HP Operations Orchestration Software

Software Version: 7.50

HP Service Desk Integration Guide

Document Release Date: March 2009

Software Release Date: March 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

All marks mentioned in this document are the property of their respective owners.

Finding or updating documentation on the Web

Documentation enhancements are a continual project at Hewlett-Packard Software. You can obtain or update the HP OO documentation set and tutorials at any time from the HP Software Product Manuals web site. You will need an HP Passport to log in to the web site.

To obtain HP OO documentation and tutorials

1. Go to the HP Software Product Manuals web site (*http://support.openview.hp.com/selfsolve/manuals*).
2. Log in with your HP Passport user name and password.

   OR

   If you do not have an HP Passport, click **New users – please register** to create an HP Passport, then return to this page and log in.

   If you need help getting an HP Passport, see your HP OO contact.
3. In the **Product** list box, scroll down to and select **Operations Orchestration**.
4. In the **Product Version** list, click the version of the manuals that you're interested in.
5. In the **Operating System** list, click the relevant operating system.
6. Click the **Search** button.
7. In the **Results** list, click the link for the file that you want.

Where to Find Help, Tutorials, and More

The HP Operations Orchestration software (HP OO) documentation set is made up of the following:

- Help for Central

  Central Help provides information to the following:

  - Finding and running flows
  - For HP OO administrators, configuring the functioning of HP OO
  - Generating and viewing the information available from the outcomes of flow runs

  The Central Help system is also available as a PDF document in the HP OO home directory, in the \Central\docs subdirectory.
- Help for Studio

  Studio Help instructs flow authors at varying levels of programming ability.

  The Studio Help system is also available as a PDF document in the HP OO home directory, in the \Studio\docs subdirectory.
- Animated tutorials for Central and Studio

  HP OO tutorials can each be completed in less than half an hour and provide basic instruction on the following:

  - In Central, finding, running, and viewing information from flows
  - In Studio, modifying flows

  The tutorials are available in the Central and Studio subdirectories of the HP OO home directory.

- Self-documentation for operations and flows in the Accelerator Packs and ITIL folders

  Self-documentation is available in the descriptions of the operations and steps that are included in the flows.

Support

For support information, including patches, troubleshooting aids, support contract management, product manuals and more, visit the following site:

- *http://support.openview.hp.com*

Table of Contents

Overview

This guide walks you through the process of integrating Hewlett-Packard Operations Orchestration (HP OO) with HP Service Desk.

Requirements for integration

Integrating HP OO with HP Service Desk requires the following software:

- HP Service Desk 4.5 client software on the JRAS
- HP OO Central and Studio 7.50

Background

HP Service Desk has one readily usable integration point in the form of the Java API provided as a Java Archive (.jar) file. This will require the use of a Java compiler. The other potential integration point is the Web user interface (by default on port 8080), but because this is often rebranded and customized in production environments, it is unlikely that any pre-built operations would work against this integration point.

Deployment

Versions supported

4.5, 5.0 (partial)

Deployment requirements

You must have the Service Desk Web API installed on a machine on which a Java-enabled Remote Action Service (RAS) is installed. The RAS that is shipped with HP OO 7.5 is enabled for both Java and .NET.

The following files should be present in the RAS \lib folder (for instance, RAS\Java\repository\lib):

- For version 4.5, web-api.jar

  OR

  For version 5.0:

  - sd-webapi.jar
  - xpl.jar
  - OvObsSDK.jar
  - OvObsWebApi-Client.jar
  - OvObsWebApi-Common.jar
  - OvObsCommon.jar

A Java compiler is required to compile these together to form a new .jar file that works with this particular installation. The version of the Java compiler that you use must be compatible with that of the Service Desk installation. The safest Java compiler version to use is the last release of 1.3. Some Service Desk installations may be running on Java 1.4, in which case it is safe to use the appropriate 1.4 compiler.

Creating a custom web-api.jar

The web-api/sd-webapi .jar files must match the patch level and configuration of the Service Desk application server. When the forms and/or schemas are changed, Service Desk becomes incompatible with the default API .jars and needs a new one to be generated. If this .jar has already been generated on the system that you are going to integrate, it is highly recommended to use it. If modifications have been made to the default schema and forms but a web-api.jar or sd-webapi.jar file has not been created, create the web-api.jar or sd-webapi.jar file before attempting the integration.

The following instructions assume that Service Desk 4.5 is installed. Service Desk 5.0 includes an Ant xml build file and some documentation on how to create .jars that work with that integration.

You will need:

- HP Service Desk client application (patched to level of Service Desk server)
- Original web-api.jar (also should be patched during patches)
- Eclipse (or another IDE for compiling Java)
- Java 1.3 SDK

To perform this integration

1. Log into the console as an administrator and choose the **Tools > System** menu option.
2. From the new window, go to **File > Generate Web API**.

   This does not produce a .jar but .Java files that must be compiled and repackaged into a new .jar.
3. Compile and repackage the java files into a new .jar with .class files from the original .jar file that was distributed with the client installation.
4. Start Eclipse and create a new project.
5. Make sure the selected compiler is Java 1.3.

   Using a 1.3 compiler avoids potential marshalling problems, because the SD server runs a 1.3 JRE by default.
6. Copy the web-api.jar from SD and add it to the Eclipse environment.
7. Add the web-api.jar to the project build path.
8. Add all the generated .Java files from Service Desk to the Eclipse project.

   This should automatically compile it all via Eclipse's automatic compilation system.
9. Open the web-api.jar file in WinZip or any other archive manager and delete everything under **com/hp/itsm/api**.
10. Copy the generated .class files from the Eclipse project into the archive under **com/hp/itsm/api**. You should see the following:
    - The **impl** directory
    - The **interfaces** directory
    - ApiEntityMap.class
    - ApiSDSession.class
11. Save the archive, copy it to the HP OO RAS's repository/lib directory.

12. Restart the RAS.

    Now you can run flows that use the basic Service Desk operations.

Usage

Basic concepts and syntax

The OO Service Desk integration operations are located in the \Library\Integrations\Hewlett-Packard\Service Desk\ folder in the OO Central repository.

A number of operations allow users to specify parameters (such as "Modify Object") to the call. These parameters allow users to use the Service Desk Java API by calling methods by their names or by specifying a number of criteria for searching (if applicable). You add extra parameters by adding inputs onto the operations. Note that you cannot simply add new parameters into a flow - you must modify the operation at the step level.

Property specifications use a simple dot object notation familiar to programmers. Here's a sample of a caller's name property from a service call.

Input Name	Value
Caller.Name	*null / nothing*

This will find the name of the caller. It will call the following equivalent Java code:

```
IServicecall servicecall = IServicecallHome.open(servicecallID);
IPerson caller = servicecall.getCaller();
String name = caller.getName();
```

You may not refer to object properties by their form labels.

For example, the following reference is bad – do not use this sort of reference:

```
User/Customer.Name
```

The following reference is okay.

```
Requester.Name
```

Multiple properties can be retrieved and even properties of those properties:

Input Name	Value
Requestor.Location.Remark	*null / nothing*
Caller.Name	*null / nothing*
Caller.Birthdate	*null / nