performed.	The request ID you provided does not exists in the system.	Provide an existing request ID.

setRequestRemoteReferenceStatus

Purpose

This operation updates the status of a remote reference in a request in PPM Center, as illustrated in the following example:



Function

This operation provides the following functions.

- Update the status of the specified remote reference (the Source), in a PPM request (the Receiver). Remote reference is identified by the reference ID and server URL.
- Execute the workflow transition, if the given value of 'Status' exactly matches the transition name as appeared on the action buttons (case sensitive). This function is supported for backward compatibility. If the status does not match any of the transition names, no workflow transition will be performed. If you only want to perform workflow transition, use the *executeWFTransitions* operation.
- Update the simple fields of the request. This function is provided for backward compatibility. It is recommended to use *setRequestFields* operation to update simple fields.

Limitations

This operation has the following limitations:

- Only updates the status field of the remote reference. There is no operation to update the other remote reference fields.
- No security check on workflow transition. Even though the user cannot perform the transition from PPM Center standard interfaces, the transition will be performed with this Web service call, as long as the status value matches the transition name.
- Even when there is no simple field change, a simple field collection is still required when you call this operation.

For more information, see:

- setRequestFields - set the value for simple fields in the request.
- executeWFTransitions - execute work flow transition for the request.

Input

The following inputs are required for this operation:

- Receiver ID, which is the Identifier for the request in PPM Center.
- Source ID, which is to identify the remote reference. ServerURL is required.
- Status
- Simple field array. Even if there is no update on simple fields, it is still required.

Return

Upon success, the operation returns the ID of the request.

Java Interface

```
SetRequestRemoteReferenceStatusResponseDocument
setRequestRemoteReferenceStatus(SetRequestRemoteReferenceStatus
Document in)
```

Parameters	Description
SetRequestRemoteReferenceStatusDocument	Wrapper of the request ID, source ID, status, and the SimpleField[].
SetRequestRemoteReferenceStatusResponseDocument	Wrapper of the return code of this operation.

Java Examples

Example: update a remote reference of an existing request.

```
public class ExampleSetRequestFields {
. . .
    private void setRequestFields(String serviceURL, String
requestId) throws Exception {
. . .
        // create the array of changed fields
        SimpleField[] fields = new SimpleField[2];

        // Set field 'Description'
       SimpleField field_A = SimpleField.Factory.newInstance();
        field_A.setToken("REQ.DESCRIPTION");
       field_A.setStringValue1Array(new String[] { "WebService
Test (Update)" });
        fields[0] = field_A;

        // Set field 'Department'
       SimpleField field_B = SimpleField.Factory.newInstance();
        field_B.setToken("REQ.DEPARTMENT_NAME");
        field_B.setStringValue1Array(new String[] { "IS" });
        fields[1] = field_B;

        // Get web service
        DemandServiceStub stub = new DemandServiceStub(ctx,
serviceURL);
        // Construct message to send
        SetRequestFieldsDocument inDoc =
SetRequestFieldsDocument.Factory.newInstance();
        SetRequestFieldsDocument.SetRequestFields
setRequestFields = inDoc
                .addNewSetRequestFields();
        setRequestFields.setRequestId(reqId);
        setRequestFields.setFieldsArray(fields);
. . .
    }
}
```

Message Code	Message	Cause(s)	Possible Corrective Action
KNTA-11186	Internal error has occurred while calling PPM Web Service. Contact PPM support with the detail information if the problem persists. (KNTA-11186) Details: Missing required element {http://mercury.com/ppm/dm/service/1.0}fields	Missing simple fields.	Although this operation is not designed to update the request simple fields, a simple field array is required when calling this operation. Refer to the example provided in this section for how to add an empty simple field array.
N/A	Internal error has occurred while calling PPM Web Service. Contact PPM support with the detail information if the problem persists. (KNTA-11186) Details: Missing required element "{http://mercury.com/ppm/common/1.0}token"(line -1, col -1, in SOAP-message)	Missing token.	For a simple field, the token cannot be NULL. If for any reason we have to leave the token empty (for example, in case we have to supply an empty simple field array), we should set the token to an empty string.
N/A	An error occurred when updating remote reference status - java.lang.Exception: Error 3: Error applying remote reference: cannot apply a remote reference with NULL server URL	Missing ServerURL.	For identifying a remote reference, the ServerURL field of the Identifier of the remote reference is required.
KNTA-11186	//mercury.com/ppm/dm/1.0}id" start tag, found "{http://mercury.com/ppm/dm/1.0}serverURL" start tag (line -1, col -1, in SOAP-message).	You have not provided the request ID.	Provide the request ID.

Message Code	Message	Cause(s)	Possible Corrective Action
N/A	Internal error has occurred while calling PPM Web Service. Contact PPM support with the detail information if the problem persists. (KNTA-11186) Details: Missing required element {http://mercury.com/ppm/dm/service/1.0}receiver	You have not set the receiver ID.	Provide the request ID for the receiver.
N/A	Internal error has occurred while calling PPM Web Service. Contact PPM support with the detail information if the problem persists. (KNTA-11186) Details: Missing required element {http://mercury.com/ppm/dm/service/1.0}source	You have not set the Source.	Provide the Identifier with ID and server URL.
N/A	//mercury.com/ppm/dm/1.0}id" start tag, found "{http://mercury.com/ppm/dm/1.0}serverURL" start tag (line -1, col -1, in SOAP-message).	You have not provide the request ID.	Provide the reference request ID.
N/A	An error occurred when updating remote reference status - java.lang.Exception: Error 3: Error applying remote reference: cannot apply a remote reference with NULL server URL	You have not set the Server URL.	Provide the server URL for the identifier.

Message Code	Message	Cause(s)	Possible Corrective Action
N/A	Internal error has occurred while calling PPM Web Service. Contact PPM support with the detail information if the problem persists. (KNTA-11186) Details: Missing required element {http://mercury.com/ppm/dm/service/1.0}status	You have not set the Status.	Provide the status for the remote reference.

addRequestNotes

Purpose

This operation adds one or more notes into a request in PPM Center.

Function

This operation adds one or more notes into a specified request. The following fields can be specified for each note:

- Author, which can be any string value and is displayed in the **originally added by** entry. The value for the **Note Author** field is the user account used when you call the web service server.
- Creation Date
- Content

The notes are grouped into a daily bucket and displayed in date order. Request note cannot be edited or deleted once added.

Input

The following inputs are required for this operation:

- Request ID
- Collection of one or more notes

Return

Upon success, the operation returns the ID of the request.

Java Interface

```
AddRequestNotesResponseDocument
addRequestNotes(AddRequestNotesDocument in)
```

Parameters	Description
AddRequestNotesDocument	Wrapper of the request ID and the Note[].
AddRequestNotesResponseDocument	Wrapper of the return code of this operation.

Java Examples

Example: add two notes to an existing request.

```
public class ExampleAddRequestNotes {
. . .
    private void addRequestNotes(String serviceURL, String
requestId) throws Exception {
. . .
        // construct the note array
        Note[] notes = new Note[2];
        notes[0] = Note.Factory.newInstance();
        notes[0].setAuthor("Admin");
        notes[0].setCreationDate(Calendar.getInstance());
        notes[0].setContent("Note 1 added thru web service");
        notes[1] = Note.Factory.newInstance();
        notes[1].setAuthor("Admin");
        notes[1].setCreationDate(Calendar.getInstance());
        notes[1].setContent("Note 2 added thru web service");
        // get Webservice handle
        DemandServiceStub stub = new DemandServiceStub(ctx,
serviceURL);
        // Construct message to send
        AddRequestNotesDocument inDoc =
AddRequestNotesDocument.Factory.newInstance();
        AddRequestNotesDocument.AddRequestNotes addNotes =
inDoc.addNewAddRequestNotes();
        addNotes.setRequestId(reqId);
        addNotes.setNotesArray(notes);
. . .
    }
}
```

Errors and Exceptions

Message Code	Message	Cause(s)	Possible Corrective Action
KNTA-11186	Internal error has occurred while calling PPM Web Service. Contact PPM support with the detail information if the problem persists. (KNTA-11186) Details: Missing required element {http://mercury.com/ppm/dm/service/1.0}requestId	Missing request ID	Add the required request ID.

Message Code	Message	Cause(s)	Possible Corrective Action
N/A	An error occurred when adding notes to request Error in loadAndLockRequest Open API Exception 9: Request with ID 30391 was not found in the system. Import was not performed	Request ID not found	Provide an existing request ID.
N/A	java.lang.NullPointerException	Missing author, creation date or content	Provide complete data (author, creation date and content).
N/A	java.lang.IllegalArgumentException: Array element null	Note array element is not initialized or is null	Initialize array element with a Note object that has complete data.

executeWFTransitions

Purpose

This operation executes the workflow transition on a request in PPM Center.

Function

This operation executes the workflow transition on a request.

Limitations

This operation has the following limitations:

- Supports only the decision step. It does not support the execution step.
- Does not support branching; it will fail if there is a branch on the same transition.

- No security check on workflow transition. Even though the user may not be allowed to perform the transition from PPM standard interfaces, or the workflow step may require reauthentication, the transition will be performed by this Web service call, as long as the status value matches the transition name.

Input

The following inputs are required for this operation:

- Receiver ID, which is the identifier for the request in PPM Center.
- Transition name, which should exactly match the value on the action buttons (case sensitive).

Return

When the transition is valid and executed, the operation returns the following string:

```
Workflow transition was successful.
```

When the transition is invalid, the operation returns the following string:

```
There were no matches for the transition name provided.
```

Java Interface

```
ExecuteWFTransitionsResponseDocument
executeWFTransitions(ExecuteWFTransitionsDocument in)
```

Parameters	Description
ExecuteWFTransitionsDocument	Wrapper of the request ID and the transition name.
ExecuteWFTransitionsResponseDocument	Wrapper of the return code of this operation.

Java Examples

Example: execute a workflow transition on an existing request.

```
public class ExampleSetExecuteWFTransitions {
. . .
```

```
   private void executeWFTransitions(String serviceURL, String
requestId,
            String transition) throws Exception {

        // construct the Identifiers
        Identifier receiverId =
Identifier.Factory.newInstance();
        receiverId.setId(requestId);

        // get Webservice handle
        DemandServiceStub stub = new DemandServiceStub(ctx,
serviceURL);

        // Construct message to send
        ExecuteWFTransitionsDocument inDoc =
ExecuteWFTransitionsDocument.Factory
                .newInstance();
      ExecuteWFTransitionsDocument.ExecuteWFTransitions exec =
inDoc
                .addNewExecuteWFTransitions();
        exec.setReceiver(receiverId);
        exec.setTransition(transition);
. . .
    }
}
```

Errors and Exceptions

Message Code	Message	Cause(s)	Possible Corrective Action
N/A	Update remote reference status Succeeded. Return code: There were no matches for the transition name provided.	The transition has been provided but there are no matches for the transition name.	Provide a matching transition for the request.
KNTA-11186	Internal error has occurred while calling PPM Web Service. Contact PPM support with the detail information if the problem persists. (KNTA-11186) Details: Missing required element {http://mercury.com/ppm/dm/service/1.0}transition	You have not provided the transition.	Provide the transition for the request

Message Code	Message	Cause(s)	Possible Corrective Action
KNTA-11186	Internal error has occurred while calling PPM Web Service. Contact PPM support with the detail information if the problem persists. (KNTA-11186) Details: Missing required element {http://mercur y.com/ppm/dm/service/ 1.0}receiver	You have not set the request ID.	Provide the request ID for the request.
KNTA-11186	Internal error has occurred while calling PPM Web Service. Contact PPM support with the detail information if the problem persists. (KNTA-11186) Details: null; nested exception is:java.lang.NullPointerException	The request ID value is invalid.	The request ID value has to be a numeric value. Correct the value and try again.

deleteRequests

Purpose

This operation deletes one or more requests in PPM Center.

Function

This operation is to delete one or more requests in PPM Center.

Limitations

No security check on this operation. The specified request(s) will be deleted by this Web service operation even if the user does not have permissions to delete the request.

Input

A collection of request IDs.

Return

Return number of requests deleted.

Java Interface

```
DeleteRequestsResponseDocument
deleteRequests(DeleteRequestsDocument in)
```

Parameters	Description
DeleteRequestsDocument	Wrapper of a list of request ID, or Identifier[].
DeleteRequestsResponseDocument	Wrapper of the return code of this operation.

Java Examples

Example: update a remote reference of an existing request.

```
public class ExampleDeleteRequests {
. . .
    private void deleteRequests(String serviceURL, String[]
requestIds) throws Exception {

        // construct the Identifier array
        Identifier[] ids = new Identifier[requestIds.length];
        for (int i = 0; i < requestIds.length; i++) {
            ids[i] = Identifier.Factory.newInstance();
            ids[i].setId(requestIds[i]);
        }

        // get Webservice handle
        DemandServiceStub stub = new DemandServiceStub(ctx,
serviceURL);
        // Construct message to send
        DeleteRequestsDocument inDoc =
DeleteRequestsDocument.Factory.newInstance();
        DeleteRequestsDocument.DeleteRequests deletes =
inDoc.addNewDeleteRequests();
        deletes.setRequestIdsArray(ids);
. . .
    }
}
```

Message Code	Message	Cause(s)	Possible Corrective Action
KNTA-11186	Internal error has occurred while calling PPM Web Service. Contact PPM support with the detail information if the problem persists. (KNTA-11186) Details: Non digit data was set for number field. Original exception message: For input string: "30364A".	The request ID was invalid (includes non-numeric data).	Provide a valid request ID.
KNTA-11186	Internal error has occurred while calling PPM Web Service. Contact PPM support with the detail information if the problem persists. (KNTA-11186) Details: Non digit data was set for number field. Original exception message: For input string: ""	The request ID is null.	Provide a valid request ID.
N/A	Delete Requests Succeeded. Return Code: 0	The request ID is not found in the system.	Provide an existing request ID.

2 HP Financial Management (Budgets) Web Services

Overview

HP Financial Management Web services for budgets provides operations to create, update, and read budget entities in PPM Center.

Budgets can be used to track financial information for a project, program, organization unit, or other entity. Budgets can be linked to these entities with varying levels of data dependencies.

The budget functionality is replaced by financial summaries in PPM Center 8.00. Therefore, the Web service operations described in this chapter are not available in PPM Center 8.00.

For more information about general HP Financial Management terms and concepts, see the *HP Financial Management User's Guide (version 7.5)*.

References

Data types and operations definitions:

`webservice_toolkit\java\conf\wsdl60\`**`Finance.wsdl`**

Java sample code:

`webservice_toolkit\java\client\src\examples\fm60\`
`BudgetServiceClient.java`

Operations History

HP Financial Management Web services for budgets provides many operations starting with PPM Center version 6.0. *Table 2-1* lists the HP Financial Management Web services for budgets operations by version.

Table 2-1. HP Financial Management Web Service for Budgets Operations

Web Service operation	6.0	7.1	7.5	8.00
create	Yes	Yes	Yes	No
update	Yes	Yes	Yes	No
read	Yes	Yes	Yes	No
ReadExtended	No	Yes	Yes	No
Read (BudgetFilter was extended to add more filter criteria)	No	Yes	Yes	No

Data Types

HP Financial Management Web services for budgets includes the following data types:

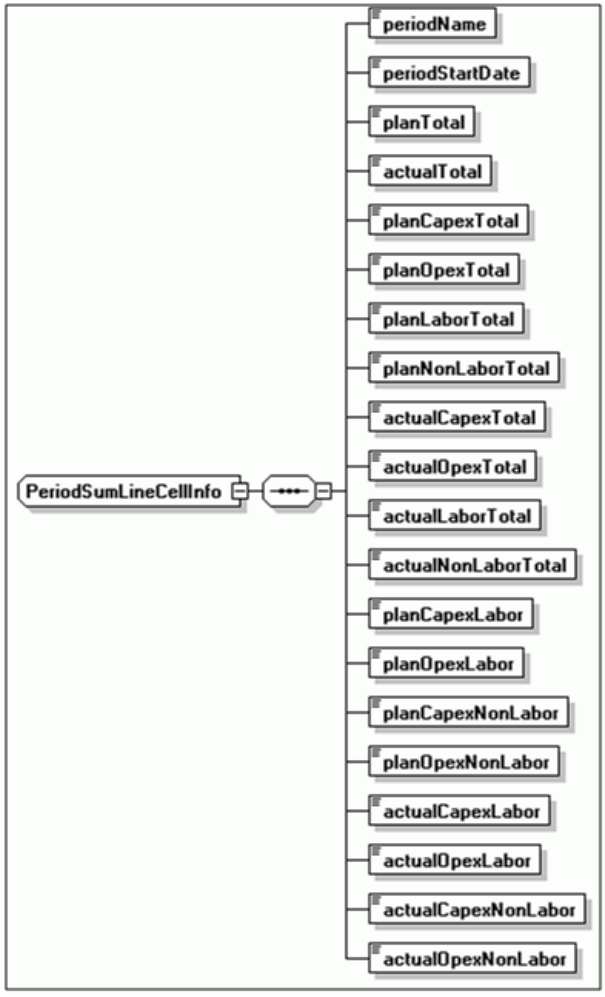
- *Budget* on page 71
- *BudgetLine* on page 74
- *BudgetLineDetail* on page 76
- *BudgetFilters* on page 76
- *ExtendedBudget* on page 82
- *BudgetPeriodSumLine* on page 84

Budget

The Budget data type is a value object representing the budget information.

For more information about the Budget data type, see:

- Budget is used as the INPUT in the following operations:
  - Operation *create*
  - Operation *update*
- Budget is used as the OUTPUT in the following operation:
  - Operation *read*



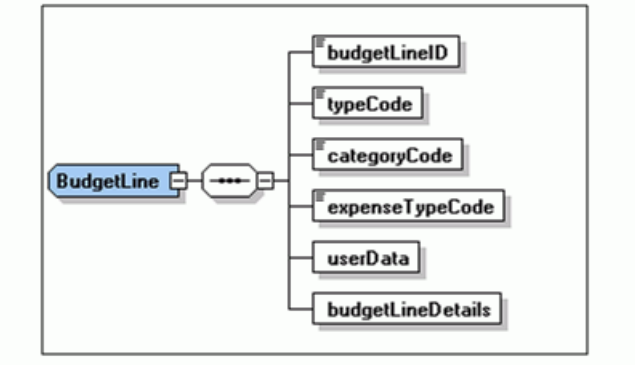
Property	Type	Description	Required	Default
budgetId	Integer	Budget ID. An optional numeric value which is normally greater than 30,000.	No	N/A
budgetName	String	Budget Name. A required string field for the name of the new budget. Must be unique.	Yes	N/A
capexOpexEnabledFlag	Boolean	A flag to specify budget capitalized cost. An optional boolean value.	No	false
actualsRolledUpCode	String	Actuals rolled up code. An optional string value. Valid values are: • MANUAL • AUTO_LABOR_ AUTO_NONLABOR • AUTO_LABOR_ MANUAL_ NONLABOR	No	MANUAL
Active	Boolean	An active flag. An optional boolean value.	No	false
associatedWithType	String	Type that budget is associated with. Valid values are: • ORGANIZATION UNIT • PROGRAM • PROJECT • PROPOSAL • ASSET	No	N/A

Property	Type	Description	Required	Default
associatedWithName	String	Name of the associated entity.	No	N/A
startPeriodName	String	Start period name. An optional string value.	No	N/A
endPeriodName	String	End period name. An optional string value.	No	N/A
startPeriodStartDate	Calendar	Start date of start period. A required date value.	Yes	N/A
endPeriodStartDate	Calendar	Start date of end period. A required date value.	Yes	N/A
periodType	String	Period Type. A required string value. Valid values are: • FISCAL_MONTH • QUARTER	Yes	N/A
description	String	Budget Description.	No	N/A
budgetStatus	String	Budget status. A required string value. Valid values are: • NEW • PROPOSED • UNDER_PREVIEW • IN_REWORK • CANCELLED • ON_HOLD • APPROVED • CLOSED	Yes	N/A

Property	Type	Description	Required	Default
regionName	String	Budget region name. A required string value that has to be a valid name.	Yes	N/A
baseCurrency	String	Budget base currency name. An optional string value.	No	N/A
userData	Field[]	Budget user data. An optional array of the Field type.	No	N/A
budgetLines	BudgetLine[]	Budget lines. An optional array of the *BudgetLine* type.	No	N/A

BudgetLine

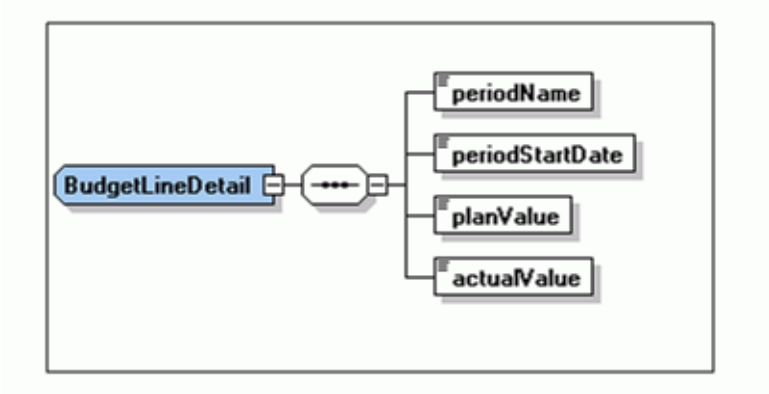
The BudgetLine data type is a value object representing the budget line information. A budget line is defined with a budget line type, the corresponding category, and the expense type.



Property	Type	Description	Required	Default
budgetLineID	Integer	Budget Line ID. A numeric value which is normally greater than 30,000. Set to NULL when it is created.	No	N/A
typeCode	String	Budget line type code. A required string value. Valid values are: • LABOR • NON_LABOR	Yes	LABOR
categoryCode	String	Budget line category code. A required string value. The line type code and line category code must match. If they do not match, the line type does not display in the UI and it might corrupt budget-based reporting. For example, the `NON_LABOR` line type code does NOT match the `EMPLOYEE` line category code.	Yes	N/A
expenseTypeCode	String	Budget line expense type code. An optional string value. Valid values are: • CAPITAL • OPERATING	Yes	OPERATING
userData	Field[]	Budget line user data. An optional array of the Field type.	No	N/A
budgetLineDetails	BudgetLineDetail[]	Budget line details. An optional array of the *BudgetLineDetail* type.	No	N/A

BudgetLineDetail

The BudgetLineDetail data type is a value object representing the budget line detail information. The budget line contains the planned budget amount and the actual budget amount of a particular budget line.



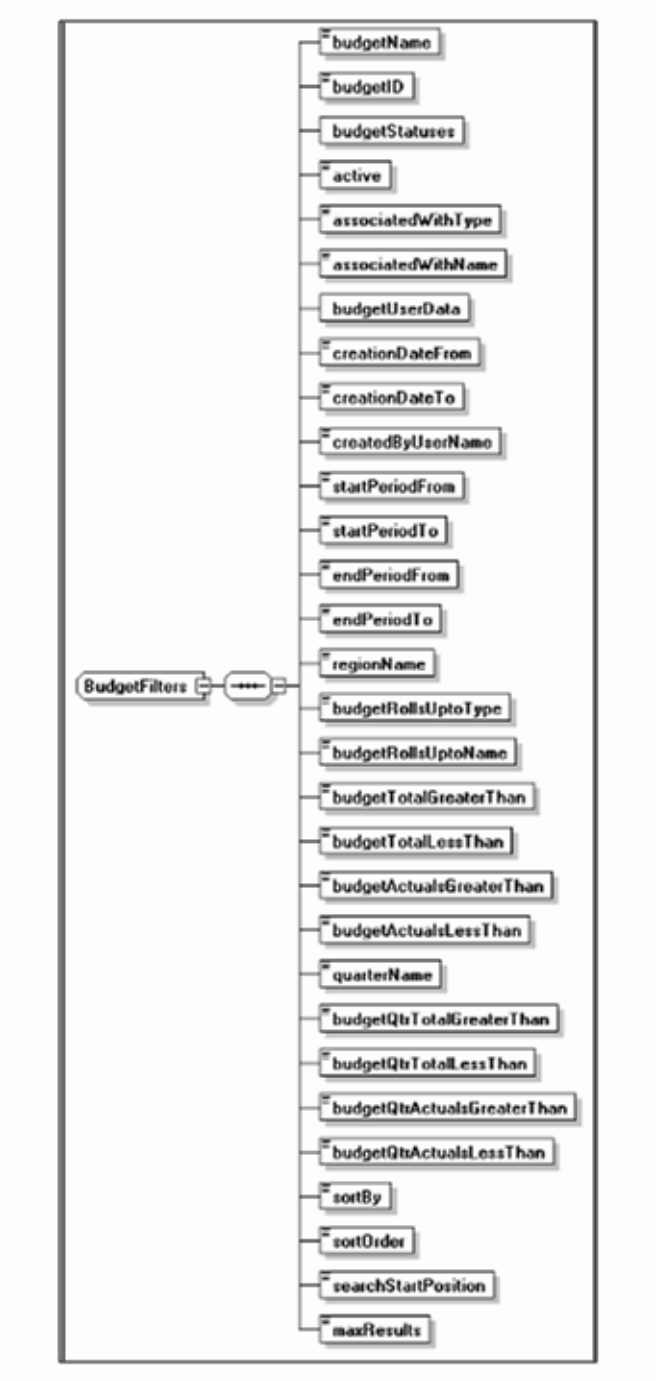
Property	Type	Description	Required	Default
periodName	String	Name of the budget line period. An optional string value.	No	N/A
periodStartDate	Calendar	Budget line period start date. A required Calendar value.	Yes	N/A
planValue	BigDecimal	Budget line planned value. An optional BigDecimal value.	No	N/A
actualValue	BigDecimal	Budget line actual value. An optional BigDecimal value.	No	N/A

BudgetFilters

The BudgetFilters bean is used to search for budgets in the system.

BudgetFilters is used as the INPUT in the following operations:

- Operation *read*
- Operation *readExtended*


BudgetFilters
budgetName
budgetID
budgetStatuses
active
associatedWithType
associatedWithName
budgetUserData
creationDateFrom
creationDateTo
createdByUserName
startPeriodFrom
startPeriodTo
endPeriodFrom
endPeriodTo
regionName
budgetRollsUptoType
budgetRollsUptoName
budgetTotalGreaterThan
budgetTotalLessThan
budgetActualsGreaterThan
budgetActualsLessThan
quarterName
budgetQtrTotalGreaterThan
budgetQtrTotalLessThan
budgetQtrActualsGreaterThan
budgetQtrActualsLessThan
sortBy
sortOrder
searchStartPosition
maxResults

Property	Type	Description	Required	Default
budgetName	String	Return budget with the specified budget name.	No	N/A
budgetID	Integer	Return budget with the specified budget ID.	No	N/A
budgetStatuses	String[]	Return budgets with the specified statuses.	No	N/A
active	Boolean	Return budgets with specified active flag.	No	N/A
associatedWithType	String	Return budgets with the specified associated type.	No	N/A
associatedWithName	String	Return budgets with the specified associated entity name.	No	N/A
userData	Field[]	Return budgets with the specified user data.	No	N/A
creationDateFrom	Date	Return budgets from the specified creation date. This search filter was introduced at PPM Center 7.5.	No	N/A
creationDateTo	Date	Return budgets up to the specified creation date. This search filter was introduced at PPM Center 7.5.	No	N/A
createdByUserName	String	Return budgets created by the specified user name. This search filter was introduced at PPM Center 7.5.	No	N/A
startPeriodFrom	String	Return budgets from the specified start period. This search filter was introduced at PPM Center 7.5.	No	N/A

Property	Type	Description	Required	Default
startPeriodTo	String	Return budgets up to the specified start period. This search filter was introduced at PPM Center 7.5.	No	N/A
endPeriodFrom	String	Return budgets from the specified end period. This search filter was introduced at PPM Center 7.5.	No	N/A
endPeriodTo	String	Return budgets up to the specified end period. This search filter was introduced at PPM Center 7.5.	No	N/A
regionName	String	Return budgets with the specified region name. This search filter was introduced at PPM Center 7.5.	No	N/A
budgetRollsUpToType	String	Return budgets with the specified roll-up type. This search filter was introduced at PPM Center 7.5.	No	N/A
budgetRollsUpToName	String	Return budgets with the specified roll-up name. This search filter was introduced at PPM Center 7.5.	No	N/A
budgetTotalGreaterThan	Integer	Return budgets with total greater than the specified amount. This search filter was introduced at PPM Center 7.5.	No	N/A

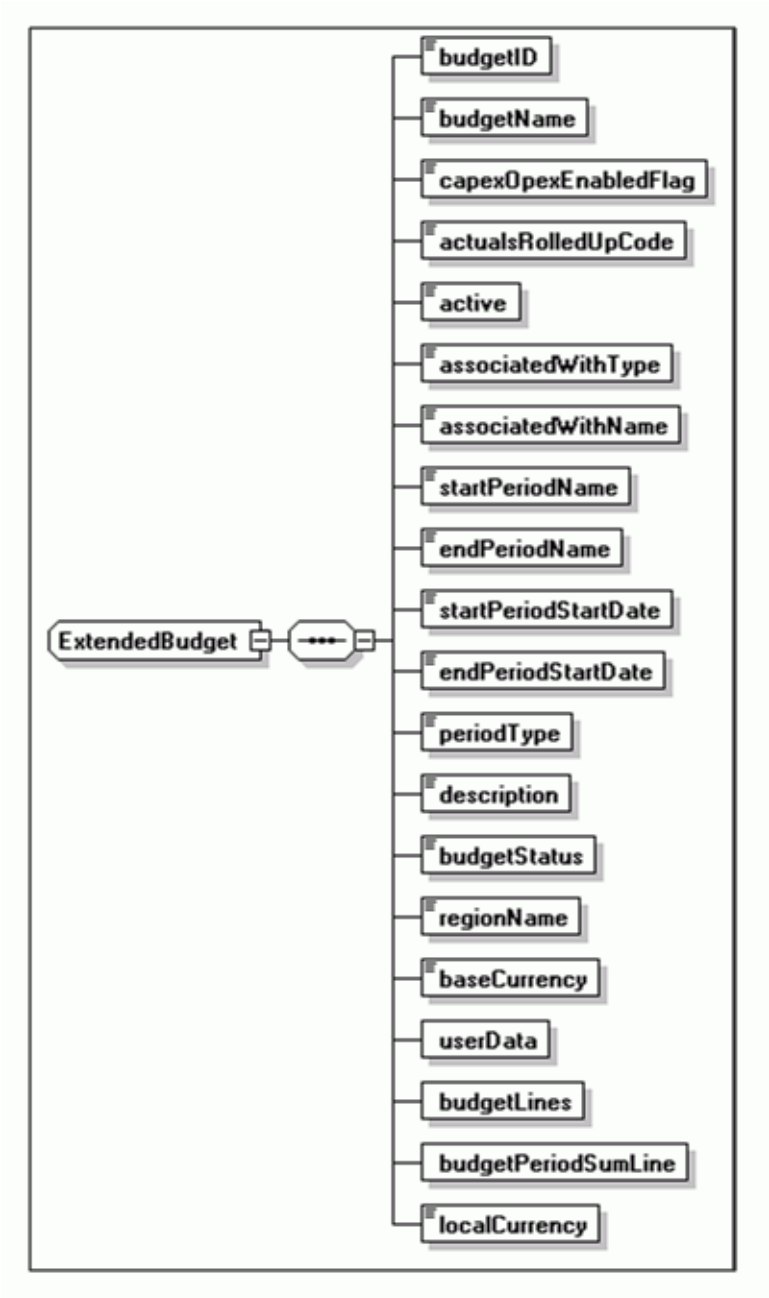
Property	Type	Description	Required	Default
budgetTotalLessThan	Integer	Return budgets with total less than the specified amount. This search filter was introduced at PPM Center 7.5.	No	N/A
budgetActualsGreaterThan	Integer	Return budgets with actuals greater than the specified amount. This search filter was introduced at PPM Center 7.5.	No	N/A
budgetActualsLessThan	Integer	Return budgets with actuals less than the specified amount. This search filter was introduced at PPM Center 7.5.	No	N/A
quarterName	String	Return budgets for the specified quarter. This search filter was introduced at PPM Center 7.5.	No	N/A
budgetQtrTotalGreaterThan	Integer	Return budgets with quarter total greater than the specified amount. This search filter was introduced at PPM Center 7.5.	No	N/A
budgetQtrTotalLessThan	Integer	Return budgets with quarter total less than the specified amount. This search filter was introduced at PPM Center 7.5.	No	N/A

Property	Type	Description	Required	Default
budgetQtrActualsGreaterThan	Integer	Return budgets with quarter actuals greater than the specified amount. This search filter was introduced at PPM Center7.5.	No	N/A
budgetQtrActualsLessThan	Integer	Return budgets with quarter actuals less than the specified amount. This search filter was introduced at PPM Center 7.5.	No	N/A
sortBy	String	Return budgets sorted by the specified field. This search filter was introduced at PPM Center 7.5.	No	N/A
sortOrder	String	Return budgets sorted by the specified order. A for ascending and Z for descending. This search filter was introduced at PPM Center 7.5.	No	N/A
searchStartPosition	Integer	Return budgets starting from the specified position. This search filter was introduced at PPM Center 7.5.	No	N/A
maxResults	Integer	Return budgets with specified maximum number of records. This search filter was introduced at PPM Center 7.5.	No	N/A

ExtendedBudget

The ExtendedBudget data type is a value object representing the budget information including the budget cost sum rows.

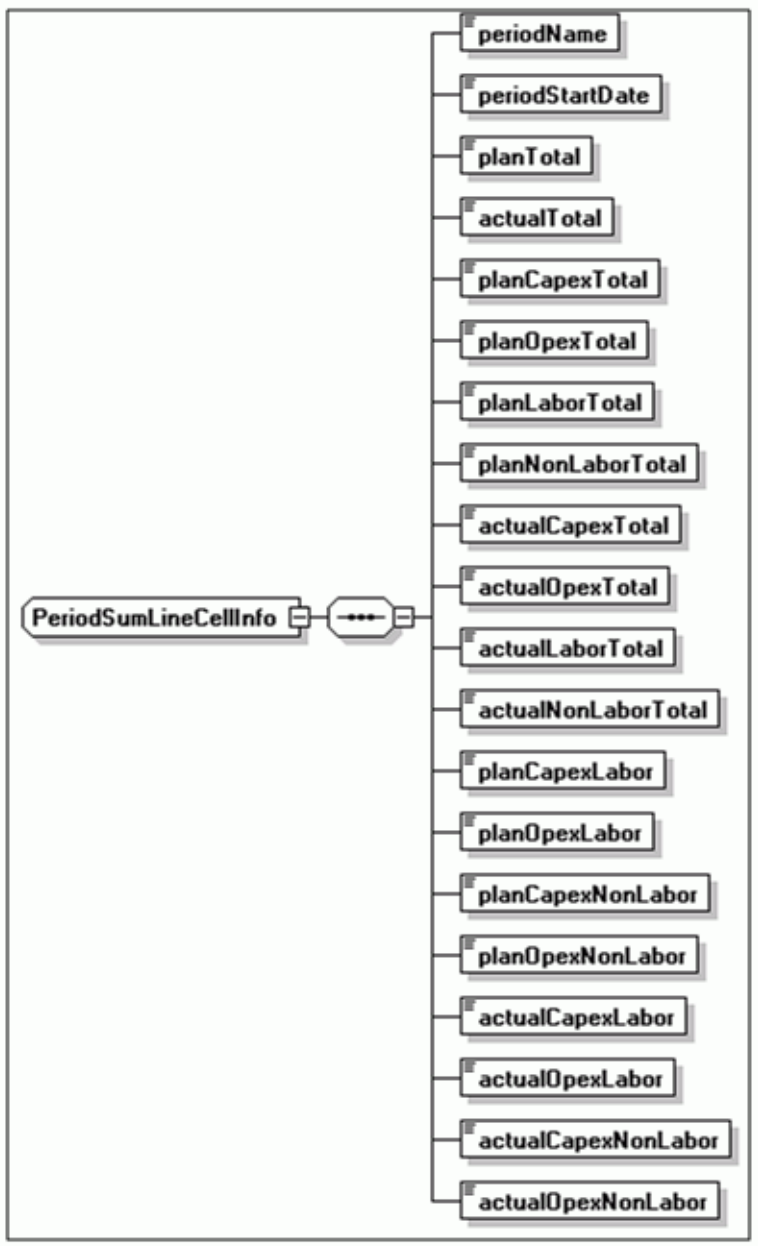
ExtendedBudget is used as the OUTPUT in the *readExtended* operation.


ExtendedBudget
budgetID
budgetName
capexOpexEnabledFlag
actualsRolledUpCode
active
associatedWithType
associatedWithName
startPeriodName
endPeriodName
startPeriodStartDate
endPeriodStartDate
periodType
description
budgetStatus
regionName
baseCurrency
userData
budgetLines
budgetPeriodSumLine
localCurrency

Property	Type	Description	Required	Default
budgetPeriodSumLine	BudgetPeriodSumLine	A value of the *BudgetPeriodSumLine* type.	No	N/A
localCurrency	String	Budget local currency name.	No	N/A

BudgetPeriodSumLine

The BudgetPeriodSumLine data type is a value object representing the budget for the planned and actual cost sum for different budget line types. This data type is used in ExtendedBudget.


periodName
periodStartDate
planTotal
actualTotal
planCapexTotal
planOpexTotal
planLaborTotal
planNonLaborTotal
actualCapexTotal
actualOpexTotal
PeriodSumLineCellInfo
actualLaborTotal
actualNonLaborTotal
planCapexLabor
planOpexLabor
planCapexNonLabor
planOpexNonLabor
actualCapexLabor
actualOpexLabor
actualCapexNonLabor
actualOpexNonLabor

Property	Type	Description	Required	Default
periodName	String	Name of a period. An optional string value.	No	N/A
periodStartDate	String	The start date of a period.	No	N/A
planTotal	decimal	The total planned value of a budget.	No	N/A
actualTotal	decimal	The total actual value of a budget.	No	N/A
planCapexTotal	decimal	The total planned CAPEX (Capitalized) value of a budget.	No	N/A
planOpexTotal	decimal	The total planned OPEX (Operating) value of a budget.	No	N/A
planLaborTotal	decimal	The total planned labor cost of a budget.	No	N/A
planNonLaborTotal	decimal	The total planned non-labor cost of a budget	No	N/A
actualCapexTotal	decimal	The total actual CAPEX value of a budget.	No	N/A
actualOpexTotal	decimal	The total actual OPEX value of a budget.	No	N/A
actualLaborTotal	decimal	The total actual labor cost of a budget.	No	N/A
actualNonLaborTotal	decimal	The total actual non-labor cost of a budget.	No	N/A
planCapexLabor	decimal	The total planned CAPEX labor cost of a budget.	No	N/A
planOpexLabor	decimal	The total planned OPEX labor cost of a budget.	No	N/A
planCapexNonLabor	decimal	The total planned CAPEX non-labor cost of a budget.	No	N/A

Property	Type	Description	Required	Default
planOpexNonLabor	decimal	The total planned OPEX non-labor cost of a budget.	No	N/A
actualCapexLabor	decimal	The total actual CAPEX labor cost of a budget.	No	N/A
actualOpexLabor	decimal	The total actual OPEX labor cost of a budget.	No	N/A
actualCapexNonLabor	decimal	The total actual CAPEX non-labor cost of a budget.	No	N/A
actualOpexNonLabor	decimal	The total actual OPEX non-labor cost of a budget.	No	N/A

Operations

The following operations are included in HP Financial Management Web services for budgets:

- *read* on page 88
- *create* on page 92
- *update* on page 96
- *readExtended* on page 100

read

Purpose

This operation is used to query PPM Center for a list of budgets.

Function

This operation allows a remote system to call PPM Center and retrieve the budget data.

Limitations

Performance can be an issue if more than a few thousand budgets are read at once.

The web service does not provide the user security check for reading budgets. This means that in cases where the log-on user does not have read permissions for budgets in PPM Center, the user can read budget data by using this operation.

Related Information

The *readExtended* operation reads budgets with cost sum rows.

Input

This operation requires the following inputs:

- ReadMessage object that contains the header.
- Filter object to describe the filter conditions for the query.

Return

This operation returns the following information:

- ReadResponse object that contains the header.
- Response message
- The budget data

Java Interface

```
ReadResponse read (ReadMessage readMessage)
```

Parameters	Description
ReadMessage	Wrapper of RequestHeader and BudgetFilters. See the following example for retrieving the bean and its fields.
ReadResponse	Wrapper of the collection of budgets, or Budget[]. See the following example for the construction.

Java Examples

Example: get data for a list of budget IDs

```
public class ExampleReadBudget {
    static String username, password;
    public static void main(String[] args) throws Exception {
System.setProperty("java.protocol.handler.pkgs","com.sun.net.ss
l.internal.www.protocol");
        Security.addProvider(new
com.sun.net.ssl.internal.ssl.Provider());
        System.setProperty("javax.net.ssl.keyStorePassword",
"password");
        if (args.length != 4) {
            System.out.println("Usage :\n" +
                    "ReadBudgetsTest \"BaseURL\" \"Username\" \
"Password\" \"parameterFile\"");
            System.exit(0);
        }
        String URL = args[0];
        username = args[1];
        password = args[2];

        Properties properties = new Properties();
```

```
        properties.load(new FileInputStream(args[3]));

        FinanceServicesServiceLocator locator = new
FinanceServicesServiceLocator();
        FinanceServices itg = locator.getFinance(new URL(URL +
"/services/Finance"));

        ReadMessage readMessage = new ReadMessage();
        readMessage.setHeader(createHeader());
        BudgetFilters filter = new BudgetFilters();
        if (properties.getProperty("budgetName") != null
                &&
!"".equals(properties.getProperty("budgetName"))) {

filter.setBudgetName(properties.getProperty("budgetName"));
        }
        if (properties.getProperty("budgetID") != null
                &&
!"".equals(properties.getProperty("budgetID"))) {
            filter.setBudgetID(new
Integer(properties.getProperty("budgetID")));
        }
        //Set the below properties according to budgetName,
budgetID
        // associatedWithType
        // associatedWithName
        // active
        // region
        // creationDateFrom
        // creationDateTo
        // startPeriodFrom
        // startPeriodTo
        // endPeriodFrom
        // endPeriodTo
        // createdBy
        // budgetRollsUptoType
        // budgetRollsUptoName
        // budgetTotalGreaterThan
        // budgetTotalLessThan
        // budgetActualsGreaterThan
        // budgetActualsLessThan
        // quarterName
        // budgetQtrTotalGreaterThan
        // budgetQtrTotalLessThan
        // budgetQtrActualsGreaterThan
        // budgetQtrActualsLessThan
// statusCount
        // sortBy
        // sortOrder
        // searchStartPosition
        // maxResults
        // userdataSize
        readMessage.setBudgetFilters(filter);
        Budget[] budgets = itg.read(readMessage).getBudget();
```

```
        if (budgets == null || budgets.length == 0) {
            System.out.println("Budget not found");
        } else if (budgets != null && budgets.length > 0) {
            for (int i = 0; i < budgets.length; i++) {

System.out.println("budget.getBudgetID()="+budget.getBudgetID()
);

System.out.println("budget.getBudgetName()="+budget.getBudgetNa
me());

            }
            System.out.println("Total " + budgets.length + "
found matching your criteria");
        }
    }
    private static RequestHeader createHeader() throws
Exception {
        RequestHeader header = new RequestHeader();
        header.setAuditNote("Submitted by " + username + " on
APIClient");
        header.setOrigin("APIClient on " +
InetAddress.getLocalHost().getHostAddress());
        header.setTransactionName(new UID().toString());
        LoginAccount loginAccount = new LoginAccount();
        loginAccount.setUsername(username);
        loginAccount.setPassword(password);
        header.setCredentials(loginAccount);

        return header;
    }
}
```

Errors and Exceptions

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
INVALID_SORT_BY_OPTION	{0} is not a valid Sort By option.	The user uses an invalid Sort By option.	The Sort By option should match the field listed in BudgetFilter

create

Purpose

This operation is used to create a new budget in PPM Center.

Function

This operation creates a new budget in PPM Center. You can specify almost all of the data fields of a budget in the input object of this function. However, if a budget with the specified name already exists, an error is generated.

Limitations

The web service does not provide the user security check for creating budgets. This means that in cases where the log-on user does not have create permissions for budgets in PPM Center, the user can create budget data by using this operation.

Related Information

The *update* operation updates fields of an existing budget.

Input

This operation requires the following inputs:

- A CreateMessage object which contains the header.
- A budget object to be created.

Return

This operation returns the following information:

- A CreateResponse object that contains the header.
- A response message
- A budget ID

Java Interface

```
CreateResponse create (CreateMessage createMessage)
```

Parameters	Description
CreateMessage	Wrapper of RequestHeader and Budget.
CreateResponse	Wrapper of the created budget ID.

Java Examples

Example: get data for a single request:

```
private static void testCreateBudget(RequestHeader header,
String budgetName)
        throws ServiceException, RemoteException,
RemoteException, ParseException {
        CreateMessage createMessage = new CreateMessage();
        createMessage.setHeader(header);
        Budget budget = new Budget();
        //Set the budget name
        budget.setBudgetName(budgetName +
System.currentTimeMillis());
        //Set the description
        budget.setDescription(budget.getBudgetName());
        //Set the capex opex enabled flag
        budget.setCapexOpexEnabledFlag(new Boolean(true));
        //Set the region
        budget.setRegionName("Enterprise");
         Calendar startDate = Calendar.getInstance();
        SimpleDateFormat sdf     = new SimpleDateFormat("yyyy-
MM-dd");
        //Set the budget start date
        Date date = sdf.parse("2005-2-1");
        startDate.setTime(date);
        budget.setStartPeriodStartDate(startDate);
        //Set the budget end date
        date = sdf.parse("2005-4-1");
        Calendar endDate = Calendar.getInstance();
        endDate.setTime(date);
        budget.setEndPeriodStartDate(endDate);
        //Set actuals roll up flag

budget.setActualsRolledUpCode(com.kintana.cst.bean.Budget.ACTUA
LS_MANUAL);
        //Set period type, only Fiscal Month is supported
        budget.setPeriodType("Fiscal Month");
        //Set budget to inactive
        budget.setActive(Boolean.FALSE);
        budget.setBudgetStatus("New");
```

```
        //budget.setAssociatedWithName("A");
        //budget.setAssociatedWithType("Proposal");
//Create one line in budget
        BudgetLine[] budgetLines = new BudgetLine[1];
        budgetLines[0] = new BudgetLine();
        //Set the budget category and labor type
        budgetLines[0].setCategoryCode("EMPLOYEE");
        budgetLines[0].setTypeCode("LABOR");
        budgetLines[0].setExpenseTypeCode("OPERATING");
        //Create the budget line details, populate 2 cells
        BudgetLineDetail[] budgetLineDetails = new
BudgetLineDetail[2];
        budgetLineDetails[0] = new BudgetLineDetail();
        budgetLineDetails[0].setPeriodStartDate(startDate);
        budgetLineDetails[0].setActualValue(new
BigDecimal(10000));
        budgetLineDetails[0].setPlanValue(new
BigDecimal(20000));

        budgetLineDetails[1] = new BudgetLineDetail();
        budgetLineDetails[1].setPeriodStartDate(startDate);
        budgetLineDetails[1].setActualValue(new
BigDecimal(30000));
        budgetLineDetails[1].setPlanValue(new
BigDecimal(40000));
        budgetLines[0].setBudgetLineDetails(budgetLineDetails);
        budget.setBudgetLines(budgetLines);
        createMessage.setBudget(budget);
        CreateResponse createResponse =
financeServices.create(createMessage);
        System.out.println("created budget id=
"+createResponse.getBudgetID());
        System.out.println("create
returns:"+createResponse.getResponseMessage().getMessage());
    }
```

Errors and Exceptions

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
PERIOD_TYPE_ UNSUPPORTED	Unsupported period specified. Please use Fiscal Month.	The period type is not set to Fiscal Month.	Use Fiscal Month as the period type.
AUTO_ ROLLUP_NOT_ POSSIBLE	This is a {0} budget. Actuals will not be automatically rolled up.	The budget type is one of the following: • Proposal • Org Unit • Asset actualRollUpCode is set to "automatically."	When you create a budget of this type, do not set actualRollUpCode to "automatically."
START_END_ PERIOD_ MANDATORY	Unable to create the budget. Please specify budget start and end date.	The startPeriodStartDate and the endPeriodStartDate are not specified.	Enter the startPeriodStartDate and the endPeriodStartDate.
BUDGET_NOT_ ASSOCIATED	Unable to make the budget active. It must first be associated with an entity.	The budget is set to "active," and no entity is associated with it.	Identify the entity for the budget before setting it to "active."
BUDGET_NOT_ ACTIVE	Unable to make the budget active. Budget has to be approved to be active.	The status of the budget is not "Approved."	Set the status of the budget to "Approved."
REGION_ MANDATORY	Unable to create the budget. Please specify the region.	The region is not specified.	Enter the region.
DUP_BUDGET	A Budget with the same name already exists.	You tried to create a budget with an already existing name.	Specify another name of the budget.

update

Purpose

This operation updates an existing budget in PPM Center.

Function

To update a budget, you must provide a valid budget ID. Otherwise, an error is generated.

To update a budget line, you must provide:

- Valid budget line ID.
- Period name for the line. The values will be overwritten only if no value is provided. This means a null value is treated as no change to the field.

The following budget fields cannot be updated:

- Budget Name
- Base Currency
- Budget Period Type
- Budget Region
- Budget Start Period
- Budget End Period
- Associated Entity

The following budget fields are required for updating:

- Budget Name
- Start Period Start Date
- End Period Start Date
- Period Type

- Budget Status
- Region Name
- Description

Limitations

The web service does not provide the user security check for updating budgets. This means that in cases where the log-on user does not have updating permissions for budgets in PPM Center, the user can update budget data by using this operation.

Related Information

The *create* operation creates new budgets.

Input

This operation requires the following inputs:

- UpdateMessage object that contains the header.
- Budget object to be updated.

Return

This operation returns the following information:

- UpdateResponse object that contains the header.
- Response message
- Budget ID

Java Interface

```
UpdateResponse update (UpdateMessage updateMessage)
```

Parameters	Description
UpdateMessage	Wrapper of RequestHeader and Budget.
UpdateResponse	Wrapper of the updated budget ID.

Java Examples

Example: update several simple fields of an existing request

```
private static void testUpdateBudget(RequestHeader header,
String budgetName)
        throws ServiceException, RemoteException,
RemoteException, ParseException {
        UpdateMessage updateMessage = new UpdateMessage();
        updateMessage.setHeader(header);

        Budget budget = new Budget();

        //budget.setBaseCurrency("United States Dollar");  //
NPE
        budget.setBudgetID(new Integer(30220));
        budget.setBudgetName("ws budget");

        Calendar startDate = new GregorianCalendar();
        startDate.setTime(date("February 1, 2008"));

        Calendar endDate = new GregorianCalendar();
        endDate.setTime(date("April 1, 2008"));

        budget.setStartPeriodStartDate(startDate);
        budget.setEndPeriodStartDate(endDate);
        budget.setPeriodType("Fiscal Month");
        budget.setBudgetStatus("Approved");
        budget.setRegionName("America");

        // now to change things that actually can be changed
        budget.setDescription("Updating again");
        budget.setCapexOpexEnabledFlag(Boolean.TRUE);

        updateMessage.setBudget(budget);
        UpdateResponse updateResponse =
financeServices.update(updateMessage);
        System.out.println("created budget id=
"+updateResponse.getBudgetID());
        System.out.println("create
returns:"+updateResponse.getResponseMessage().getMessage());
    }
```

Errors and Exceptions

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
BUDGET_ID_ MANDATORY	Unable to update budget. Budget ID not specified.	The budget ID is not specified.	Enter the budget ID.
BUDGET_NAME_ READ_ONLY	Unable to update budget name. It is a read-only property.	The user tries to update the budget name.	The budget name cannot be updated.
BUDGET_ CURRENCY_READ_ ONLY	Unable to update currency. It is a read-only property.	The user tries to update the budget currency.	The budget currency cannot be updated.
BUDGET_ PERIODTYPE_ READ_ONLY	Unable to update period type. It is a read-only property.	The user tries to update the budget period type.	The budget period type cannot be updated.
BUDGET_REGION_ READ_ONLY	Unable to update region. It is a read-only property.	The user tries to update the budget region.	The budget region cannot be updated.
BUDGET_ STARTPERIOD_ READ_ONLY	Unable to update start period. It is a read-only property.	The user tries to update the budget start period.	The budget start period cannot be updated.
BUDGET_ ENDPERIOD_ READ_ONLY	Unable to update end period. It is a read-only property.	The user tries to update the budget end period.	The budget end period cannot be updated.
BUDGET_ ASSOCIATEDTYPE_ READ_ONLY	Unable to update associated entity type. It is a read-only property.	The user tries to update the budget associated entity type.	The budget associated entity type cannot be updated.

Message Code	Message	Cause(s)	Possible Corrective Action
BUDGET_ ASSOCIATEDNAME _READ_ONLY	Unable to update associated entity name. It is a read-only property.	The user tries to update the budget associated entity name.	The budget associated entity name cannot be updated.
AUTO_ROLLUP_ NOT_POSSIBLE	This is a {0} budget. Actuals will not be automatically rolled up.	The budget type is one of the following: • Proposal • Org Unit • Asset actualRollUpCode is set to “automatically.”	When you create a budget of these types, do not set actualRollUpCode to “automatically.”
BUDGET_NOT_ ASSOCIATED	Unable to make the budget active. It must first be associated with an entity.	The budget is set to “active,” and no entity is associated with it.	Identify the entity for the budget before setting if “active.”
BUDGET_NOT_ ACTIVE	Unable to make the budget active. Budget has to be approved to be active.	The status of the budget is not “Approved.”	Set the status of the budget to “Approved.”

readExtended

Purpose

This operation is used to query PPM Center for a list of budgets with cost sum rows.

Function

This operation allows a remote system to call PPM Center and retrieve the budget data.

Limitations

Performance can be an issue if more than a few thousand budgets are read at once.

The web service does not provide the user security check for reading extended budgets. This means that in cases where the log-on user does not have read permissions for extended budgets in PPM Center, the user can read extended budget data by using this operation.

Related Information

The *read* operation reads budget lists.

Input

This operation requires the following inputs:

- UpdateMessage object that contains the header.
- Filter object to describe the filter conditions for the query.

Return

This operation returns the following information:

- UpdateResponse object that contains the header.
- Response message
- ExtendedBudget data

Java Interface

```
ReadResponse read (ReadMessage readMessage)
```

Parameters	Description
ReadMessage	Wrapper of RequestHeader and BudgetFilters. See the following example for retrieving the bean and its fields.
ReadResponse	Wrapper of the collection of budgets, or ExtendedBudget. See the following example for the construction.

Java Examples

Example: get data for a list of budget IDs

```
public static Integer testReadExtend(String url, String user,
String password, String budgetName) throws Exception {

    // get finance service
    FinanceServicesService locator = new
FinanceServicesServiceLocator();
    FinanceServices svc = locator.getFinance(new URL(url));

    // Construct request header
    RequestHeader header = new RequestHeader();
    header.setAuditNote("Test");
    header.setOrigin("PPM on " +
InetAddress.getLocalHost().getHostAddress());
    header.setCredentials(new LoginAccount(user, password));
    header.setTransactionName(new UID().toString());


    // Construct filter
    // if just pass the empty filter, all existing budget will
be returned.
    BudgetFilters filter = new BudgetFilters();
    filter.setBudgetName(budgetName);

    // example of filter by user data
    //Field[] ud = new Field[2]; // specify two user data field
    //ud[0] = new Field();
    //ud[0].setToken("BGT.APPROVER"); // field token
    //ud[0].setValue(new SingleValue("FOO")); // field value
    //ud[1] = new Field();
    //ud[1].setToken("BGT.APPROVE_DATE");
    //ud[1].setValue(new DateValue(new
java.util.GregorianCalendar()));
    //filter.setBudgetUserData(ud);

    // Construct read message
```

```
    ReadExtendedMessage readMsg = new
ReadExtendedMessage(header, filter);

    // invoke service: read
    ReadExtendedResponse readResp = svc.readExtended(readMsg);

    // print result
    Integer lastID = new Integer(0);
    ExtendedBudget[] budgets = readResp.getExtendedBudget();
    if (budgets != null && budgets.length > 0) {
    for(int i=0;i<budgets.length;i++) {
            //lastID = budgets[i].getBudgetID();
            ExtendedBudget budget = budgets[i];
    System.out.println("Get budget: " + budget.getBudgetName()
+ "(" + budget.getBudgetID() + ")");
    BudgetPeriodSumLine budgetPeriodSumLine =
budget.getBudgetPeriodSumLine();
    PeriodSumLineCellInfo[] cells =
budgetPeriodSumLine.getPeriodSumLineCells();
    System.out.println("Period Sum Info is ");
    for (int j=0;j<cells.length;j++){
    PeriodSumLineCellInfo cell = cells[j];
    System.out.println(cell.getPeriodName() + "-- TotalPlan:("
+ cell.getPlanTotal() + ") TotalActual:(" +
cell.getActualTotal() + ")");
    }
    }
    } else {
    System.out.println("No budget found");
    }
    // return the budget id of the last one (test purpose only)
    return lastID;
    }
```

Errors and Exceptions

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
INVALID_SORT_BY_OPTION	{0} is not a valid Sort By option.	The user uses an invalid Sort By option	The Sort By option should match the fields listed in BudgetFilter.

3 HP Financial Management (Cost Rules) Web Services

Overview

HP Financial Management Web services for cost rules provides operations to create, update, read, and delete cost rules. Operations to set and get cost factors in PPM Center are also supported.

Cost rules are used to assign costs to various units of work, such as the time spent working on a project task. The applicability of a cost rule to a given transaction is determined by its cost factors. A cost rule may have different rates for one or more non-overlapping periods of time. Exactly one rule in the system is the default rule. The default rule may not be deleted and its set of cost factor values may not be altered.

For more information about general HP Financial Management terms and concepts, see the *HP Financial Management User's Guide*.

References

Data types definition:

`webservice_toolkit\java\conf\wsdl60\`**`Finance.xsd`**

Operations definition:

`webservice_toolkit\java\conf\wsdl\`**`FinanceService.wsdl`**

Java sample code:

`webservice_toolkit\java\client\src\examples\fm`

.NET sample code:

`webservice_toolkit\MicrosoftDotNet\FinancialSummaryTest`

Operations History

HP Financial Management Web services for cost rules provides many operations starting with PPM Center version 7.1. *Table 3-1* on page 106 lists the HP Financial Management Web services for cost rules operations by version.

Table 3-1. HP Financial Management Web Service Cost Rules Operations

Web Service operation	7.1	7.5	8.00
createCostRules	Yes	Yes	Yes
searchCostRules	Yes	Yes	Yes
getCostRules	Yes	Yes	Yes
updateCostRules	Yes	Yes	Yes
deleteCostRules	Yes	Yes	Yes
getCostFactors	Yes	Yes	Yes
setCostFactors	Yes	Yes	Yes

Terms and Concepts

Cost Rules

Cost rules encapsulate the business logic that determines how the labor cost is calculated.

Cost Factor

Cost factors are the factors that determine cost rates. The same factors generally determine cost rates in any organization. These common cost factors are Time period, Region, Role, Skills, specific Resource, and so on.

Cost Factor Value

A cost factor value is the specific value for a given cost factor."

For example, for the **Department** cost factor, the cost factor value could be "Finance," whereas for the **Region** cost factor, the value could be "America."

Data Types

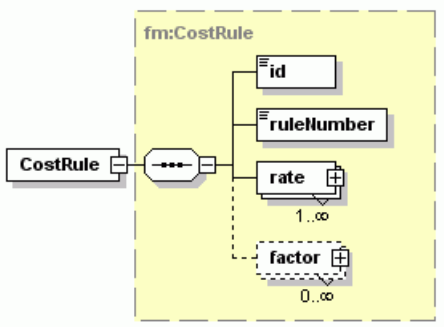
HP Financial Management Web services for cost rules includes the following data types:

WSCostRuleBean

WSCostRuleBean is a value object representing a cost rule. This bean is used to create cost rules in the system along with cost rate and cost factor values.

For more information about the WSCostRuleBean, see:

- WSCostRuleBean is used as the INPUT in the following operations:
  - Operation *createCostRules*
  - Operation *updateCostRules*
- WSCostRuleBean is used as the OUTPUT in the following operation:
  - Operation *createCostRules*

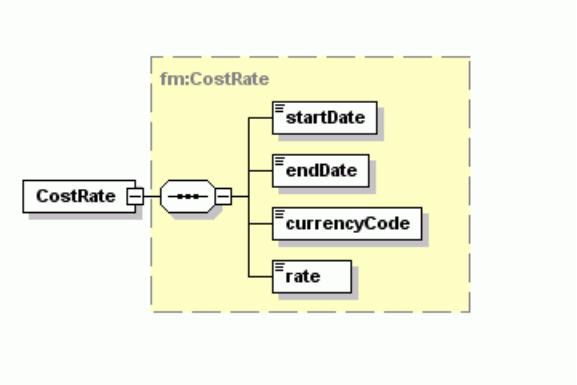


Property	Type	Description	Required	Default
Id	Long	Numeric value that is typically greater than 30,000.This value is automatically generated on the creation of a Cost Rule object in PPM Center.	No	N/A
ruleNumber	int	Numeric value used to identify the cost rule. If you do not provide a value, the value is automatically incremented.	No	N/A
costRates	List	List of *CostRateBean*s that contains the effective dates, rate, and currency code.	Yes	N/A
costFactorValues	List	list of *WSCostFactorValueBean*s that contains the cost factors values for the cost rule.	No	N/A

CostRateBean

CostRateBean is a value object representing the time-phased rate of a cost rule. A cost rule may have different rates for one or more non-overlapping periods of time.

.



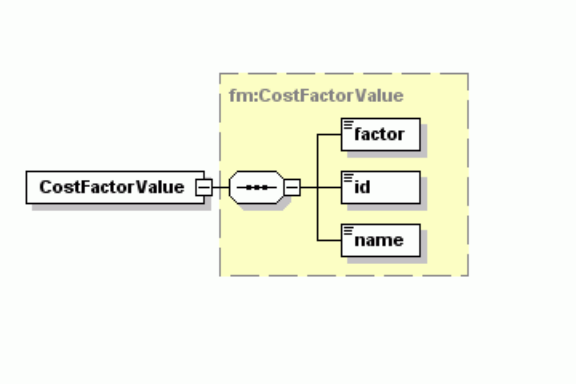
Property	Type	Description	Required	Default
effectiveStartDate	Date	Date value that specifies the start date for the cost rate. The value can be blank to indicate that the effective date is from way before. Note that the effective dates should not be overlapping with the effective dates of other cost rates that exist for this rule.	No	N/A
effectiveEndDate	Date	Date value that specifies the end date for the cost rate. The value can be blank to indicate that there is no end date. Note that the effective dates should not be overlapping with the effective dates of other cost rates that exist for this rule.	No	N/A
currencyCode	String	Currency code to be used with this cost rate. For example, the USD currency code.	Yes	N/A

Property	Type	Description	Required	Default
rate	BigDecimal	This property is the actual numeric value of the rate for this rule. This property is used in association with the currency code.	Yes	N/A

WSCostFactorValueBean

WSCostFactorValueBean is a value object that specifies which cost factor is used for the cost rule. This value object also specifies the ID and the name associated with the cost factor.

For example, if the cost factor used for the cost rule is Resource, you should specify the resource ID or resource name on which the cost rule should be applied.



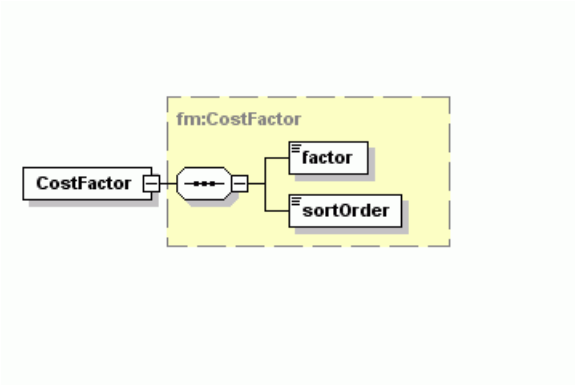
Property	Type	Description	Required	Default
costFactorName	String	Cost factor that is already set for the cost rules in the system. Both the id property and the name property are associated with this factor.	Yes	N/A

Property	Type	Description	Required	Default
id	String	Numeric value which is normally greater than 30,000. The value is associated with the selected cost factor.	No	N/A
name	String	Cost factor value that is associated with the selected cost factor.	No	N/A

WSCostFactorBean

WSCostFactorBean is used to select the cost factors and to specify the order by using the already seeded cost factors in PPM Center.

.

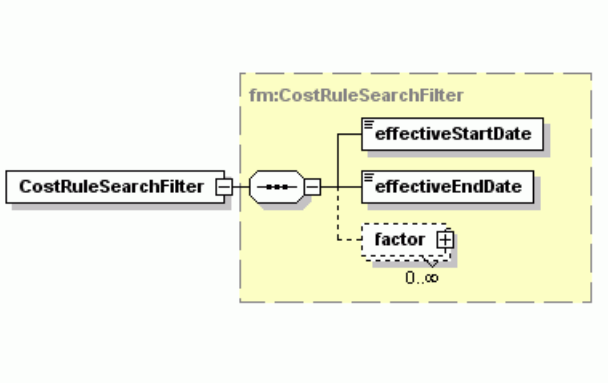


Property	Type	Description	Required	Default
costFactorName	String	One of the names of the cost factors that are already seeded in PPM Center. All the cost factors that you set through this method are applied for all the cost rules that are created in the system.	Yes	N/A
sortOrder	Integer	A numeric value used to identify the sort order. Lower numbers indicate more significance in the cost rule.	Yes	N/A

CostRuleSearchFilter

CostRuleSearchFilter bean is used to search for cost rules in the system.

.



Property	Type	Description	Required	Default
effectiveStartDate	Date	Specifies the effective start date for the cost rules that you search.	No	N/A
effectiveEndDate	Date	Specifies the effective end date for the cost rules that you search	No	N/A
costFactorValues	List	List of WSCostFactorValueBeans. Specifies the cost factor values that are contained in the cost rule that you search.	No	N/A

Operations

Supported operations include reading, writing, creating, and deleting cost rules, as well as managing the global set of cost factors.

The system may perform many validations before you proceed with any operations or changes. For example, a rule cannot be created if it has the same cost factor values as another rule that already exists. However, the cost factor values themselves are not validated as this is the responsibility of the client.

The following operations are included in HP Financial Management Web services for cost rules:

createCostRules

Purpose

Creates a collection of new cost rules in the system.

Do not use the createCostRules operation to migrate a large volume of cost rule data from a legacy system into PPM Center as it will take too much time.

Creating a cost rule or any other operation is expected to take just slightly less time than the same operation run from within the PPM application.

Function

Creates a new set of cost rules in the system. The cost rule update service picks up these new cost rules and modifies the costs on any affected entities depending on the corresponding effective dates.

Limitations

The creation of cost rules using this operation occurs in a single transaction, where if one cost rule fails, the entire operation is rolled back.

Related Information

updateCostRules updates a collection of cost rules.

Input

An array of WSCostRuleBeans.

For date values entered, the format should be a parsable string by Java DateFormat class. Otherwise a ParseException will be thrown by the DateFormat class, which will occur on the client side and not on the server side.

Return

An array of WSCostRuleBeans.

Java Interface

```
CreateCostRulesResponseDocument createCostRules
(CreateCostRulesDocument in)
```

Parameters	Description
CreateCostRulesDocument	Wrapper for the WSCostRuleBean array. See the following example for the construction. The bean includes the following fields: • Long id • int ruleNumber • List costRates// List<*CostRateBean*> • List costFactorValues// List<*WSCostFactorValueBean*>
CreateCostRulesResponseDocument	Wrapper for the array of created cost rule ids. See the following example for retrieving the bean and its fields.

Java Examples

Example: create a new set of cost rules.

```
public void testCreateCostRule() throws Exception {
```

```
    enableAllCostFactors();
    CostRule costRule = CostRule.Factory.newInstance();
    costRule.setFactorArray(new CostFactorValue[] {
    costFactorValue(DEPARTMENT, "Finance"),
    costFactorValue(REGION, "America"),
    costFactorValue(MISC_WORK_ITEMS, "Vacation"),
    costFactorValue(REQUEST_TYPE, "Bug")
    }
    );
    costRule.setRateArray(new CostRate[] {
    costRate(null, "Oct 13, 1994", 1.0f),
    costRate("Oct 14, 1994", "Jan 1, 2000", 2.0f),
    costRate("Jan 1, 2001", "Jan 1, 2001", 3.0f),
    costRate("Jan 2, 2001", "June 28, 2006", 3.7f)
    }
    );

    createCostRules(new CostRule[] { costRule });
    }

    CostRule[] createCostRules(CostRule[] costRules) throws
Exception {
    FinanceServiceStub FM = new FinanceServiceStub(ctx, WSURL);
    CreateCostRulesDocument createCostRulesDoc =
CreateCostRulesDocument.Factory.newInstance();

createCostRulesDoc.addNewCreateCostRules().setCostRuleArray(cos
tRules);
    CreateCostRulesResponseDocument response =
FM.createCostRules(createCostRulesDoc);
    return
response.getCreateCostRulesResponse().getCostRuleArray();
    }
```

Errors and Exceptions

When an error occurs with this operation, you will see a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault:
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Root Cause Descriptions

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
exception.authorization	You do not have the privilege to take this action. Please consult your PPM Administrator.	User not authorized to read the cost rule.	Check the user licenses and privileges (access grants).
error.ws.rateOverlap	One of the cost rules being created has two rates with overlapping effective ranges. Please check the rule and fix the rate overlap.	One of the rules has two or more rates with overlapping effective ranges.	Check the effective date ranges specified for the cost rate in the cost rule and make sure the dates are not overlapping.
error.conflictingCostFactors	You are adding a rule that has an illegal cost factor combination. No cost rule may have values for both the Resource and Role cost factor.	One of the rules has a combination of cost factors that is not allowed.	Avoid using the combination of cost factors mentioned in the message.
error.conflictingRules	You are adding a rule that has a conflict with an existing rule (Rule#30061). Please make sure the new rule has at least one cost factor with a different value.	One of the rules has the same set of cost factors as another rule in the system.	Change the cost factor values or use different cost factors.

Message Code	Message	Cause(s)	Possible Corrective Action
validationError.nameNotFound	Could not find a Department whose name is MANUFACTUR. Please verify that a Department with this name exists in the system. Alternatively, you may specify the Department by ID instead of name.	One of the rules has a cost factor that is not in the system wide selected cost factor set.	Verify the cost factor values given for a cost rule. These values should be already seeded in the system.
ex.InvalidCurrency	Could not find a Currency with id USWD. Please verify that a Currency with this id exists in the system.	Specified currency does not exist in the PPM system.	Verify that a currency with this ID exists in the system. Alternatively, you may specify the Currency by name instead of ID.
ex.InvalidDateRange	The effective end date {0} is before the start date {1}.	Specified start date is after the end date.	Specify the start date before the end date.
error.ws.notSelectedCostFactor	The cost factor {Project} is not selected for use. The selected cost factors are {Resource, Role, Region.}	Specified cost factor is not selected for use.	Use selected cost factors.

Message Code	Message	Cause(s)	Possible Corrective Action
validationError.nameNotFound	Could not find a {0} whose name is {1} or the {0} with name {1} is disabled. Please verify that a {0} with this name exists in the system or enable it. Alternatively, you may specify the {0} by id instead of name.	Specified cost factor value you selected does not exist or disable.	Verify that a cost factor value with this name exists in the system.
validationError.nameAndIdNull	When specifying a {0} you must provide either its name or id. If you do not want to specify a {0} please omit it entirely.	The user selects a cost factor, but does not input the factor id or name.	Specify a value either for the name of the ID.

updateCostRules

Purpose

Updates the cost rates of the collection of existing cost rules in the system.

Function

Updates a collection of cost rules. Each rule to update is identified by its primary key and must exist in the database. The cost factor values of a rule cannot be changed after the rule is created. Only the collection of cost rates may be altered. You must record the updated rates as CostRuleUpdates so that the updates are visible to the Cost Rate Rule Update service, which triggers the service to update the costs of any affected entities.

Limitations

Updates can be made only to the cost rate for an existing cost rule. The cost factors cannot be updated. This is a functional limitation.

Related Information

createCostRules creates a collection of cost rules.

Input

An object of an array of WSCostRuleBeans, which holds the cost rule IDs and the rates to update.

Return

None

Java Interface

```
UpdateCostRulesResponseDocument updateCostRules
(UpdateCostRulesDocument in)
```

Parameters	Description
UpdateCostRulesDocument	Wrapper for WSCostRuleBean array. See the following example for the construction. The following fields in the bean are required: • Long id • List costRates// List<*CostRateBean*>

Java Examples

Example: update a given set of cost rules.

```
public void testUpdateCostRule() throws Exception {

    CostRule costRule = CostRule.Factory.newInstance();
    costRule.setId (30010); //Set the cost rule id.
    costRule.setRateArray(new CostRate[] {
    costRate(null, "Oct 13, 1994", 1.0f),
    costRate("Oct 14, 1994", "Jan 1, 2000", 2.0f),
    costRate("Jan 1, 2001", "Jan 1, 2001", 3.0f),
    costRate("Jan 2, 2001", "June 28, 2006", 3.7f)
    }
    );

    updateCostRules(new CostRule[] { costRule });
    }
```

```
    void updateCostRules(CostRule[] costRules) throws Exception
{
    FinanceServiceStub FM = new FinanceServiceStub(ctx, WSURL);
    UpdateCostRulesDocument updateCostRulesDoc =
UpdateCostRulesDocument.Factory.newInstance();

updateCostRulesDoc.addNewUpdateCostRules().setCostRuleArray(cos
tRules);
    FM.updateCostRules(updateCostRulesDoc);
    }
```

Errors and Exceptions

When an error occurs on this operation, you will see a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault:
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Root Cause Descriptions

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
exception.authorization	You do not have the privilege to take this action. Consult your PPM Administrator.	User is not authorized to read the cost rule.	Check the user licenses and privileges (access grants).
error.ws.rateOverlap	One of the cost rules being created has two rates with overlapping effective ranges. Check the rule and fix the rate overlap.	One of the rules has two or more rates with overlapping effective ranges.	Check the effective date ranges specified for the cost rate in the cost rule and make sure the dates are not overlapping.

Message Code	Message	Cause(s)	Possible Corrective Action
error.ws.update.idRequired	A cost rule ID is required when updating cost rules. Please fill in the ID for each of the rules you wish to modify.	The rule ID is not specified for the updated cost rule.	Specify the rule ID.
ex.ObjectNotFound	Cannot load the {0} with the specified ID: {1}. {0} may be deleted by another user.	Cost rule is not found for any of the IDs in the costRuleBeans.	Check to read the cost rule with the same ID to see if the cost rule exists.

getCostRules

Purpose

Gets the details of the cost rules for the given set of IDs.

Function

For a given set of IDs, this operation reads the details of the cost rules that are persisted in the system.

Limitations

Performance might be an issue if more than a few thousand cost rules read at one time.

Input

An object with an array of existing cost rule IDs.

Return

An object with an array of WSCostRuleBean corresponding to the given set of IDs.

Java Interface

```
GetCostRulesResponseDocument getCostRules (GetCostRulesDocument
in)
```

Parameters	Description
GetCostRulesDocument	Wrapper for cost rule ids array. See the following example for the construction. The following field is required: • Long[] ids
GetCostRulesResponseDocument	Wrapper for the array of cost rule beans (WSCostRuleBean) obtained performing this operation.

Java Examples

Example: Read cost rules for a given set of IDs.

```
CostRule[] getCostRules(long[] ids) throws Exception {
    FinanceServiceStub FM = new FinanceServiceStub(ctx, WSURL);
    GetCostRulesDocument getCostRulesDoc =
GetCostRulesDocument.Factory.newInstance();

getCostRulesDoc.addNewGetCostRules().setCostRuleIdArray(ids);
    GetCostRulesResponseDocument response =
FM.getCostRules(getCostRulesDoc);
    return
response.getGetCostRulesResponse().getCostRuleArray();
    }
```

Errors and Exceptions

When an error occurs on this operation, you will see a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault:
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>

</exception:exceptionDetails>
```

Root Cause Descriptions

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
exception.authorization	You do not have the privilege to take this action. Consult your PPM Administrator.	User is not authorized to read the cost rule.	Check the user licenses and privileges (access grants).
ex.ObjectNotFound	Cannot load the {0} with the specified ID: {1}. {0} may be deleted by another user.	Cost rule is not found for any of the IDs in the array.	This means the cost rule with this ID does not exist in the PPM system. Try to read with another existing ID.

deleteCostRules

Purpose

Deletes the cost rules for a given set of cost rule IDs.

Function

Deletes the given cost rules. If a rule is not found for a given ID it is silently ignored. The default rule cannot be deleted, and attempts to delete it are ignored.

After the cost rule is deleted, the costs related with this cost rule will be recalculated for all entities in the system. This operation is not performed in the “frozen” state. This is done asynchronously by the Cost Rate Rule Update service.

Related Information

getCostRules reads a collection of cost rules.

Input

An object with an array of cost rule IDs.

Return

None

Java Interface

```
deleteCostRules (DeleteCostRulesDocument in)
```

Parameters	Description
DeleteCostRulesDocument	Wrapper for cost rule ids array. See the following example for the construction. The following field is required: • Long[] ids

Java Examples

Example: delete a set of cost rules.

```
void deleteCostRules(long[] ids) throws Exception {
    FinanceServiceStub FM = new FinanceServiceStub(ctx, WSURL);
    DeleteCostRulesDocument deleteCostRulesDoc =
DeleteCostRulesDocument.Factory.newInstance();

deleteCostRulesDoc.addNewDeleteCostRules().setCostRuleIdArray(i
ds);
    FM.deleteCostRules(deleteCostRulesDoc);
    }
```

Errors and Exceptions

When an error occurs on this operation, you will see a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault:
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
```

```
</exception:exceptionDetails>
```

Root Cause Descriptions

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
exception.authorization	You do not have the privilege to take this action. Consult your PPM Administrator.	User is not authorized to delete the cost rule.	Check the user licenses and privileges (access grants).

If the ID you provide to delete a cost rule is not found, the operation is silently ignored, and no exception is thrown.

searchCostRules

Purpose

Searches all the existing cost rules in the system that satisfy the search criteria provided.

Function

For a given search criteria, this operation gets all the cost rules satisfying the criteria. The criteria include the effective dates and the cost rate for the cost rules.

Limitations

The maximum number of results cannot exceed 1,000

Related Information

getCostRules gets all the cost rules for a given set of IDs.

Input

CostRuleSearchFilter

Return

An object with an array of WSCostRuleBeans corresponding to the given search filter.

Java Interface

```
SearchCostRulesResponseDocument searchCostRules
(SearchCostRulesDocument in)
```

Parameters	Description
SearchCostRulesDocument	Wrapper for cost rule search filter (CostRuleSearchFilter).
SearchCostRulesResponseDocument	Wrapper for the array of cost rule beans (WSCostRuleBean) that the caller obtains by performing the search operation.

Java Examples

Example: Search cost rules for a given set of IDs.

```
CostRule[] searchAllCostRules() throws Exception {
    FinanceServiceStub FM = new FinanceServiceStub(ctx, WSURL);
    SearchCostRulesDocument searchCostRulesDoc =
SearchCostRulesDocument.Factory.newInstance();
   searchCostRulesDoc.addNewSearchCostRules().addNewFilter();
// empty filter finds ALL rules
    SearchCostRulesResponse searchResponse =
FM.searchCostRules(searchCostRulesDoc).getSearchCostRulesRespon
se();
    return searchResponse.getCostRuleArray();
    }
```

Errors and Exceptions

When an error occurs on this operation, you will see a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault:
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>

</exception:exceptionDetails>
```

Root Cause Descriptions

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
exception.authorization	You do not have the privilege to take this action. Consult your PPM Administrator.	User is not authorized to search cost rules.	Check the user licenses and privileges (access grants).

getCostFactors

Purpose

Gets the cost factors that are in PPM Center.

Function

Gets the cost factors defined in the system. Cost factors are returned in a list with the most significant factor first and the least significant factor last. For more information about the significance of cost factors, refer to the *HP Financial Management User's Guide.*

Limitations

None

Related Information

setCostFactors sets the cost factors in PPM Center.

Input

None

Return

An object with an array of WSCostFactorBeans.

Java Interface

```
GetCostRulesResponseDocument getCostRules (GetCostRulesDocument
in)
```

Parameters	Description
GetCostFactorsResponseDocument	Wrapper for the array of cost factor beans (WSCostFactorBean) that the caller obtains by performing this operation.

Java Examples

Example: Read the cost factors that are in PPM Center.

```
CostFactor[] getCostFactors() throws Exception {
    FinanceServiceStub FM = new FinanceServiceStub(ctx, WSURL);
    GetCostFactorsDocument getCostFactorsDoc =
GetCostFactorsDocument.Factory.newInstance();
                getCostFactorsDoc.addNewGetCostFactors();
    GetCostFactorsResponseDocument response =
FM.getCostFactors(getCostFactorsDoc);
    return
response.getGetCostFactorsResponse().getCostFactorArray();
    }
```

Errors and Exceptions

When an error occurs on this operation, you will see a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault:
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
```

```
</exception:exceptionDetails>
```

Root Cause Descriptions

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
exception.authorization	You do not have the privilege to take this action. Consult your PPM Administrator.	User is not authorized to read the cost factors.	Check the user licenses and privileges (access grants).

setCostFactors

Purpose

Sets the cost factors in PPM Center to be used by cost rules.

Function

Sets the cost factors for the system. The precedence of cost factors is determined by their ordering in the list, the most significant cost factor is the first element in the list and the least significant factor is the last.

Setting of cost factors does not mean adding cost factors to an already existing list. This operation removes the existing list of cost factors and creates an entirely new list.

Changing the order of cost factors necessitates that costs be recalculated for all entities in the system that are not in a “frozen” state. This is done asynchronously by the Cost Rate Rule Update service. Costs do not need to be recalculated when cost factors are added or deleted. Adding new cost factors does not affect costing because no rule in the system has a value for the new factor. If a rule is created with a non-null value for the new factor, a CostRuleUpdate is created for it by PPM Center automatically. Deleting cost factors also does not require cost recalculation because only unused cost factors may be removed.

Related Information

getCostFactors gets the cost factors in PPM Center.

Input

An object with an array of WSCostFactorBeans.

Return

None

Java Interface

```
SetCostFactorsResponseDocument
setCostFactors(SetCostFactorsDocument in)
```

Parameters	Description
SetCostFactorsDocument	Wrapper for the array of cost factor beans (WSCostFactorBean) that the caller obtains by performing this operation.

Java Examples

Example: Set the cost factors in PPM Center.

```
CostFactor[] setCostFactors() throws Exception {
    SetCostFactorsDocument setCostFactorsDoc =
SetCostFactorsDocument.Factory.newInstance();
    SetCostFactors setCostFactors =
setCostFactorsDoc.addNewSetCostFactors();
    List newFactors = shuffleCostFactors();
    populate(setCostFactors, newFactors);
    if (VERBOSE) print(setCostFactors);
    FM.setCostFactors(setCostFactorsDoc);
    }

    List shuffleCostFactors() {
    List factors = new ArrayList();
    Random r = new Random();
    for (int i = 0; i < COST_FACTORS.length; i++) {
    if (r.nextFloat() > .2) {  // more likeley to keep it then
not
    factors.add(COST_FACTORS[i]);
    }
```

```
    }

    Collections.shuffle(factors);
    return factors;
    }

    void populate(SetCostFactors setCostFactors, List factors)
{
    for (int i = 0; i < factors.size(); i++ ) {
    CostFactor f = setCostFactors.addNewCostFactor();
    f.setFactor((String) factors.get(i));
    f.setSortOrder(BigInteger.valueOf(i));
    }
    }
```

Errors and Exceptions

When an error occurs on this operation, you will see a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault:
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>

</exception:exceptionDetails>
```

Root Cause Descriptions

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
ex.Authorization	You do not have the privilege to take this action. Consult your PPM Administrator.	User is not authorized to read the cost factors.	Check the user licenses and privileges (access grants).

Message Code	Message	Cause(s)	Possible Corrective Action
ex.CostFactorInUse	You are removing a cost factor - Request Type - that is still in use by cost rules in the system. Please remove these rules before proceeding.	An attempt is made to deactivate a cost factor that is being used by one or more cost rules. In other words, a CostFactorInUseException is thrown if one or more rules exist in the system with a non-null value for this factor.	Do NOT deactivate or remove the cost factors that are being used by cost rules. Identify the cost rule associated with this cost factor. Then, determine if the cost rule should be removed. If the cost rule can be removed, do so, and then remove the cost factor. Note that when you are trying to set the cost factors, it is as though you are deleting the existing list and setting a whole new set of factors.

4 HP Financial Management (Finance Data) Web Services

Overview

HP Financial Management Web services for finance data provides interfaces for accessing and updating finance data in PPM Center. These include operations for modifying cost rules, cost factors, financial summaries, and financial data.

For more information about general HP Financial Management terms and concepts, see the *HP Financial Management User's Guide.*

References

Data types definition:

`webservice_toolkit\java\conf\xsd\`**`Finance.xsd`**

Operations definition:

`webservice_toolkit\java\conf\wsdl\`**`FinanceService.wsdl`**

Java sample code:

- `webservice_toolkit\java\client\src\examples\fm\`**`FinanceServiceClient.java`**
- `webservice_toolkit\java\client\src\examples\fm\`**`FinancialSummaryClient.java`**
- `webservice_toolkit\java\client\src\examples\fm\`**`FinancialDataClient.java`**

.NET sample code:

- `webservice_toolkit\MicrosoftDotNet\FinancialDataTest`
- `webservice_toolkit\MicrosoftDotNet\FinancialSummaryTest`

Operations History

HP Financial Management Web services for finance data provides many operations starting with PPM Center version 7.1. *Table 4-1* on page 134 lists the HP Financial Management Web services for finance data operations by version.

Table 4-1. HP Financial Management Web Service Finance Data Operations

Web Service operation	7.1	7.5	8.00
createCostRules	Yes	Yes	Yes
searchCostRules	Yes	Yes	Yes
getCostRules	Yes	Yes	Yes
deleteCostRules	Yes	Yes	Yes
getCostFactors	Yes	Yes	Yes
setCostFactors	Yes	Yes	Yes
readFinancialSummary	No	No	Yes
readFinancialSummarySnapshot	No	No	Yes
updateFinancialSummary	No	No	Yes
createFinancialSummarySnapshot	No	No	Yes
readFinancialSummaryACL	No	No	Yes
updateFinancialSummaryACL	No	No	Yes
readFinancialData	No	No	Yes
updateFinancialData	No	No	Yes

Table 4-1. HP Financial Management Web Service Finance Data Operations

Web Service operation	7.1	7.5	8.00
createFinancialData	No	No	Yes
readFinancialDataACL	No	No	Yes
updateFinancialDataACL	No	No	Yes

Data Types

HP Financial Management Web services for finance includes the following data types:

- *AccessListInfo* on page 151

FinancialSummaryInfo

This type defines the financial summary.

.

Property	Type	Description	Required	Default
id	Integer	Financial summary ID.	No	N/A
name	String	Financial summary name.	No	N/A
description	String	Description of the financial summary.	No	N/A
parent	parentInfo	The parent entity associated with this financial summary. A parent entity can be one of the following: • Project • Proposal • Asset • Program • Org Unit	No	N/A
localCurrencyCode	String	Local Currency Code.	No	N/A
baseCurrencyCode	String	Base Currency Code.	No	N/A
approvedBudgets	approvedBudgetInfo[]	A list of approved budget entries.	No	N/A
forecastActual	forecastActualInfo	Planned and actual costs.	No	N/A
benefit	benefitInfo	Benefits.	No	N/A
snapshots	snapshotInfo[]	A list of snapshot headers.	No	N/A

FinancialDataInfo

This type defines the financial data.

Property	Type	Description	Required	Default
id	Integer	Financial data ID.	No	N/A
name	String	Financial data name.	No	N/A
description	String	Description of the financial data.	No	N/A
parent	financialDataParentInfo	The parent entity associated with this financial data. A parent entity can be one of the following: • Project • Proposal • Asset • Program • Org Unit	No	N/A
localCurrencyCode	String	Currency Code.	No	N/A
baseCurrencyCode	String	Currency Code.	No	N/A
forecastActual	forecastActualInfo	Planned and actual costs.	No	N/A
benefit	benefitInfo	Benefits.	No	N/A

FinancialSummarySnapshotInfo

This type defines the financial summary snapshot.

Property	Type	Description	Required	Default
id	Integer	Financial summary snapshot ID.	No	N/A
name	String	Financial summary snapshot name.	No	N/A

Property	Type	Description	Required	Default
description	String	Description of the financial summary snapshot.	No	N/A
parent	parentInfo	The parent entity associated with this financial summary. A parent entity can be one of the following: • Project • Proposal • Asset • Program • Org Unit	No	N/A
localCurrencyCode	String	Currency Code.	No	N/A
baseCurrencyCode	String	Currency Code.	No	N/A
activeFSId	Integer	Active Financial Summary ID.	No	N/A
isPlanOfRecord	Boolean	Indicates if this snapshot is the plan of record	No	N/A
creationDate	Date	Date when the snapshot was taken.	No	N/A
approvedBudgets	approvedBudgetInfo[]	List of approved budget entries.	No	N/A
forecastActual	forecastActualInfo	Planned and actual costs.	No	N/A
benefit	benefitInfo	Benefits.	No	N/A

SnapshotInfo

This type defines the financial summary snapshot.

Property	Type	Description	Required	Default
id	Integer	Financial summary snapshot ID.	No	N/A
name	String	Financial summary snapshot name	No	N/A
description	String	Description of the financial summary snapshot.	No	N/A
isPlanOfRecord	Boolean	Indicates if this snapshot is the plan of record.	No	N/A
creationDate	Date	Date when the snapshot was taken.	No	N/A

UserDataInfo

This type defines the user data type.

Property	Type	Description	Required	Default
userData1	String	User data string values.	No	N/A
userData2	String	User data string values.	No	N/A
userData3	String	User data string values.	No	N/A
userData4	String	User data string values.	No	N/A
userData5	String	User data string values.	No	N/A
userData6	String	User data string values.	No	N/A
userData7	String	User data string values.	No	N/A
userData8	String	User data string values.	No	N/A
userData9	String	User data string values.	No	N/A
userData10	String	User data string values.	No	N/A
userData11	String	User data string values.	No	N/A
userData12	String	User data string values.	No	N/A

Property	Type	Description	Required	Default
userData13	String	User data string values.	No	N/A
userData14	String	User data string values.	No	N/A
userData15	String	User data string values.	No	N/A
userData16	String	User data string values.	No	N/A
userData17	String	User data string values.	No	N/A
userData18	String	User data string values.	No	N/A
userData19	String	User data string values.	No	N/A
userData20	String	User data string values.	No	N/A
visUserData1	String	Visible user data string values.	No	N/A
visUserData2	String	Visible user data string values.	No	N/A
visUserData3	String	Visible user data string values.	No	N/A
visUserData4	String	Visible user data string values.	No	N/A
visUserData5	String	Visible user data string values.	No	N/A
visUserData6	String	Visible user data string values.	No	N/A
visUserData7	String	Visible user data string values.	No	N/A
visUserData8	String	Visible user data string values.	No	N/A
visUserData9	String	Visible user data string values.	No	N/A
visUserData10	String	Visible user data string values.	No	N/A
visUserData11	String	Visible user data string values.	No	N/A

Property	Type	Description	Required	Default
visUserData12	String	Visible user data string values.	No	N/A
visUserData13	String	Visible user data string values.	No	N/A
visUserData14	String	Visible user data string values.	No	N/A
visUserData15	String	Visible user data string values.	No	N/A
visUserData16	String	Visible user data string values.	No	N/A
visUserData17	String	Visible user data string values.	No	N/A
visUserData18	String	Visible user data string values.	No	N/A
visUserData19	String	Visible user data string values.	No	N/A
visUserData20	String	Visible user data string values.	No	N/A

ApprovedBudgetInfo

This type defines the approved budget:

Property	Type	Description	Required	Default
id	Integer	Budget ID.	No	N/A
name	String	Budget name.	No	N/A
description	String	Budget description.	No	N/A
period	periodInfo	Budget date period.	No	N/A
amountLCL	Double	Budget amount in local currency.	No	N/A

Property	Type	Description	Required	Default
amountBSE	Double	Budget amount in base currency.	No	N/A
approvalDate	Date	Approval date of the budget.	No	N/A
approvedBy	String	The user name of the user who approves the budget.	No	N/A

ForecastActualInfo

Thistype defines the forecast actual.

Property	Type	Description	Required	Default
id	Integer	Forecast actual ID.	No	N/A
actualRollupCode	String	• MANUAL • AUTO • PARTIAL	No	N/A
isCapexOpexEnabled	Boolean	Indicates if planned and actual costs are capitalized.	No	N/A

Property	Type	Description	Required	Default

lines	line[]	Actual lines. A line contains the following:
		• laborType: String
		• category: String
		• expenseType: String
		• syncSourceCode:
		  • NO_SYNC (Default)
		  • STAFFING_PROFILE
		  • PROJECT
		  • ASSET
		  • PROPOSAL
		  • cells: financialLineCellInfo[]
		• userData: userDataInfo
		See Finance.xsd for more information.
			No	N/A

periodSum	costPeriodSumCellInfo[]	Sum of planned and actual costs by type (Labor, Non-Labor) and expenseType (Capitalized, Operating) over a given period	No	N/A
userData	userDataInfo	User data information.	No	N/A

BenefitInfo

This type defines the financial benefit.

Property	Type	Description	Required	Default
id	Integer	Financial benefit ID.	No	N/A
lines	line[]	Actual lines. A line contains: • ID: Integer • benefitType: String • category: String • cells: financialLineCellInfo[] • userData: userDataInfo See Finance.xsd for more info	No	N/A
periodSum	costPeriodSumCellInfo[]	Sum of benefit planned and actual values by period.	No	N/A
userData	userDataInfo	User data info.	No	N/A

FinancialLineCellInfo

This type defines financial benefit or actual cells:

Property	Type	Description	Required	Default
period	periodInfo	Date period.	No	N/A
actualValueBSE	Double	Actual cost or benefit value in the base currency.	No	N/A
actualValueLCL	Double	Actual cost or benefit value in a local currency.	No	N/A
planValueBSE	Double	Planned cost or benefit value in the base currency.	No	N/A
planValueLCL	Double	Planned cost or benefit value in a local currency.	No	N/A

CostPeriodSumCellInfo

This type defines total planned and actual costs for a given period summed by type (labor, non-Labor) and expenseType (capitalized, operating) in the base currency and the local currency.

Property	Type	Description	Required	Default
period	periodInfo	Date period.	No	N/A
projectedTotal	Double	Total cost.	No	N/A
capexProjectedTotal	Double	CAPEX (Capitalized) total cost.	No	N/A
opexProjectedTotal	Double	OPEX (Operating) total cost.	No	N/A
planTotalLCL	Double	Planned total cost or benefit value in local currency.	No	N/A
planTotalBSE	Double	Planned cost or benefit value in base currency.	No	N/A
actualTotalLCL	Double	Actual total cost or benefit value in local currency.	No	N/A
actualTotalBSE	Double	Actual total cost or benefit value in local currency.	No	N/A
planCapexTotalLCL	Double	Planned total CAPEX cost or benefit value in local currency.	No	N/A
planCapexTotalBSE	Double	Planned total CAPEX cost or benefit value in base currency.	No	N/A
planOpexTotalLCL	Double	Planned total OPEX in local currency.	No	N/A
planOpexTotalBSE	Double	Planned total OPEX in base currency.	No	N/A

Property	Type	Description	Required	Default
planLaborTotalLCL	Double	Planned total labor cost value in local currency.	No	N/A
planLaborTotalBSE	Double	Planned total labor cost value in base currency.	No	N/A
planNonLaborTotalLCL	Double	Planned total non-labor cost value in local currency.	No	N/A
planNonLaborTotalBSE	Double	Planned total non-labor cost value in base currency.	No	N/A
actualCapexTotalLCL	Double	Actual CAPEX cost or benefit value in local currency.	No	N/A
actualCapexTotalBSE	Double	Actual CAPEX cost or benefit value in base currency.	No	N/A
actualOpexTotalLCL	Double	Actual OPEX cost or benefit value in local currency.	No	N/A
actualOpexTotalBSE	Double	Actual OPEX cost or benefit value in base currency.	No	N/A
actualLaborTotalLCL	Double	Actual total labor cost value in local currency.	No	N/A
actualLaborTotalBSE	Double	Actual total labor cost value in base currency.	No	N/A
actualNonLaborTotalLCL	Double	Actual total non-labor cost value in local currency.	No	N/A
actualNonLaborTotalBSE	Double	Actual total non-labor cost value in base currency.	No	N/A

Property	Type	Description	Required	Default
planCapexLaborLCL	Double	Planned CAPEX labor cost value in local currency.	No	N/A
planCapexLaborBSE	Double	Planned CAPEX labor cost value in base currency	No	N/A
planOpexLaborLCL	Double	Planned OPEX labor cost value in local currency.	No	N/A
planOpexLaborBSE	Double	Planned OPEX labor cost value in base currency	No	N/A
planCapexNonLaborLCL	Double	Planned CAPEX non-labor cost value in local currency.	No	N/A
planCapexNonLaborBSE	Double	Planned CAPEX non-labor cost value in base currency	No	N/A
planOpexNonLaborLCL	Double	Planned OPEX non-labor cost value in local currency.	No	N/A
planOpexNonLaborBSE	Double	Planned OPEX non-labor cost value in base currency	No	N/A
actualCapexLaborLCL	Double	Actual CAPEX non-labor cost value in local currency.	No	N/A
actualCapexLaborBSE	Double	Actual CAPEX non-labor cost value in base currency.	No	N/A
actualOpexLaborLCL	Double	Actual OPEX labor cost value in local currency.	No	N/A
actualOpexLaborBSE	Double	Actual OPEX labor cost value in base currency.	No	N/A

Property	Type	Description	Required	Default
actualCapexNonLaborLCL	Double	Actual CAPEX non-labor cost value in local currency.	No	N/A
actualCapexNonLaborBSE	Double	Actual CAPEX non-labor cost value in base currency.	No	N/A
actualOpexNonLaborLCL	Double	Actual OPEX non-labor cost value in local currency.	No	N/A
actualOpexNonLaborBSE	Double	Actual OPEX non-labor cost value in base currency.	No	N/A

BenefitPeriodSumCellInfo

This type defines the benefit period summary cell.

Property	Type	Description	Required	Default
period	periodInfo	Date period	No	N/A
actualValueBSE	Double	Actual cost or benefit value in the base currency	No	N/A
actualValueLCL	Double	Actual cost or benefit value in a local currency	No	N/A
planValueBSE	Double	Planned cost or benefit value in the base currency	No	N/A
planValueLCL	Double	Planned cost or benefit value in a local currency	No	N/A
projectedTotal	Double	Total benefit	No	N/A

ParentInfo

This type is used by the parent reference.

Property	Type	Description	Required	Default
parentType	String	Enum of: • PROPOSAL • PROJECT • ASSET • ORG_UNIT • PROGRAM	No	N/A
parentIdentifier	Long/String	ID or name that uniquely identifies the associated parent entity.	No	N/A

FinancialDataParentInfo

This type is used by the parent reference for financial data.

Property	Type	Description	Required	Default
parentType	String	Enum of: • REQUEST	No	N/A
parentId	Long	Parent ID.	No	N/A
token	String	The token that uniquely identifies the financial data field on a demand management request type.	No	N/A

PeriodInfo

This type defines the period.

Property	Type	Description	Required	Default
periodType	String	A string representing the fiscal period type. Currently, a value of “MONTH” is supported.	Yes	N/A
periodStartDate	Date	Period Start Date.	Yes	N/A

AccessListInfo

This type defines the access list.

Property	Type	Description	Required	Default
entry	(list)	A list of the following entities: • username: String or securityGroupName: String • viewCosts: Boolean • editCosts: Boolean • viewBenefits: Boolean • editBenefits: Boolean • viewApprovedBudget: Boolean • editApprovedBudget: Boolean • setPlanOfRecord: Boolean • editAccess: Boolean	No	N/A

Operations

The following operations are included in HP Financial Management Web services for finance data:

- *createFinancialData* on page 159
- *readFinancialDataACL* on page 160
- *updateFinancialDataACL* on page 161

readFinancialSummary

Purpose

This operation reads a financial summary.

Related Information

Data Type: *ParentInfo*, *FinancialSummaryInfo*

Input

This operation requires the following input:

- financialSummaryParent: ParentInfo, with the following required information:
  - parentType
  - parentIdentifier

Return

FinancialSummaryInfo

A financial summary will be returned if this operation is successful.

Java Examples

See webservice_toolkit:

`java\client\src\examples\fm\`**`FinancialSummaryClient.java`**

readFinancialSummary ()

```
ReadFinancialSummaryDocument requestDoc =
ReadFinancialSummaryDocument.Factory.newInstance();
```

```
ReadFinancialSummary fs =
requestDoc.addNewReadFinancialSummary();

// Set up parent information.
ParentInfo parent =
populateParentInfo(fs.addNewFinancialSummaryParent(),
parentType, parentId);

// Read the financial summary.
FinanceServiceStub stub = new FinanceServiceStub(context,
serviceURL);
ReadFinancialSummaryResponseDocument responseDoc =
stub.readFinancialSummary(requestDoc);
ReadFinancialSummaryResponse response =
responseDoc.getReadFinancialSummaryResponse();

return response.getFinancialSummary();



ParentInfo populateParentInfo (ParentInfo parentInfo, String
parentType, Long parentId)
{
    ParentInfo.ParentType.Enum parentTypeEnum =
ParentInfo.ParentType.Enum.forString(parentType);
    parentInfo.setParentType(parentTypeEnum);

    ParentIdentifier identifier =
ParentIdentifier.Factory.newInstance();
    identifier.setId(new BigInteger(parentId.toString()));
    parentInfo.setParentIdentifier(identifier);
    return parentInfo;
}
```

readFinancialSummarySnapshot

Purpose

This operation reads a financial summary snapshot.

Related Information

Data type: *ParentInfo*

Input

This operation requires the following inputs:

- financialSummaryParent: ParentInfo, with the following required information:
  - parentType
  - parentIdentifier
- planOfRecord: Boolean
- creationDate: Date

Return

FinancialSummarySnapshotInfo

Java Examples

See webservice_toolkit:

`java\client\src\examples\fm\`**`FinancialSummaryClient.java`**

readSnapshot ()

updateFinancialSummary

Purpose

This operation updates an existing financial summary.

Related Information

Operation: *readFinancialSummary*

Input

This operation requires the following input:

- financialSummary: FinancialSummaryInfo

Return

If the operation is successful, nothing will be returned to the client.

Java Examples

See webservice_toolkit:

`java\client\src\examples\fm\`**`FinancialSummaryClient.java`**

updateFinancialSummary ()

createFinancialSummarySnapshot

Purpose

This operation creates a financial summary snapshot.

Input

This operation requires the following inputs:

- financialSummaryParent: parentInfo (required)
- snapshot: snapshotInfo

Return

If the operation is successful, nothing will be returned to the client.

Java Examples

See webservice_toolkit:

`java\client\src\examples\fm\`**`FinancialSummaryClient.java`**

createSnapshot ()

readFinancialSummaryACL

Purpose

This operation reads a financial summary access control list (ACL).

Related Information

Operation: *readFinancialSummary*

Input

This operation requires the following input:

- financialSummaryParent: ParentInfo

Return

AccessListInfo

Java Examples

See webservice_toolkit: `java\client\src\examples\fm\`**`FinancialSummaryClient.java`**

readACL()

```
ReadFinancialSummaryACLDocument requestDoc =
ReadFinancialSummaryACLDocument.Factory.newInstance();
ReadFinancialSummaryACL root =
requestDoc.addNewReadFinancialSummaryACL();

ParentInfo parent =
populateParentInfo(root.addNewFinancialSummaryParent(),
parentType, parentId);

FinanceServiceStub stub = new FinanceServiceStub(context,
serviceURL);
ReadFinancialSummaryACLResponseDocument responseDoc =
stub.readFinancialSummaryACL(requestDoc);
ReadFinancialSummaryACLResponse response =
responseDoc.getReadFinancialSummaryACLResponse();

return response.getAccessList();
```

updateFinancialSummaryACL

Purpose

This operation updates a financial summary ACL.

Input

This operation requires the following input:

- financialSummaryParent: ParentInfo
- accessList: AccessListInfo

Return

If the operation is successful, nothing will be returned to the client.

Java Examples

See webservice_toolkit:

`java\client\src\examples\fm\`**`FinancialSummaryClient.java`**

updateACL()

readFinancialData

Purpose

This operation reads financial data.

Input

This operation requires the following input:

- financialDataParent: financialDataParentInfo

Return

Data as defined in *FinancialDataInfo*.

Java Examples

See webservice_toolkit:

`java\client\src\examples\fm\`**`FinancialDataClient.java`**

readFinancialData ()

updateFinancialData

Purpose

This operation updates financial data.

Related Information

Operation: *readFinancialData*

Input

This operation requires the following input:

- financialData: FinancialDataInfo

Return

If the operation is successful, nothing will be returned to the client.

Java Examples

See webservice_toolkit:

`java\client\src\examples\fm\`**`FinancialDataClient.java`**

updateFinancialData ()

```
// Modify the financial summary name and description.
fd.setName("Toolkit Financial Data");
```

```
fd.setDescription(makeUnique("Last Modified: "));

// Set the local currency.
fd.setLocalCurrencyCode("USD");

// Create or update benefits.
BenefitInfo benefit = null;
BenefitInfo.Lines benefitLines = fd.getBenefit().getLines();
if (isEmpty(benefitLines)) {
    benefit = createBenefit(fd.getBenefit());
}
else {
    benefit = updateBenefit(fd.getBenefit());
}
fd.setBenefit(benefit);

UpdateFinancialDataDocument requestDoc =
UpdateFinancialDataDocument.Factory.newInstance();
requestDoc.addNewUpdateFinancialData().setFinancialData(fd);

FinanceServiceStub stub = new FinanceServiceStub(context,
serviceURL);
stub.updateFinancialData(requestDoc);
```

createFinancialData

Purpose

This operation creates financial data.

Related Information

Operations: *readFinancialData*, *updateFinancialData*

Input

This operation requires the following inputs:

- financialData: FinancialDataInfo
- regionName: String

Return

If the operation is successful, nothing will be returned to the client.

Java Examples

See webservice_toolkit:

`java\client\src\examples\fm\`**`FinancialDataClient.java`**

createFinancialData ()

readFinancialDataACL

Purpose

This operation reads a financial data ACL.

Related Information

Operation: *readFinancialSummaryACL*

Input

This operation requires the following input:

- financialDataParent: financialDataParentInfo

Return

Data as defined in *AccessListInfo*.

Java Examples

See webservice_toolkit:

`java\client\src\examples\fm\`**`FinancialDataClient.java`**

readACL ()

updateFinancialDataACL

Purpose

This operation updates a financial data ACL.

Related Information

Operation: *updateFinancialSummaryACL*

Input

This operation requires the following inputs:

- financialDataParent: financialDataParentInfo
- accessList: AccessListInfo

Return

If the operation is successful, nothing will be returned to the client.

Java Examples

See webservice_toolkit:

`java\client\src\examples\fm\`**`FinancialDataClient.java`**

updateACL()

5 HP Project Management Web Services

Overview

HP Project Management Web services provides interfaces for accessing, creating, updating, and transitioning projects through workflow steps in PPM Center. Operations for creating work plans, adding tasks to work plans, accessing task information and modifying actual time for various task assignments, are also supported. Additionally, operations such as searching for tasks and projects are provided.

For more information about general HP Project Management terms and concepts (such as, work plan, and project), see the *HP Project Management User's Guide.*

References

Data types definition:

`webservice_toolkit\java\conf\xsd\`**`Project.xsd`**

Operations definition:

`webservice_toolkit\java\conf\wsdl\`**`ProjectService.wsdl`**

Java sample code:

`webservice_toolkit\java\client\src\examples\pm\`
`ProjectServiceClient.java`

.NET sample code:

Operations History

HP Project Management Web services provides many operations starting with PPM Center version 7.1. Several new operations were added to PPM Center version 7.5. No new operations were added to PPM Center version 8.00. *Figure 5-1* lists the HP Project Management Web service operations by version.

Table 5-1. HP Project Management Web Service operations by Version

Web Service operation	7.1	7.5	8.00
createProject	Yes	Yes	Yes
updateProject	Yes	Yes	Yes
executeWorkFlowTransition	Yes	Yes	Yes
createBlankWorkPlan	Yes	Yes	Yes
createWorkPlanFromTemplate	Yes	Yes	Yes
importWorkPlanTasks	Yes	Yes	Yes
addTasksToExistingWorkPlan	Yes	Yes	Yes
updateTaskActuals	Yes	Yes	Yes
readTasks	Yes	Yes	Yes
searchProjects	Yes	Yes	Yes
searchTasks	Yes	Yes	Yes
getProjectDetails	No	Yes	Yes
bulkImportProjects	No	Yes	Yes
updateWorkPlanStatus	No	Yes	Yes
exportWorkPlanFromProject	No	Yes	Yes

Data Types

HP Project Management Web services includes the following data types:

ProjectType

This common data type is used in many project-related operations.

Property	Type	Description	Required	Default
projectName	String	Name of the project.	No	N/A
projectTypeName [a]	String	Name of the project type for the project.	No	N/A
projectTypeId [a]	Long	Numeric value used to identify a project type.	No	N/A
projectManagerUserName	String[]	List of all the project managers for the project.	No	N/A
plannedStartPeriodFullName	String	Full name of the project start period.	No	N/A
plannedFinishPeriodFullName	String	Full name of the project finish period.	No	N/A
regionName	String	Name of the project's region.	No	N/A
staffingProfileName	String	Name of the associated staffing profile.	No	N/A
budgetName	String	Name of the associated budget.	No	N/A
financialBenefitName	String	Name of the associated financial benefit.	No	N/A
simpleFields	SimpleFields	An array list of the following elements: • token • stringValue • dateValue	No	N/A
tables	Table	An array list of token and columns elements.	No	N/A

Property	Type	Description	Required	Default

pmReferences	pmReference	A list of references for the project. References contain the following information:
		• targetId (Integer)
		• targetTypeCode (Integer)
		• refRelationshipId (Integer)
		• referenceName (String)
		• relBehaviorCode (String):
		  • SUCCESSOR
		  • INFORMATIONAL
		  • PREDECESSOR
		  • FF_PREDECESSOR
		  • AUTO_UPDATE
		  • UPDATED_BY_STEP
		• refDescirption (String)
		The 'attachment' reference type is not yet supported through web services.
			No	N/A

notes	Note	Refer to Demand Management.	No	N/A
fieldChangeNotes	fieldChangeNote	Refer to Demand Management.	No	N/A

a. When you make a web service call, you can either specify the projectTypeName property or the projectTypeId property, but not both of them. When the projectType object is returned in response to the web service call, only the projectTypeName is populated.

CreateProjectResultType

The CreateProjectResult data type returned when a new project is created.

Property	Type	Description	Required	Default
projectName [a]	String	Name of the project.	No	N/A
projectId	Long	A numeric value used to identify a project.	Yes	N/A
requestId	Integer	A numeric value used to identify a request corresponding to the newly created project.	Yes	N/A

a. The projectName property is only present when you perform a bulkImportProject operation. This property is not set when you create a single project.

ProjectInputType

This data type is used to identify a project.

Property	Type	Description	Required	Default
projectName [a]	String	Name of the project.	No	N/A
projectId [a]	Long	Numeric value used to identify a project.	No	N/A
requestId [a]	Integer	Numeric value used to identify a request corresponding to the newly-created project.	No	N/A

a. You can either specify the projectName property, the projectId property, or the requestId property associated with the project, but not all of them. None of these properties is “required,” but at least one must be specified.

WorkPlanInputType

The WordPlanInputType data type is typically used for operations on a project's work plan.

Property	Type	Description	Required	Default
projectName [a]	String	Name of the project.	No	N/A
projectId [a]	Long	Numeric value used to identify a project.	No	N/A

a. You can either specify the projectName property or the projectId property, but not both of them. Neither of these properties is "required," but at least one must be specified.

Anchortype

This data type is used to order and outline tasks.

Property	Type	Description	Required	Default
taskSequenceNumber	Integer	Sequence number of the task. Task sequence starts from 0, which is the root task.	Yes	N/A
outLineLevel	Long	The outline level of the task. Task outLineLevel starts from 1, which is the root task.	Yes	N/A

taskAnchors

The TaskAnchors data type is used when you specify anchors for following operation: addingTasksToExistingWorkPlan.

Property	Type	Description	Required	Default
topAnchor	anchorType	Top anchor task. New tasks should be added after the top anchor task.	Yes	N/A
bottomAnchor	anchorType	Bottom anchor task. New tasks should be added before the bottom anchor task.	No	N/A

AddTaskResultType

The AddTaskResultType data type is the returned result for tasks that are added to a work plan.

Property	Type	Description	Required	Default
taskSequence Number	Long	Sequence number of the new task in the work plan.	Yes	N/A
taskId	Long	Numeric value used to identify the task that is being added to the work plan.	Yes	N/A

SearchProjectPreferenceType

The SearchProjectPreference type is used for project search.

Property	Type	Description	Required	Default
startSearchPosition	Integer	Start position of the project in the search result. This is used in paged search. Starts at position 1.	Yes	0
maximumProjectsToShow	Integer	Number of projects to be returned for the search criteria. Values range from 1 to 1,000 inclusively.	Yes	N/A
projectNameContains	String	String contained in the project name of the projects that you want to search for.	No	N/A
projectTypeNames	String[]	List of project type names you want to search for.	No	N/A
projectRegionNames	String[]	List of project region names of the projects that you want to search for.	No	N/A
associatedProgramNames	String[]	List of program names associated with the projects that you want to search for.	No	N/A
projectManagerUserNames	String	List of project manager user names of the projects you want to search for.	No	N/A
isIncludeFinishProject	boolean	Determines whether the search result should include finished projects.	No	N/A

Property	Type	Description	Required	Default
plannedStartFrom	dateTime	Start from date for the project you want to search for.	No	N/A
plannedStartTo	dateTime	Start to date for the project you want to search for.	No	N/A
plannedFinishFrom	dateTime	Finish from date for the project you want to search for.	No	N/A
plannedFinishTo	dateTime	Finish to date for the project you want to search for.	No	N/A
projectHealth	String[]	List of project health information you want to search for. Valid values for project health are: • RED • YELLOW • GREEN • NONE	No	N/A

SearchTaskPreferenceType

The SearchTaskPreferenceType data type is used for task search:

Property	Type	Description	Required	Default
startSearchPosition	Integer	Start position of the task in the search result. This is used in paged search. Starts at position 1.	Yes	N/A
maximumTasksToShow	Integer	Number of tasks to be returned for the search criteria. Values range from 1 to 1,000 inclusively.	Yes	N/A
taskNamePrefix	String	String the task name starts with.	No	N/A
projectNames	String[]	List of project names of the tasks that you want to search for.	No	N/A
resourceUserNames	String[]	List of resource user names of the tasks that you want to search for.	No	N/A
showOnlyMileStone	boolean	Determines whether the search result should show only milestone tasks.	No	N/A
showOnlyExceptionTask	boolean	Determines whether the search result should show only tasks with exceptions.	No	N/A
includeFinishedTask	boolean	Determines whether the search result should include finished tasks.	No	N/A
scheduledStartFrom	dateTime	Schedules the Start from date for the task you want to search for.	No	N/A

Property	Type	Description	Required	Default
scheduledStartTo	dateTime	Schedules the Start to date for the task you want to search for.	No	N/A
scheduledFinishFrom	dateTime	Schedules the Finish from date for the task you want to search for.	No	N/A
scheduledFinishTo	dateTime	Schedules the Finish to date for the task you want to search for.	No	N/A

AssignmentType

The Assignment Type data type is the returned result for tasks that are added to a work plan. This type is part of the *UpdateActualsInput* data type.

Property	Type	Description	Required	Default
taskId	Long	ID of the task, to which the assignment belongs.	No	N/A
scheduledEffort	Double	Scheduled effort for the assignment.	No	N/A
resource	Resource Type	Resources assigned to the task.	No	N/A
actualDuration	Double	Duration for the assignment.	No	N/A
percentComplete	Double	Completion percentage of the assignment, which is a value ranging from 0 to 100.	No	N/A
actualEffort	Double	Actual effort of the assignment.	No	N/A
estimatedRemainingEffort	Double	Estimated remaining effort of the assignment	No	N/A

Property	Type	Description	Required	Default
actualStart	DateTime	Actual start date for the assignment.	No	N/A
actualFinish	DateTime	Actual finish date for the assignment.	No	N/A
estimatedFinish	DateTime	Estimated finish date for the assignment.	No	N/A

UpdateActualsInput

The UpdateActualsInput data type defines the assignment's (or work unit's) actuals information in relation to a task:

Property	Type	Description	Required	Default
assignment	AssignmentType	Assignment of the task, for which you want to update actuals.	Yes	N/A
taskId	Long	ID of the task, for which you want to update actuals.	Yes	N/A

ResourceType

The ResourceType data type is the returned result for tasks being added to a work plan.

Property	Type	Description	Required	Default
resourceId	Long	ID of the resource assigned to a task.	Yes	N/A
firstName	String	First name of the resource assigned to a task.	No	N/A
lastName	String	Last name of the resource assigned to a task.	No	N/A

TaskType

The TaskType data type defines tasks in a work plan.

Property	Type	Description	Required	Default
taskId	Long	ID of the task.	No	N/A
taskName	String	Name of the task.	Yes	N/A
taskSequence	Long	Sequence number of the task.	Yes	N/A
outlineLevel	Long	Outline level of a task within a work plan.	Yes	N/A
taskStatus	String	The status of a task with the following valid values: • in-progress • completed • in-planning • ready • on-hold • active • cancelled • pending-predecessor • completed-pending-predecessor • completed-pending-request • pending-request	No	N/A
description	String	Description of the task.	No	N/A
priority	Long	Task priority.	No	N/A
isMileStone	boolean	Whether this task is a milestone.	No	N/A

Property	Type	Description	Required	Default
isMajorMilestone	boolean	Whether this task is a major milestone.	No	N/A
isMilestoneAutomaticallyCompletes	boolean	Whether this Milestone automatically completes.	No	N/A
isMilestoneManualConversion	boolean	Whether this milestone is a manual conversion.	No	N/A
isRequired	boolean	Whether this task is required for the work plan.	No	N/A
activity	activityType	Activity for this task.	No	N/A
owners	resourceType	A list of owners of the task.	No	N/A
workUnits	assignmentType	A list of assignments for the task.	No	N/A
actuals	taskActualType	Task level actuals.	No	N/A
roleBean	roleInfo	Task role.	No	N/A
schedulingBean	scheduleInfo	Task schedule.	Yes	N/A
costBean	costInfo	Task cost.	No	N/A
predecessors	dependencyInfo	Task predecessors.	No	N/A
notificationSetupBean	notificationSetupInfo	Task notification setup.	No	N/A
userData	userDataInfo	Task user Data.	No	N/A
skillProficiencies	skillProficiencyInfo	Task skill proficiencies.	No	N/A
notes	taskNoteInfo	Task notes.	No	N/A

Property	Type	Description	Required	Default
pmReferences	pmReference	Task references. Note that attachment is not a supported reference.	No	N/A

ActivityType

This type defines the activity for a task.

Property	Type	Description	Required	Default
activityId	Long	ID of the activity.	Yes	N/A
activityName	String	Name of the activity.	No	N/A

ScheduleInfo

The ScheduleInfo data type defines schedule information for a task.

Property	Type	Description	Required	Default
scheduledDuration	Double	Scheduled duration of a task.	No	N/A
scheduledEffort	Double	Scheduled effort of a task.	Yes	N/A
scheduledStart	DateTime	Scheduled start date of a task.	No	N/A
scheduledFinish	DateTime	Scheduled finish date of a task.	No	N/A
constraintType	String	Constraint Type of a task with the following valid values: • as-soon-as-possible • as-late-as-possible • start-no-earlier-than • start-no-later-than • finish-no-earlier-than • finish-no-later-than • must-start-on • must-finish-on	No	N/A
constraintDate	DateTime	Constraint date of the task constraint.	No	N/A

TaskActualType

The TaskActualType data type defines the actual information for a task.

Property	Type	Description	Required	Default
actualDuration	Double	Actual duration of a task.	No	N/A
percentComplete	Double	Completion percentage of a task.	No	N/A
actualEffort	Double	Actual effort of a task.	No	N/A
estimatedRemainingEffort	Double	Estimated remaining effort of a task	No	N/A
actualStart	DateTime	Actual start date of task.	No	N/A
estimatedFinish	DateTime	Estimated finish date of task.	No	N/A
actualFinish	DateTime	Actual finish date of task.	No	N/A

MoneyInfo

The MoneyInfo data type defines the money information related to task cost.

Property	Type	Description	Required	Default
localValue	Double	Money value corresponding to the currency code.	Yes	N/A
localCurrencyCode	String	Currency code used for the value.	Yes	N/A

CostBean

The CostBean data type defines the task cost information.

Property	Type	Description	Required	Default
actualCapLaborMoney	MoneyInfo	Actual capitalized labor money of the task.	No	N/A

Property	Type	Description	Required	Default
actualCapNonLaborMoney	MoneyInfo	Actual capitalized non-labor money for the task.	No	N/A
actualOpLaborMoney	MoneyInfo	Actual operating labor money of the task.	No	N/A
actualOpNonLaborMoney	MoneyInfo	Actual operating non-labor money for the task.	No	N/A
plannedCapLaborMoney	MoneyInfo	Planned capitalized labor money for the task.	No	N/A
plannedCapNonLaborMoney	MoneyInfo	Planned capitalized non-labor money for the task.	No	N/A
plannedOpNonLaborMoney	MoneyInfo	Planned operating non-labor money for the task.	No	N/A
plannedOpLaborMoney	MoneyInfo	Planned operating labor money for the task.	No	N/A

RoleInfo

The RoleInfo data type defines the role information for a task.

Property	Type	Description	Required	Default
roleId	Long	ID of the task role.	Yes	N/A
roleName	String	Name of the task role.	No	N/A

DependencyInfo

The DependencyInfo data type defines the role information for a task.

Property	Type	Description	Required	Default
taskName	String	Name of the predecessor task.	No	N/A
taskId	Long	ID of the predecessor task.	No	N/A

Property	Type	Description	Required	Default
isExternalPredecessor	boolean	Whether it is an external predecessor.	No	N/A
predTaskSeq	Long	Sequence number of the predecessor task.	Yes	N/A
predRelationType	String	Predecessor relationship with the following valid values: • start-to-finish • finish-to-finish • start-to-start • finish-to-start	Yes	N/A
lagInDays	Double	Lag in days for the predecessor.	No	N/A

Operations

The following operations are included in HP Project Management Web services:

- *createProject* on page 183
- *bulkImportProjects* on page 184
- *updateProject* on page 186
- *getProjectDetails* on page 187
- *executeWorkflowTransition* on page 188
- *createWorkPlanFromTemplate* on page 189
- *createBlankWorkPlan* on page 190
- *importWorkPlanTasks* on page 191
- *addTasksToExistingWorkPlan* on page 193
- *exportWorkPlanFromProject* on page 194

- *readTasks* on page 195
- *updateTaskActuals* on page 196
- *updateWorkPlanStatus* on page 197
- *searchProjects* on page 198
- *searchTasks* on page 199

createProject

Purpose

Create a project and its associated request in PPM Center.

Function

Create a project with name, project type, and period.

Related Information

Data Type: *ProjectType*

Input

ProjectType with the following required information:

- Project name
- Project type name or ID
- A list of project manager user names. At least one user name should be provided.
- Planned start period full name
- Planned finish period full name
- Region name

All other information is optional. Specify the information only when you need to create a project with the corresponding information.

Return

CreateProjectResultType

The project ID and request ID are returned if the project is created successfully.

Java Examples

See webservice_toolkit:

`java\client\src\examples\pm\`**`ProjectServiceClient.java`**

createProject()

```
CreateProjectDocument createProjDoc =
CreateProjectDocument.Factory.newInstance();
ProjectType projectBean =
createProjDoc.addNewCreateProject().addNewProjectBean();

projectBean.setProjectTypeName("Enterprise");
projectBean.setRegionName("US West Coast");
projectBean.addProjectManagerUserName("user1");
projectBean.setPlannedStartPeriodFullName("January 2007");
projectBean.setPlannedFinishPeriodFullName("June 2007");
projectBean.setProjectName(projectName);

ProjectServiceStub stub = new ProjectServiceStub(ctx, WSURL);
CreateProjectResponseDocument createProjectResponseDoc =
stub.createProject(createProjDoc);

CreateProjectResultType cpResult =
createProjectResponseDoc.getCreateProjectResponse().getReturn()
;

return cpResult;
```

bulkImportProjects

Purpose

Create multiple projects at one time.

Function

This operation creates projects and their associated requests according to the given input. All projects are created in one transaction.

There is no limitation on the maximum number of projects that can be imported at the same time. However, if too many projects are created at the same time (for example, more than 1,000 projects), the operation can fail when there is a network issue.

Related Information

Operation: *createProject*

Return

CreateProjectResultType[]

A list of project IDs, request IDs, and the corresponding project names are returned if all projects are created successfully. Otherwise, a SOAP fault with a detailed error message is returned.

Java Examples

```
BulkImportProjectsDocument importProjDoc =
BulkImportProjectsDocument.Factory.newInstance();
BulkImportProjects bip =
importProjDoc.addNewBulkImportProjects();
ProjectType projectBean = bip.addNewProjects();
… // Set projectBean for the first project

// Add the second input project bean
projectBean = bip.addNewProjects();
… // Set projectBean for the second project

BulkImportProjectsResponseDocument importProjectsResponseDoc =
stub.bulkImportProjects(importProjDoc);
// check the response
CreateProjectResultType[] cpResults =
importProjectsResponseDoc.getBulkImportProjectsResponse().getRe
turnArray();
```

updateProject

Purpose

Update an existing project.

Related Information

Operation: *createProject*

Input

The following inputs are required:

- One of the following:
  - Project name
  - Project ID
- ProjectType detail

Return

If the operation is successful, no results are returned to the client.

Java Examples

See webservice_toolkit:

`java\client\src\examples\pm\`**`ProjectServiceClient.java`**

updateProject()

```
UpdateProjectDocument updateProjDoc =
UpdateProjectDocument.Factory.newInstance();
UpdateProject up = updateProjDoc.addNewUpdateProject();
ProjectInputType pit = up.addNewProjectInput();
// Set the project name for update
pit.setProjectName(projectName);

ProjectType pb = up.addNewProjectBean();

// Update the project info
pb.addProjectManagerUserName("admin");
```

```
pb.setPlannedStartPeriodFullName("March 2007");

ProjectServiceStub stub = new ProjectServiceStub(ctx, WSURL);
stub.updateProject(updateProjDoc);
```

getProjectDetails

Purpose

This operation collects detailed information for a project.

Function

Not all information related to a project and its request is returned. Only the information relevant to ProjectType (refer to the *ProjectType* data type) is returned. If ProjectType details for a project do not exist, no results are returned.

Input

One of the following inputs is required:

- Project name
- Project ID.

Return

The project together with its associated request detailed information is returned.

Java Examples

```
GetProjectDetailsDocument gpdDoc =
GetProjectDetailsDocument.Factory.newInstance();
GetProjectDetails gpd = gpdDoc.addNewGetProjectDetails();
ProjectInputType pit = gpd.addNewProjectInput();
pit.setProjectName(projectName);

GetProjectDetailsResponseDocument gpdResDoc =
stub.getProjectDetails(gpdDoc);
```

```
ProjectType pt =
gpdResDoc.getGetProjectDetailsResponse().getReturn();
```

executeWorkflowTransition

Purpose

This operation executes one workflow transition step from the current available steps.

Input

The following inputs are required:

- Transition step name

  The step name must be one of the visible step names available on the **Project Details** tab in the PPM Center user interface.

- One of the following:
  - Project ID
  - Project name
  - Request ID

Return

If the operation is successful, no results are returned to the client.

Java Examples

See webservice_toolkit:

`java\client\src\examples\pm\`**`ProjectServiceClient.java`**

executeWorkflowTransition ()

```
ExecuteWorkflowTransitionDocument updateProjStatusDoc =
       ExecuteWorkflowTransitionDocument.Factory.newInstance();

ExecuteWorkflowTransition ups =
updateProjStatusDoc.addNewExecuteWorkflowTransition();
```

```
// create projectInputType
ProjectInputType pit = ups.addNewProjectInput();
pit.setProjectName(projectName);

ups.setTransition(transition);

stub.executeWorkflowTransition(updateProjStatusDoc)
```

createWorkPlanFromTemplate

Purpose

This operation creates a work plan for a specified project from a given template.

Input

The following inputs are required:

- Work plan template name.
- One of the following:
  - Project ID
  - Project name

Return

If the operation is successful, no results are returned to the client.

Java Examples

```
CreateWorkPlanFromTemplateDocument crtblkWpDoc =
CreateWorkPlanFromTemplateDocument.Factory.newInstance();
CreateWorkPlanFromTemplate cbwp =
crtblkWpDoc.addNewCreateWorkPlanFromTemplate();

WorkPlanInputType wpit = cbwp.addNewProjectInput();

// Set the value for WorkPlanInputType element's projectName
wpit.setProjectName(projectName);
```

```
// Set the value for CreateWorkPlanFromTemplate element's
templateName
cbwp.setTemplateName(templateName);
// Invoke service method
stub.createWorkPlanFromTemplate(crtblkWpDoc);
```

createBlankWorkPlan

Purpose

This operation creates a blank work plan for the specified project.

Input

One of the following:

- Project ID
- Project name

Return

If the operation is successful, no results are returned to the client.

Java Examples

See webservice_toolkit:

`java\client\src\examples\pm\`**`ProjectServiceClient.java`**

createBlankWorkPlan ()

```
// Create inpput document
CreateBlankWorkPlanDocument crtblkWpDoc =
CreateBlankWorkPlanDocument.Factory.newInstance();

// create and add an empty CreateBlankWorkPlan element
CreateBlankWorkPlan cbwp =
crtblkWpDoc.addNewCreateBlankWorkPlan();

// Create and add an empty WorkPlanInputType element
WorkPlanInputType wpit = cbwp.addNewProjectInput();

// Set the value for this element
wpit.setProjectName(projectName);
```

```
stub.createBlankWorkPlan(crtblkWpDoc);
```

importWorkPlanTasks

Purpose

This operation creates and imports all the tasks to the work plan for a specified project.

Function

This operation requires that the specified project not have a work plan created. If a work plan already exists, a SOAP fault is returned to the client. In this operation, you do not need to specify actual values in TaskActualType. Instead, actual values should be specified in the assignment for each resource. The assignment actual values are then finally rolled up to the task.

Input

The following inputs are required:

- One of the following:
  - Project ID
  - Project name
- TaskType details with the following required information:
  - Task name
  - Task sequence number
  - Task outline level
  - Task schedule information

Return

AddTaskResultType[]

If the operation is successful, a list of task sequence and task IDs corresponding to the tasks that are newly imported to the work plan is returned.

Java Examples

```
ImportWorkPlanTasksDocument impWPDoc =
ImportWorkPlanTasksDocument.Factory.newInstance();
ImportWorkPlanTasks impWP =
impWPDoc.addNewImportWorkPlanTasks();
WorkPlanInputType wpit = impWP.addNewWorkPlanInput();

// Set the value for WorkPlanInput element's projectName
wpit.setProjectName(projectName);

// Create and add an empty tasks element
TaskType task1 = impWP.addNewTasks();

// Set required properties for task
task1.setOutlineLevel(2);
task1.setTaskSequence(1);
task1.setTaskName("pm ws test importTask1" + uniqueSuffix);

// create and add task scheduling bean to task.
ScheduleInfo sif = task1.addNewSchedulingBean();
sif.setScheduledDuration(4);
sif.setScheduledEffort(34);
sif.setScheduledStart(taskScheduleStart);
sif.setScheduledFinish(taskScheduleFinish);
sif.setConstraintType(ConstraintType.AS_SOON_AS_POSSIBLE);

AssignmentType assignment = task1.addNewWorkUnits();
// Add resource to assignment
ResourceType resource = assignment.addNewResource();
resource.setResourceId(resourceId2.longValue());

assignment.setScheduledEffort(10);
assignment.setActualEffort(0);
assignment.setEstimatedRemainingEffort(10);
assignment.setPercentComplete(0);

// Create and add another empty tasks
TaskType task2 = impWP.addNewTasks();
… // Info for task2

// Calling service
ImportWorkPlanTasksResponseDocument impResponseDoc =
stub.importWorkPlanTasks(impWPDoc);

// Check the response and make sure we are getting it back ok
AddTaskResultType[] addedTasks =
impResponseDoc.getImportWorkPlanTasksResponse().getReturnArray(
);
```

addTasksToExistingWorkPlan

Purpose

This operation adds tasks to the work plan of the project.

Function

The position is defined by a top anchor and a bottom anchor. The top anchor is always required, but the bottom anchor can be null if you want to add tasks after the last existing task in the work plan.

This operation requires that the project have an existing work plan. In this operation, you do not need to specify actual values in TaskActualType. Instead, actual values should be specified in the assignment for each resource. The assignment actual values are then finally rolled up to the task.

Related Information

Operation: *importWorkPlanTasks*

Input

The following inputs are required:

- Project name or project ID
- TaskType details with the following required information:
  - Task name
  - Task sequence number
  - Task outline level
  - Task schedule information
- Position (topAnchor and bottomAnchor)

Return

AddTaskResultType[]

If the operation is successful, a list of task sequence and task IDs corresponding to the tasks newly added to the work plan is returned.

Java Examples

See webservice_toolkit:

`java\client\src\examples\pm\`**`ProjectServiceClient.java`**

addTaskToExistingWorkPlanWithTopAnchor ()

exportWorkPlanFromProject

Purpose

This operation exports all the tasks in a work plan.

Function

Rather than all data, only those properties that are defined in the TaskType data type are exported.

Input

One of the following:

- Project name
- Project ID

Return

TaskType[]

If the operation is successful, a list of tasks as defined in the TaskType is returned to the client.

Java Examples

```
ExportWorkPlanFromProjectDocument expWPDoc =
ExportWorkPlanFromProjectDocument.Factory.newInstance();

ExportWorkPlanFromProject expWP =
expWPDoc.addNewExportWorkPlanFromProject();
WorkPlanInputType wpit = expWP.addNewWorkPlanInput();

wpit.setProjectName(projectName);

ExportWorkPlanFromProjectResponseDocument expWPResDoc  =
stub.exportWorkPlanFromProject(expWPDoc);

TaskType[] tasks =
expWPResDoc.getExportWorkPlanFromProjectResponse().getReturnArr
ay();
```

readTasks

Purpose

This operation reads a list of tasks information corresponding to the given task IDs.

Input

The input specifies a list of task IDs for which you want to collect task information. The specified tasks do not need to belong to the same project.

Return

TaskType[]

If the operation is successful, a list of tasks as defined in the TaskType is returned to the client.

Java Examples

See webservice_toolkit:

`java\client\src\examples\pm\`**`ProjectServiceClient.java`**

readTasks ()

```
ReadTasksDocument readTskDoc =
ReadTasksDocument.Factory.newInstance();

ReadTasks  readTsks = readTskDoc.addNewReadTasks();

readTsks.addTaskId(taskId);

ReadTasksResponseDocument readTskresponseDoc =
stub.readTasks(readTskDoc);

TaskType[] tasks =
readTskresponseDoc.getReadTasksResponse().getReturnArray();
```

updateTaskActuals

Purpose

This operation allows the actuals of task assignments to be updated.

Function

Some actual values cannot be changed if the resource is a Time Management user and has actuals logged in PPM Center already. In this case, refer to the user interface behavior in PPM Center.

Input

The following inputs are required:

- Task ID
- Resource ID of the resource to which the assignment belongs
- Other actual values

Return

If the operation is successful, no results are returned to the client.

Java Examples

See webservice_toolkit:

`java\client\src\examples\pm\`**`ProjectServiceClient.java`**

updateTaskActuals ()

updateWorkPlanStatus

Purpose

This operation updates the status of the root task of a given project's work plan.

Function

You cannot give arbitrary values for the status. Basically, only those statuses available in the root tasks of the PPM Center user interface are allowed. If the status is invalid, a SOAP fault with detailed error message is returned to the client.

Input

The following inputs are required:

- One of the following:
  - Project name
  - Project ID
- Task status (Note: this is the status you want the root task of the work plan to be.)

Return

If the operation is successful, no results are returned to the client.

Java Examples

```
UpdateWorkPlanStatusDocument uwpsDoc =
UpdateWorkPlanStatusDocument.Factory.newInstance();
UpdateWorkPlanStatus uwps=
uwpsDoc.addNewUpdateWorkPlanStatus();
WorkPlanInputType wpi = uwps.addNewWorkPlanInput();

wpi.setProjectName(projectName);
// new status to set
uwps.setNewStatus("active");
stub.updateWorkPlanStatus(uwpsDoc);
```

searchProjects

Purpose

This operation returns project IDs based on the search criteria.

Input

The input specifies the project search criteria. In addition to the search criteria, you must specify the startSearchPosition property, which should begin with 1 and the maximumProjectsToShow property, which should be within the range of 1 to 1,000.

Return

long[]

A list of project IDs corresponding to the search criteria.

Java Examples

```
SearchProjectsDocument schProjDoc =
SearchProjectsDocument.Factory.newInstance();

SearchProjects schProjs = schProjDoc.addNewSearchProjects();
SearchProjectPreferenceType searchPreference =
schProjs.addNewSearchPreferences();
// Set the search preferences
searchPreference.setStartSearchPosition(start);
searchPreference.setMaximumProjectsToShow(maxResult);
// searchPreference.setProjectNameContains("pm ws");
```

```
SearchProjectsResponseDocument schProjRespDoc =
stub.searchProjects(schProjDoc);

long[] projIds =
schProjRespDoc.getSearchProjectsResponse().getProjectIdsArray()
;
```

searchTasks

Purpose

This operation returns task IDs based on the search criteria.

Input

The input specifies the task search criteria. In addition to the search criteria, you must specify the startSearchPosition property, which should begin with 1, and the maximumTasksToShow property, which should be within the range of 1 to 1,000.

Return

long[]

A list of task IDs corresponding to the search criteria.

Java Examples

See webservice_toolkit:

`java\client\src\examples\pm\`**`ProjectServiceClient.java`**

searchTasks ()

```
SearchTasksDocument schTsksDoc =
SearchTasksDocument.Factory.newInstance();
SearchTasks  schTsks = schTsksDoc.addNewSearchTasks();

// Create SearchPreference element
SearchTaskPreferenceType searchPreference =
schTsks.addNewSearchPreferences();

// searchPreference
searchPreference.setMaximumTasksToShow(10);
searchPreference.setStartSearchPosition(1);
```

```
searchPreference.setScheduledStartFrom(taskScheduleStart);
// searchPreference
searchPreference.setScheduledStartTo(taskScheduleFinish);

SearchTasksResponseDocument schTskRespDoc =
stub.searchTasks(schTsksDoc);
long[] taskids =
schTskRespDoc.getSearchTasksResponse().getTaskIdArray();
```

6 HP Resource Management Web Services

Overview

HP Resource Management Web services provides the following operations in PPM Center:

- Create, update, search, and fetch resource pools
- Get and set resource participation in resource pools
- Create roles and skills

For more information about general HP Resource Management terms and concepts, see the *HP Resource Management User's Guide*.

References

- Data types:

  `webservice_toolkit\java\conf\xsd\`**`Resource.xsd`**

- Operations:

  `webservice_toolkit\java\conf\wsdl\`**`ResourceService.wsdl`**

- Sample code:

  `webservice_toolkit\java\client\src\examples\rm`

Operations History

HP Resource Management Web services provides many operations starting with PPM Center version 7.1. No new operations were added to PPM Center version 8.00. *Table 6-1* lists the HP Resource Management Web service operations by version.

Table 6-1. HP Resource Management Web service operations by Version

Web Service Operation	7.1	7.5	8.00
createResourcePools	Yes	Yes	Yes
searchResourcePools	Yes	Yes	Yes
getResourcePools	Yes	Yes	Yes
updateResourcePools	Yes	Yes	Yes
getResourceParticipation	Yes	Yes	Yes
setResourceParticipation	Yes	Yes	Yes
createRoles	No	Yes	Yes
createSkills	No	Yes	Yes

Data Types

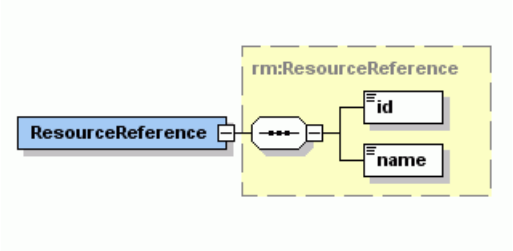
HP Resource Management Web services includes the following data types:

ResourceReference

ResourceReference is a value object that specifies the ID and the name of a resource. It is used in *ResourcePool* and *ResourceParticipation*.

WSCostRuleBean is used as the INPUT in the following operation:

- *getResourceParticipation*

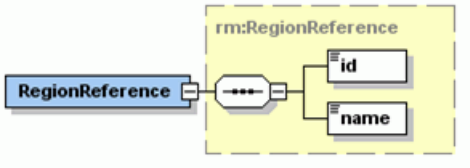


Property	Type	Description	Required	Default
id	Long	ID of the resource. A value that is typically greater than 30,000.	No	N/A
name	String	Name of the resource.	No	N/A

RegionReference

RegionReference is a value object that specifies the ID and the name of a region. It is used in *ResourcePoolSearchFilter* and *ResourcePool*.

For more information about the RegionReference, see *ResourcePool*.

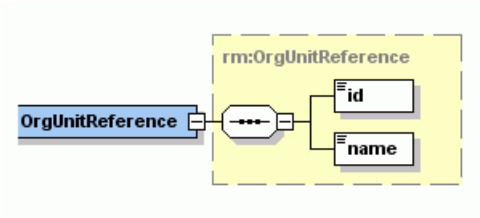


Property	Type	Description	Required	Default
id	Long	ID of the region.	No	N/A
name	String	Name of the region.	No	N/A

OrgUnitReference

OrgUnitReference is a value object that specifies the ID and the name of an organizational unit. It is used in *ResourcePoolSearchFilter* and *ResourcePool*.

For more information about the OrgUnitReference, see *ResourcePool*.



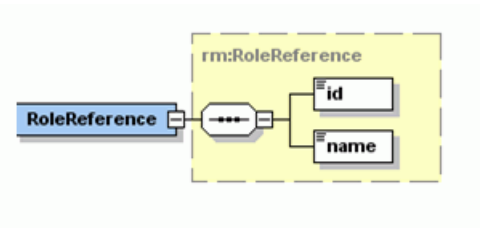
Property	Type	Description	Required	Default
id	Long	ID of the org unit.	No	N/A
name	String	Name of the org unit.	No	N/A

RoleReference

RoleReference is a value object that specifies the ID and the name of a role.

RoleReference is used as the OUTPUT in the following operation:

- Operation *createRoles*



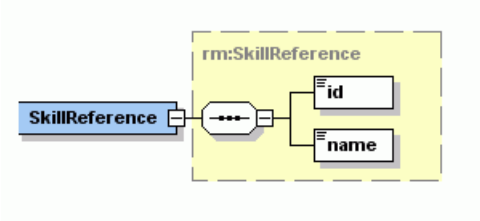
Property	Type	Description	Required	Default
id	Long	ID of the role.	No	N/A
name	String	Name of the role.	No	N/A

SkillReference

SkillReference is a value object that specifies the ID and the name of a skill.

SkillReference is used as the OUTPUT in the following operation:

- Operation *createRoles*



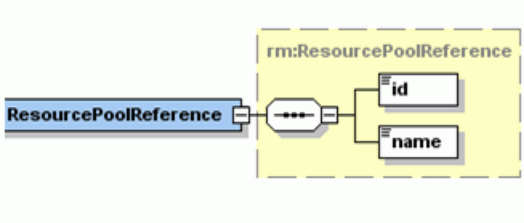
Property	Type	Description	Required	Default
id	Long	ID of the skill.	No	N/A
name	String	Name of the skill.	No	N/A

ResourcePoolReference

ResourcePoolReference is a value object that specifies the ID and the name of a resource pool.

ResourcePoolReference is used as the OUTPUT in the following operation:

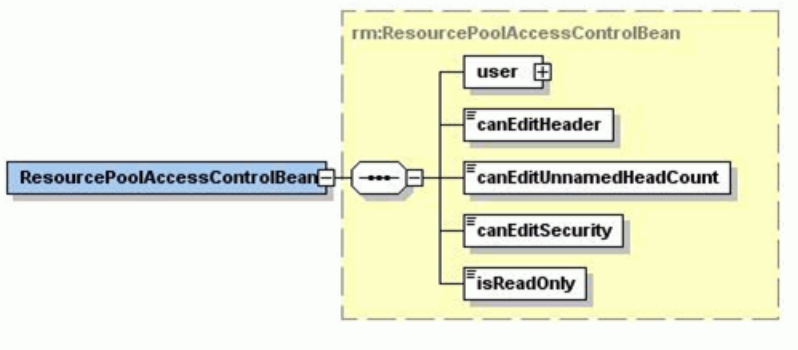
- Operation *createResourcePools*



Property	Type	Description	Required	Default
id	Long	ID of the resource pool.	No	N/A
name	String	Name of the resource pool.	No	N/A

ResourcePoolAccessControlBean

ResourcePoolAccessControlBean is a value object that specifies the user access privileges of a resource pool.



Property	Type	Description	Required	Default
user	Resource Reference	User/resource whose privileges the bean describes.	Yes	N/A
canEditHeader	boolean	Value that specifies whether the user can edit the resource pool header.	No	N/A
canEditUnnamed HeadCount	boolean	Value that specifies whether the user can edit unnamed head count in resource pool.	No	N/A
canEditSecurity	boolean	Value that specifies whether the user can edit resource pool security settings.	No	N/A
isReadOnly	boolean	Value that specifies whether the user is read only.	No	N/A

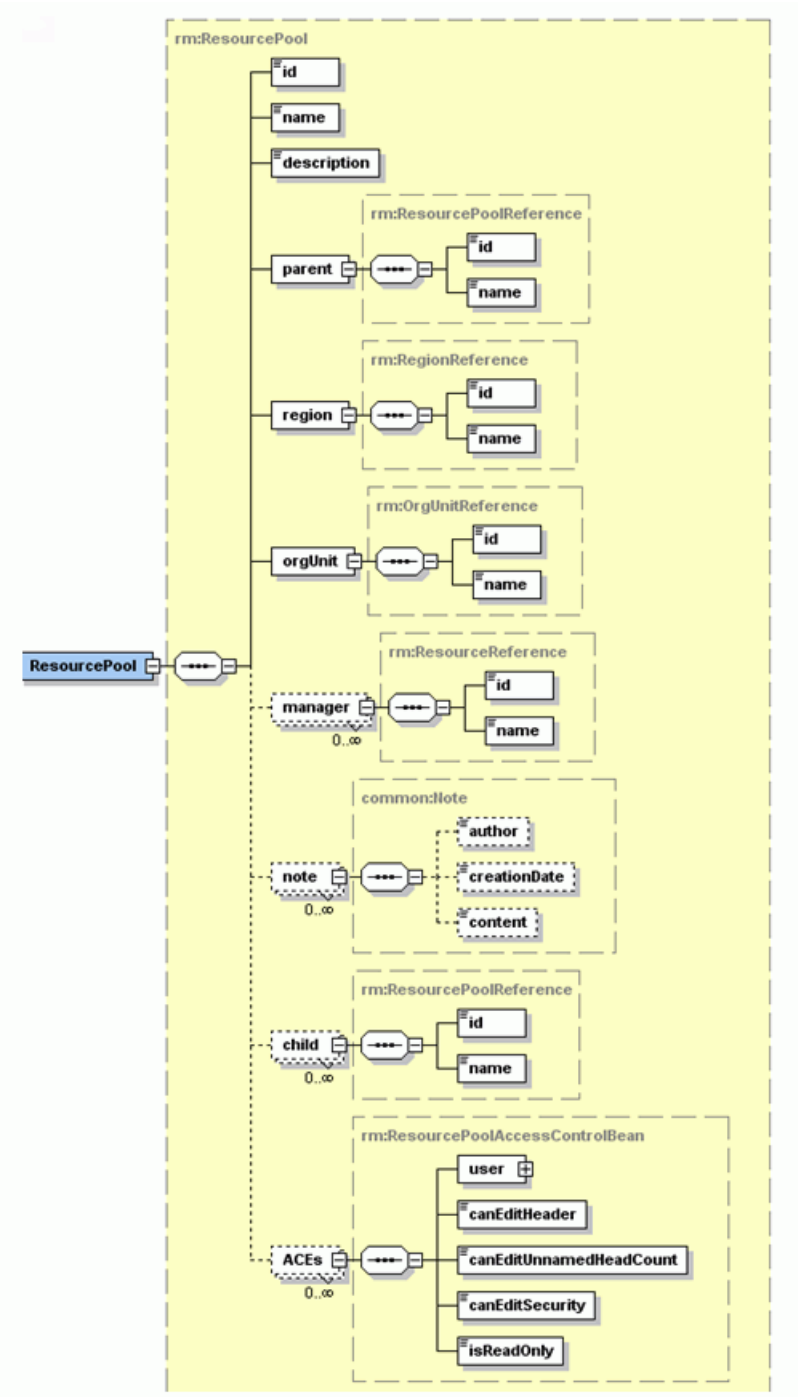
ResourcePool

ResourcePool is a value object that specifies the resource pool information. The bean is used to create resource pools in the system.

- ResourcePool is used as the INPUT in the following operations:
  - Operation *createResourcePools*
  - Operation *updateResourcePools*

- ResourcePool is used as the OUTPUT in the following operations:
  - Operation *searchResourcePools*
  - Operation *getResourcePools*

.


rm:ResourcePool
id
name
description
rm:ResourcePoolReference
parent
id
name
rm:RegionReference
region
id
name
rm:OrgUnitReference
orgUnit
id
name
ResourcePool
rm:ResourceReference
manager
0..∞
id
name
common:Note
note
0..∞
author
creationDate
content
rm:ResourcePoolReference
child
0..∞
id
name
rm:ResourcePoolAccessControlBean
ACEs
0..∞
user
canEditHeader
canEditUnnamedHeadCount
canEditSecurity
isReadOnly

Property	Type	Description	Required	Default
id	Long	ID of the resource pool. A value which is typically greater than 30,000.	No	N/A
name	String	Name of the resource pool. Name cannot be more than 260 characters.	Yes	N/A
description	String	Description of the resource pool. Description cannot be more than 650 characters.	No	N/A
parent	ResourcePoolRefernce	This is a ResourcePoolRefernce value that contains the ID and name of the parent Resource Pool.	No	N/A
region	RegionReference	This is a RegionReference value that contains the ID and name of the pool's region.	Yes	N/A
orgUnit	OrgUnitReference	This is a OrgUnitReference value that contains the ID and name of the org unit the resource is used for.	No	N/A
manager	List	This is a list of ResourceReference values that contains the ID and names of the resource pool managers.	No	N/A
note	List	This is a list of NoteBean values that contains Author, Content, and CreationDate.	No	N/A

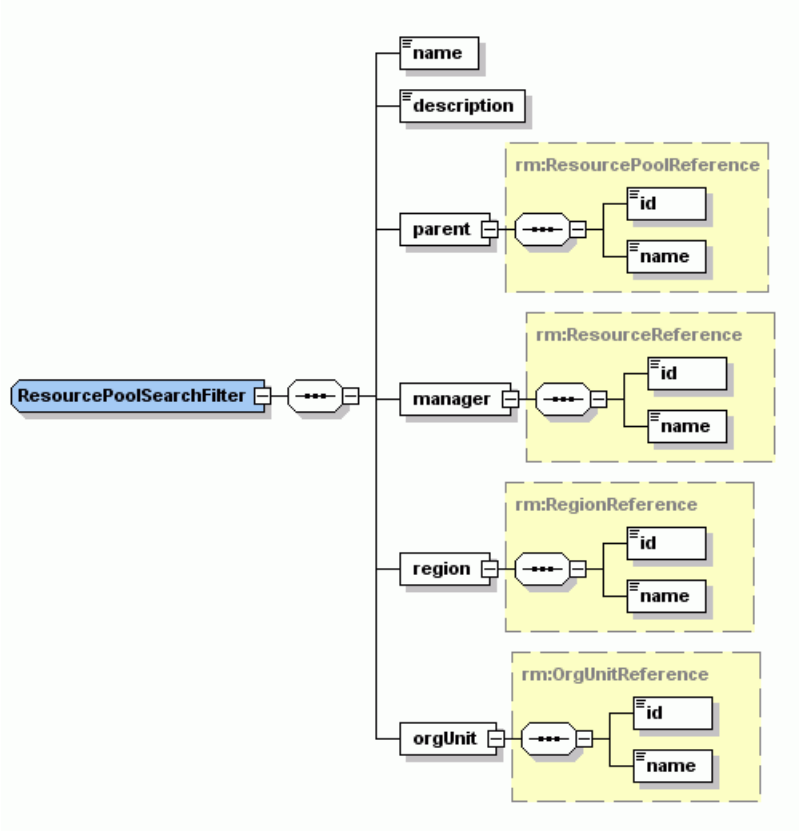
Property	Type	Description	Required	Default
child	List	This is a list of ResourcePoolRefernce values that contains the IDs and names of the children resource pools.	No	N/A
ACEs	List	This is a list of ResourcePoolAccessControlBean values.	No	N/A

ResourcePoolSearchFilter

The search filter criteria bean that web services uses to search for resource pools in the system.

ResourcePoolSearchFilter is used as the INPUT in the following operations:

- Operation *searchResourcePools*



Property	Type	Description	Required	Default
name	String	Name of the resource pool.	No	N/A
description	String	Description of the resource pool.	No	N/A
parent	ResourcePoolRefernce	This is a ResourcePoolRefernce value that contains the ID and name of the parent Resource Pool.	No	N/A
region	RegionReference	This is a RegionReference value that contains the ID and name of the pool's region.	Yes	N/A

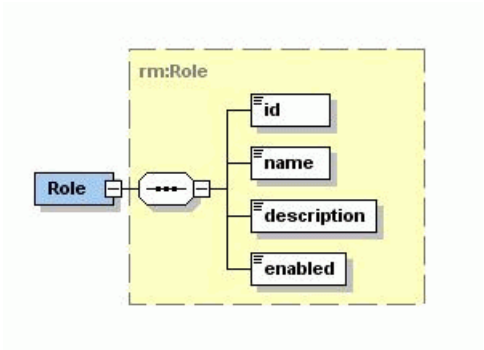
Property	Type	Description	Required	Default
orgUnit	OrgUnitReference	This is a OrgUnitReference value that contains the ID and name of the org unit the resource belongs to.	No	N/A
manager	List	This is a list of ResourceReference values that contains the ID and name of the resource pool manager.	No	N/A

Role

Role is a value object that specifies the role information. It is also used to create roles.

Role is used as the INPUT in the following operations:

- Operation *createRoles*



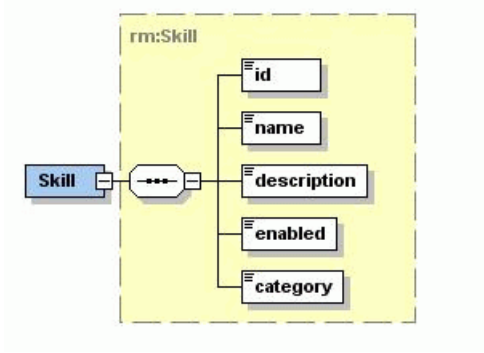
Property	Type	Description	Required	Default
id	Long	ID of the role.	No	N/A
name	String	Name of the role.	Yes	N/A
description	String	Description of the role.	No	N/A
enabled	boolean	Boolean value - true if the role is enabled.	Yes	N/A

Skill

Skill is a value object that specifies the skill information. It can also be used to create skills.

Skill is used as the INPUT in the following operations:

- Operation *createSkills*

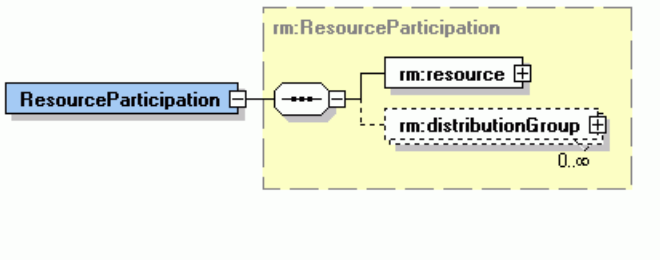


Property	Type	Description	Required	Default
id	Long	ID of the skill.	No	N/A
name	String	Name of the skill.	Yes	N/A
description	String	Description of the skill.	No	N/A
enabled	boolean	Boolean value - true if the skill is enabled.	Yes	N/A
category	String	Category of the skill.	No	N/A

ResourceParticipation

ResourceParticipation is a value object that specifies the participation of a resource in one or more resource pools. It can also be used to get resource participations.

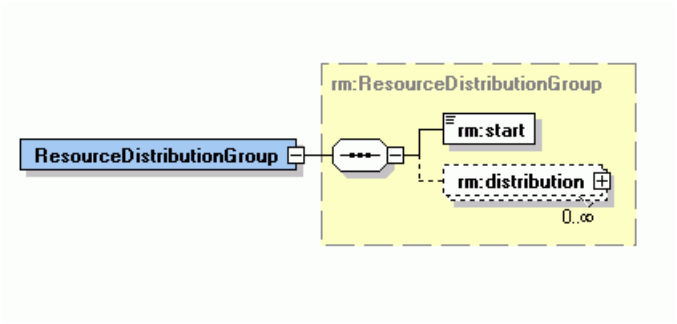
- ResourceParticipation is used as the INPUT in the following operations:
  - Operation *setResourceParticipation*
- ResourceParticipation is used as the OUTPUT in the following operation:
  - Operation *getResourceParticipation*



Property	Type	Description	Required	Default
resource	*RoleReference*	This is a ResourceReference value that contains the ID and name of the resource.	Yes	N/A
distribution Group	*ResourceDistributionGroup*	This is a ResourceDistributionGroup value that contains the startDate and the distributions of the resource pools.	No	N/A

ResourceDistributionGroup

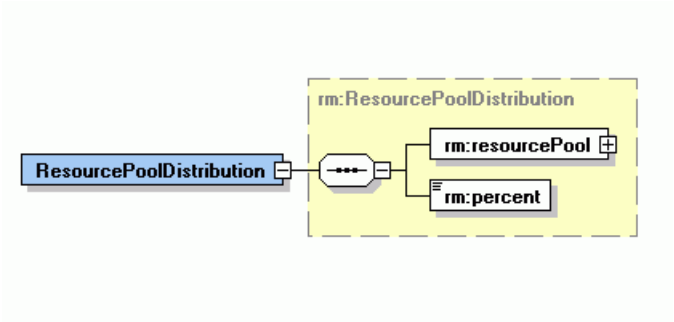
ResourceDistributionGroup is a value object that specifies the distribution of a resource among one or more resource pools starting on a given date.



Property	Type	Description	Required	Default
start	Date	This value specifies the start date for the given resource.	No	N/A
distribution	*ResourceDistributionGroup*	This is a ResourcePoolDistribution value that contains the resource pool information and the allocation percentage.	No	N/A

ResourcePoolDistribution

ResourcePoolDistribution is a value object that specifies the allocation of a resource to a specific resource pool.



Property	Type	Description	Required	Default
resourcePool	*ResourcePool Reference*	This is a ResourcePoolRefernce value that contains the ID and name of the specific Resource Pool.	No	N/A
percent	Double	This value specifies how much percentage a resource is allocated to the resource pool.	No	N/A

Operations

The following operations are included in HP Resource Management Web services:

createResourcePools

Purpose

This operation creates one or more new resource pool objects in PPM Center.

Function and Parameters

This operation creates one or more (maximum 1,000) new resource pool objects, identified by resource pool ID in PPM Center.

The user performing this operation must have the following access grants:

- Create Resource Pools
- Edit Resource Pool or Edit All Resource Pools

Required Fields:

- Name
- Region

Optional Fields:

- ID
- Description
- Parent
- Org unit
- Manager
- Note
- Child
- Access Control

If these optional fields are not set, no defaults are used.

Limitations

This operation has the following limitations:

- User data cannot be created.
- This operation assumes basic data exists in the database. The following data must exist in any production PPM Center instance.
  - Resources
  - Users
  - Parent resource pool
  - Org Unit
  - Region

All values provided in the parameters must comply with what is expected in PPM Center. The possible values are provided above when each parameter is described.

- If a data problem occurs with any data provided for the resource pools, the entire data set is rejected. No resource pools are created.

Related Information

updateResourcePools - update fields of an existing resource pool.

Input

An array of *ResourcePools*.

Return

An array of *ResourcePoolReferences*.

Java Interface

```
CreateResourcePoolsResponseDocument createResourcePools(
CreateResourcePoolsDocument in)
```

Parameters	Description
CreateResourcePoolsDocument	Wrapper of the resource pool array object (WSResourcePoolBean[]). See the following example for the construction. The bean includes the following fields: • Long id; • String name; • String description; • ResourcePoolRefernce parent; • RegionReference region; • OrgUnitReference orgUnit; • List manager; • List Notes; • ResourcePoolRefernce[] child; • ResourcePoolAccessControlBean[] aces;
CreateResourcePoolsResponseDocument	Wrapper for the resource pool ref array (ResourcePoolReference[]). See the following example for retrieving the bean and its fields.

Java Examples

Example: create a new resource pool.

```
/**
     * Test creating a resource pool with values for all fields,
reading it back out and searching for it.
     */
    public long testCreateResourcePool() throws Exception {

        // create the resource pool object
ResourcePool resourcePool1 =
createTestResourcePoolObject("TestResourcePool1");
ResourcePool resourcePool2 =
createTestResourcePoolObject("TestResourcePool2");

        // call the service
```

```
ResourcePoolReference[] resourcePoolReference =
createResourcePools(new ResourcePool[] { resourcePool1,
resourcePool2 });

        // return  first pool ID
return resourcePoolReference[0].getId();

    }
/**
     * This is a wrapper method around the createResourcePools
web service. It handles the details of creating
     * the document, invoking the web service and unwrapping the
response.
     *
     * @param resourcePools    Zero or more resource pools to
create in PPM.
     * @return A resource pool reference for each object that
was successfully created.
     * @throws Exception
     */
    ResourcePoolReference[] createResourcePools(ResourcePool[]
resourcePools) throws Exception {
        ResourceServiceStub service = new
ResourceServiceStub(ctx, WSURL);
    CreateResourcePoolsDocument createResourcePoolsDoc =
CreateResourcePoolsDocument.Factory.newInstance();

createResourcePoolsDoc.addNewCreateResourcePools().setResourceP
oolArray(resourcePools);
    CreateResourcePoolsResponseDocument responseDocCreate =
service.createResourcePools(createResourcePoolsDoc);

CreateResourcePoolsResponse responseCreate =
responseDocCreate.getCreateResourcePoolsResponse();

debugPrint(responseCreate, "create response");

return responseCreate.getResourcePoolRefArray();
    }

    public static ResourcePool
createTestResourcePoolObject(String name) {

        ResourcePool resourcePool =
ResourcePool.Factory.newInstance();

        // resource pool name
        resourcePool.setName(name);

        // description
        resourcePool.setDescription("A resource pool created
programmatically through web services ");

        // region
```

```
        resourcePool.addNewRegion().setName("America");

        // pool managers
        resourcePool.addNewManager().setName("admin");

        // set parent pool
         resourcePool.addNewParent().setName("parent pool
name");

        // add children pool
         resourcePool.addNewChild().setName("child pool name");


        // org unit
        resourcePool.addNewOrgUnit().setName(" org unit name");

        // add notes
         Note note = resourcePool.addNewNote();
         note.setContent("Note content");
         note.setAuthor("admin");

        // set access control list

        for (int i = 0; i < usersInACL.length; i++) {
            ResourcePoolAccessControlBean acb =
resourcePool.addNewACEs();
            ResourceReference aceUser = acb.addNewUser();
            aceUser.setName(usersInACL[i]);
            acb.setCanEditHeader(true);
            acb.setCanEditSecurity(true);
            acb.setCanEditUnnamedHeadCount(false);
        }


        return resourcePool;
    }
}
```

Errors and Exceptions

When an error occurs on this operation, you will receive a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault:
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Possible root cause descriptions:

:

Message Code	Message	Cause(s)	Possible Corrective Action
error.ws.maxResourcePools	The maximum number of resource pools to process in a single web service transaction may not exceed 1000.	The number of resource pools exceeds 1,000.	Break up the list of resource pools into smaller chunks and process over multiple transactions.
error.ws.create.duplicateName	A resource pool with the name "{0}" already exists. Enter a different name.	A resource pool with the name already exists.	Enter a different name.
error.ws.create.cycle	Cannot create resource pool as it would introduce a cyclical parent-child relationship between the resource pools {0} and {1}.	Two resource pools have each other as parents.	Make sure that the resource pools do not have each other as parents.
error.ws.regionRequired	Resource pool region is required. Specify a region for each resource pool.	The region for a resource pool is not specified.	Specify the region for the resource pool.
error.ws.nameRequired	Resource pool name cannot be blank. Provide a name for each resource pool.	The resource pool name is not set for a pool.	Set the resource pool name.
error.ws.descriptionTooLong	Description cannot be longer than 650 characters.	Description of a resource pool is more than 650 characters.	Reduce length of description.

Message Code	Message	Cause(s)	Possible Corrective Action
error.ws.nameTooLong	Name cannot be longer than 260 characters.	Name of a resource pool is great than 260.	Reduce length of name.
N/A	Entity Validation errors if region, manager, resource pool does not exist.	Resource pool region, manager, or parent resource pool does not exist.	Ensure that Resource pool region, manager, and parent resource pool exist for all pools to be created before this web service call.

searchResourcePools

Purpose

This operation searches for resource pool objects in PPM Center based on certain criteria.

Function and Parameters

This operation returns all resource pool objects matching the search criteria from PPM Center.

The user performing this operation must have one of the following access grants:

- View Resource Pools
- View All Resource Pools
- Edit Resource Pool
- Edit All Resource Pools

Required Fields: None

Optional Fields:

- Name
- Description
- Parent
- Org unit
- Manager
- Region

If none of the optional fields are set, all resource pools are returned. If more than one criteria are specified, they are combined with each other by the AND logic.

Limitations

Performance might be an issue if more than a few thousand resource pools are read at once.

Input

ResourcePoolSearchFilter

Return

ResourcePool matching the criteria.

Java Interface

```
CreateResourcePoolsResponseDocument createResourcePools(
CreateResourcePoolsDocument in)
```

Parameters	Description
SearchResourcePoolsDocument	Wrapper of the resource pool search filter. (ResourcePoolSearchFilter)

Parameters	Description
SearchResourcePoolsResponseDocument	Wrapper for the resource pool array. (WSResourcePoolBean[])

Java Examples

Example: search resource pools.

```
/**
 * Create a search filter with all fields set to match the non-
null fields of a    *specific resource pool, thereby creating a
search filter that should match only that *one resource pool.
* @param target    The resource pool that the filter will be
made for.
* @return the search filter.
*/
ResourcePoolSearchFilter createSearchFilter(ResourcePool
target) {
  ResourcePoolSearchFilter filter =
ResourcePoolSearchFilter.Factory.newInstance();
  filter.setName(target.getName());    // required field
  filter.setDescription(target.getDescription());
 filter.addNewRegion().setName(target.getRegion().getName());
// required field
  if (target.sizeOfManagerArray() > 0) {
      // use the first of the resource pool's managers in the
search query

filter.addNewManager().setName(target.getManagerArray(0).getNam
e());
  }
  if (target.getOrgUnit() != null) {

filter.addNewOrgUnit().setName(target.getOrgUnit().getName());
  }
  if (target.getParent() != null) {

filter.addNewParent().setName(target.getParent().getName());
  }
  return filter;
}


/**
 * This is a wrapper method around the searchResourcePools web
service, which reads *resource pools based on filter criteria.
This method handles the details of creating *the document,
invoking the service and unwrapping the response.
*
```

```
* @param filter    A bunch of filter fields to narrow down the
search.
* @return        Zero or more resource pools that match the
search criteria.
* @throws Exception
*/
ResourcePool[] searchResourcePools(ResourcePoolSearchFilter
filter) throws Exception {
    ResourceServiceStub service = new ResourceServiceStub(ctx,
WSURL);
    SearchResourcePoolsDocument searchResourcePoolsDoc =
SearchResourcePoolsDocument.Factory.newInstance();

searchResourcePoolsDoc.addNewSearchResourcePools().setFilter(fi
lter);

    SearchResourcePoolsResponse searchResponse =
service.searchResourcePools(searchResourcePoolsDoc).getSearchRe
sourcePoolsResponse();

    return searchResponse.getResourcePoolArray();
}
```

Errors and Exceptions

When an error occurs on this operation, you will see a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault:
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Possible root cause descriptions
:

Message Code	Message	Cause(s)	Possible Corrective Action
exception.authorization	You do not have the privilege to take this action. Consult your PPM Administrator.	You do not have the required access grants for this operation.	Verify that your account has the required access grants.

getResourcePools

Purpose

This operation fetches resource pool objects in PPM Center based on the resource pool IDs.

Function and Parameters

This operation returns one or more resource pool objects according to a specified list of resource pool IDs from PPM Center. If no IDs are specified or if there are no resource pools with those IDs, no records are returned.

The user performing this operation must have one of the following access grants:

- View Resource Pools
- View All Resource Pools
- Edit Resource Pool
- Edit All Resource Pools

Required Fields:

- Resource pool IDs

Optional Fields: None

Limitations

Performance might be an issue if more than a few thousand resource pools are read at once.

Input

Resource pool IDs

Return

An array of *ResourcePool*.

Java Interface

```
CreateResourcePoolsResponseDocument createResourcePools(
CreateResourcePoolsDocument in)
```

Parameters	Description
GetResourcePoolsDocument	Wrapper array of resource pool IDs. To construct this parameter: long [] ids;
GetResourcePoolsResponseDocument	Wrapper for the resource pool array. (WSResourcePoolBean[])

Java Examples

Example: read resource pools.

```
/**
 * This is a wrapper method around the getResourcePools web
service, which reads resource pools by ID. This
 * method handles the details of creating the document, invoking
the web service and unwrapping the response.
 *
* @param ids    The primary keys of the resource pools to
retrieve.
* @return    The resource pools that match the given primary
keys; primary keys that don't correspond to entities in the
database are silently ignored.
* @throws Exception
*/
ResourcePool[] getResourcePools(long[] ids) throws Exception {
    ResourceServiceStub service = new ResourceServiceStub(ctx,
WSURL);

GetResourcePoolsDocument getResourcePoolsDoc =
GetResourcePoolsDocument.Factory.newInstance();

GetResourcePools getResourcePools =
getResourcePoolsDoc.addNewGetResourcePools();

getResourcePools.setResourcePoolIdArray(ids);

GetResourcePoolsResponseDocument responseGet =
service.getResourcePools(getResourcePoolsDoc);
    debugPrint(responseGet, "get response");
    return
responseGet.getGetResourcePoolsResponse().getResourcePoolArray(
);
}
```

Errors and Exceptions

When an error occurs on this operation, you will see a description of the root cause in the log or in the response message:

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault:
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
exception.authorization	You do not have the privilege to take this action. Consult your PPM Administrator.	You do not have the required access grants for this operation.	Verify that your account has the required access grants.

updateResourcePools

Purpose

This operation updates existing resource pool objects in PPM Center.

Function and Parameters

This operation returns one or more (maximum 1,000) updated resource pool objects from PPM Center according to a specified list of resource pool objects. The user performing this operation must have one of the following access grants:

- Edit Resource Pool
- Edit All Resource Pools

Required Fields:

- ID

Optional Fields:

- Name
- Region
- Description
- Parent
- Org unit
- Manager
- Note
- Child
- ACEs

Limitations

This operation has the following limitations:

- User data cannot be created.
- This operation assumes basic data exists in the database. The following data must exist in any production PPM Center instance:
  - Resources
  - Users
  - Parent resource pool
  - Org Unit
  - Region

  All values provided in the parameters must comply with what is expected in PPM Center. The possible values are provided above when each parameter is described.
- If a data problem occurs with any data provided for the resource pools, the entire data set is rejected. No resource pools are created.

Related Information

createResourcePools- to create new resource pools.

Input

ResourcePool

Return

None

Java Interface

```
UpdateResourcePoolsResponseDocument
updateResourcePools(UpdateResourcePoolsDocument in)
```

Parameters	Description
UpdateResourcePoolsDocument	Wrapper array of resource pool objects. (WSResourcePoolBean[])

Java Examples

Example: update a resource pool

```
/**
* Test updating a resource pool.
*/
public void testUpdateResourcePool(long poolId) throws
Exception {

    // query the resource pool
    ResourcePool[] resourcePools = getResourcePools(new long[]
{poolId});
    ResourcePool pool = resourcePools[0];

    // update the pool object
    pool.setDescription("New version of the pool");

    // remove existing manager
    pool.setManagerArray(new ResourceReference[0]);

    // add new manager
    pool.addNewManager().setName("user1");

    // remove note
    pool.setNoteArray(new Note[] {});

    // call the service
    updateResourcePools(new ResourcePool[] { pool} );

}

/**
* This is a wrapper method around the updateResourcePools web
service, which modifies existing resource
* pools. This method handles the details of creating the
document, invoking the web service and unwrapping
* the response.
* @param resourcePools
* The changes that should be made to existing resource pools.
Resource pools are
* identified by primary key, so the ID of each resource pool
must be non-null.
* @throws Exception
*/
```

```
void updateResourcePools(ResourcePool[] resourcePools) throws
Exception {
    ResourceServiceStub service = new ResourceServiceStub(ctx,
WSURL);
    UpdateResourcePoolsDocument updateResourcePoolsDoc =
UpdateResourcePoolsDocument.Factory.newInstance();

updateResourcePoolsDoc.addNewUpdateResourcePools().setResourceP
oolArray(resourcePools);
    service.updateResourcePools(updateResourcePoolsDoc);
}
```

Errors and Exceptions

When an error occurs on this operation, you will see a description of the root cause in the log or in the response message:

```
Exception in thread "main" org.apache.axis2.AxisFault:
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Possible root cause descriptions:

:

Message Code	Message	Cause(s)	Possible Corrective Action
error.ws.update.id Required	When updating a resource pool you must provide the ID of the resource pool you wish to modify. The resource pool name is not sufficient.	The pending update Resource pool ID is null.	Enter the resource pool ID.
error.ws.maxResourcePools	The maximum number of resource pools to process in a single web service transaction may not exceed 1000.	The number of resource pools exceeds 1,000.	Break up the list of resource pools into smaller chunks and process over multiple transactions.

Message Code	Message	Cause(s)	Possible Corrective Action
error.ws.create.duplicateName	A resource pool with the name "{0}" already exists. Enter a different name.	A resource pool with the name already exists.	Enter a different name.
error.ws.create.cycle	Cannot create resource pool as it would introduce a cyclical parent-child relationship between the resource pools {0} and {1}.	Two resource pools have each other as parents.	Make sure that the resource pools do not have each other as parents.
error.ws.regionRequired	Resource pool region is required. Specify a region for each resource pool.	The region for a resource pool is not specified.	Specify the region.
error.ws.nameRequired	Resource pool name cannot be blank. Provide a name for each resource pool.	The resource pool name is not set for a pool.	Set the resource pool name.
error.ws.descriptionTooLong	Description cannot be longer than 650 characters.	The description of a resource pool contains more than 650 characters	Reduce length of description.
error.ws.nameTooLong	Name cannot be longer than 260 characters.	The name of a resource pool is greater than 260.	Reduce length of name.
N/A	Entity Validation errors if region, manager, resource pool does not exist.	The resource pool region, manager, or parent resource pool does not exist.	Ensure that Resource pool region, manager, and parent resource pool exist for all pools to be created before this web service call.

Message Code	Message	Cause(s)	Possible Corrective Action
exception.cannot LoadEntity	Cannot load the resource pool with the specified id: {1}. Resource Pool may be deleted by another user.	The resource pool does not exist.	Ensure the pending update resource pool exists.

getResourceParticipation

Purpose

This operation fetches participations of resources in various resource pools.

Function and Parameters

This operation returns an array of resource distribution groups from PPM Center according to a specified list of (maximum 1,000) resource IDs. The user performing this operation must have the following access grant:

- View Resource

Required Fields:

- *ResourceReference*

Optional Fields: None

Limitations:

- Maximum 1,000 ResourceReferences can be passed in a single invocation.

Related Information

setResourceParticipation - to set resource participations

Input

An array of *ResourceReference*.

Return

An array of *ResourceParticipation*.

Java Interface

```
GetResourceParticipationResponseDocument
getResourceParticipation (GetResourceParticipationDocument in)
```

Parameters	Description
GetResourceParticipationDocument	Wrapper array of resource references. (ResourceReference[])
GetResourceParticipationResponseDocument	Wrapper for the resource participation array. (ResourceParticipationBean[])

Java Examples

Example: get resource participation.

```
/**
     * This is a wrapper method around the
getResourceParticipation web service, which reads the
participation of
     * resources in resource pools. This method handles the
details of creating the document, invoking the web service
     * and unwrapping the response.
     *
     * @param participation        Zero or more
ResourceParticipation objects, which encapsulate the
participation details
     *                             for a resource.
     * @throws Exception
     */

    public ResourceParticipation[]
getResourceParticipation(ResourceReference[] resources) throws
Exception {
        ResourceServiceStub service = new
ResourceServiceStub(ctx, RESOURCE_SERVICE);
GetResourceParticipationDocument getParticipationDoc =
GetResourceParticipationDocument.Factory.newInstance();

getParticipationDoc.addNewGetResourceParticipation().setResourc
eArray(resources);
```

```
return
service.getResourceParticipation(getParticipationDoc).getGetRes
ourceParticipationResponse().getResourceParticipationArray();
    }
/**
     * Return a resource reference object with the given name
and ID.
     */
    ResourceReference resourceRef(Long id, String name) {
        ResourceReference ref =
ResourceReference.Factory.newInstance();
        if (id != null) ref.setId(id.longValue());
        ref.setName(name);
        return ref;
    }
```

Errors and Exceptions

When an error occurs on this operation, you will see a description of the root cause in the log or in the response message:

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault:
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
exception.authorization	You do not have the privilege to take this action. Consult your PPM Administrator.	You do not have the required access grants for this operation.	See above for list of access grants required for this operation.
error.ws.maxResources	The maximum number of resource to process in a single web service transaction may not exceed 1000.	The number of resource exceeds 1,000.	Break up the list of resource into smaller chunks and process over multiple transactions.

setResourceParticipation

Purpose

This operation sets the participation of resources in various resource pools.

Function and Parameters

This operation takes an array of (maximum 1,000) resource participation objects to set the participation of resources in different resource pools. The user performing this operation must have one of the following access grants:

- Edit Resource Pools
- Edit All Resource Pools

The start date of each resource distribution group in the same participation must be unique.

The sum of percentages in each resource distribution group must be in the range of 0 to 100.

Resource distribution group start date cannot be outside the range of resource start and end date in PPM Center.

Each resource distribution group may contain at most one distribution per resource pool.

Required Fields:

- *ResourceParticipation*

Optional Fields: None

Limitations

Up to 1,000 resource participation objects can pass in a single invocation.

Related Information

getResourceParticipation - to get resource participations.

Input

ResourceParticipation

Return

None

Java Interface

```
SetResourceParticipationResponseDocument
setResourceParticipation (SetResourceParticipationDocument in)
```

Parameters	Description
SetResourceParticipationDocument	Wrapper array of resource participations. (ResourceParticipationBean[])

Java Examples

Example: set resource participation

```
/**
     * Test creating a resource participation for a resource and
then modifying it.
     */
    public void testSetResourceParticipation() throws Exception
{
        String resource = "user1";

        // create participation object
        ResourceParticipation p =
ResourceParticipation.Factory.newInstance();
        p.addNewResource().setName(resource);

    ResourceDistributionGroup group1 =
ResourceDistributionGroup.Factory.newInstance();
        group1.setStart(calendar("Jan 1, 2005"));
        group1.setDistributionArray(new
ResourcePoolDistribution[] {
                createDistribution("TestResourcePool1", 25),
                createDistribution("TestResourcePool2", 50)
        });

    ResourceDistributionGroup group2 =
ResourceDistributionGroup.Factory.newInstance();
        group2.setStart(calendar("Jan 1, 2006"));
```

```
        group2.setDistributionArray(new
ResourcePoolDistribution[] {
                createDistribution("TestResourcePool1", 5)
        });

    ResourceDistributionGroup group3 =
ResourceDistributionGroup.Factory.newInstance();
        group3.setStart(calendar("Jan 1, 2007"));
        group3.setDistributionArray(new
ResourcePoolDistribution[] {
                createDistribution("TestResourcePool1", 90),
                createDistribution("TestResourcePool2", 10)
        });

        p.setDistributionGroupArray(new
ResourceDistributionGroup[] { group1, group2, group3 });

        // call service
       setResourceParticipation(new ResourceParticipation[] { p
});

    }
    /**
     * This is a wrapper method around the
setResourceParticipation web service, which manages the
participation of
     * resources in resource pools. This method handles the
details of creating the document, invoking the web service
     * and unwrapping the response.
     *
     * @param participation Zero or more ResourceParticipation
objects, which encapsulate the participation details
     * for a resource.
     * @throws Exception
     */
    public void
setResourceParticipation(ResourceParticipation[] participation)
throws Exception {
        ResourceServiceStub service = new
ResourceServiceStub(ctx, WSURL);
    SetResourceParticipationDocument setParticipationDoc =
SetResourceParticipationDocument.Factory.newInstance();

setParticipationDoc.addNewSetResourceParticipation().setResourc
eParticipationArray(participation);
        service.setResourceParticipation(setParticipationDoc);
    }
```

Errors and Exceptions

When an error occurs on this operation, you will see a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault:
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
error.ws.participation.badPercent	The resource distribution group with start date {0} for resource {1} has an invalid total percentage. The sum of percentages in each group must be in the range 0 to 100	The sum of percentages in the resource distribution group exceeds 100.	Ensure the sum of percentages is between 0 and 100.
error.ws.participation.duplicateStartDate	Duplicate start date: {0}. The start date of each resource distribution group in the same participation should be unique.	The start date of each resource distribution group in the same participation is not unique.	The start date of each resource distribution group in the same participation should be unique.

Message Code	Message	Cause(s)	Possible Corrective Action
error.ws.participation.duplicateDistribution	The resource distribution group with start date {0} for the resource {1} has multiple distributions for the same resource pool. Each group may contain at most one distribution per resource pool.	The resource distribution group with start date {0} for the resource {1} has multiple distributions for the same resource pool.	Each group may contain at most one distribution per resource pool.
error.ws.participation.resourceUnavailable	The PPM start or end date of the resource {0} makes the resource unable to participate in a distribution group that starts on {1}.	Distribution group start date is outside the range of the resource start and end date in PPM Center.	Adjust either the start and end date for the resource in PPM Center or the distribution group start date.
exception.authorization	You do not have the privilege to take this action. Consult your PPM Administrator.	You do not have the required access grants for this operation.	Verify that your account has the required access grants.

createRoles

Purpose

This operation creates multiple new roles in PPM Center.

Function and Parameters

This operation returns zero or more role reference objects from PPM Center according to a list of (maximum 1,000) role beans to create roles. The user performing this operation must have the following access grant:

- Edit All Roles

Required Fields:

- Name
- Enabled

Optional Fields:

- Description
- ID

Limitations

This operation has the following limitations:

- Maximum 1,000 roles can be created in a single invocation.
- Creation of new roles using this operation occurs in a single transaction and as an atomic operation. If the creation of one role fails, the whole operation is rolled back.

Input

An array of *Role*.

Return

An array of *RoleReference*.

Java Interface

```
CreateRolesResponseDocument createRoles(CreateRolesDocument in)
```

Parameters	Description
CreateRolesDocument	Wrapper array of role bean. (RoleBean[])
CreateRolesResponseDocument	Wrapper for the roles reference array. (RoleReference[])

Java Examples

Example: create new roles.

```
public static void main(String[] args) throws Exception {
      // check parameter
         if (args.length < 1) {
             System.out.println("Usage: java
ResourceServiceClient <service URL> [<true/false>]");
             System.exit(1);
         }

         System.out.println("Starting Resource Service
tests...");
         ResourceServiceClient rm = new ResourceServiceClient();
         rm.WSURL = args[0];
        if (args.length > 1 && args[1].equalsIgnoreCase("true"))
{
             rm.DEBUG = true;
         }


         System.out.println("Test create role ...");
              Role role = Role.Factory.newInstance();
                 role.setName("Test Role " +
System.currentTimeMillis());
                 role.setDescription("New Role");
                 role.setEnabled(true);
              RoleReference[] roleReferences = createRoles(new
Role[]{role});

         System.out.println("Resource Service tests complete.");
    }
```

```
  RoleReference[] createRoles(Role[] roles) throws Exception {
        ResourceServiceStub service = new
ResourceServiceStub(ctx, RESOURCE_SERVICE);
        CreateRolesDocument createRolesDoc =
CreateRolesDocument.Factory.newInstance();
        createRolesDoc.addNewCreateRoles().setRoleArray(roles);
        CreateRolesResponseDocument responseDocCreate =
service.createRoles(createRolesDoc);
        CreateRolesResponse responseCreate =
responseDocCreate.getCreateRolesResponse();
        debugPrint(responseCreate, "create response");
        return responseCreate.getRoleRefArray();
    }
```

Errors and Exceptions

When an error occurs on this operation, you will see a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault:
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
error.ws.maxRoles	The maximum number of roles to process in a single web service transaction may not exceed 1000.	The role list has more than 1,000 roles.	Break up the list into smaller chunks and process over multiple transactions.
error.duplicateName	A role with the name "{0}" already exists. Enter a different name.	The role already exists.	Enter a different name.

Message Code	Message	Cause(s)	Possible Corrective Action
error.ws.nameRequiredRole	Name cannot be blank. Provide a name for each role.	At least one of the roles does not have a name.	Ensure all roles have a name.
error.ws.descriptionTooLong	Description cannot be longer than 650 characters.	The description of a role is more than 650 characters.	Reduce length of description.
error.ws.nameTooLong	Name cannot be longer than 260 characters.	The name of a role is great than 260.	Reduce length of name.

createSkills

Purpose

This operation creates multiple new skills in PPM Center.

Function and Parameters

This operation returns zero or more skill reference objects from PPM Center according to a specified list of (maximum 1,000) skill beans to create skills. The user performing this operation must have the following access grant:

- Edit All Skills

Required Fields:

- Name
- Enabled

Optional Fields:

- Description
- ID
- category

Limitations

This operation has the following limitations:

- Maximum 1,000 skills can be created in a single invocation.
- Creation of skills using this operation occur in one single transaction and as an atomic operation. If the creation of one skill fails, the whole operation is rolled back.

Input

An array of *Skill*.

Return

An array of *SkillReference*.

Java Interface

```
CreateSkillsResponseDocument createSkills(CreateSkillsDocument
in)
```

Parameters	Description
CreateSkillsDocument	Wrapper array of skill bean. (SkillBean[])
CreateSkillsResponseDocument	Wrapper for the skill reference array. (SkillReference [])

Java Examples

Example: create new skills.

```
public static void main(String[] args) throws Exception {
        // check parameter
        if (args.length < 1) {
            System.out.println("Usage: java
ResourceServiceClient <service URL> [<true/false>]");
            System.exit(1);
        }

        System.out.println("Starting Resource Service
tests...");
        ResourceServiceClient rm = new ResourceServiceClient();
        rm.WSURL = args[0];
       if (args.length > 1 && args[1].equalsIgnoreCase("true"))
{
            rm.DEBUG = true;
        }


        System.out.println("Test create skill ...");

      Skill skill = Skill.Factory.newInstance();
          skill.setName("Test Skill " +
System.currentTimeMillis());
          skill.setDescription("Test Skill");
          skill.setEnabled(true);
          SkillReference[] skillReferences = createSkills(new
Skill[]{skill});

        System.out.println("Resource Service tests complete.");
```

```
    }

SkillReference[] createSkills(Skill[] skills) throws Exception
{
    ResourceServiceStub service = new ResourceServiceStub(ctx,
RESOURCE_SERVICE);
    CreateSkillsDocument createSkillsDoc =
CreateSkillsDocument.Factory.newInstance();
    createSkillsDoc.addNewCreateSkills().setSkillArray(skills);
    CreateSkillsResponseDocument responseDocCreate =
service.createSkills(createSkillsDoc);
    CreateSkillsResponse responseCreate =
responseDocCreate.getCreateSkillsResponse();
    debugPrint(responseCreate, "create response");
    return responseCreate.getSkillRefArray();
}
```

Errors and Exceptions

When an error occurs with this operation, you will see a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault:
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
error.ws.maxSkills	The maximum number of skills to process in a single web service transaction may not exceed 1000.	The skill list has more than 1,000 skills.	Break up the list into smaller chunks and process over multiple transactions.

Message Code	Message	Cause(s)	Possible Corrective Action
error.duplicateName	A skill with the name "{0}" already exists. Enter a different name.	The skill of the name is duplicated with an existing one.	Enter a different name.
error.ws.nameRequiredSkill	Name cannot be blank. Provide a name for each skill.	At least one of the skills does not have a name.	Ensure all skills have a name.
error.ws.descriptionTooLong	Description cannot be longer than 650 characters.	The description of a skill is more than 650 characters.	Reduce length of description.
error.ws.nameTooLong	Name cannot be longer than 260 characters.	The name of a skill is great than 260.	Reduce length of name

7 HP Time Management Web Services

Overview

HP Time Management Web services provides interfaces for accessing, creating, and updating time sheets in PPM Center. Operations for searching, submitting, approving, rejecting, reworking, freezing, closing, and canceling time sheets are also supported. Additionally, operations such as getting the actual time for work items are provided.

References

Data types definition:

`webservice_toolkit\java\conf\xsd\`**`Time.xsd`**

Operations definition:

`webservice_toolkit\java\conf\wsdl\`**`TimeService.wsdl`**

Java sample code:

`webservice_toolkit\java\client\src\examples\tm\`
`TimeServiceClient.java`

Operations History

HP Time Management Web services provides many operations starting with PPM Center version 7.1. *Table 7-1* lists the HP Time Management Web service operations by version.

Table 7-1. HP Time Management Web Service operations by Version

Web Service Operation	7.1	7.5	8.00
getActualTime	Yes	Yes	Yes
createTimeSheet	No	Yes	Yes
updateTimeSheet	No	Yes	Yes
getTimeSheet	No	Yes	Yes
searchTimeSheets	No	Yes	Yes
getTimeSheetPolicy	No	Yes	Yes
submitTimeSheet	No	Yes	Yes
approveTimeSheet	No	Yes	Yes
approveTimeSheetLine	No	Yes	Yes
rejectTimeSheet	No	Yes	Yes
rejectTimeSheetLine	No	Yes	Yes
reworkTimeSheetLine	No	Yes	Yes
freezeTimeSheet	No	Yes	Yes
closeTimeSheet	No	Yes	Yes
cancelTimeSheet	No	Yes	Yes

Terms and Concepts

Time Sheet Policy

Time sheet policies are rules that control the creation and operation of time sheets. You can configure different time sheet policies, and then apply them to different resources individually as needed. For example, you may want to apply different time sheet policies to different resources depending on their business units or whether they are employees or contractors. Every resource is assigned a time sheet policy. One time sheet policy serves as the default (global) time sheet policy.

Period Type

A period type specifies the interval over which time sheets report the time that users worked on work items. The available period types are:

- Weekly

  If the system uses this period type, each time period covers one week, starting by default on a Monday.

- Bi-Weekly

  If the system uses this period type, each time period covers two weeks, starting by default on a Monday.

- Semi-Monthly

  If the system uses this period type, there are two time periods per month, and the first time period always ends on the 15th of the month.

- Monthly

  If the system uses this period type, each time period covers a full month.

Time Period

A time period (period) is a particular date range of a period type.

Charge Code

Charge codes are entities used as links between work items and charge accounts.

Activity

Activities are used to categorize work performed against a work item, such as design work or coding. Activities can also be used to classify work as depreciable for financial accounting.

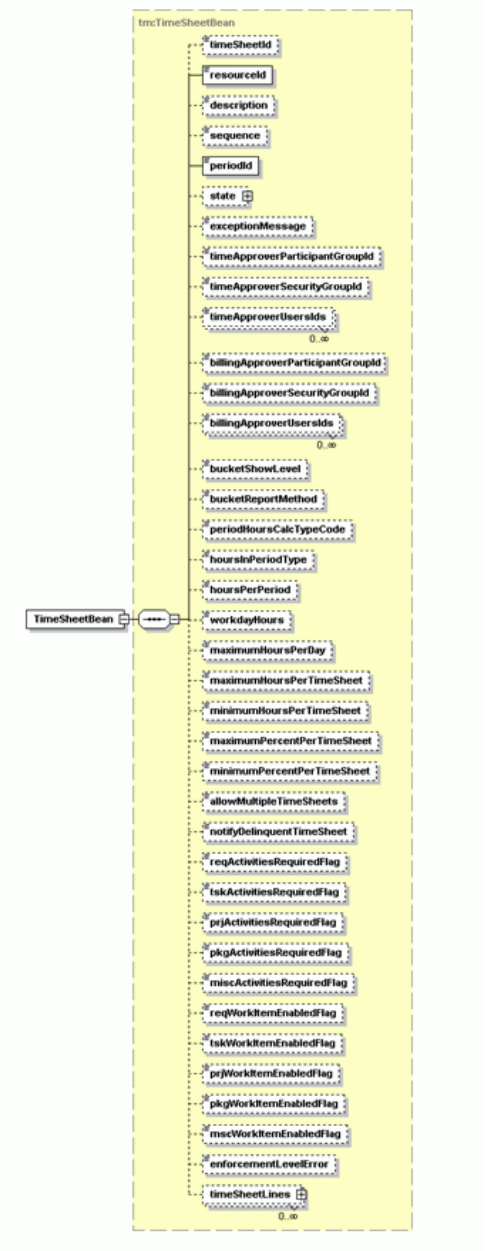
Data Types

HP Time Management Web services includes the following data types:

- *TimeSheetBean* on page 257
- *TimeSheetLineBean* on page 269
- *TimeActualsBean* on page 276
- *ChargeCodeBean* on page 278
- *UserDataBean* on page 279
- *TimeSheetStatus* on page 281
- *TimeSheetLineStatus* on page 282
- *TimeSheetPolicyBean* on page 284
- *TimeSheetSearchCriteriaBean* on page 291
- *TimeFilter* on page 293
- *WorkItemActualTime* on page 295

TimeSheetBean

This is the common type in operations with time sheets. The policy fields are optional.



Property	Type	Description	Required	Default
timeSheetId	Long	Generated automatically when the time sheet object is created in PPM Center. This field is not required in the createTimeSheet operation, but it is required for other operations that use TimeSheetBean.	No	N/A
resourceId	Long	ID of the resource of the time sheet.	Yes	N/A
periodId	Long	Period ID of the Time Period of the time sheet.	Yes	N/A
description	String	Description of the time sheet. If you do not provide this property when you create a time sheet, it is automatically created with the default value: resourceName - periodName #sequence	No	resourceName - periodName #sequence
sequence	Integer	Read-only field. Numeric value starting from 1. Unique per resource and time period. If the policy allows multiple time sheets per resource and period, each subsequent time sheet that you create is automatically increment the sequence number.	No	1 (the first one)
state	TimeSheetStatus	Status of the time sheet. On the time sheet creation, the default value is “unsubmitted.” See *TimeSheetStatus* for more details.	No	unsubmitted
timeSheetLines	List	List of TimeSheetLineBean objects. See *TimeSheetLineBean*	No	N/A

Property	Type	Description	Required	Default
bucketShowLevel	String	Part of policy fields. The unit of measurement of each time bucket. Valid values: • PERIOD • DAY • HOURS It is the only field that the system checks to determine whether policy fields are set. If you do not set this field, the system discards the values in the following fields marked with policy fields.	No	N/A
bucketReportMethod	String	Part of policy fields. Defines how time sheets report time. Valid values: • HOURS • PERCENT It is required when bucketShowLevel is set.	No	N/A
hoursPerPeriod	Double	Part of policy fields. Numeric value of how many hours should be per period. This field is relevant only when all of the following conditions are true: • The value for bucketShowLevel is PERIOD. • The value for bucketReportMethod is PERCENT. • The value for periodHoursCalcTypeCode is FIXED.	No	N/A

Property	Type	Description	Required	Default
periodHoursCalcTypeCode	String	Part of policy fields. Defines how period in hours should be calculated. Valid values: • FIXED (for period) • WORKDAY (for days). This field is relevant only when both of the following conditions are true: • The value for bucketShowLevel is PERIOD. • The value for bucketReportMethod is PERCENT.	No	N/A
workdayHours	Double	Part of policy fields. Numeric value of how many hours should be in a workday. This field is relevant only when all of the following conditions are true: • The value for bucketShowLevel is PERIOD. • The value for bucketReportMethod is PERCENT. • The value for periodHoursCalcTypeCode is WORKDAY.	No	N/A
allowMultipleTimeSheets	Boolean	Part of policy fields. Boolean value - Set this field to true if the policy allows multiple time sheets. The default value for this field is False, which means multiple time sheets are not allowed by default. This field is required if you set the bucketShowLevel field.	No	False

Property	Type	Description	Required	Default
hoursInPeriodType	String	Part of policy fields. Defines how period in hours should be calculated. Valid values: • FIXED (for period) • WORKDAY (for hours)	No	N/A
notifyDelinquentTimeSheet	Boolean	Part of policy fields. Boolean value - Set this field to true if you need the notification of delinquent time sheets. False otherwise. It is required if you set the bucketShowLevel field.	No	False
maximumHoursPerDay	Double	Part of policy fields. Numeric value of the max hours per day to enforce. This field is relevant only when both of the following conditions are true: • The value for bucketShowLevel is Day. • The value for bucketReportMethod is HOURS.	No	N/A
maximumHoursPerTimeSheet	Double	Part of policy fields. Numeric value of the max hours per time sheet to enforce. This field is relevant only when the value for bucketReportMethod is HOURS.	No	N/A
minimumHoursPerTimeSheet	Double	Part of policy fields. Numeric value of the minimum hours per time sheet to enforce. This field is relevant only when the value for bucketReportMethod is HOURS.	No	N/A

Property	Type	Description	Required	Default
maximumPercentPerTimeSheet	Double	Part of policy fields. Numeric value of the maximum percent per time sheet to enforce. This field is relevant only when both of the following conditions are true: • The value for bucketShowLevel is PERIOD • The value for bucketReportMethod is PERCENT	No	N/A
minimumPercentPerTimeSheet	Double	Part of policy fields. Numeric value of the min percent per time sheet to enforce. This field is relevant only when both of the following conditions are true: • The value for bucketShowLevel is PERIOD. • The value for bucketReportMethod is PERCENT.	No	N/A
enforcementLevelError	Boolean	Part of policy fields. Boolean value - Set this field to True if you want the system to prevent the submission of a time sheet when the submission violates the policy enforcements. Set it to False if you want the system to display only a warning when a violation occurs. It is required if you set the bucketShowLevel field.	No	False
reqActivitiesRequiredFlag	Boolean	Part of policy fields. Boolean value - Set this field to True if activities are required for Request work items, False otherwise. It is required if you set the bucketShowLevel field.	No	False

Property	Type	Description	Required	Default
tskActivitiesRequiredFlag	Boolean	Part of policy fields. Boolean value - Set this field to True if activities are required for Task work items, False otherwise). It is required if you set the bucketShowLevel field.	No	False
prjActivitiesRequiredFlag	Boolean	Part of policy fields. Boolean value - Set this field to True if activities are required for Project work items, False otherwise. It is required if you set the bucketShowLevel field.	No	False
pkgActivitiesRequiredFlag	Boolean	Part of policy fields. Boolean value - Set this field to True if activities are required for Package work items, False otherwise. It is required when bucketShowLevel is set.	No	False
miscActivitiesRequiredFlag	Boolean	Part of policy fields. Boolean value - Set this field to True if activities are required for Misc work items, False otherwise. It is required if you set the bucketShowLevel field.	No	False

Property	Type	Description	Required	Default
reqWorkItemEnabledFlag	Boolean	Part of policy fields. Boolean value - Set this field to True if you want to track Request work items for this time sheet, False otherwise. It is required if you set the bucketShowLevel field. The default value is taken from server.conf. com.kintana.core.server.PURGE_SERVER_PROPERTIES_TABLE=true com.kintana.core.server.ENABLE_TM_WORK_ITEM_REQUESTS=true	No	Taken from server.conf
tskWorkItemEnabledFlag	Boolean	Part of policy fields. Boolean value - Set this field to True if you want to track Task work items for this time sheet, False otherwise. It is required if you set the bucketShowLevel field. The default value is taken from server.conf. com.kintana.core.server.PURGE_SERVER_PROPERTIES_TABLE=true com.kintana.core.server.ENABLE_TM_WORK_ITEM_TASKS=true	No	Taken from server.conf

Property	Type	Description	Required	Default
prjWorkItemEnabledFlag	Boolean	Part of policy fields. Boolean value - Set this field to True if you want to track Project work items for this time sheet, False otherwise. It is required if you set the bucketShowLevel field. The default value is taken from server.conf. com.kintana.core.server.PURGE_SERVER_PROPERTIES_TABLE=true com.kintana.core.server.ENABLE_TM_WORK_ITEM_PROJECTS=true	No	Taken from server.conf
pkgWorkItemEnabledFlag	Boolean	Part of policy fields. Boolean value - Set this field to True if you want to track Package work items in this time sheet, False otherwise. It is required if you set the bucketShowLevel field. The default value is taken from server.conf.com.kintana.core.server. PURGE_SERVER_PROPERTIES_TABLE=true com.kintana.core.server. ENABLE_TM_WORK_ITEM_PACKAGES=true	No	Taken from server.conf

Property	Type	Description	Required	Default
mscWorkItemEnabledFlag	Boolean	Part of policy fields. Boolean value - True if you want to track Misc work items in this time sheet, False otherwise). It is required when bucketShowLevel is set. The default value is taken from server.conf. com.kintana.core.server.PURGE_SERVER_PROPERTIES_TABLE=true com.kintana.core.server. ENABLE_TM_WORK_ITEM_MISC =true	No	Taken from server.conf
exceptionMessage	String	Read-only field. If there is a violation of the policy, this field is set by the system automatically when you save the Time Sheet.	No	N/A
timeApproverParticipantGroupId	Long	Read-only field. The system reevaluates possible approvers when you save and load the time sheet. At the same time, the system sets this field automatically. This field is set only if the possible Time Approver of all the lines in the time sheet is a Participant Group (such as Project Managers). Otherwise it is null.	No	N/A
timeApproverSecurityGroupId	Long	Read-only field. The system reevaluates possible approvers when you save and load the time sheet. At the same time, the system sets this field automatically. This field is set only if the possible Time Approver of all the lines in the time sheet is a Security Group. Otherwise it is null.	No	N/A

Property	Type	Description	Required	Default
timeApproverUserIds	List	Read-only field. List of Long object values, each representing a User ID. The system reevaluates possible approvers when you save and load the time sheet. At the same time, the system sets this field automatically. This field must be set on reevaluation of approvers using the following order of precedence: • If timeApproverParticipantGroupId is not null, this field is set to the IDs of all the users in this participant group. • Else, if timeApproverSecurityGroupId is not null, this field is set to the IDs of all the users in this security group. • Else, if all lines have the same user as possible time approver, this field is set to the IDs of this user ID. • Else, this field is set to the ID of the default time approver of the resource of this time sheet.	No	N/A
billingApproverParticipantGroupId	Long	Read-only field. The system reevaluates possible approvers when you save and load the time sheet. At the same time, the system sets this field automatically. This field is set only if the possible Billing Approver of this time sheet is a Participant Group (such as Project Managers). Otherwise it is null.	No	N/A

Property	Type	Description	Required	Default
billingApproverSecurityGroupId	Long	Read-only field. The system reevaluates possible approvers when you save and load the time sheet. At the same time, the system sets this field automatically. This field is set only if the possible Billing Approver of this time sheet is a Security Group. Otherwise it is null.	No	N/A
billingApproverUsersIds	List	Read-only field. List of Long object values, each representing a User ID. The system reevaluates possible approvers when you save and load the time sheet. At the same time, the system sets this field automatically. This field must be set on re-evaluation of approvers using this order of precedence: • If billingApproverParticipantGroupId is not null, this field is set to the IDs of all the users in this participant group. • Else, if billingApproverSecurityGroupId is not null, this field is set to the IDs of all the users in this security group. • Else, this field is set to the ID of the default billing approver as determined by the re-evaluation of the approvers.	No	N/A

Related Information

TimeSheetBean is used as the INPUT in the following operations:

- createTimeSheet

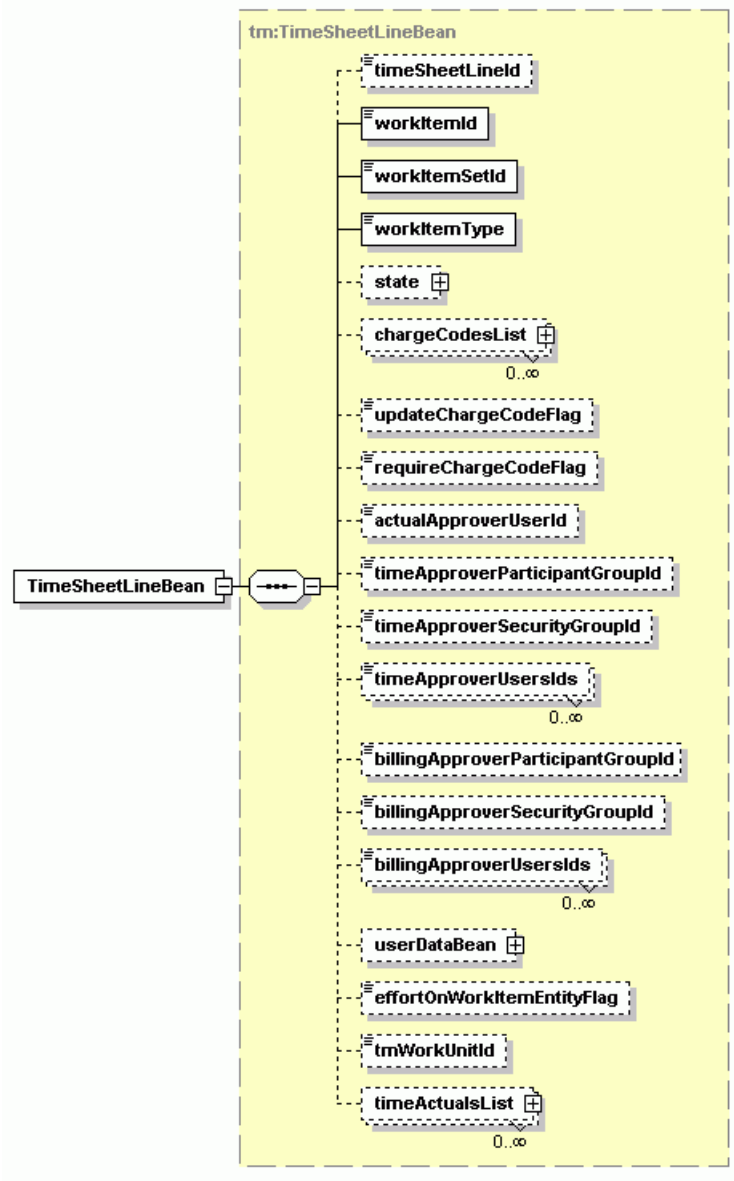
- updateTimeSheet
- approveTimeSheet
- approveTimeSheetLine
- rejectTimeSheet
- rejectTimeSheetLine
- reworkTimeSheetLine
- freezeTimeSheet
- closeTimeSheet
- cancelTimeSheet

TimeSheetBean is used as the RETURN type in the following operations:

- createTimeSheet
- updateTimeSheet
- approveTimeSheet
- approveTimeSheetLine
- rejectTimeSheet
- rejectTimeSheetLine
- reworkTimeSheetLine
- freezeTimeSheet
- closeTimeSheet
- cancelTimeSheet

TimeSheetLineBean

TimeSheetLineBean represents the time sheet line in a time sheet. It is used in TimeSheetBean.


tm:TimeSheetLineBean
timeSheetLineId
workItemId
workItemSetId
workItemType
state
chargeCodesList
0..∞
updateChargeCodeFlag
requireChargeCodeFlag
actualApproverUserId
timeApproverParticipantGroupId
timeApproverSecurityGroupId
timeApproverUsersIds
0..∞
billingApproverParticipantGroupId
billingApproverSecurityGroupId
billingApproverUsersIds
0..∞
userDataBean
effortOnWorkItemEntityFlag
tmWorkUnitId
timeActualsList
0..∞
TimeSheetLineBean

Property	Type	Description	Required	Default
timeSheetLineId	Long	Generated automatically by the system when a time sheet line object in PPM Center is created.	Yes if the timesheet line has already been existed	N/A
workItemId	String	String object which holds the ID of the work item of the line.	Yes	N/A
workItemSetId	String	String object which holds the ID of the parent of the work item of the line. The parent of a request should be the request type. The parent of a task should be the project. The parent of a project should be the same project. The parent of a package should be the workflow. The parent of a misc should be the constant MISC.	Yes	N/A
workItemType	String	String object which holds the type of the work item of the line. Valid values: • REQUEST • TASK • PROJECT • PACKAGE • MISC	Yes	N/A
State	TimeSheetLineStatus	This field is set by default to be “unsubmitted” during the creation of a line. See *TimeSheetLineStatus* for more details.	No	unsubmitted

Property	Type	Description	Required	Default
timeActualsList	List	List of TimeActualsBean objects. See *TimeActualsBean* for more details. For each Line there must be at least two records of Time Actuals list. The two records include same actuals with different total flag Y and N. If there are more than one non-total records (records with total flag as N), each non-total record must be associated with an Activity ID. If there is only one non-total record, it may or may not be associated with an Activity ID.	No	N/A
chargeCodesList	List	List of ChargeCodeBean objects. See *ChargeCodeBean* for more details. The sum of all percentages in charge code list must be 100, otherwise an exception (ex.chargeCodePercentages MustSum100) is thrown.	No	N/A
updateChargeCodeFlag	Boolean	Boolean value. True if charge codes can be updated in the time sheet, False otherwise.	No	True
requireChargeCodeFlag	Boolean	Boolean value. True if charge codes are required in the time sheet, False otherwise.	No	False

Property	Type	Description	Required	Default
userDataBean	UserDataBean	UserDataBean object. See *UserDataBean* for more details. Holds an array of up to 20 user data hidden values and a corresponding array of up to 20 user data visible values.	No	N/A
actualApproverUserId	Long	Long value of the user ID of the user who approves the actual time logged in the line. It is required when the state is "approved."	Yes If line is approved	N/A
effortOnWorkItemEntityFlag	Boolean	Boolean value. True if effort must be synched to HP Project Management or HP Demand Management when you save the effort, False otherwise.	No	False
tmWorkUnitId	Long	Long value of the ID of the TM work unit, which is a record in the staging table that helps in synching the effort between HP Time Management and HP Project Management or HP Demand Management.	After first save	N/A
timeApproverParticipantGroupId	Long	Read-only field. The system reevaluates possible approvers when you save and load the time sheet. At the same time, the system sets this field automatically. This field is set only if the possible Time Approver of the line is a Participant Group (such as Project Managers). Otherwise it is null.	No	N/A

Property	Type	Description	Required	Default
timeApproverSecurityGroupId	Long	Read-only field. The system reevaluates possible approvers when you save and load the time sheet. At the same time, the system sets this field automatically. This field is set only if the possible Time Approver of the line is a Security Group. Otherwise it is null.	No	N/A
timeApproverUsersIds	List	Read-only field. List of Long object values, each representing a User ID. The system reevaluates possible approvers when you save and load the time sheet. At the same time, the system sets this field automatically. This field must be set on re-evaluation of approvers using this order of precedence: • If timeApproverParticipantGroupId is not null, this field is set to the IDs of all the users in this participant group. • Else, if timeApproverSecurityGroupId is not null, this field is set to the IDs of all the users in this security group. • Else, this field is set to the ID of the time approver of the line as determined by the re-evaluation of the approvers.	No	N/A

Property	Type	Description	Required	Default
billingApproverParticipantGroupId	Long	Read-only field. The system reevaluates possible approvers when you save and load the time sheet. At the same time, the system sets this field automatically. This field is set only if the possible Billing Approver of the time sheet of this line is a Participant Group (such as Project Managers). Otherwise it is null.	No	N/A
billingApproverSecurityGroupId	Long	Read-only field. The system reevaluates possible approvers when you save and load the time sheet. At the same time, the system sets this field automatically. This field is set only if the possible Billing Approver of the time sheet of this line is a Security Group. Otherwise it is null.	No	N/A

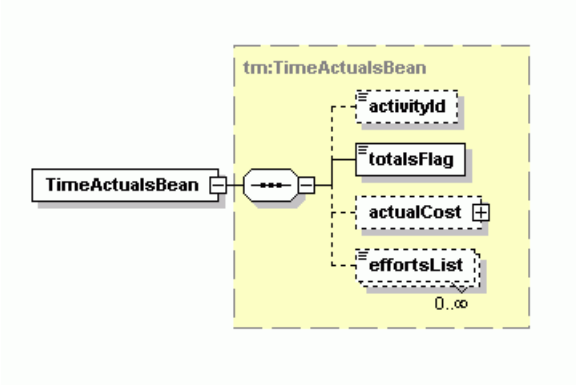
Property	Type	Description	Required	Default
billingApproverUsersIds	List	Read-only field. List of Long object values, each representing User ID. The system reevaluates possible approvers when you save and load the time sheet. At the same time, the system sets this field automatically. This field must be set on re-evaluation of approvers using this order of precedence: • If billingApproverParticipantGroupId is not null, this field is set to the IDs of all the users in this participant group. • Else, if billingApproverSecurityGroupId is not null, this field is set to the IDs of all the users in this security group. • Else, this field is set to the ID of the time approver of the line as determined by the re-evaluation of the approvers.	No	N/A

Related Information

TimeSheetLineBean is used in *TimeSheetBean*.

TimeActualsBean

TimeActualsBean is used to hold the actual effort for each time sheet line. It is used in TimeSheetLineBean.



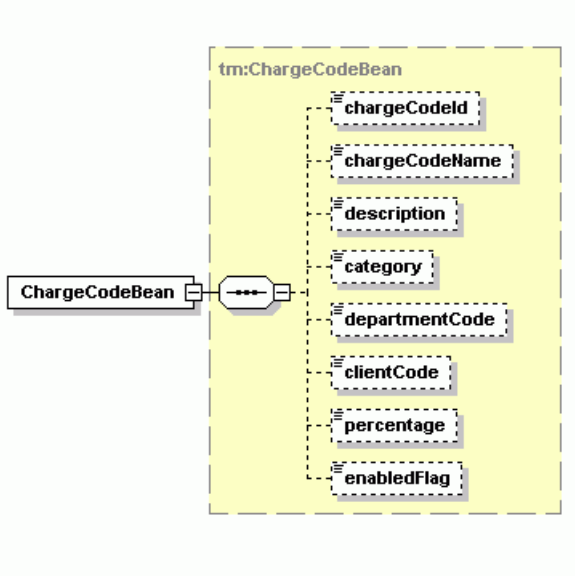
Property	Type	Description	Required	Default
activityId	Long	ID of the activity in PPM Center.	No	N/A
effortsList	List	List of objects with the Double type in the size of the time period of the time sheet. Each element holds the effort in hours for the corresponding day by order. The effort is always in hours irrespective of the value you set in bucketShowLevel and bucketReportMethod.	No	All zeros
totalsFlag	Boolean	Boolean value. True if this record is the totals record of the line, False otherwise.	Yes	N/A
actualCost	TaskCost Bean	Object of *CostBean*. See Web Services for HP Project Management.	No	N/A

Related Information

TimeActualsBean is used in *TimeSheetLineBean*.

ChargeCodeBean

ChargeCodeBean is used to hold the charge code and the percentage in a time sheet line. It is used in TimeSheetLineBean.



Property	Type	Description	Required	Default
chargeCodeId	Long	ID of the charge code.	Yes	N/A
chargeCodeName	String	String value of the charge code name.	No	N/A
Description	String	String value of the charge code description.	No	N/A
enabledFlag	Boolean	Boolean value. True if this charge code is enabled, False otherwise.	Yes	N/A
category	String	String value of the charge code category.	No	N/A
departmentCode	String	String value of the charge code department code.	No	N/A
clientCode	String	String value of the charge code client code.	No	N/A

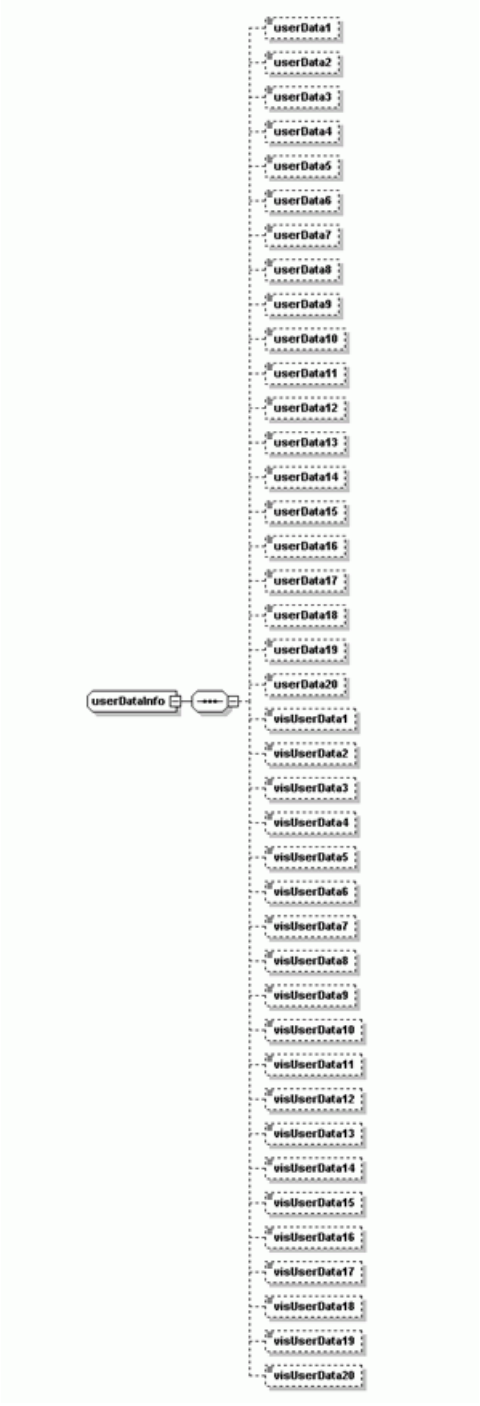
Property	Type	Description	Required	Default
Percentage	Double	Double value of the percentage of this charge code with the range of 0 to 100. The sum of all percentages in the Charge Codes list must be 100.	Yes	N/A

Related Information

ChargeCodeBean is used in *TimeSheetLineBean*.

UserDataBean

UserDataBean holds the additional fields in time sheet lines. It is used in TimeSheetLineBean.


userDataInfo
userData1
userData2
userData3
userData4
userData5
userData6
userData7
userData8
userData9
userData10
userData11
userData12
userData13
userData14
userData15
userData16
userData17
userData18
userData19
userData20
visUserData1
visUserData2
visUserData3
visUserData4
visUserData5
visUserData6
visUserData7
visUserData8
visUserData9
visUserData10
visUserData11
visUserData12
visUserData13
visUserData14
visUserData15
visUserData16
visUserData17
visUserData18
visUserData19
visUserData20

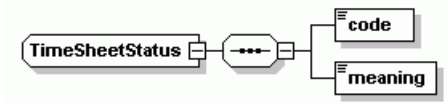
Property	Type	Description	Required	Default
userData	String[]	Array of 20 String values of the hidden value of user data.	No	N/A
visibleUserData	String[]	Array of 20 String values of the visible value of user data. Each corresponds to the hidden value in the userData array, by order.	No	N/A

Related Information

UserDataBean is used in *TimeSheetLineBean*.

TimeSheetStatus

TimeSheetStatus is used to hold the status for each time sheet. It is used in TimeSheetBean. The code property value and of the meaning property value must match with each other.



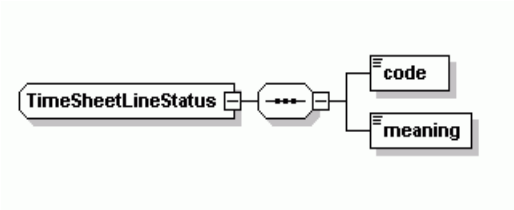
Property	Type	Description	Required	Default
code	int	Numeric value. Valid numeric values are (the corresponding meanings are listed in the brackets): • 1 (for unsubmitted) • 2 (for pending-approval) • 3 (for in-rework) • 4 (for approved) • 5 (for cancelled) • 6 (for frozen) • 7 (for closed)	Yes	N/A
meaning	String	String value. Valid values are: • unsubmitted • pending-approval • in-rework • approved • cancelled • frozen • closed	Yes	N/A

Related Information

TimeSheetStatus is used in *TimeSheetBean*.

TimeSheetLineStatus

TimeSheetLineStatus is used to hold the status for a time sheet line. It is used in TimeSheetLineBean. The code property value and of the meaning property value must match with each other.



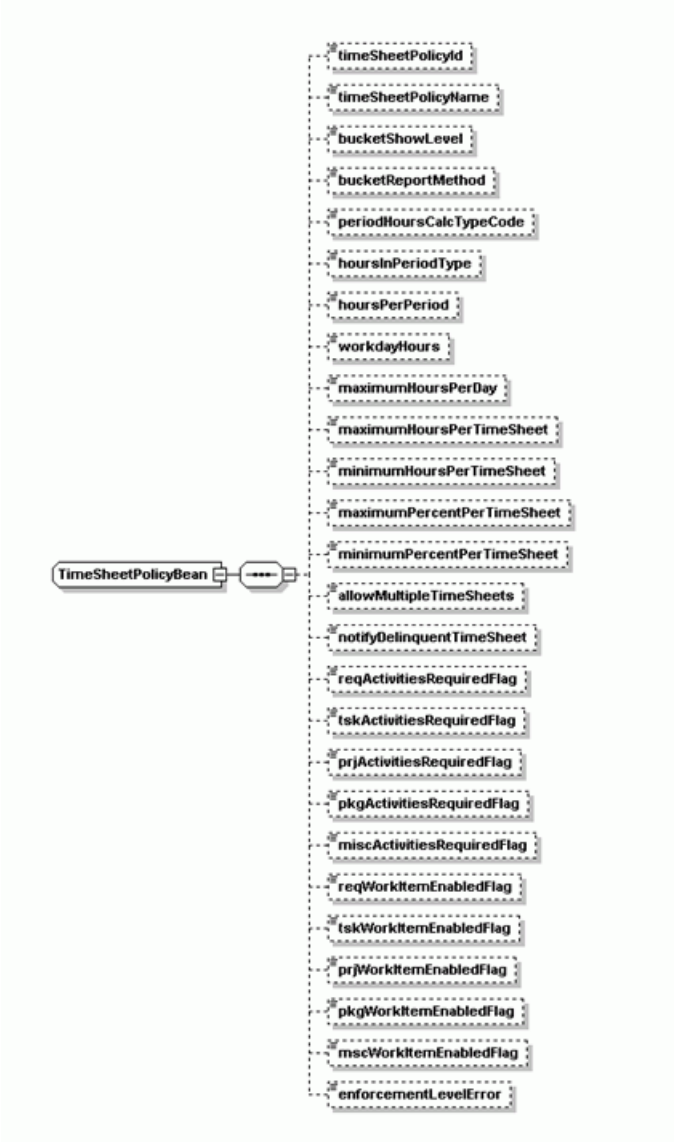
Property	Type	Description	Required	Default
code	int	Numeric value. Valid numeric values are (the corresponding meanings are listed in the brackets): • 1 (for unsubmitted) • 2 (for pending-approval) • 3 (for in-rework) • 4 (for approved) • 5 (for cancelled) • 6 (for frozen) • 7 (for closed)	Yes	N/A
meaning	String	String value. Valid values are: • unsubmitted • pending-approval • in-rework • approved • cancelled • frozen • closed	Yes	N/A

Related Information

TimeSheetLineStatus is used in *TimeSheetLineBean*.

TimeSheetPolicyBean

TimeSheetPolicyBean is used to hold the information for an existing time sheet policy.



Property	Type	Description	Required	Default
timeSheetPolicyId	Long	ID of the time sheet policy.	Yes	N/A
timeSheetPolicyName	String	String value of the time sheet policy name.	Yes	N/A
bucketShowLevel	String	Unit of measurement for each time bucket. Valid values: • PERIOD • DAY • HOURS	Yes	N/A
bucketReportMethod	String	Defines how time sheets report time. Valid values: • HOURS • PERCENT	Yes	N/A
hoursPerPeriod	Double	Numeric value that defines how many hours a single period contains. This field is relevant only when all of the following conditions are true: • The value for bucketShowLevel is PERIOD • The value for bucketReportMethod is PERCENT • The value for periodHoursCalcTypeCode is FIXED	No	N/A

Property	Type	Description	Required	Default
periodHoursCalcTypeCode	String	Defines how period in hours should be calculated. Valid values: • FIXED (for period) • WORKDAY (for days). This field is relevant only when both of the following conditions are true: • The value for bucketShowLevel is PERIOD • The value for bucketReportMethod is PERCENT	No	N/A
workdayHours	Double	Numeric value of how many hours are in a workday. Relevant when time is measured by day or hours. This field is relevant only when all of the following conditions are true: • The value for bucketShowLevel is PERIOD. • The value for bucketReportMethod is PERCENT. • The value for periodHoursCalcTypeCode is WORKDAY.	No	N/A
allowMultipleTimeSheets	Boolean	Boolean value - True if the policy allows multiple time sheets, False otherwise.	Yes	False

Property	Type	Description	Required	Default
hoursInPeriodType	String	Defines how period in hours should be calculated. Valid values: • FIXED (for period) • WORKDAY (for hours)	No	N/A
notifyDelinquentTimeSheet	Boolean	Boolean value - True if you need the notification of delinquent time, False otherwise.	Yes	False
maximumHoursPerDay	Double	Numeric value of the max hours per day to enforce. Relevant when time is measured by day. This field is relevant only when both of the following conditions are true: • The value for bucketShowLevel is Day. • The value for bucketReportMethod is HOURS.	No	N/A
maximumHoursPerTimeSheet	Double	Numeric value of the max hours per time sheet to enforce. This field is relevant only when the value for bucketReportMethod is HOURS.	No	N/A
minimumHoursPerTimeSheet	Double	Numeric value of the min hours per time sheet to enforce. This field is relevant only when the value for bucketReportMethod is HOURS.	No	N/A

Property	Type	Description	Required	Default
maximumPercentPerTimeSheet	Double	Numeric value of the max percent per time sheet to enforce. This field is relevant only when both of the following conditions are true: • The value for bucketShowLevel is PERIOD. • The value for bucketReportMethod is PERCENT.	No	N/A
minimumPercentPerTimeSheet	Double	Numeric value of the min percent per time sheet to enforce. This field is relevant only when both of the following conditions are true: • The value for bucketShowLevel is PERIOD. • The value for bucketReportMethod is PERCENT.	No	N/A
enforcementLevelError	Boolean	Boolean value - Set this field to True if you want the system to prevent the submission of a time sheet when the submission violates the policy enforcements. Set it to False if you want the system to display only a warning when a violation occurs.	Yes	False
reqActivitiesRequiredFlag	Boolean	Boolean value - True if activities are required for Request work items, False otherwise.	Yes	False

Property	Type	Description	Required	Default
tskActivitiesRequiredFlag	Boolean	Boolean value - True if activities are required for Task work items, False otherwise.	Yes	False
prjActivitiesRequiredFlag	Boolean	Boolean value - True if activities are required for Project work items, False otherwise.	Yes	False
pkgActivitiesRequiredFlag	Boolean	Boolean value - True if activities are required for Package work items, False otherwise.	Yes	False
miscActivitiesRequiredFlag	Boolean	Boolean value - True if activities are required for Misc work items, False otherwise.	Yes	False
reqWorkItemEnabledFlag	Boolean	Boolean value - True if Request work items are enabled for this time sheet, False otherwise. The default value is taken from server.conf. com.kintana.core.server.PURGE_SERVER_PROPERTIES_TABLE=true com.kintana.core.server.ENABLE_TM_WORK_ITEM_REQUESTS=true	Yes	Taken from server.conf

Property	Type	Description	Required	Default
tskWorkItemEnabledFlag	Boolean	Boolean value - True if Task work items are enabled for this time sheet, False otherwise The default value is taken from server.conf. com.kintana.core.server.PURGE_SERVER_PROPERTIES_TABLE=true com.kintana.core.server.ENABLE_TM_WORK_ITEM_TASKS=true	Yes	Taken from server.conf
prjWorkItemEnabledFlag	Boolean	Boolean value - True if Project work items are enabled for this time sheet, False otherwise. The default value is taken from server.conf. com.kintana.core.server.PURGE_SERVER_PROPERTIES_TABLE=true com.kintana.core.server.ENABLE_TM_WORK_ITEM_PROJECTS=true	Yes	Taken from server.conf
pkgWorkItemEnabledFlag	Boolean	Boolean value - True if Package work items are enabled for this time sheet, False otherwise. The default value is taken from server.conf. com.kintana.core.server.PURGE_SERVER_PROPERTIES_TABLE=true com.kintana.core.server.ENABLE_TM_WORK_ITEM_PACKAGES =true	Yes	Taken from server.conf

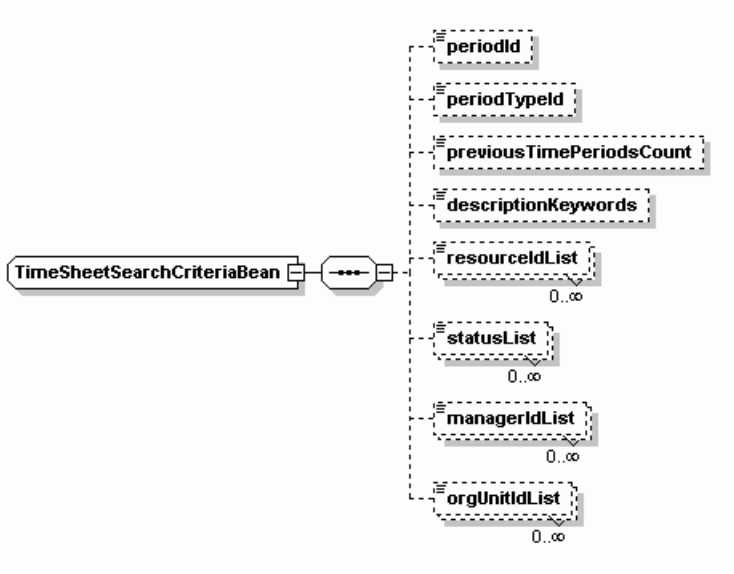
Property	Type	Description	Required	Default
mscWorkItemEnabledFlag	Boolean	Boolean value - True if Misc work items are enabled for this time sheet, False otherwise. The default value is taken from server.conf. com.kintana.core.server.PURGE_SERVER_PROPERTIES_TABLE=true com.kintana.core.server.ENABLE_TM_WORK_ITEM_MISC =true	Yes	Taken from server.conf

Related Information

TimeSheetPolicyBean is used as the RETURN type in *getTimeSheetPolicy*.

TimeSheetSearchCriteriaBean

TimeSheetSearchCriteriaBean holds the parameters which are used for searching time sheets.



Property	Type	Description	Required	Default
periodTypeId	Long	Numeric value of a Time Period Type ID.	No	N/A
periodId	Long	Numeric value of a Time Period ID. Must correspond to the period type ID, if the period type ID exists.	No	N/A
previousTimePeriods Count	Long	Numeric value that defines how many previous time periods to include in the results.	No	0
resourceIdList	List	List of Long values of Resource IDs.	No	N/A
descriptionKeywords	String	String value of keywords to search from time sheets descriptions.	No	N/A

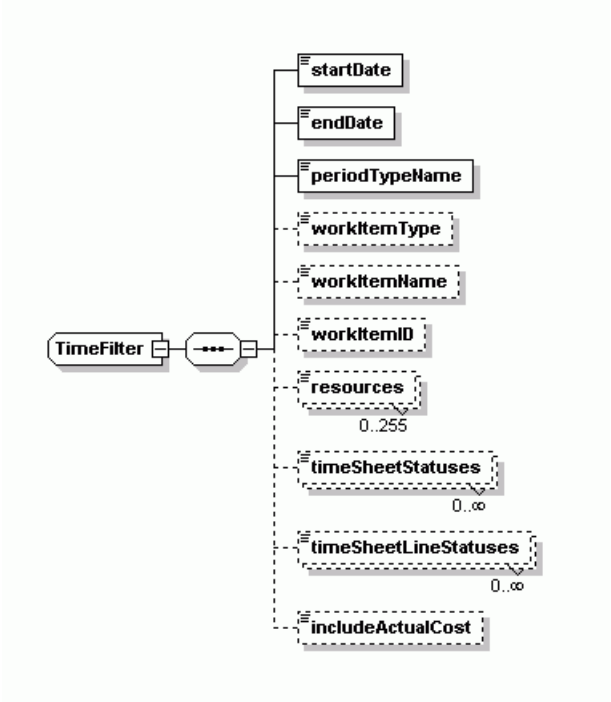
Property	Type	Description	Required	Default
statusList	List	List of Long values of statuses. Must correspond to the statuses constants in the code field of *TimeSheetStatus*.	No	N/A
managerIdList	List	List of Long values of User IDs of resource managers.	No	N/A
orgUnitIdList	List	List of Long values of Org Unit IDs.	No	N/A

Related Information

TimeSheetSearchCriteriaBean is used as the INPUT type in *searchTimeSheets*.

TimeFilter

TimeFilter is holds the parameters which are used to get the work item actuals.



Property	Type	Description	Required	Default
startDate	Date	Date value of the start of the work item effort.	Yes	N/A
endDate	Date	Date value of the end of the work item effort.	Yes	N/A
periodTypeName	String	String value of the time period type name. Valid values: • Weekly • Bi-Weekly • Semi-Monthly • Monthly	Yes	N/A

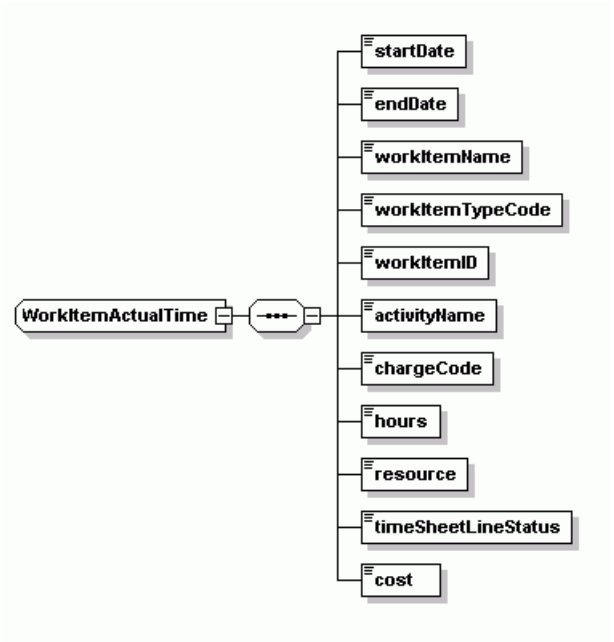
Property	Type	Description	Required	Default
workItemType	String	String value of the work item type. Valid values: • REQUEST • TASK • PROJECT • PACKAGE • MISC	No	N/A
workItemName	String	String value of the work item name.	No	N/A
workItemID	String	String value of the work item ID.	No	N/A
resources	List	List of resources usernames.	No	N/A
timeSheetStatuses	List	List of time sheet statuses. Constant must correspond to the statuses constants in meaning field of *TimeSheetStatus*.	No	N/A
timeSheetLineStatuses	List	List of time sheet line statuses Constant must correspond to the statuses constants in meaning field of *TimeSheetLineStatus*.	No	N/A
includeActualCost	boolean	True if you want to get the actual cost information in the results, False otherwise.	No	N/A

Related Information

TimeFilter is used as the INPUT type in *getActualTime*.

WorkItemActualTime

It is used to hold the results of the getActualTime operation.



Property	Type	Description	Required	Default
startDate	Date	Date value of the start of the work item effort.	No	N/A
endDate	Date	Date value of the end of the work item effort.	No	N/A
workItemName	String	String value of the work item name.	No	N/A
workItemTypeCode	String	String value of work item type code. Valid values: • REQUEST • TASK • PROJECT • PACKAGE • MISC	No	N/A
workItemID	String	String value of the work item ID.	No	N/A

Property	Type	Description	Required	Default
activityName	String	String value of an activity name.	No	N/A
chargeCode	String	String value of a charge code name.	No	N/A
Hours	String	String value of amount of hours for this work item.	No	N/A
Resource	String	String value of the resource username.	No	N/A
timeSheetLineStatus	String	String value of the statuses of the time sheet lines. Must correspond to the constants in code field in *TimeSheetLineStatus*.	No	N/A
Cost	String	String value of the cost of the effort.	No	N/A

Related Information

TimeFilter is used as the RETURN type in *getActualTime*.

Operations

The following operations are included in HP Time Management Web services:

- *createTimeSheet* on page 298
- *updateTimeSheet* on page 312
- *getTimeSheet* on page 321
- *searchTimeSheets* on page 323
- *getActualTime* on page 325

createTimeSheet

Purpose

Create a new time sheet, including time sheet lines in PPM Center.

HP Time Management Web services was not built for the purpose of migrating large volume of time sheets data from a legacy system into PPM Center. To migrate large volume of time sheets, use TM Bulk Importer.

Creating a time sheet or any other operation is expected to take just slightly less than the same operation in the PPM application. Migration of many time sheets takes too much time with the current web services.

Function

This operation creates a new time sheet in PPM Center.

The user performing this operation must have the Time Mgmt: Edit Time Sheets access grant.

The operation can do the following:

- Create a time sheet.
- Attach time sheet policy to this time sheet.
- Create time sheet lines in the time sheet.

The system creates audit information throughout the whole create and update process.

Once the new time sheet is created, the system performs the following functionalities:

- The status of the time sheet is reevaluated according to the statuses of its Lines.
- If the status of the time sheet is “unsubmitted,” “pending-approval” or “in-rework,” then the possible time and billing approvers are reevaluated.

  This process fills the following fields:

  - timeApproverParticipantGroupId
  - timeApproverSecurityGroupId
  - timeApproverUsersIds
  - billingApproverParticipantGroupId
  - billingApproverSecurityGroupId
  - billingApproverUsersIds
- The actual effort set to Task or Request work items is synched with those Tasks or Requests in HP Project Management or HP Demand Management accordingly.

The following field is set only by the functionality of this operation, and cannot be set by the user:

- Sequence: The sequence number set in the new time sheet would be the next sequence number available, meaning the system automatically increments the number of time sheets that exist for this resource and period by 1.

If the policy does not allow multiple time sheets per resource and period, and a time sheet has already existed for this resource and period, an exception (ex.timesheetExist) is thrown.

Input

TimeSheetBean

The following fields are must be set to complete this operation:

- resourceId: Existing Resource ID in PPM Center for whom you log the time sheet.
- periodId: Existing Time Period ID in PPM Center with which you log the time period.

Settings for the other fields depend on the functionality you want to achieve. For details, refer to the following:

- *Create a Time Sheet*
- *Attach a Time Sheet Policy*
- *Create a Time Sheet Line*

Create a Time Sheet

To create a time sheet, set the following fields.

- description: If the description filed is not set, a default description is given, composed of Resource name - Time Period name and #Sequence.
- state: If no time sheet status is set, the default time sheet status is “unsubmitted.”

Attach a Time Sheet Policy

To attach a time sheet policy, set the following fields.

- Policy fields

  (bucketShowLevel, bucketReportMethod, periodHoursCalcTypeCode, hoursInPeriodType, hoursPerPeriod, workdayHours, maximumHoursPerDay, maximumHoursPerTimeSheet,

minimumHoursPerTimeSheet, maximumPercentPerTimeSheet, minimumPercentPerTimeSheet, allowMultipleTimeSheets, notifyDelinquentTimeSheet, reqActivitiesRequiredFlag, tskActivitiesRequiredFlag, prjActivitiesRequiredFlag, pkgActivitiesRequiredFlag, miscActivitiesRequiredFlag, reqWorkItemEnabledFlag, tskWorkItemEnabledFlag, prjWorkItemEnabledFlag, pkgWorkItemEnabledFlag, mscWorkItemEnabledFlag, enforcementLevelError)

The BucketShowLevel field is used to determine if policy fields are set. If BucketShowLevel field is not set, the system considers you specify no policy fields.

- If no policy fields are specified, the policy set for this time sheet is the policy associated with the resource. If no policy is associated with the resource, the derived policy is based on the global policy.
- If policy fields are specified, then those fields are set as the policy fields in the time sheet.

In both cases, the Time Period Type ID of the policy must match the Period Type ID associated with the Time Period ID which is a required parameter in this operation. All of the fields must be set in order to avoid confusion with the default policy fields that would be set on initial creation.

Create a Time Sheet Line

To create a time sheet line, set the following fields:

- timeSheetLines:
- Each new time sheet line must have the following required fields set:
  - workItemId
  - workItemSetId

For the other TimeSheetLineBean fields regarding a time sheet line, refer to the following.

To set time sheet line Status, set the following field.

- state: If no status is set for the line when it is created, it is set with the default “unsubmitted.”

It is recommended that you update the time sheet line status by using the *approveTimeSheetLine*, *rejectTimeSheetLine*, and *reworkTimeSheetLine* operations.

To add Charge Codes in a time sheet line, set the following field:

- chargeCodesList: Each charge code parameter must have an ID of an existing Charge Code in PPM Center, and a percentage attached to it.

The percentages of the charge codes in the charge code list should sum up to 100.

To input actual effort in the time sheet line, set the following field.

- timeActualsList

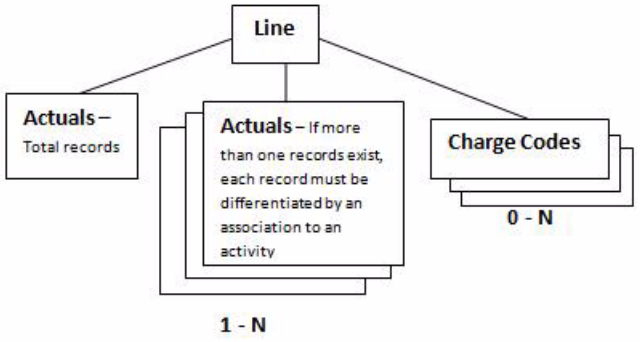
  For each line, there must be at least two records of Time Actuals: one record must be the totals record while the others are the non-total records.

Non-total records:

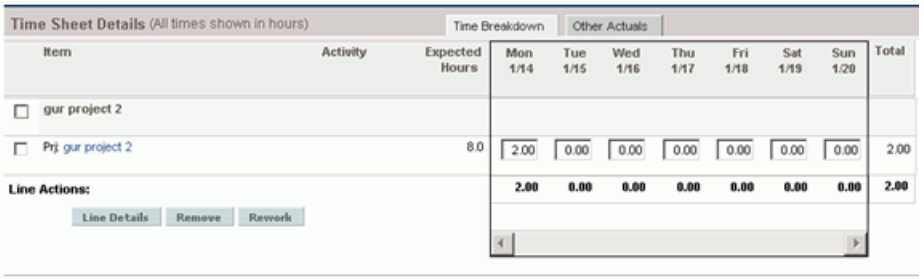
There can be one or more non-total records. If there is one non-total record, it may or may not be associated with an activity. If there are more than one non-total records, each record must be associated with an Activity ID.

When data is added to each non-total record, the totals record is automatically recalculated on the update of the time sheet object.

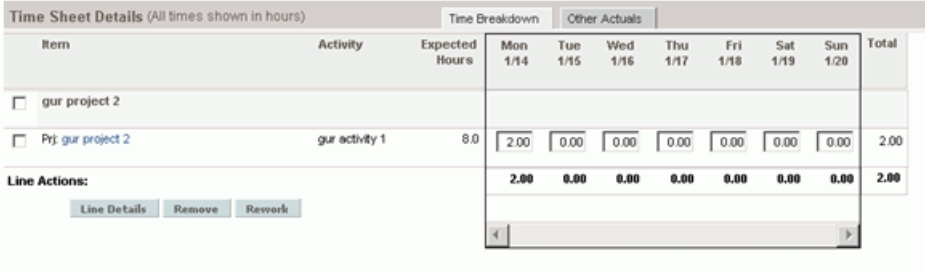
These relationships are shown in the following diagrams.



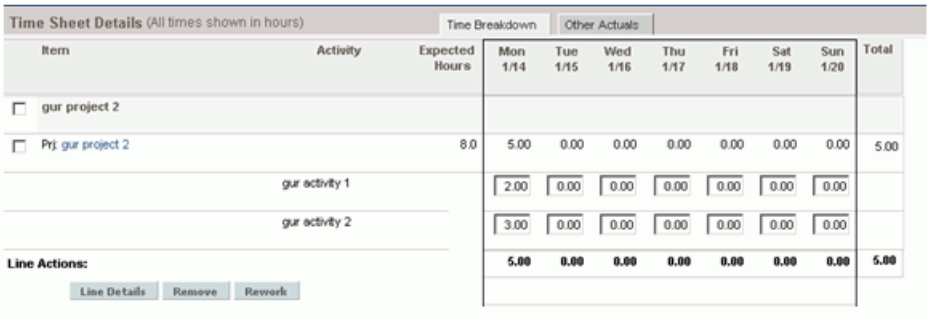
A line with one non-total actuals record, which is not associated with an activity:



A line with one non-total actuals record, which is associated with one activity:



A line with several non-total actuals records, where each is associated with one activity:



The following are additional fields that can be set for a time sheet line:

- actualApproverUserId: Actual Approver User ID (If the status for the line is “approved” or higher).
- updateChargeCodeFlag: Flag that defines whether charge codes can be updateable (Y/N).
- requireChargeCodeFlag: Flag that defines whether charge codes are required (Y/N).
- userDataBean: An array of up to 20 user data hidden values and a corresponding array of up to 20 user data visible values.

Return

An object of TimeSheetBean with updated fields.

Limitations

- All data, specified in the properties, must comply with what is expected in PPM Center - no data validation is performed by this Web service operation.
- Notes are not implemented in this Web service operation.

Related Information

updateTimeSheet updates fields of an existing time sheet.

Java Interface

```
CreateTimeSheetResponseDocument
createTimeSheet(CreateTimeSheetDocument in)
```

Parameters	Description

CreateTimeSheetDocument	Wrapper for TimeSheetBean. See the following example for the construction.The bean includes:
	• Long timeSheetId;
	• Long resourceId;
	• Long periodId;
	• String description;
	• Integer sequence;
	• TimeSheetStatus state; // constants correspond to those in model.TimeSheet
	• List timeSheetLines; // list of TimeSheetLineBean objects
	• String bucketShowLevel[a];
	• String bucketReportMethod;[a]
	• String periodHoursCalcTypeCode;[a]
	• Double hoursPerPeriod;[a]
	• Double workdayHours;[a]
	• Boolean allowMultipleTimeSheets;[a]
	• String hoursInPeriodType;[a]
	• Double maximumHoursPerDay;[a]
	• Double maximumHoursPerTimeSheet;[a]
	• Double minimumHoursPerTimeSheet;[a]
	• Double maximumPercentPerTimeSheet;[a]
	• Double minimumPercentPerTimeSheet;[a]
	• Boolean notifyDelinquentTimeSheet;[a]
	• Boolean reqActivitiesRequiredFlag;[a]
	• Boolean tskActivitiesRequiredFlag;[a]
	• Boolean prjActivitiesRequiredFlag;[a]
	• Boolean pkgActivitiesRequiredFlag;[a]
	• Boolean miscActivitiesRequiredFlag;[a]
	• Boolean reqWorkItemEnabledFlag;[a]
	• Boolean tskWorkItemEnabledFlag;[a]
	• Boolean prjWorkItemEnabledFlag;[a]
	• Boolean pkgWorkItemEnabledFlag;[a]
	• Boolean mscWorkItemEnabledFlag;[a]
	• Boolean enforcementLevelError;[a]
	• String exceptionMessage;[b]
	• Long timeApproverParticipantGroupId;[b]
	• Long timeApproverSecurityGroupId;[b]
	• List timeApproverUsersIds;[b] // list of Long user ids
	• Long billingApproverParticipantGroupId;[b]
	• Long billingApproverSecurityGroupId;[b]
	• List billingApproverUsersIds; [b]// list of Long user ids

Parameters	Description
CreateTimeSheetResponseDocument	Wrapper for TimeSheetBean. See the following example for retrieving the bean and its fields.

a. Policy fields - for creation only

b. Read-only fields

Java Examples

Example: create a new time sheet.

```
/**
 *
 * We are assuming existence of basic data that must exist in
the app. Meaning:
 *   - admin resource with user id 1
 *   - time periods starting with period id 30000, at least one
 *   - base time policy named 'Semi-Monthly - Day - Hours'
 *
 */

    TimeSheetBean createdTimeSheetBean = null;
    public void testCreateTimeSheet() {
        System.out.println("testCreateTimeSheet started ...");
        try {
            TimeServiceStub stub = new TimeServiceStub(ctx,
WSURL);
            CreateTimeSheetDocument createTimeSheetDocument =
CreateTimeSheetDocument.Factory.newInstance();
       final TimeSheetBean newTimeSheetBean =
createTimeSheetDocument.addNewCreateTimeSheet().addNewTimeSheet
Bean();

          newTimeSheetBean.setResourceId(1); // assuming admin
user exists with id 1
            newTimeSheetBean.setPeriodId(30000);  // assuming
this time period exists

            newTimeSheetBean.setBucketShowLevel("DAY");
            newTimeSheetBean.setBucketReportMethod("HOURS");

newTimeSheetBean.setPeriodHoursCalcTypeCode("WORKDAY");
            newTimeSheetBean.setHoursPerPeriod(8);
            newTimeSheetBean.setWorkdayHours(5.5);
           newTimeSheetBean.setReqActivitiesRequiredFlag(true);
           newTimeSheetBean.setTskActivitiesRequiredFlag(true);
           newTimeSheetBean.setPkgActivitiesRequiredFlag(true);

newTimeSheetBean.setMiscActivitiesRequiredFlag(true);
            newTimeSheetBean.setAllowMultipleTimeSheets(true);
```

```
            newTimeSheetBean.setEnforcementLevelError(true);

            TimeSheetLineBean timeSheetLineBean =
newTimeSheetBean.addNewTimeSheetLines();
            timeSheetLineBean.setWorkItemId("2");
            timeSheetLineBean.setWorkItemSetId("1");
            timeSheetLineBean.setWorkItemType("MISC");
            TimeSheetLineStatus timeSheetLineStatus =
timeSheetLineBean.addNewState();
            timeSheetLineStatus.setCode(new BigInteger("1"));
            timeSheetLineStatus.setMeaning("unsubmitted");

            ChargeCodeBean chargeCodeBean =
timeSheetLineBean.addNewChargeCodesList();
            chargeCodeBean.setChargeCodeId(30000);
            chargeCodeBean.setChargeCodeName("my cc name");
            chargeCodeBean.setCategory("BILLABLE");
            chargeCodeBean.setClientCode("CLIENT_1");
            chargeCodeBean.setDepartmentCode("FINANCE");
            chargeCodeBean.setDescription("my cc name");
            chargeCodeBean.setEnabledFlag(true);
            chargeCodeBean.setPercentage(100.d);

            timeSheetLineBean.setUpdateChargeCodeFlag(true);
            timeSheetLineBean.setRequireChargeCodeFlag(false);
            timeSheetLineBean.setActualApproverUserId(1);
            UserDataInfo ud =
timeSheetLineBean.addNewUserDataBean();
            ud.setUserData1("UD1");
            ud.setVisUserData1("VisUD1");

            TimeActualsBean timeActualsBean1 =
timeSheetLineBean.addNewTimeActualsList();
            timeActualsBean1.setTotalsFlag(true);
            timeActualsBean1.setEffortsListArray(new double[]
{2,4,6,8,10,0,0,2,4,6,8,10,0,0,20});

            TimeActualsBean timeActualsBean2 =
timeSheetLineBean.addNewTimeActualsList();
            timeActualsBean2.setTotalsFlag(false);
            timeActualsBean2.setEffortsListArray(new double[]
{1,2,3,4,5,0,0,1,2,3,4,5,0,0,10});

            TimeActualsBean timeActualsBean3 =
timeSheetLineBean.addNewTimeActualsList();
            timeActualsBean3.setTotalsFlag(false);
            timeActualsBean2.setActivityId(30000);
            timeActualsBean3.setEffortsListArray(new double[]
{1,2,3,4,5,0,0,1,2,3,4,5,0,0,10});

            CreateTimeSheetResponseDocument response =
stub.createTimeSheet(createTimeSheetDocument);
            createdTimeSheetBean =
response.getCreateTimeSheetResponse().getReturn();
```

```
            System.out.println("Newly created TimeSheet with id
" + createdTimeSheetBean.getTimeSheetId());
        }
        catch(Exception e) {
            e.printStackTrace();
        }
        System.out.println("testCreateTimeSheet completed");
    }
```

Errors and Exceptions

When an error occurs during this operation, a description of the root cause will be logged and a response message generated.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault: [root
cause description]
```

Response message:

```
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
ex.cannotEditTimesheet	User cannot edit Timesheet	You may not have the required access grant or Time Management License.	Check your licenses and privileges (access grants).
ex.resourceNotExist	Resource does not exist	Resource ID is invalid or resource does not exist anymore in PPM Center.	Check the validity of the resource ID.
ex.periodNotExist	Time Period does not exist	Time Period ID is invalid or Time Period ID does not exist anymore in the database.	Check the validity of the time period ID.

Message Code	Message	Cause(s)	Possible Corrective Action
ex.globalPolicyNotExist	Global timesheet policy does not exist	Corrupt database.	Check for an existence of the global policy; must exist in PPM Center.
ex.periodTypeNotMatching	Period type in time period does not match period type in timesheet policy	Invalid Time Period data.	Check for the correspondence between the time period and the period type of the policy.
ex.timesheetExist	Timesheet already exists for this resource and period, policy does not allow multiple timesheets	You try to create multiple time sheets when the policy does not allow it.	Verify that either the policy allows multiple time sheets or a time sheet does not already exist for the period and resource for which you are trying to create a new time sheet.
ex.workitemInvalid	Timesheet line could not be created, item added is not valid for this resource	Invalid data in the new time sheet line. It could be caused by a certain field or one of the following these causes: • Work item ID does not exist in the type. • Work item ID does not exist in the work item set ID.	If work item type is 'PROJECT' or 'TASK', verify that he resource has access to the work item ID; or if the work item type is 'REQUEST', verify that the work item set ID is the request type ID.

Message Code	Message	Cause(s)	Possible Corrective Action
ex.workitemProblem	Timesheet line could not be created, problem probably with work item ID or type	Invalid data in the new time sheet line. It could be caused by a certain field or one of these causes: • Work item ID does not exist in the type. • Work item ID does not exist in the work item set ID. • Activity ID does not exist. • Invalid numeric value for the effort. • Count of efforts does not match the number of days in the period type of the time sheet.	Check every field of the work item in the line that failed for its validity.
ex.chargeCodeNotExist	Charge code does not exist	Charge code ID is invalid or charge code does not exist anymore in the database.	Check the validity of the charge code ID.
ex.chargeCodePercentagesMustSum100	Charge codes percentages must sum to 100	For each line, a list of charge codes can be defined. Each charge code must have a percentage. The sum of all the percentages per each line must sum to 100.	Check the sum of all charge codes percentages for each line.

updateTimeSheet

Purpose

Update an existing time sheet in PPM Center.

Function

This operation updates an existing time sheet object, identified by time sheet ID in PPM Center.

The user performing this operation must have the permission to edit this particular time sheet, meaning that the user must meet the following conditions:

- Have the Time Mgmt: Edit Time Sheets access grant.
- Be one of the following:
  - The resource of this time sheet.
  - The delegate of the resource.
  - The manager of the resource.

This operation can do the following:

- Update the description of the time sheet.
- Update the time sheet Status.
- Delete the time sheet line.
- Add the time sheet line.
- Update the time sheet lines.

The following fields cannot be updated through this operation.

- Resource
- Time Period
- Policy fields

- Sequence

The system creates audit information throughout the whole process.

Once the time sheet is updated, the system performs the following functionalities:

- The system reevaluates the status of the time sheet according to the statuses of its lines.
- If the status of the time sheet is “unsubmitted,” “pending-approval,” or “in-rework,” the system reevaluates the possible time and billing approvers.

  This process fills the following fields:

  - timeApproverParticipantGroupId
  - timeApproverSecurityGroupId
  - timeApproverUsersIds
  - billingApproverParticipantGroupId
  - billingApproverSecurityGroupId
  - billingApproverUsersIds
- The actual effort set to Task or Request work items is synchronized with those Tasks or Requests in HP Project Management or HP Demand Management accordingly.

Input

An object of TimeSheetBean that must have the following fields:

- timeSheetID: Existing time sheet ID in PPM Center.

Settings for the other fields depend on the functionality you want to achieve. For details, refer to the following:

- *Update the Description for a Time Sheet*
- *Update the Time Sheet Status*

- *Delete a Time Sheet Line*
- *Add a new Time Sheet Line*

Update the Description for a Time Sheet

To update the description for a time sheet, set the following field.

- description: If the description field is null or blank, the time sheet keeps the original description.

Update the Time Sheet Status

To update the time sheet status, set the following field.

- state: If the Status field is null or blank, the time sheet keeps the original status. Note that the status of a time sheet is set according to the time sheet line status in this time sheet.

It is recommended that you update the time sheet line status by using *submitTimeSheet*, *cancelTimeSheet*, *rejectTimeSheet*, *approveTimeSheet*, *freezeTimeSheet*, and *closeTimeSheet* operations.

Delete a Time Sheet Line

To delete a specific time sheet line, remove the TimeSheetLineBean together with the specific line ID from TimeSheetBean.

Add a new Time Sheet Line

To add a new time sheet line, refer to the *Create a Time Sheet Line* section in operation *createTimeSheet*.

Important: If you want to update a time sheet or a time sheet line within the existing TimeSheetBean, you have to input the whole TimeSheetBean. For example, if the time sheet contains 100 lines, you should input all 100 lines using Web service, not only the one that you want to update. Otherwise, all the lines that you did not input will be missing from the time sheet.

Return

The updated TimeSheetBean.

Limitations

- All data, specified in the properties, must comply with what is expected in PPM Center, no data validation is performed by the Web service operation.
- Notes are not implemented in this Web service operation.

Java Interface

```
UpdateTimeSheetResponseDocument
updateTimeSheet(UpdateTimeSheetDocument in)
```

Parameters	Description

UpdateTimeSheetDocument	Wrapper for TimeSheetBean. See the following example for the construction.The bean includes:
	• Long timeSheetId;
	• Long resourceId;
	• Long periodId;
	• String description;
	• Integer sequence;
	• TimeSheetStatus state; // constants correspond to those in model.TimeSheet
	• List timeSheetLines; // list of TimeSheetLineBean objects
	• String bucketShowLevel[a];
	• String bucketReportMethod;[a]
	• String periodHoursCalcTypeCode;[a]
	• Double hoursPerPeriod;[a]
	• Double workdayHours;[a]
	• Boolean allowMultipleTimeSheets;[a]
	• String hoursInPeriodType;[a]
	• Double maximumHoursPerDay;[a]
	• Double maximumHoursPerTimeSheet;[a]
	• Double minimumHoursPerTimeSheet;[a]
	• Double maximumPercentPerTimeSheet;[a]
	• Double minimumPercentPerTimeSheet;[a]
	• Boolean notifyDelinquentTimeSheet;[a]
	• Boolean reqActivitiesRequiredFlag;[a]
	• Boolean tskActivitiesRequiredFlag;[a]
	• Boolean prjActivitiesRequiredFlag;[a]
	• Boolean pkgActivitiesRequiredFlag;[a]
	• Boolean miscActivitiesRequiredFlag;[a]
	• Boolean reqWorkItemEnabledFlag;[a]
	• Boolean tskWorkItemEnabledFlag;[a]
	• Boolean prjWorkItemEnabledFlag;[a]
	• Boolean pkgWorkItemEnabledFlag;[a]
	• Boolean mscWorkItemEnabledFlag;[a]
	• Boolean enforcementLevelError;[a]
	• String exceptionMessage;[b]
	• Long timeApproverParticipantGroupId;[b]
	• Long timeApproverSecurityGroupId;[b]
	• List timeApproverUsersIds;[b] // list of Long user ids
	• Long billingApproverParticipantGroupId;[b]
	• Long billingApproverSecurityGroupId;[b]
	• List billingApproverUsersIds; [b]// list of Long user ids

UpdateTimeSheetResponseDocument	Wrapper for TimeSheetBean. See the following example for retrieving the bean and its fields.

a. Policy fields - for creation only

b. Read-only fields

Java Examples

Example: update an existing time sheet.

```
TimeSheetBean createdTimeSheetBean = null;
    public void testUpdateTimeSheet() {
        System.out.println("testUpdateTimeSheet started ...");
        try {
            if(createdTimeSheetBean == null)
                testCreateTimeSheet();

            TimeServiceStub stub = new TimeServiceStub(ctx,
WSURL);
            UpdateTimeSheetDocument updateTimeSheetDocument =
UpdateTimeSheetDocument.Factory.newInstance();
    final TimeSheetBean updateTimeSheetBean =
updateTimeSheetDocument.addNewUpdateTimeSheet().addNewTimeSheet
Bean();

            // set required fields

updateTimeSheetBean.setTimeSheetId(createdTimeSheetBean.getTime
SheetId());

updateTimeSheetBean.setResourceId(createdTimeSheetBean.getResou
rceId());

updateTimeSheetBean.setPeriodId(createdTimeSheetBean.getPeriodI
d());

            // set all other fields
            updateTimeSheetBean.setDescription("my desc");
            TimeSheetStatus timeSheetStatus =
updateTimeSheetBean.addNewState();
            timeSheetStatus.setCode(new BigInteger("1"));
            timeSheetStatus.setMeaning("unsubmitted");
            TimeSheetLineBean timeSheetLineBean =
updateTimeSheetBean.addNewTimeSheetLines();

            // set required fields
timeSheetLineBean.setTimeSheetLineId(createdTimeSheetBean.getTi
meSheetLinesArray(0).getTimeSheetLineId());

            timeSheetLineBean.setWorkItemId("2");
            timeSheetLineBean.setWorkItemSetId("1");
           timeSheetLineBean.setWorkItemType("MISC"); // TASK,
REQUEST, PACKAGE, MISC
            TimeSheetLineStatus timeSheetLineStatus =
timeSheetLineBean.addNewState();
            timeSheetLineStatus.setCode(new BigInteger("1"));
            timeSheetLineStatus.setMeaning("unsubmitted");

            UserDataInfo userDataInfo =
timeSheetLineBean.addNewUserDataBean();
            userDataInfo.setUserData1("my userdata1");
            userDataInfo.setVisUserData1("my visible
userdata1");
```

```
            TimeActualsBean timeActualsBean1 =
timeSheetLineBean.addNewTimeActualsList();
            timeActualsBean1.setTotalsFlag(true);
            timeActualsBean1.setEffortsListArray(new double[]
{1,2,3,4,5,0,0});
            TimeActualsBean timeActualsBean2 =
timeSheetLineBean.addNewTimeActualsList();
            timeActualsBean2.setTotalsFlag(false);
            timeActualsBean2.setEffortsListArray(new double[]
{1,2,3,4,5,0,0});

            timeActualsBean2.setEffortsListArray(new double[]
{1,2,3,4,5,0,0});

            UpdateTimeSheetResponseDocument response =
stub.updateTimeSheet(updateTimeSheetDocument);
            TimeSheetBean returnTimeSheetBean =
response.getUpdateTimeSheetResponse().getReturn();

            System.out.println("Time Sheet was updated " +
returnTimeSheetBean.getTimeSheetId());
        }
        catch(Exception e) {
            e.printStackTrace();
        }
        System.out.println("testUpdateTimeSheet completed");
    }
```

Errors and Exceptions

When an error occurs on this operation, you will receive a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault: [root
cause description]
```

Response message:

```
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
ex.cannotEditTimesheet	User cannot edit Timesheet	You may not have the required access grant or Time Management License.	Check the user licenses and privileges (access grants).
ex.timesheetNotExist	Timesheet does not exist	Time Sheet ID is invalid (Cancelled time sheets must still be kept in the database).	Check the validity of the time sheet ID.
ex.workitemInvalid	Timesheet line could not be created, item added is not valid for this resource	Invalid data in the new time sheet line. It could be caused by a certain field or one of the following causes: • Work item ID does not exist in the type. • Work item ID does not exist in the work item set ID.	If work item type is 'PROJECT' or 'TASK', check if the resource has the access to the work item ID; or if the work item type is 'REQUEST', check if the work item set ID is the request type ID.

Message Code	Message	Cause(s)	Possible Corrective Action
ex.workitemProblem	Timesheet line could not be created, problem probably with work item ID or type	Invalid data in the new time sheet line. It could be caused by a certain field or one of these causes: • Work item ID does not exist in the type. • Work item ID does not exist in the work item set ID. • Activity ID does not exist. • Invalid numeric value for the effort. • Count of efforts does not match the number of days in the period type of the time sheet.	Check every field of the work item in the line that failed for its validity.
ex.chargeCodeNotExist	Charge code does not exist	Charge code ID is invalid or charge code does not exist anymore in the database.	Check the validity of the charge code ID.
ex.chargeCodePercentagesMustSum100	Charge codes percentages must sum to 100, but are {0}	For each line, a list of charge codes can be defined. Each charge code must have a percentage. The sum of all the percentages per each line must sum to 100.	Check the sum of all charge codes percentages for each line.

getTimeSheet

Purpose

Read an existing time sheet object in PPM Center.

Function

This operation reads an existing time sheet object, identified by time sheet ID in PPM Center.

The user performing this operation must have one of the following access grants:

- Time Mgmt: View Time Sheets
- Time Mgmt: Edit Time Sheets

Input

A long value of an ID of an existing time sheet.

Return

An object of TimeSheetBean, with all fields values completed.

Java Interface

```
GetTimeSheetResponseDocument getTimeSheet(GetTimeSheetDocument
in)
```

Parameters	Description
GetTimeSheetDocument	Wrapper for a long value which holds the TimeSheetId. See the following example for the construction.
GetTimeSheetResponseDocument	Wrapper for TimeSheetBean. See the following example for retrieving the bean and its fields.

Java Examples

Example: read an existing time sheet.

```
TimeSheetBean createdTimeSheetBean = null;
    public void testGetTimeSheet() {
        System.out.println("testGetTimeSheet started ...");
        try {
            if(createdTimeSheetBean == null)
                testCreateTimeSheet();

            TimeServiceStub stub = new TimeServiceStub(ctx,
WSURL);
            GetTimeSheetDocument getTimeSheetDocument =
GetTimeSheetDocument.Factory.newInstance();

getTimeSheetDocument.addNewGetTimeSheet().setTimeSheetId(create
dTimeSheetBean.getTimeSheetId());

            GetTimeSheetResponseDocument response =
stub.getTimeSheet(getTimeSheetDocument);
            final TimeSheetBean returnTimeSheetBean =
response.getGetTimeSheetResponse().getReturn();

            System.out.println("Time Sheet was found " +
returnTimeSheetBean.getTimeSheetId());

        }
        catch(Exception e) {
            e.printStackTrace();
        }
        System.out.println("testGetTimeSheet completed");
    }
```

Errors and Exceptions

When an error occurs on this operation, you will receive a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault: [root
cause description]
```

Response message:

```
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
ex.timesheetNotExist	Timesheet does not exist	Time sheet ID is invalid.	Check the validity of the time sheet ID.

searchTimeSheets

Purpose

Search for a list of time sheets in PPM Center by using specific criteria.

Function

This operation searches for time sheets, according to the search criteria.

The user performing this operation must have one of the following access grants:

- Time Mgmt: View Time Sheets
- Time Mgmt: Edit Time Sheets

Input

TimeSheetSearchCriteriaBean

Return

A list of String values of the time sheet IDs.

Limitation

The maximum number of results cannot exceed 1,000.

Java Interface

```
SearchTimeSheetsResponseDocument
searchTimeSheets(SearchTimeSheetsDocument in)
```

Parameters	Description

SearchTimeSheetsDocument	Wrapper for TimeSheetSearchCriteriaBean. See the following example for the construction.
	The criteria bean includes the following fields:
	• Long periodTypeId;
	• Long periodId;
	• Long previousTimePeriodsCount;
	• List resourceIdList; // list of Long resource ids
	• String descriptionKeywords;
	• List statusList; // list of Long, statuses from model.TimeSheet
	• List managerIdList; // list of Long user ids
	• List orgUnitIdList; // list of Long orgUnit ids

SearchTimeSheetsResponseDocument	Wrapper for an array of long values, each holding a time sheet ID. See the following example for retrieving the array.

Java Examples

Example: search for time sheets.

```
public void testSearchTimeSheets() {
        System.out.println("testSearchTimeSheets started ...");
        try {
            TimeServiceStub stub = new TimeServiceStub(ctx,
WSURL);
           SearchTimeSheetsDocument searchTimeSheetsDocument =
SearchTimeSheetsDocument.Factory.newInstance();
            TimeSheetSearchCriteriaBean
timeSheetSearchCriteriaBean =
searchTimeSheetsDocument.addNewSearchTimeSheets().addNewSearchC
riteriaBean();


timeSheetSearchCriteriaBean.setPeriodTypeId(tmPeriod.getPeriodT
ypeId().longValue());
```

```
timeSheetSearchCriteriaBean.setPeriodId(tmPeriod.getPeriodId().
longValue());

timeSheetSearchCriteriaBean.setPreviousTimePeriodsCount(10);

timeSheetSearchCriteriaBean.setResourceIdListArray(new
long[]{1}); // list of Long resource ids // admin

timeSheetSearchCriteriaBean.setDescriptionKeywords("description
");
            timeSheetSearchCriteriaBean.setStatusListArray(new
long[]{1});// TimeSheet.STATUS_UNSUBMITTED from model.TimeSheet
            SearchTimeSheetsResponseDocument response =
stub.searchTimeSheets(searchTimeSheetsDocument);
            long[] returnArray =
response.getSearchTimeSheetsResponse().getTimeSheetIdArray();

            System.out.println("Time Sheet search brought " +
returnArray.length + " results");
        }
        catch(Exception e) {
            e.printStackTrace();
        }
        System.out.println("testSearchTimeSheets completed");
    }
```

Errors and Exceptions

N/A

getActualTime

Purpose

Read actual time data of the work items between a specific time range in PPM Center.

Function

This operation filters for actual data of the work items in Time Management in PPM Center.

The user performing this operation must have one of the following access grants:

- Time Mgmt: View Time Sheets
- Time Mgmt: Edit Time Sheets

Input

An object of TimeFilter, which holds all the fields described in the TimeFilter data type.

Return

An array of WorkItemActualTime objects.

Java Interface

```
GetActualTimeResponseDocument
getActualTime(GetActualTimeDocument in)
```

Parameters	Description
GetActualTimeDocument	Wrapper for a TimeFilter object. See the following example for the construction. The TimeFilter object includes the following fields: • Date startDate • Date endDate • String periodTypeName • String workItemType • String workItemName • String workItemID • List resources • List timeSheetStatuses • List timeSheetLineStatuses • boolean includeActualCost

Parameters	Description
GetActualTimeResponseDocument	Wrapper for an array of WorkItemActualTime objects. See the following example for retrieving the array. Each WorkItemActualTime includes the following fields: • Date startDate • Date endDate • String workItemName • String workItemTypeCode • String workItemID • String activityName • String chargeCode • String hours • String resource • String timeSheetLineStatus • String cost

Java Examples

Example: get actual time data.

```
public void testGetActualTime() throws Exception {
        System.out.println("testGetActualTime started ...");

        // get stub
    TimeServiceStub stub = new TimeServiceStub(ctx, WSURL);

    // create input message
    GetActualTimeDocument getActualTimeDoc =
GetActualTimeDocument.Factory.newInstance();

    // set filter
    TimeFilter filter =
getActualTimeDoc.addNewGetActualTime().addNewParam0();

    // setup the filters
    Calendar startDate = Calendar.getInstance();
    Calendar endDate = Calendar.getInstance();
    endDate.add(Calendar.DATE, 14);
    filter.setStartDate(startDate);
    filter.setEndDate(endDate);
```

```
    filter.setPeriodTypeName("Semi-Monthly");
    filter.setIncludeActualCost(true);
    filter.addResources("admin");

    // invoke the service
    GetActualTimeResponseDocument response =
stub.getActualTime(getActualTimeDoc);

    // process response
    System.out.println("Read " +
response.getGetActualTimeResponse().sizeOfReturnArray() + "
work item rows");
    WorkItemActualTime[] workItems =
response.getGetActualTimeResponse().getReturnArray();
    if (workItems.length > 0) {
    WorkItemActualTime workItem = workItems[0];
    System.out.println("total hours for misc. item = " +
workItem.getHours());
    }
          System.out.println("testGetActualTime completed");
    }
```

Errors and Exceptions

When an error occurs on this operation, you will receive a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault: [root
cause description]
```

Response message:

```
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
ex.cannotViewTimesheet	User cannot view Timesheet	You may not have the required access grant or Time Management License.	Check the user licenses and privileges (access grants).

getTimeSheetPolicy

Purpose

Read an existing Time Sheet Policy object in PPM Center.

Function

This operation reads an existing Time Sheet Policy object, identified by the unique time sheet policy name in PPM Center.

Input

A String value of the name of an existing time sheet policy

Return

An object of TimeSheetPolicyBean

Java Interface

```
GetTimeSheetPolicyResponseDocument
getTimeSheetPolicy(GetTimeSheetPolicyDocument in)
```

Parameters	Description
GetTimeSheetPolicyDocument	Wrapper for a String value which holds the policy name. See the following example for the construction.

Parameters	Description
GetTimeSheetPolicyResponseDocument	Wrapper for a TimeSheetPolicyBean. See the following example for retrieving the bean and its fields. Each bean includes the following fields: • Long timeSheetPolicyId • String timeSheetPolicyName • String bucketShowLevel • String bucketReportMethod • String periodHoursCalcTypeCode • Double hoursPerPeriod • Double workdayHours • Boolean allowMultipleTimeSheets • String hoursInPeriodType • Double maximumHoursPerDay • Double maximumHoursPerTimeSheet • Double minimumHoursPerTimeSheet • Double maximumPercentPerTimeSheet • Double minimumPercentPerTimeSheet • Boolean notifyDelinquentTimeSheet • Boolean reqActivitiesRequiredFlag • Boolean tskActivitiesRequiredFlag • Boolean prjActivitiesRequiredFlag • Boolean pkgActivitiesRequiredFlag • Boolean miscActivitiesRequiredFlag • Boolean reqWorkItemEnabledFlag • Boolean tskWorkItemEnabledFlag • Boolean prjWorkItemEnabledFlag • Boolean pkgWorkItemEnabledFlag • Boolean mscWorkItemEnabledFlag • Boolean enforcementLevelError

Java Examples

Example: read an existing time sheet policy.

```
public void testGetTimeSheetPolicy() {
        System.out.println("testGetTimeSheetPolicy started
...");
        try {
            TimeServiceStub stub = new TimeServiceStub(ctx,
WSURL);
```

```
          GetTimeSheetPolicyDocument getTimeSheetPolicyDocument
= GetTimeSheetPolicyDocument.Factory.newInstance();

getTimeSheetPolicyDocument.addNewGetTimeSheetPolicy().setPolicy
Name("Semi-Monthly - Day - Hours");

            GetTimeSheetPolicyResponseDocument response =
stub.getTimeSheetPolicy(getTimeSheetPolicyDocument);
            final TimeSheetPolicyBean returntimesheetPolicy =
response.getGetTimeSheetPolicyResponse().getReturn();

           System.out.println("Time Sheet Policy was found " +
returntimesheetPolicy.getTimeSheetPolicyId());
        }
        catch(Exception e) {
            e.printStackTrace();
        }
        System.out.println("testGetTimeSheetPolicy completed");
    }
```

Errors and Exceptions

When an error occurs on this operation, you will receive a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault: [root
cause description]
```

Response message:

```
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
ex.policyNotExist	Policy does not exist with name {0}	Policy does not exist, or corrupt databse.	Check the validity of the policy name (must be unique in PPM Center)

* All the following operations are convenience helper methods to change the status of a time sheet or a time sheet line:

- *submitTimeSheet*
- *approveTimeSheet*
- *approveTimeSheetLine*
- *rejectTimeSheet*
- *rejectTimeSheetLine*
- *reworkTimeSheetLine*
- *freezeTimeSheet*
- *closeTimeSheet*
- *cancelTimeSheet*

Since status change can occur only from a specific existing status, you must perform a validity check for this existing status first. If the existing status does not accord with the functionality you want to achieve (for example, you try to submit an approved time sheet), an exception is thrown.

If you want to update a time sheet or a time sheet line within the existing TimeSheetBean, you must input the whole TimeSheetBean. For example, if time sheet contains 100 lines, you must input all 100 lines by using Web service, not only the one that you want to update. Otherwise, all the lines that you did not input will be missing from the time sheet.

submitTimeSheet

Purpose

Submit an existing Time Sheet object in PPM Center.

Function

This operation updates the status of all the lines of an existing time sheet, and the status of the time sheet to “submitted.”

Before this operation, the status of the time sheet must be “unsubmitted” or “in-rework.”

The user performing this operation must have the permission to edit this particular time sheet, meaning that the user must meet the following conditions:

- Have the Time Mgmt: Edit Time Sheets access grant.
- Be one of the following:
  - The resource of this time sheet.
  - The delegate of the resource.
  - The manager of the resource.

Input

An object of whole TimeSheetBean to submit.

Return

An object of TimeSheetBean.

Java Interface

```
SubmitTimeSheetResponseDocument
submitTimeSheet(SubmitTimeSheetDocument in)
```

Parameters	Description
SubmitTimeSheetDocument	Wrapper for TimeSheetBean. See the following example for the construction.
SubmitTimeSheetResponseDocument	Wrapper for TimeSheetBean. See the following example on how to retrieve the bean.

Java Examples

Example: submit an existing time sheet.

```
TimeSheetBean createdTimeSheetBean = null;
    public void testSubmitTimeSheet() {
        System.out.println("testSubmitTimeSheet started ...");
        try {
            if(createdTimeSheetBean == null)
                testCreateTimeSheet();

            TimeServiceStub stub = new TimeServiceStub(ctx,
WSURL);
            SubmitTimeSheetDocument submitTimeSheetDocument =
SubmitTimeSheetDocument.Factory.newInstance();
          TimeSheetBean submitTimeSheetBean =
submitTimeSheetDocument.addNewSubmitTimeSheet().addNewTimeSheet
Bean();

submitTimeSheetBean.setTimeSheetId(createdTimeSheetBean.getTime
SheetId());

submitTimeSheetBean.setResourceId(createdTimeSheetBean.getResou
rceId());

submitTimeSheetBean.setPeriodId(createdTimeSheetBean.getPeriodI
d());
            TimeSheetStatus timeSheetStatus =
submitTimeSheetBean.addNewState();
            timeSheetStatus.setCode(new BigInteger("1"));
            timeSheetStatus.setMeaning("unsubmitted");
            TimeSheetLineBean timeSheetLineBean =
submitTimeSheetBean.addNewTimeSheetLines();

timeSheetLineBean.setTimeSheetLineId(createdTimeSheetBean.getTi
meSheetLinesArray(0).getTimeSheetLineId());

timeSheetLineBean.setWorkItemId(createdTimeSheetBean.getTimeShe
etLinesArray(0).getWorkItemId());

timeSheetLineBean.setWorkItemSetId(createdTimeSheetBean.getTime
SheetLinesArray(0).getWorkItemSetId());

timeSheetLineBean.setWorkItemType(createdTimeSheetBean.getTimeS
heetLinesArray(0).getWorkItemType());
            TimeActualsBean timeActualsBean1 =
timeSheetLineBean.addNewTimeActualsList();
            timeActualsBean1.setTotalsFlag(true);
            timeActualsBean1.setEffortsListArray(new double[]
{1,2,3,4,5,0,0,1,2,3,4,5,0,0,10});

            TimeActualsBean timeActualsBean2 =
timeSheetLineBean.addNewTimeActualsList();
            timeActualsBean2.setTotalsFlag(false);
            timeActualsBean2.setEffortsListArray(new double[]
{1,2,3,4,5,0,0,1,2,3,4,5,0,0,10});
```

```
            SubmitTimeSheetResponseDocument response =
stub.submitTimeSheet(submitTimeSheetDocument);
            TimeSheetBean returnTimeSheet =
response.getSubmitTimeSheetResponse().getReturn();

            System.out.println("Time Sheet was submitted " +
returnTimeSheet.getTimeSheetId());
        }
        catch(Exception e) {
            e.printStackTrace();
        }
        System.out.println("testSubmitTimeSheet completed");
    }
```

Errors and Exceptions

When an error occurs on this operation, you will receive a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault: [root
cause description]
```

Response message:

```
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
ex.cannotEditTimesheet	User cannot edit Timesheet	User may not have the required access grant or Time Management License.	Check the user licenses and privileges (access grants).
ex.timesheetCannotbeSubmitted	Timesheet is not in unsubmitted or rework state, it cannot be submitted	Time sheets can be submitted only if they are unsubmitted or in rework.	Check the state of the time sheet that you are trying to submit.

Message Code	Message	Cause(s)	Possible Corrective Action
ex.timesheetNullCannotbeSubmitted	Timesheet is null, cannot be submitted	Time sheet ID is invalid, database might be corrupt.	Check the validity of the time sheet ID.

approveTimeSheet

Purpose

Approve an existing Time Sheet object in PPM Center.

Function

This operation updates the status of all the lines of an existing time sheet, and the status of the time sheet to “approved.”

Before this operation, the status of the time sheet must be “pending-approval” or “in-rework” and the statuses of each Line must be “submitted” or “rejected.”

The user performing this operation must have the permission to edit this particular time sheet, meaning that the user must meet the following conditions:

- Have the Time Mgmt: Edit Time Sheets access grant.
- Be one of the following:
  - The resource of this time sheet.
  - The delegate of the resource.
  - The manager of the resource.

This operation does not have a strict checking on user permissions, which causes a discrepancy on user access grants between Web services and user interfaces. This discrepancy is resolved in PPM Center version 9.10.

An additional field updated in this operation is the Actual Approver, which is set to be the Current User ID.

Input

An object of whole TimeSheetBean to approve.

Return

An object of TimeSheetBean, complete with all the fields.

Java Interface

```
ApproveTimeSheetResponseDocument
approveTimeSheet(ApproveTimeSheetDocument in)
```

Parameters	Description
ApproveTimeSheetDocument	Wrapper for TimeSheetBean. See the following example for the construction.
ApproveTimeSheetResponseDocument	Wrapper for TimeSheetBean. See the following example on how to retrieve the bean.

Java Examples

Example: approve an existing time sheet.

```
TimeSheetBean createdTimeSheetBean = null;
    public void testApproveTimeSheetAllSubmitted() {
        System.out.println("testApproveTimeSheetAllSubmitted
started ...");
        try {
            if(createdTimeSheetBean == null)
                testSubmitTimeSheet();

            TimeServiceStub stub = new TimeServiceStub(ctx,
WSURL);
          ApproveTimeSheetDocument approveTimeSheetDocument =
ApproveTimeSheetDocument.Factory.newInstance();
       TimeSheetBean approveTimeSheetBean =
approveTimeSheetDocument.addNewApproveTimeSheet().addNewTimeShe
etBean();

approveTimeSheetBean.setTimeSheetId(createdTimeSheetBean.getTim
eSheetId());

approveTimeSheetBean.setResourceId(createdTimeSheetBean.getReso
urceId());
```

```
approveTimeSheetBean.setPeriodId(createdTimeSheetBean.getPeriod
Id());
            TimeSheetStatus timeSheetStatus =
approveTimeSheetBean.addNewState();
            timeSheetStatus.setCode(new BigInteger("2"));
            timeSheetStatus.setMeaning("pending-approval");
            TimeSheetLineBean timeSheetLineBean =
approveTimeSheetBean.addNewTimeSheetLines();

timeSheetLineBean.setTimeSheetLineId(createdTimeSheetBean.getTi
meSheetLinesArray(0).getTimeSheetLineId());

timeSheetLineBean.setWorkItemId(createdTimeSheetBean.getTimeShe
etLinesArray(0).getWorkItemId());

timeSheetLineBean.setWorkItemSetId(createdTimeSheetBean.getTime
SheetLinesArray(0).getWorkItemSetId());

timeSheetLineBean.setWorkItemType(createdTimeSheetBean.getTimeS
heetLinesArray(0).getWorkItemType());
            TimeSheetLineStatus timeSheetLineStatus =
timeSheetLineBean.addNewState();
            timeSheetLineStatus.setCode(new BigInteger("2"));
            timeSheetLineStatus.setMeaning("submitted");
            TimeActualsBean timeActualsBean1 =
timeSheetLineBean.addNewTimeActualsList();
            timeActualsBean1.setTotalsFlag(true);
            timeActualsBean1.setEffortsListArray(new double[]
{1,2,3,4,5,0,0,1,2,3,4,5,0,0,10});

            TimeActualsBean timeActualsBean2 =
timeSheetLineBean.addNewTimeActualsList();
            timeActualsBean2.setTotalsFlag(false);
            timeActualsBean2.setEffortsListArray(new double[]
{1,2,3,4,5,0,0,1,2,3,4,5,0,0,10});

            ApproveTimeSheetResponseDocument response =
stub.approveTimeSheet(approveTimeSheetDocument);
            TimeSheetBean returnTimeSheet =
response.getApproveTimeSheetResponse().getReturn();

            System.out.println("Time Sheet was approved " +
returnTimeSheet.getTimeSheetId());
        }
        catch(Exception e) {
            e.printStackTrace();
        }
        System.out.println("testApproveTimeSheetAllSubmitted
completed");
    }
```

Errors and Exceptions

When an error occurs on this operation, you will receive a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault: [root
cause description]
```

Response message:

```
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
ex.cannotApprovefTimesheet	User cannot approve Timesheet	You may not have the required access grant or Time Management License.	Check the user licenses and privileges (access grants).
ex.timesheetCannotbeApproved	Timesheet is not in pending approval state, it cannot be approved	The time sheet that you try to approve is not pending approval.	Check the state of the time sheet that you are trying to approve.
ex.timesheetNullCannotbeApproved	Timesheet is null, cannot be approved	Time sheet ID is invalid, or database might be corrupt.	Check the validity of the time sheet ID.

approveTimeSheetLine

Purpose

Approve an existing time sheet line object in PPM Center.

Function

This operation updates the status of a time sheet line to be “approved.”

Before this operation, the status of the Line must be “submitted” or “rejected.”

The user performing this operation must have the permission to edit this particular time sheet, meaning that the user must meet the following conditions:

- Have the Time Mgmt: Edit Time Sheets access grant.
- Be one of the following:
  - The resource of this time sheet.
  - The delegate of the resource.
  - The manager of the resource.

This operation does not have a strict checking on user permissions, which causes a discrepancy on user access grants between Web services and user interfaces. This discrepancy is resolved in PPM Center version 9.10.

This operation updates the status of the line and calls the *updateTimeSheet* operation.

An additional field this operation updates is the Actual Approver, which is set to the Current User ID.

Input

- A long value of the time sheet line ID.
- An object of whole TimeSheetBean where this line exists.

Return

An object of TimeSheetBean with all the fields completed.

Java Interface

```
ApproveTimeSheetLineResponseDocument
approveTimeSheetLine(ApproveTimeSheetLineDocument in)
```

Parameters	Description
TimeSheetLineID	Long value of the time sheet line ID.
ApproveTimeSheetLineDocument	Wrapper for TimeSheetBean and a long value for a time sheet line ID. See the following example for the construction.
ApproveTimeSheetLineResponse Document	Wrapper for TimeSheetBean. See the following example on how to retrieve the bean.

Java Examples

Example: approve an existing time sheet line.

```
TimeSheetBean createdTimeSheetBean = null;
    public void testApproveTimeSheetNotAllApproved() {
        System.out.println("testApproveTimeSheetNotAllApproved
started ...");
        try {
            if(createdTimeSheetBean == null)
                testSubmitTimeSheet();

            TimeServiceStub stub = new TimeServiceStub(ctx,
WSURL);
           ApproveTimeSheetDocument approveTimeSheetDocument =
ApproveTimeSheetDocument.Factory.newInstance();
       TimeSheetBean approveTimeSheetBean =
approveTimeSheetDocument.addNewApproveTimeSheet().addNewTimeShe
etBean();

approveTimeSheetBean.setTimeSheetId(createdTimeSheetBean.getTim
eSheetId());

approveTimeSheetBean.setResourceId(createdTimeSheetBean.getReso
urceId());

approveTimeSheetBean.setPeriodId(createdTimeSheetBean.getPeriod
Id());
            TimeSheetStatus timeSheetStatus =
approveTimeSheetBean.addNewState();
            timeSheetStatus.setCode(new BigInteger("2"));
            timeSheetStatus.setMeaning("pending-approval");

            TimeSheetLineBean timeSheetLineBean1 =
approveTimeSheetBean.addNewTimeSheetLines();

timeSheetLineBean1.setTimeSheetLineId(createdTimeSheetBean.getT
imeSheetLinesArray(0).getTimeSheetLineId());
```

```
timeSheetLineBean1.setWorkItemId(createdTimeSheetBean.getTimeSh
eetLinesArray(0).getWorkItemId());

timeSheetLineBean1.setWorkItemSetId(createdTimeSheetBean.getTim
eSheetLinesArray(0).getWorkItemSetId());

timeSheetLineBean1.setWorkItemType(createdTimeSheetBean.getTime
SheetLinesArray(0).getWorkItemType());
            TimeSheetLineStatus timeSheetLineStatus1 =
timeSheetLineBean1.addNewState();
            timeSheetLineStatus1.setCode(new BigInteger("4"));
            timeSheetLineStatus1.setMeaning("rejected");
            TimeActualsBean timeActualsBean11 =
timeSheetLineBean1.addNewTimeActualsList();
            timeActualsBean11.setTotalsFlag(true);
            timeActualsBean11.setEffortsListArray(new double[]
{1,2,3,4,5,0,0,1,2,3,4,5,0,0,10});

            TimeActualsBean timeActualsBean21 =
timeSheetLineBean1.addNewTimeActualsList();
            timeActualsBean21.setTotalsFlag(false);
            timeActualsBean21.setEffortsListArray(new double[]
{1,2,3,4,5,0,0,1,2,3,4,5,0,0,10});

            TimeSheetLineBean timeSheetLineBean2 =
approveTimeSheetBean.addNewTimeSheetLines();

timeSheetLineBean2.setTimeSheetLineId(createdTimeSheetBean.getT
imeSheetLinesArray(0).getTimeSheetLineId());

timeSheetLineBean2.setWorkItemId(createdTimeSheetBean.getTimeSh
eetLinesArray(0).getWorkItemId());

timeSheetLineBean2.setWorkItemSetId(createdTimeSheetBean.getTim
eSheetLinesArray(0).getWorkItemSetId());

timeSheetLineBean2.setWorkItemType(createdTimeSheetBean.getTime
SheetLinesArray(0).getWorkItemType());
            TimeSheetLineStatus timeSheetLineStatus2 =
timeSheetLineBean2.addNewState();
            timeSheetLineStatus2.setCode(new BigInteger("5"));
            timeSheetLineStatus2.setMeaning("cancelled");
            TimeActualsBean timeActualsBean12 =
timeSheetLineBean2.addNewTimeActualsList();
            timeActualsBean12.setTotalsFlag(true);
            timeActualsBean12.setEffortsListArray(new double[]
{1,2,3,4,5,0,0,1,2,3,4,5,0,0,10});

            TimeActualsBean timeActualsBean22 =
timeSheetLineBean2.addNewTimeActualsList();
            timeActualsBean22.setTotalsFlag(false);
            timeActualsBean22.setEffortsListArray(new double[]
{1,2,3,4,5,0,0,1,2,3,4,5,0,0,10});
```

```
            ApproveTimeSheetResponseDocument response =
stub.approveTimeSheet(approveTimeSheetDocument);
            TimeSheetBean returnTimeSheet =
response.getApproveTimeSheetResponse().getReturn();

           System.out.println("Time Sheet lines were approved "
+ returnTimeSheet.getTimeSheetId());
        }
        catch(Exception e) {
            e.printStackTrace();
        }
        System.out.println("testApproveTimeSheetNotAllApproved
completed");
    }
```

Errors and Exceptions

When an error occurs on this operation, you will receive a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault: [root
cause description]
```

Response message:

```
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
ex.cannotApproveTimesheetLine	User cannot approve Timesheet Line	You may not have the required access grant or Time Management License.	Check the user licenses and privileges (access grants).

Message Code	Message	Cause(s)	Possible Corrective Action
ex.timesheetLineCannotbeApproved	Timesheet line is not in submitted or rejected state, it cannot be approved	You try to approve a time sheet line that is neither submitted nor rejected.	Check the state of the time sheet line that you are trying to approve.
ex.timesheetLineNotExist	Timesheet line does not exist in this timesheet	The time sheet ID or line ID is invalid, or database might be corrupt.	Check the validity of the time sheet ID and line ID.
ex.timesheetLineNullCannotbeApproved	Timesheet line or timesheet is null, cannot be approved	Time sheet ID or line ID is invalid, or database might be corrupt.	Check the validity of the time sheet ID and line ID.

rejectTimeSheet

Purpose

Reject an existing time sheet object in PPM Center.

Function

This operation updates the status of all the lines of an existing time sheet to “rejected,” and the status of the time sheet to be “in-rework.”

Before this operation, the status of the time sheet must be “pending-approval.”

The user performing this operation must have the permission to edit this particular time sheet, meaning that the user must meet the following conditions:

- Have the Time Mgmt: Edit Time Sheets access grant.
- Be one of the following:
  - The resource of this time sheet.

- The delegate of the resource.
- The manager of the resource.

This operation does not have a strict checking on user permissions, which causes a discrepancy on user access grants between Web services and user interfaces. This discrepancy is resolved in PPM Center version 9.10.

An additional field this operation updates is the Actual Approver, which is set to null.

Input

An object of whole TimeSheetBean to reject.

Return

An object of TimeSheetBean, with all the fields completed.

Java Interface

```
RejectTimeSheetResponseDocument
rejectTimeSheet(RejectTimeSheetDocument in)
```

Parameters	Description
RejectTimeSheetDocument	Wrapper for TimeSheetBean. See the following example for the construction.
RejectTimeSheetResponseDocument	Wrapper for TimeSheetBean. See the following example on how to retrieve the bean.

Java Examples

Example: reject an existing time sheet.

```
TimeSheetBean createdTimeSheetBean = null;
    public void testRejectTimeSheet() {
        System.out.println("testRejectTimeSheet started ...");
        try {
            if(createdTimeSheetBean == null)
                testSubmitTimeSheet();
```

```
            TimeServiceStub stub = new TimeServiceStub(ctx,
WSURL);
            RejectTimeSheetDocument rejectTimeSheetDocument =
RejectTimeSheetDocument.Factory.newInstance();
          TimeSheetBean rejectTimeSheetBean =
rejectTimeSheetDocument.addNewRejectTimeSheet().addNewTimeSheet
Bean();

rejectTimeSheetBean.setTimeSheetId(createdTimeSheetBean.getTime
SheetId());

rejectTimeSheetBean.setResourceId(createdTimeSheetBean.getResou
rceId());

rejectTimeSheetBean.setPeriodId(createdTimeSheetBean.getPeriodI
d());
            TimeSheetStatus timeSheetStatus =
rejectTimeSheetBean.addNewState();
            timeSheetStatus.setCode(new BigInteger("2"));
            timeSheetStatus.setMeaning("pending-approval");
            TimeSheetLineBean timeSheetLineBean =
rejectTimeSheetBean.addNewTimeSheetLines();

timeSheetLineBean.setTimeSheetLineId(createdTimeSheetBean.getTi
meSheetLinesArray(0).getTimeSheetLineId());

timeSheetLineBean.setWorkItemId(createdTimeSheetBean.getTimeShe
etLinesArray(0).getWorkItemId());

timeSheetLineBean.setWorkItemSetId(createdTimeSheetBean.getTime
SheetLinesArray(0).getWorkItemSetId());

timeSheetLineBean.setWorkItemType(createdTimeSheetBean.getTimeS
heetLinesArray(0).getWorkItemType());
            TimeActualsBean timeActualsBean1 =
timeSheetLineBean.addNewTimeActualsList();
            timeActualsBean1.setTotalsFlag(true);
            timeActualsBean1.setEffortsListArray(new double[]
{1,2,3,4,5,0,0,1,2,3,4,5,0,0,10});

            TimeActualsBean timeActualsBean2 =
timeSheetLineBean.addNewTimeActualsList();
            timeActualsBean2.setTotalsFlag(false);
            timeActualsBean2.setEffortsListArray(new double[]
{1,2,3,4,5,0,0,1,2,3,4,5,0,0,10});

            RejectTimeSheetResponseDocument response =
stub.rejectTimeSheet(rejectTimeSheetDocument);
            TimeSheetBean returnTimeSheet =
response.getRejectTimeSheetResponse().getReturn();

            System.out.println("Time Sheet was rejected " +
returnTimeSheet.getTimeSheetId());
        }
```

```
        catch(Exception e) {
            e.printStackTrace();
        }
        System.out.println("testRejectTimeSheet completed");
    }
```

Errors and Exceptions

When an error occurs on this operation, you will receive a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault: [root
cause description]
```

Response message:

```
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
ex.cannotRejectTimesheet	User cannot reject Timesheet	You may not have the required access grant or Time Management License.	Check the user licenses and privileges (access grants).
ex.timesheetCannotbeRejected	Timesheet is not in pending approval state, it cannot be rejected	You try to reject a time sheet that is not pending approval.	Check the state of the time sheet that you are trying to reject.
ex.timesheetNullCannotbeRejected	Timesheet is null, cannot be rejected	Time sheet ID is invalid, or database might be corrupt.	Check the validity of the time sheet ID.

rejectTimeSheetLine

Purpose

Reject an existing time sheet line object in PPM Center.

Function

This operation updates the status of a time sheet line to be “rejected.”

Before this operation, the status of the line must be “submitted” or “approved.”

The user performing this operation must have the permission to edit this particular time sheet, meaning that the user must meet the following conditions:

- Have the Time Mgmt: Edit Time Sheets access grant.
- Be one of the following:
  - The resource of this time sheet.
  - The delegate of the resource.
  - The manager of the resource.

This operation does not have a strict checking on user permissions, which causes a discrepancy on user access grants between Web services and user interfaces. This discrepancy is resolved in PPM Center version 9.10.

This operation updates the status of the line and calls the *updateTimeSheet* operation.

An additional field this operation updates is the Actual Approver, which is set to null.

Input

A long value of the time sheet line ID.

An object of whole TimeSheetBean where this line exists.

Return

An object of TimeSheetBean, with all the fields completed.

Java Interface

```
RejectTimeSheetLineResponseDocument
rejectTimeSheetLine(RejectTimeSheetLineDocument in)
```

Parameters	Description
RejectTimeSheetLineDocument	Wrapper for TimeSheetBean and a long value for a time sheet line ID. See the following example for the construction.
RejectTimeSheetLineResponseDocument	Wrapper for TimeSheetBean. See the following example on how to retrieve the bean.

Java Examples

Example: reject an existing time sheet line

```
TimeSheetBean createdTimeSheetBean = null;
        public void testRejectTimeSheetLine() {
        System.out.println("testRejectTimeSheetLine started
...");
        try {
            if(createdTimeSheetBean == null)
                testSubmitTimeSheet();

            TimeServiceStub stub = new TimeServiceStub(ctx,
WSURL);
            RejectTimeSheetDocument rejectTimeSheetDocument =
RejectTimeSheetDocument.Factory.newInstance();
          TimeSheetBean rejectTimeSheetBean =
rejectTimeSheetDocument.addNewRejectTimeSheet().addNewTimeSheet
Bean();

rejectTimeSheetBean.setTimeSheetId(createdTimeSheetBean.getTime
SheetId());

rejectTimeSheetBean.setResourceId(createdTimeSheetBean.getResou
rceId());

rejectTimeSheetBean.setPeriodId(createdTimeSheetBean.getPeriodI
d());
```

```
            TimeSheetStatus timeSheetStatus =
rejectTimeSheetBean.addNewState();
            timeSheetStatus.setCode(new BigInteger("2"));
            timeSheetStatus.setMeaning("pending-approval");
            TimeSheetLineBean timeSheetLineBean =
rejectTimeSheetBean.addNewTimeSheetLines();

timeSheetLineBean.setTimeSheetLineId(createdTimeSheetBean.getTi
meSheetLinesArray(0).getTimeSheetLineId());

timeSheetLineBean.setWorkItemId(createdTimeSheetBean.getTimeShe
etLinesArray(0).getWorkItemId());

timeSheetLineBean.setWorkItemSetId(createdTimeSheetBean.getTime
SheetLinesArray(0).getWorkItemSetId());

timeSheetLineBean.setWorkItemType(createdTimeSheetBean.getTimeS
heetLinesArray(0).getWorkItemType());
            TimeActualsBean timeActualsBean1 =
timeSheetLineBean.addNewTimeActualsList();
            timeActualsBean1.setTotalsFlag(true);
            timeActualsBean1.setEffortsListArray(new double[]
{1,2,3,4,5,0,0,1,2,3,4,5,0,0,10});

            TimeActualsBean timeActualsBean2 =
timeSheetLineBean.addNewTimeActualsList();
            timeActualsBean2.setTotalsFlag(false);
            timeActualsBean2.setEffortsListArray(new double[]
{1,2,3,4,5,0,0,1,2,3,4,5,0,0,10});

            RejectTimeSheetResponseDocument response =
stub.rejectTimeSheet(rejectTimeSheetDocument);
            TimeSheetBean returnTimeSheet =
response.getRejectTimeSheetResponse().getReturn();

            System.out.println("Time Sheet was rejected " +
returnTimeSheet.getTimeSheetId());
        }
        catch(Exception e) {
            e.printStackTrace();
        }
        System.out.println("testRejectTimeSheetLine
completed");
    }
```

Errors and Exceptions

When an error occurs on this operation, you will receive a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault: [root
cause description]
```

Response message:

```
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
ex.cannotRejectTimesheetLine	User cannot reject Timesheet Line	You may not have the required access grant or Time Management License.	Check the user licenses and privileges (access grants).
ex.timesheetLineCannotbeRejected	Timesheet line is not in submitted or approved state, it cannot be rejected	You try to reject a time sheet line that is neither submitted nor approved.	Check the state of the time sheet line that you are trying to reject.
ex.timesheetLineNotExist	Timesheet line does not exist in this timesheet	Time sheet ID or line ID is invalid, or database might be corrupt.	Check the validity of the time sheet ID and line ID.
ex.timesheetLineNullCannotbeRejected	Timesheet line or timesheet is null, cannot be rejected	Time sheet ID or line ID is invalid, or database might be corrupt.	Check the validity of the time sheet ID and line ID.

reworkTimeSheetLine

Purpose

Rework an existing time sheet line object in PPM Center.

Function

This operation updates the status of a time sheet line to be “unsubmitted” and updates the time sheet to “in-rework.”

Before this operation, the status of the line must be “submitted,” “approved,” or “rejected.”

The user performing this operation must have the permission to edit this particular time sheet, meaning that the user must meet the following conditions:

- Have the Time Mgmt: Edit Time Sheets access grant.
- Be one of the following:
  - The resource of this time sheet.
  - The delegate of the resource.
  - The manager of the resource.

This operation updates the status of the line and calls the *updateTimeSheet* operation.

An additional field this operation updates is the Actual Approver, which is set to null.

Input

A long value of the time sheet line ID.

An object of whole TimeSheetBean where this line exists.

Return

An object of TimeSheetBean, complete with all the fields.

Java Interface

```
ReworkTimeSheetLineResponseDocument
reworkTimeSheetLine(ReworkTimeSheetLineDocument in)
```

Parameters	Description
ReworkTimeSheetLineDocument	Wrapper for TimeSheetBean and a long value for a time sheet line ID. See the following example for the construction.
ReworkTimeSheetLineResponseDocument	Wrapper for TimeSheetBean. See the following example on how to retrieve the bean.

Java Examples

Example: rework an existing time sheet line.

```
TimeSheetBean createdTimeSheetBean = null;
        public void testReworkTimeSheetLine() {
        System.out.println("testReworkTimeSheetLine started
...");
        try {
            if(createdTimeSheetBean == null)
                testSubmitTimeSheet();

            TimeServiceStub stub = new TimeServiceStub(ctx,
WSURL);
            ReworkTimeSheetDocument reworkTimeSheetDocument =
ReworkTimeSheetDocument.Factory.newInstance();
          TimeSheetBean reworkTimeSheetBean =
reworkTimeSheetDocument.addNewReworkTimeSheet().addNewTimeSheet
Bean();

reworkTimeSheetBean.setTimeSheetId(createdTimeSheetBean.getTime
SheetId());

reworkTimeSheetBean.setResourceId(createdTimeSheetBean.getResou
rceId());

reworkTimeSheetBean.setPeriodId(createdTimeSheetBean.getPeriodI
d());
            TimeSheetStatus timeSheetStatus =
reworkTimeSheetBean.addNewState();
            timeSheetStatus.setCode(new BigInteger("2"));
            timeSheetStatus.setMeaning("pending-approval");
            TimeSheetLineBean timeSheetLineBean =
reworkTimeSheetBean.addNewTimeSheetLines();

timeSheetLineBean.setTimeSheetLineId(createdTimeSheetBean.getTi
meSheetLinesArray(0).getTimeSheetLineId());
```

```
timeSheetLineBean.setWorkItemId(createdTimeSheetBean.getTimeShe
etLinesArray(0).getWorkItemId());

timeSheetLineBean.setWorkItemSetId(createdTimeSheetBean.getTime
SheetLinesArray(0).getWorkItemSetId());

timeSheetLineBean.setWorkItemType(createdTimeSheetBean.getTimeS
heetLinesArray(0).getWorkItemType());
            TimeActualsBean timeActualsBean1 =
timeSheetLineBean.addNewTimeActualsList();
            timeActualsBean1.setTotalsFlag(true);
            timeActualsBean1.setEffortsListArray(new double[]
{1,2,3,4,5,0,0,1,2,3,4,5,0,0,10});

            TimeActualsBean timeActualsBean2 =
timeSheetLineBean.addNewTimeActualsList();
            timeActualsBean2.setTotalsFlag(false);
            timeActualsBean2.setEffortsListArray(new double[]
{1,2,3,4,5,0,0,1,2,3,4,5,0,0,10});

            ReworkTimeSheetResponseDocument response =
stub.reworkTimeSheet(reworkTimeSheetDocument);
            TimeSheetBean returnTimeSheet =
response.getReworkTimeSheetResponse().getReturn();

            System.out.println("Time Sheet was reworked " +
returnTimeSheet.getTimeSheetId());
        }
        catch(Exception e) {
            e.printStackTrace();
        }
        System.out.println("testReworkTimeSheetLine
completed");
    }
```

Errors and Exceptions

When an error occurs on this operation, you will receive a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault: [root
cause description]
```

Response message:

```
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
ex.cannotReworkTimesheetLine	User cannot rework Timesheet Line	You may not have the required access grant or Time Management License.	Check the user licenses and privileges (access grants).
ex.timesheetLineCannotbeReworked	Timesheet line is not in submitted, rejected or approved state, it cannot be reworked	You try to rework a time sheet line that is not submitted, rejected, or approved.	Check the state of the time sheet line that you are trying to rework.
ex.timesheetLineNotExist	Timesheet line does not exist in this timesheet	Time sheet ID or line ID is invalid, database might be corrupt.	Check the validity of the time sheet ID and line ID.
ex.timesheetLineNullCannotbeReworked	Timesheet line or timesheet is null, cannot be reworked	Time sheet ID or line ID is invalid, database might be corrupt	Check the validity of the time sheet ID and line ID.

freezeTimeSheet

Purpose

Freeze an existing time sheet object in PPM Center.

Function

This operation updates the status of all the lines of an existing time sheet, and the status of the time sheet to “frozen.”

Before this operation, the status of the time sheet must be “approved.”

The user performing this operation must have the permission to edit this particular time sheet, meaning that the user must meet the following conditions:

- Have the Time Mgmt: Edit Time Sheets access grant.
- Be one of the following:
  - The resource of this time sheet.
  - The delegate of the resource.
  - The manager of the resource.

This operation does not have a strict checking on user permissions, which causes a discrepancy on user access grants between Web services and user interfaces. This discrepancy is resolved in PPM Center version 9.10.

Input

An object of whole TimeSheetBean to freeze.

Return

An object of TimeSheetBean, with all the fields completed.

Java Interface

```
FreezeTimeSheetResponseDocument
freezeTimeSheet(FreezeTimeSheetDocument in)
```

Parameters	Description
FreezeTimeSheetDocument	Wrapper for TimeSheetBean. See the following example for the construction.
FreezeTimeSheetResponseDocument	Wrapper for TimeSheetBean. See the following example on how to retrieve the bean.

Java Examples

Example: freeze an existing time sheet.

```
TimeSheetBean createdTimeSheetBean = null;
    public void testFreezeTimeSheet() {
        System.out.println("testFreezeTimeSheet started ...");
        try {
            if(createdTimeSheetBean == null)
                testApproveTimeSheetAllApproved();

            TimeServiceStub stub = new TimeServiceStub(ctx,
WSURL);
            FreezeTimeSheetDocument freezeTimeSheetDocument =
FreezeTimeSheetDocument.Factory.newInstance();
            TimeSheetBean freezeTimeSheetBean =
freezeTimeSheetDocument.addNewFreezeTimeSheet().addNewTimeSheet
Bean();

freezeTimeSheetBean.setTimeSheetId(createdTimeSheetBean.getTime
SheetId());

freezeTimeSheetBean.setResourceId(createdTimeSheetBean.getResou
rceId());

freezeTimeSheetBean.setPeriodId(createdTimeSheetBean.getPeriodI
d());
            TimeSheetStatus timeSheetStatus =
freezeTimeSheetBean.addNewState();
            timeSheetStatus.setCode(new BigInteger("4"));
            timeSheetStatus.setMeaning("approved");
            TimeSheetLineBean timeSheetLineBean =
freezeTimeSheetBean.addNewTimeSheetLines();

timeSheetLineBean.setTimeSheetLineId(createdTimeSheetBean.getTi
meSheetLinesArray(0).getTimeSheetLineId());

timeSheetLineBean.setWorkItemId(createdTimeSheetBean.getTimeShe
etLinesArray(0).getWorkItemId());

timeSheetLineBean.setWorkItemSetId(createdTimeSheetBean.getTime
SheetLinesArray(0).getWorkItemSetId());

timeSheetLineBean.setWorkItemType(createdTimeSheetBean.getTimeS
heetLinesArray(0).getWorkItemType());
            TimeActualsBean timeActualsBean1 =
timeSheetLineBean.addNewTimeActualsList();
            timeActualsBean1.setTotalsFlag(true);
            timeActualsBean1.setEffortsListArray(new double[]
{1,2,3,4,5,0,0,1,2,3,4,5,0,0,10});

            TimeActualsBean timeActualsBean2 =
timeSheetLineBean.addNewTimeActualsList();
            timeActualsBean2.setTotalsFlag(false);
            timeActualsBean2.setEffortsListArray(new double[]
{1,2,3,4,5,0,0,1,2,3,4,5,0,0,10});
```

```
            FreezeTimeSheetResponseDocument response =
stub.freezeTimeSheet(freezeTimeSheetDocument);
            TimeSheetBean returnTimeSheet =
response.getFreezeTimeSheetResponse().getReturn();

            System.out.println("Time Sheet was frozen " +
returnTimeSheet.getTimeSheetId());
        }
        catch(Exception e) {
            e.printStackTrace();
        }
        System.out.println("testFreezeTimeSheet completed");
    }
```

Errors and Exceptions

When an error occurs on this operation, you will receive a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault: [root
cause description]
```

Response message:

```
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
ex.cannotFreezeTimesheet	User cannot freeze Timesheet	You may not have the required access grant or Time Management License.	Check the user licenses and privileges (access grants).
ex.timesheetCannotbeFrozen	Timesheet is not in approved state, it cannot be frozen	You try to freeze a time sheet that is not approved.	Check the state of the time sheet that you are trying to freeze.
ex.timesheetNullCannotbeFrozen	Timesheet is null, cannot be frozen	Time sheet ID is invalid, database might be corrupt.	Check the validity of the time sheet ID.

closeTimeSheet

Purpose

Close an existing time sheet object in PPM Center.

Function

This operation updates the status of all the lines of an existing time sheet and the status of the time sheet to “closed.”

Before this operation, the status of the time sheet must be “approved” or The user performing this operation must have the permission to edit this particular time sheet, meaning that the user must meet the following conditions:

- Have the Time Mgmt: Edit Time Sheets access grant.
- Be one of the following:
  - The resource of this time sheet.
  - The delegate of the resource.
  - The manager of the resource.

This operation does not have a strict checking on user permissions, which causes a discrepancy on user access grants between Web services and user interfaces. This discrepancy is resolved in PPM Center version 9.10.

Input

An object of whole TimeSheetBean to close

Return

An object of TimeSheetBean, complete with all the fields.

Java Interface

```
CloseTimeSheetResponseDocument
closeTimeSheet(CloseTimeSheetDocument in)
```

Parameters	Description
CloseTimeSheetDocument	Wrapper for TimeSheetBean. See the following example for the construction.
CloseTimeSheetResponseDocument	Wrapper for TimeSheetBean. See the following example on how to retrieve the bean.

Java Examples

Example: close an existing time sheet

```
TimeSheetBean createdTimeSheetBean = null;
    public void testCloseTimeSheet() {
        System.out.println("testCloseTimeSheet started ...");
        try {
            if(createdTimeSheetBean == null)
                testFreezeTimeSheet();

            TimeServiceStub stub = new TimeServiceStub(ctx,
WSURL);
            CloseTimeSheetDocument closeTimeSheetDocument =
CloseTimeSheetDocument.Factory.newInstance();
            TimeSheetBean closeTimeSheetBean =
closeTimeSheetDocument.addNewCloseTimeSheet().addNewTimeSheetBe
an();

closeTimeSheetBean.setTimeSheetId(createdTimeSheetBean.getTimeS
heetId());

closeTimeSheetBean.setResourceId(createdTimeSheetBean.getResour
ceId());

closeTimeSheetBean.setPeriodId(createdTimeSheetBean.getPeriodId
());
            TimeSheetStatus timeSheetStatus =
closeTimeSheetBean.addNewState();
            timeSheetStatus.setCode(new BigInteger("6"));
            timeSheetStatus.setMeaning("frozen");
            TimeSheetLineBean timeSheetLineBean =
closeTimeSheetBean.addNewTimeSheetLines();

timeSheetLineBean.setTimeSheetLineId(createdTimeSheetBean.getTi
meSheetLinesArray(0).getTimeSheetLineId());

timeSheetLineBean.setWorkItemId(createdTimeSheetBean.getTimeShe
etLinesArray(0).getWorkItemId());
```

```
timeSheetLineBean.setWorkItemSetId(createdTimeSheetBean.getTime
SheetLinesArray(0).getWorkItemSetId());

timeSheetLineBean.setWorkItemType(createdTimeSheetBean.getTimeS
heetLinesArray(0).getWorkItemType());
            TimeActualsBean timeActualsBean1 =
timeSheetLineBean.addNewTimeActualsList();
            timeActualsBean1.setTotalsFlag(true);
            timeActualsBean1.setEffortsListArray(new double[]
{1,2,3,4,5,0,0,1,2,3,4,5,0,0,10});

            TimeActualsBean timeActualsBean2 =
timeSheetLineBean.addNewTimeActualsList();
            timeActualsBean2.setTotalsFlag(false);
            timeActualsBean2.setEffortsListArray(new double[]
{1,2,3,4,5,0,0,1,2,3,4,5,0,0,10});

            CloseTimeSheetResponseDocument response =
stub.closeTimeSheet(closeTimeSheetDocument);
            TimeSheetBean returnTimeSheet =
response.getCloseTimeSheetResponse().getReturn();

            System.out.println("Time Sheet was closed " +
returnTimeSheet.getTimeSheetId());
        }
        catch(Exception e) {
            e.printStackTrace();
        }
        System.out.println("testCloseTimeSheet completed");
    }
```

Errors and Exceptions

When an error occurs on this operation, you will receive a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault: [root
cause description]
```

Response message:

```
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
ex.cannotCloseTimesheet	User cannot close Timesheet	You may not have the required access grant or Time Management License.	Check the user licenses and privileges (access grants).
ex.timesheetCannotbeClosed	Timesheet is not in approved or frozen state, it cannot be closed	You try to close a time sheet that is neither approved, nor frozen.	Check the state of the time sheet that you are trying to close.
ex.timesheetNullCannotbeClosed	Timesheet is null, cannot be closed	Time sheet ID is invalid, database might be corrupt.	Check the validity of the time sheet ID.

cancelTimeSheet

Purpose

Cancel an existing Time Sheet object in PPM Center.

Function

This operation updates the status of all the lines of an existing time sheet and the status of the time sheet to “cancelled.”

Before this operation, the status of the time sheet must be “unsubmitted.”

The user performing this operation must have the permission to edit this particular time sheet, meaning that the user must meet the following conditions:

- Have the Time Mgmt: Edit Time Sheets access grant.
- Be one of the following:
  - The resource of this time sheet.
  - The delegate of the resource.

- The manager of the resource.

This operation does not have a strict checking on user permissions, which causes a discrepancy on user access grants between Web services and user interfaces. This discrepancy is resolved in PPM Center version 9.10.

Input

An object of the whole TimeSheetBean to cancel

Return

An object of TimeSheetBean, with all the fields completed

Java Interface

```
CancelTimeSheetResponseDocument
cancelTimeSheet(CancelTimeSheetDocument in)
```

Parameters	Description
CancelTimeSheetDocument	Wrapper for TimeSheetBean. See the following example for the construction.
CancelTimeSheetResponseDocument	Wrapper for TimeSheetBean. See the following example on how to retrieve the bean.

Java Examples

Example: cancel an existing time sheet.

```
TimeSheetBean createdTimeSheetBean = null;
    public void testCancelTimeSheet() {
        System.out.println("testCancelTimeSheet started ...");
        try {

            if(createdTimeSheetBean == null)
                testCreateTimeSheet();

            TimeServiceStub stub = new TimeServiceStub(ctx,
WSURL);
            CancelTimeSheetDocument cancelTimeSheetDocument =
CancelTimeSheetDocument.Factory.newInstance();
```

```
          TimeSheetBean cancelTimeSheetBean =
cancelTimeSheetDocument.addNewCancelTimeSheet().addNewTimeSheet
Bean();

cancelTimeSheetBean.setTimeSheetId(createdTimeSheetBean.getTime
SheetId());

cancelTimeSheetBean.setResourceId(createdTimeSheetBean.getResou
rceId());

cancelTimeSheetBean.setPeriodId(createdTimeSheetBean.getPeriodI
d());
            TimeSheetStatus timeSheetStatus =
cancelTimeSheetBean.addNewState();
            timeSheetStatus.setCode(new BigInteger("1"));
            timeSheetStatus.setMeaning("unsubmitted");
            TimeSheetLineBean timeSheetLineBean =
cancelTimeSheetBean.addNewTimeSheetLines();

timeSheetLineBean.setTimeSheetLineId(createdTimeSheetBean.getTi
meSheetLinesArray(0).getTimeSheetLineId());

timeSheetLineBean.setWorkItemId(createdTimeSheetBean.getTimeShe
etLinesArray(0).getWorkItemId());

timeSheetLineBean.setWorkItemSetId(createdTimeSheetBean.getTime
SheetLinesArray(0).getWorkItemSetId());

timeSheetLineBean.setWorkItemType(createdTimeSheetBean.getTimeS
heetLinesArray(0).getWorkItemType());
            TimeActualsBean timeActualsBean1 =
timeSheetLineBean.addNewTimeActualsList();
            timeActualsBean1.setTotalsFlag(true);
            timeActualsBean1.setEffortsListArray(new double[]
{1,2,3,4,5,0,0,1,2,3,4,5,0,0,10});

            TimeActualsBean timeActualsBean2 =
timeSheetLineBean.addNewTimeActualsList();
            timeActualsBean2.setTotalsFlag(false);
            timeActualsBean2.setEffortsListArray(new double[]
{1,2,3,4,5,0,0,1,2,3,4,5,0,0,10});

            CancelTimeSheetResponseDocument response =
stub.cancelTimeSheet(cancelTimeSheetDocument);
            TimeSheetBean returnTimeSheet =
response.getCancelTimeSheetResponse().getReturn();

            System.out.println("Time Sheet was cancelled " +
returnTimeSheet.getTimeSheetId());
        }
        catch(Exception e) {
            e.printStackTrace();
        }
        System.out.println("testCancelTimeSheet completed");
```

```
}
```

Errors and Exceptions

When an error occurs on this operation, you will receive a description of the root cause in the log or in the response message.

The server log file content is similar to the following:

```
Exception in thread "main" org.apache.axis2.AxisFault: [root
cause description]
```

Response message:

```
<exception:exceptionDetails xmlns:exception="http://
www.mercury.com/ppm/ws/exception">
<exception:detail>[root cause description] </exception:detail>
</exception:exceptionDetails>
```

Possible root cause descriptions:

Message Code	Message	Cause(s)	Possible Corrective Action
ex.cannotEditTimesheet	User cannot edit Timesheet	You may not have the required access grant or Time Management License.	Check the user licenses and privileges (access grants).
ex.timesheetCannotbeCancelled	Timesheet is not in unsubmitted state, it cannot be cancelled	You try to cancel a Time sheet that is not unsubmitted.	Check the state of the time sheet that you are trying to cancel.
ex.timesheetNullCannotbeCancelled	Timesheet is null, cannot be cancelled	Time sheet ID is invalid, database might be corrupt.	Check the validity of the time sheet ID.

8 Multilingual User Interface Support in Web Services

Overview

PPM Center supports a multilingual user interface (MLU) to facilitate the needs of global enterprises. PPM Center Web services is also enhanced to support MLU so that users can specify their preferred languages when invoking a Web service operation.

Users can use their preferred language even if that language is different from the specified system language or session language. When calling PPM Center through Web services, you can specify the preferred language in the Simple Object Access Protocol (SOAP) header.

In the PPM Center standard interface, a user can choose a session language during log-in and a Web service caller can specify a preferred language in the SOAP header that will override the session language that the user specified.

Operations History

MLU support for PPM Center Web services is newly introduced in PPM Center 8.00.

Preferred Language Setting

Specifying the Session Language

The following example specifies the session language in the SOAP header.

```
<soap:Header>
    <common:UserLocaleHeader xmlns:common="http://mercury.com/
ppm/common/1.0">
        <common:LanguageLocale>de</common:LanguageLocale>
    </common:UserLocaleHeader>

    <wsseecurity xmlns:wsse="http://docs.oasis-open.org/wss/
2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd"
soap:mustUnderstand="1">
        <wsu:Timestamp xmlns:wsu="http://docs.oasis-open.org/
wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
wsu:Id="Timestamp-6557466">
            <wsu:Created>2009-08-21T18:29:15.687Z</wsu:Created>
            <wsu:Expires>2009-08-21T18:34:15.687Z</wsu:Expires>
        </wsu:Timestamp>
        <wsse:UsernameToken xmlns:wsu="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-
1.0.xsd" wsu:Id="UsernameToken-7433399">
            <wsse:Username>admin</wsse:Username>
            <wsseassword Type="http://docs.oasis-open.org/wss/
2004/01/oasis-200401-wss-username-token-profile-
1.0#PasswordText">admin</wsseassword>
        </wsse:UsernameToken>
    </wsseecurity>
</soap:Header>
```

When invoking a Web service operation, a caller can specify the preferred language in the UserLocaleHeader SOAP header element. To make it easier for the caller to set this SOAP header element, the Web service toolkit has enhanced the SOAPHeaderCreater class by adding the setUserLocaleHeader method.

SOAPHeaderCreator

Purpose

This class provides interfaces to set SOAP header elements for PPM Center Web services.

Function

setAuditHeader

This method is used to add some auditing information through an Audit header.

setUserLocaleHeader

This method is used to set a preferred language when a caller invokes a Web service operation. For example, if you want to create requests in a language other than the system language, you can specify the preferred locale by using this method.

Limitations

Data in Multiple Languages

In a single web service operation, there is no support for working with data in multiple languages. To work with data in multiple languages, you must perform multiple, separate Web service transactions, each specifying the desired session language.

Supported Languages

The language code you set in the method must be supported (and enabled) in PPM Center. If the translation for a certain value you want to specify does not exist in that language, PPM Center passes the value in the system language.

Java Interface

Add the Audit header for auditing information:

```
SOAPHeaderCreator.setAuditHeader(stub, "Submitted By:
TEST","PPM on " +
InetAddress.getLocalHost().getHostAddress(),"createRequest");
```

Parameters	Description
stub	Generated by another Web service operation. For example: `DemandServiceStub stub = new DemandServiceStub(ctx, serviceURL);`

Add `UserLocaleHeader` to set the language locale to a specified locale:

```
SOAPHeaderCreator.setUserLocaleHeader(stub, locale);
```

Parameters	Description
stub	Generated by another Web service operation. For example: `DemandServiceStub stub = new DemandServiceStub(ctx, serviceURL);`
locale	Language code of the language you want to set the session to, for example, **de** (for German) or **ko** (for Korean).

Java Example

This is an example for DemandService that demonstrates how to get the MLU functionality from DemandService in a PPM Center instance that supports MLU.

```
import examples.dm.DemandServiceClient;
import examples.util.SOAPHeaderCreator;


public class DemandServiceMLUClient {

    protected ConfigurationContext ctx = null;

    public DemandServiceMLUClient() {
        String repositoryPath =
System.getProperty("client.repository.dir");
        String axis2 = repositoryPath + "/conf/client-
axis2.xml";
        File file = new File(axis2);
        if (file.exists()) {
            try {
                ctx =
ConfigurationContextFactory.createConfigurationContextFromFileS
ystem(repositoryPath, axis2);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }

    /**
     * The main program
     *
     * Parameter: args[0] - service URL. e.g.
```

```
     * http://server:port/itg/ppmservices/DemandService
     * args[1] - language locale e.g. de or en, ko etc.
     *
     */
    public static void main(String[] args) throws Exception {
        // check parameter
        if (args.length < 1) {
           System.out.println("Usage: java DemandServiceClient
<service URL> <language locale>");
            System.exit(1);
        }

        System.out.println("Starting Demand Service MLU
tests...");

        // get server URL
        String serviceURL = args[0];
        // get preferred language
        String locale = null;
        if (args.length > 1) locale = args[1];

        // Test Create Request
        DemandServiceMLUClient dm = new
DemandServiceMLUClient();
        String requestId = dm.createRequest(serviceURL);

        // Test Get Request
        dm.getRequests(serviceURL, locale, requestId);

        System.out.println("Demand Service MLU tests
complete.");
    }

    /**
     * This method creates a request and it shows the
instruction  on how to
     * create the request in a language other than the system
language
     * @param serviceURL
     * @return
     * @throws Exception
     */
    private String createRequest(String serviceURL) throws
Exception {

    // Get web service
        DemandServiceStub stub = new DemandServiceStub(ctx,
serviceURL);

       // Add the Audit header for auditing information
        SOAPHeaderCreator.setAuditHeader(stub, "Submitted By:
TEST","PPM on " +
InetAddress.getLocalHost().getHostAddress(),"createRequest");
```

```
/
**************************************************************
*
* Note: If you want to create a request in a language other than
the System language, you must specify the preferred locale by
uncommenting the following line:
SOAPHeaderCreator.setUserLocaleHeader(stub, locale);

Then, make sure to pass the token values in the translated
language instead of the System language. This is because when
you set the LanguageLocale field in UserLocaleHeader to a
specific locale, PPM web service expects the token values in the
corresponding language if the translation for that value in the
preferred language exists in the system. If the translation
doesn't exist in that language, you must pass the values in the
System language.

Take the following scenario as an example:

o    You set the preferred locale to "de" and you want to set
the value for the REQ.DEPARTMENT_NAME token to 'Manufacturing'
(English.
o    The German translation for 'Manufacturing' exists in the
system.

In this scenario, you must specify it as 'Herstellung', which is
the German translated value of 'Manufacturing' and so forth for
other token values as well.
************************************************************/

        // Add UserLocaleHeader to set the language locale to
the specified locale
       // SOAPHeaderCreator.setUserLocaleHeader(stub, locale);

        // Construct a request object
        Request oRequest = Request.Factory.newInstance();
        oRequest.setRequestType("Bug");
        SimpleField[] fields = new SimpleField[2];

        // Set values for the fields of the request object

        // Set field 'Description'
       SimpleField field_A = SimpleField.Factory.newInstance();
        field_A.setToken("REQ.DESCRIPTION");
       field_A.setStringValue1Array(new String[] { "WebService
Test" });
        fields[0] = field_A;

        // Set field 'Department'
       SimpleField field_B = SimpleField.Factory.newInstance();
        field_B.setToken("REQ.DEPARTMENT_NAME");
        field_B.setStringValue1Array(new String[]{"Finance"});
        fields[1] = field_B;
```

```
        // Add all the fields to request object
        oRequest.setSimpleFieldsArray(fields);

    }

    /**
     * This method invokes getRequest operation with language
locale set to specified locale
     * @param serviceURL
     * @param language
     * @param requestId
     * @throws Exception
     */
    private void getRequests(String serviceURL, String
language, String requestId)
            throws Exception {

        // Set Identifier
        Identifier[] ids = new Identifier[1];
        Identifier reqId = Identifier.Factory.newInstance();
        reqId.setId(requestId);
        reqId.setServerURL(serviceURL);
        ids[0] = reqId;

        // Get web service
        DemandServiceStub stub = new DemandServiceStub(ctx,
serviceURL);
        // Add the UserLocaleHeader SOAP header
        SOAPHeaderCreator.setUserLocaleHeader(stub, language);

        // Construct message to send
        GetRequestsDocument inDoc =
GetRequestsDocument.Factory.newInstance();
        GetRequestsDocument.GetRequests getRequests =
inDoc.addNewGetRequests();
        getRequests.setRequestIdsArray(ids);

        // Invoke web service
        GetRequestsResponseDocument outDoc =
stub.getRequests(inDoc);

        // Process return message
        Request[] requests =
outDoc.getGetRequestsResponse().getReturnArray();
        System.out.println("getRequests Succeeded");
        System.out.println("Returned Request: " +
requests[0].getId());
    }
}
```

Errors and Exceptions

There are no special exceptions for the SOAPHeaderCreator class.

9 Web Service Security

Overview

Authentication

PPM Center Web services uses the Web services Security specification (WS-Security) to secure SOAP message exchanges. PPM Center Web services relies on a Rampart module integrated with Axis2 Web service engine to provide WS-Security support.

For more information about WS-Security specification, go to the following site:

http://www.oasis-open.org/specs/index.php#wssv1.1

The WS-Security specification defines a set of standard SOAP headers to provide quality of protection through the following mechanisms:

- Message integrity (XML signature)
- Message confidentiality (XML encryption)
- Single message authentication (User name token authentication, Kerberos authentication, X509 certificate authentication, and so forth.)

These mechanisms can be used to accommodate a wide variety of security models. The WS-Security specification is considered a message-level authentication protocol because all security information is carried within the SOAP message.

Out of the box, PPM Center supports WS-Security user name token authentication, timestamp validation, and encryption of WS-Security headers. PPM Center also supports HTTP basic authentication (HTTP transport-level authentication protocol), as well as HTTPS (secure) authentication.

PPM Center Web services can also be integrated with third-party single sign-on software such as SiteMinder.

Authorization

PPM Center Web services follows the same authorization model as Web applications. Refer to the *Security Model Guide and Reference* for details on specific functional areas. This chapter focuses only on authentication.

Web Service Security on PPM Server

This section describes the *WS-Security Authentication* and the *HTTP Basic Authentication*.

WS-Security Authentication

WS-Security authentication includes user name token authentication, timestamp validation, and encryption of WS-Security headers. This section provides you with several examples of authentication configuration.

Enable/Disable WS-Security Authentication

WS-Security user name token configuration can be found in:
`<PPM_Home>/server/<PPM_Server_Name>/deploy/itg.war/WEB-INF/conf/axis2.xml`.

.

Variable	Meaning
<PPM_Home>	Represents the path where your PPM Center instance is installed. For example: xyzserver/E/PPMServer.
<PPM_Server_Name>	Represents the name assigned to your PPM Server during installation. For example: xyzProduction. This corresponds to the KINTANA_SERVER_NAME server.conf parameter value and does not necessarily reflect the actual host name of the server.

By default, WS-Security authentication is enabled.

axis2.xml

In the axis2.xml file, the following XML configuration enables WS-Security authentication. To disable WS-Security authentication, comment out all of this XML.

```
<module ref="rampart" />
<parameter name="InflowSecurity">
    <action>
    <items>
        UsernameToken Timestamp Encrypt
        </items>
        <passwordCallbackClass>

com.mercury.itg.ws.core.handlers.security.PasswordCallbackHandl
er
        </passwordCallbackClass>
        <decryptionPropFile>
            service.properties
        </decryptionPropFile>
    </action>
</parameter>
```

In the InflowSecurity section, the following three action items are defined:

- UsernameToken: Specifies that the UsernameToken security credentials should be expected in received SOAP messages.

  The UsernameToken profile defines a set of SOAP headers to carry the username/password from the client to the server.

- Timestamp: Specifies that the Timestamp element should be validated.

  The default clock skew tolerance is five minutes.

- Encrypt: Specifies that encrypted messages should be decrypted.

SOAP Header Without Timestamp Data or Encryption

The following example shows a SOAP header that does not include timestamp data or encryption:

```
<soap:Header>
    <wsse:Security    xmlns:wsse="http://docs.oasis-open.org/
wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd"
        soap:mustUnderstand="1">
        <wsse:UsernameToken
            xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/
oasis-200401-wss-wssecurity-utility-1.0.xsd"
            wsu:Id="UsernameToken-25699763">
            <wsse:Username>admin</wsse:Username>
            <wsse:Password
                Type="http://docs.oasis-open.org/wss/2004/01/
oasis-200401-wss-username-token-profile-
1.0#PasswordText">admin</wsse:Password>
        </wsse:UsernameToken>
    </wsse:Security>
</soap:Header>
```

Refer to the Axis2 Rampart module's documentation for more information:

http://ws.apache.org/axis2/modules/rampart/1_0/security-module.html

WS-Security Timestamp

By default, PPM Center enables Timestamp validation.

To disable Timestamp validation on the server side, remove “Timestamp” from the action items list as shown in the following example:

```
<module ref="rampart" />
<parameter name="InflowSecurity">
    <action>
    <items>
        UsernameToken Encrypt
        </items>
        <passwordCallbackClass>
```

```
com.mercury.itg.ws.core.handlers.security.PasswordCallbackHandl
er
        </passwordCallbackClass>
        <decryptionPropFile>
            service.properties
        </decryptionPropFile>
    </action>
</parameter>
```

WS-Security Encryption

By default, PPM Center encrypts WS-Security user name token headers.

To disable encryption, remove “Encrypt” from the action items list as shown in the following example:

```
<module ref="rampart" />
<parameter name="InflowSecurity">
    <action>
    <items>
        UsernameToken Timestamp
        </items>
        <passwordCallbackClass>

com.mercury.itg.ws.core.handlers.security.PasswordCallbackHandl
er
        </passwordCallbackClass>
        <decryptionPropFile>
            service.properties
        </decryptionPropFile>
    </action>
</parameter>
```

The following content shows an example of the service.properties file:

```
org.apache.ws.security.crypto.provider=org.apache.ws.security.c
omponents.crypto.Merlin
org.apache.ws.security.crypto.merlin.keystore.type=jks
org.apache.ws.security.crypto.merlin.keystore.password=ppmservi
ce
org.apache.ws.security.crypto.merlin.file=service.jks
```

In this example, the java key store file is defined as the security properties file, and the password to the key store file is ppmservice.

HTTP Basic Authentication

In a scenario where it is not convenient to use WS-Security, it is possible to configure PPM Center to accept user credentials passed by using HTTP basic authentication headers.

1. Open the axis2.xml file.
2. Locate “InflowBasicAuth” section.
3. Change the value of “Enforced” to true:
```
<parameter name="InFlowBasicAuth">
     <Enforced>true</Enforced>
</parameter>
```
4. Save and close the axis2.xml file.

When HTTP basic authentication is enabled on the PPM Server, the credential carried in HTTP authentication header is authenticated against PPM Center users' credentials.

The following example shows an http header using the HTTP basic authentication:

```
POST /itg/ppmservices/ProjectService HTTP/1.1
User-Agent: Crosscheck Networks SOAPSonar
Content-Type: text/xml; charset=utf-8
SOAPAction: "urn:GetProjectDetails"
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:8088
Content-Length: 542
Expect: 100-continue
Connection: Keep-Alive
```

Web Service Authentication for Web Service Toolkit

In the Web service toolkit, the WS-Security configuration is defined in `<Webservice_toolkit>/java/conf/`**`client_axis2.xml`**.

Specify a User Through Configuration File

By default, as a PPM Center server the PPM Center Web service toolkit enables UsernameToken, Timestamp, and Encrypt.

Set Headers

In the client_axis2.xml file, the following “OutflowSecurity” section defines how WS-Security headers are set on an outgoing SOAP message generated by the toolkit.

```
<module ref="rampart" />
<parameter name="OutflowSecurity">
    <action>
        <items>UsernameToken Encrypt Timestamp</items>
        <user>admin</user>
        <passwordCallbackClass>
             examples.security.PasswordCallbackHandler
        </passwordCallbackClass>
        <passwordType>
             PasswordText
        </passwordType>
        <encryptionParts>{Element}{http://docs.oasis-open.org/
wss/2004/01/oasis-200401-wss-wssecurity-secext-
1.0.xsd}UsernameToken</encryptionParts>
        <encryptionUser>ppmservice</encryptionUser>
        <encryptionPropFile>
             client.properties
        </encryptionPropFile>
     </action>
 </parameter>
```

To specify a fixed user name for the web service call, simply change the value of the <user> element in the client_axis2.xml file.

Set Password

To set the password for this particular user, following these steps:

1. Open `<webservice_toolkit>/java/ client/src/examples/security/`**`PasswordCallbackHandler.java`**
2. Change the default password “admin” to the desired password.

Example:

```
public class PasswordCallbackHandler implements CallbackHandler
{

String username = null;

    public void handle(Callback[] callbacks) throws
IOException,
            UnsupportedCallbackException {

        for (int i = 0; i < callbacks.length; i++) {
            WSPasswordCallback callback =
               (WSPasswordCallback)callbacks[i];

            // obtain password. This can be customized to obtain
            // password from any desire source and apply any
            // necessary algorithm.
            //
            // if your logic requires the username, you can get
the
            // user name by:
            //    String username = callback.getIdentifier();
            //
            String password = "admin";


            // set the obtained password
            callback.setPassword(password);
        }
    }
}
```

Specify User

The following code describes how to specify the user through a configuration file.

```
String repositoryPath =
System.getProperty("client.repository.dir");
        String axis2 = repositoryPath + "/conf/client-
axis2.xml";
        File file = new File(axis2);
        if (file.exists()) {
            try {
                ctx = ConfigurationContextFactory

.createConfigurationContextFromFileSystem(
                                    repositoryPath, axis2);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
```

HTTP Basic Authentication

The PPM Center Web service client can be programmed so that instead of WS-Security headers, you can use HTTP basic authentication headers to pass the user credential.

The following examples illustrate how to pass the user credential by using HTTP basic authentication headers:

```
public static void setHttpBasicAuthHeader(Stub stub, String
username, String password) {
       Options options = stub._getServiceClient().getOptions();
        if (options == null) return;
        HttpTransportProperties.Authenticator auth = new
            HttpTransportProperties.Authenticator();
        auth.setUsername(username);
        auth.setPassword(password);
        auth.setPreemptiveAuthentication(true);
        options.setProperty(HTTPConstants.AUTHENTICATE, auth);
        stub._getServiceClient().setOptions(options);
    }
```

NTLM Authentication

The PPM Center Web service client can also be programmed to pass NTLM credentials.

The following example illustrates how to pass NTLM credentials:

```
public static void setNTLMAuthHeader(Stub stub,
            String domain,
            String username,
            String password,
            String host) {
       Options options = stub._getServiceClient().getOptions();
        if (options == null) return;
        HttpTransportProperties.Authenticator auth = new
            HttpTransportProperties.Authenticator();
        List schemes = new ArrayList();

schemes.add(HttpTransportProperties.Authenticator.NTLM);
auth.setAuthSchemes(schemes);
auth.setPreemptiveAuthentication(true);
        auth.setUsername(username);
        auth.setPassword(password);
        auth.setDomain(domain);
        auth.setHost(host);
    options.setProperty(HTTPConstants.AUTHENTICATE, auth);
```

```
        stub._getServiceClient().setOptions(options);
    }
```

Configure Web Service Client for HTTPS

To enable a Web service client for HTTPS, do the following:

1. Get the certification authority (CA) root certificate used by the Web server with which PPM Center is integrated.
2. Perform one of the following options to import the certificate into the key store.
   - Create a custom key store at the client side to import the CA root certificate into it as trusted CA certificate, and specify the key store for the Web service client.
   - Or, directly import the CA certificate into the JDK's default key store. In this option, you do not need to specify the key store for the Web service client.

Custom Key Store

After you obtain the CA root certificate used by the web server with which PPM Center is integrated, you can run the following command to import the certificate into your new key store.

```
keytool -import  -keystore {keystore file} -alias {entryAlias}
-file {certfile}
```

Notify the Web service client of the location of the certificate by using system property “javax.net.ssl.trustStore” as shown in the following example:

```
java -Dclient.repository.dir=%WSCLIENT_HOME% -classpath %CPATH%
-Djavax.net.ssl.trustStore="C:/toolkit/java/conf/client.jks"
examples.pm.ProjectServiceClient https://localhost:8443/itg/
ppmservices/ProjectService "kevin8"
```

Or, you can use code inside your program as shown in the following example:

```
system.setProperty("javax.net.ssl.trustStore","full-path-of-
keystore-file" );
```

SSL-Client Authentication

If Secure Sockets Layer (SSL)-client authentication is required by the Web server, the javax.net.ssl.keyStore system property should be set to a key store file that contains the client's personal certificate.

Example:

```
java -Dclient.repository.dir=%WSCLIENT_HOME% -classpath %CPATH%
-Djavax.net.ssl.trustStore="C:/toolkit/java/conf/client.jks"
-Djavax.net.ssl.keyStore="C:/toolkit/java/conf/client.jks"
examples.pm.ProjectServiceClient https://localhost:8443/itg/
ppmservices/ProjectService "kevin8"
```

The personal key store and the trust key store can point to the same key store file or a different one.

JDK Default Key Store

Another place to keep the trusted CA certificate is in the JDK's default key store, which can be found at `$JRE_HOME/lib/security/cacerts`. The default password to the JDK key store is 'changeit'.

Use the following command to import the certificate into the JDK default key store:

```
keytool -import -trustcacerts -keystore {$JRE_HOME/lib/
security/cacerts} -alias {entryAlias} -file {certfile}
```

Make sure you are updating the right version of JDK on your machine if multiple JDKs are installed.

Java Virtual Machine (JVM) will load this key store when the program starts, so no additional Web service client configuration is needed if the CA certificate is loaded into the key store.

Web Service Single Sign-On

As with many other PPM Center Web components, PPM Center Web services is able to integrate with most industry-standard single sign-on (SSO) systems such as CA SiteMinder, Oracle Identity Management, RSA Sign-On Manager, and IBM Tivoli Access Manager through pluggable authentication

frameworks. PPM Center provides a log-in module for SiteMinder. For other SSO systems, additional customization may be required.

You can integrate with SiteMinder using the PPM Center SiteMinder Log-in Module. When this authentication mode is used, PPM Center authenticates users to SiteMinder, and does not store user passwords in the PPM Center database.

PPM Center Server Configuration

From the PPM Center server side, you can add the following parameters into the server.conf file:

- To allow SiteMinder Login Module to be invoked for Web service user authentication, set the following parameter:

  com.kintana.core.server.ENABLE_WEBSERVICE_SSO=true

- To choose SiteMinder to be the authentication mode, set the following parameter:

  com.kintana.core.server.authethentication_mode=SiteMinder
  Note: If SiteMinder is chosen as the only authentication mode, any individual user's authentication mode that was set through the workbench user page would be overwritten by this mode.

- To make PPM Center Web application use single sign-on mode, set the following parameter:

  com.kintana.core.server.SINGLE_SIGN_ON_
  PLUGIN=com.kintana.sc.security.auth.SiteMinderSingleSignOn.

Integration with a Client-Side Log-In Module

To complement the integration with client-side log-in module, follow these steps:

1. Develop a Java Authentication and Authorization Service (JAAS) log-in module that authenticates with the SSO system and receives an SSO token. The token could be set as a private credential in the Subject class.

   PPM Center has already provided such a module:

```
com.kintana.sc.security.auth.SiteMinderLoginModule
```

2. Create the JAAS configuration file under the `$WebServiceToolkit/java/conf` directory

   Example:

```
#authentication.conf
SiteMinder {
com.kintana.sc.security.auth.SiteMinderLoginModule required
debug=true;}
```

3. Specify the JAAS login configure system property in the command lines used to invoke the Web services in the compile_client.bat file

   Example:

```
java -Dclient.repository.dir=%WSCLIENT_HOME% -classpath
%CPATH%
-Djava.security.auth.login.config==%WSCLIENT_HOME%/conf/
authentication.conf
examples.pm.ProjectServiceClient https://localhost:8443/itg/
ppmservices/ProjectService "kevin8"
```

4. Add logic in the Web service client to invoke JAAS login

   Examples:

```
public Subject login() {
    LoginContext lc = null;
          lc = new LoginContext(
    " MyCustomModule ",
    myCallbackHandler
);
    lc.login();
    return lc.getSubject();
    }
```

5. Add the SSO token as cookie in the Web service client.

   Examples: Set a HTTP cookie in axis2 Web service client.

```
public void setSSOCookie(Stub stub, String ssoToken) {
         List headers = new ArrayList();

         //Set the required session variable for SSO system
        Header header = new Header(
             "Cookie",
             "SMSESSION=" + ssoToken
         );
        headers.add(header);

         ServiceClient client = stub._getServiceClient();
         Options option = client.getOptions();
```

```
            option.setProperty(HTTPConstants.HTTP_HEADERS,
headers);
        }
```

6. Call the corresponding method to set the SSO cookie after a stub is created.
7. Make the desired Web service request with the SSO cookie you set.

Working with Proxy Servers and Proxy Authentication

In some cases, the Web service client must go through a proxy server to reach the PPM Center server. In this case, a proxy server must be specified when you establish a connection. Otherwise, a connection time out error or no connection exception will occur.

There are two solutions available (depending on whether you want all or some calls to go through the proxy):

- Configure proxy by using client_axis2.xml.
- Configure proxy through java code.

All Web Service Calls Go Through Proxy: Configure Proxy By Using client_axis2.xml

This configuration forces all Web service calls from the Web service toolkit to go through the proxy. This file exists in:
`<webservice_toolkit>/java/conf/`**`client-axis2.xml`**

Add following XML configuration:

```
<transportSender
name=""
class="org.apache.axis2.transport.http.CommonsHTTPTransportSend
er">
    <parameter name="PROTOCOL" locked="false">HTTP/1.1</
parameter>
<parameter
    name="PROXY"
    proxy_host="proxy_host_name"
    proxy_port="proxy_host_port"
        locked="true>
                userName:domain:password
    </parameter>
</transportSender>
```

If authentication is not available, fill

“userName:domain:password” as “anonymous:anonymous:anonymous.”

Some Web Service Calls Go Through Proxy: Configure Proxy By Using Java Code

If only a selected number of operations in the toolkit should go through the proxy, choose this solution.

Add the following method in the client code:

```
public void setProxy(Stub stub, String proxyHost, int
proxyPort) {
      // get options
      Options options = stub._getServiceClient().getOptions();
      if (options == null) {
         options = new Options();
         stub._getServiceClient().setOptions(options);
      }

     HttpTransportProperties.ProxyProperties proxyProperties =
                      new
HttpTransportProperties.ProxyProperties();
      proxyProperties.setProxyName(proxyHost);
      proxyProperties.setProxyPort(proxyPort);
      options.setProperty(HTTPConstants.PROXY,
proxyProperties);

   }
```

Call this method before you invoke a Web service operation. For example:

```
DemandServiceStub stub = new DemandServiceStub(ctx,
serviceURL);
setProxy(stub, "proxy.hp.com", 8888);
…
```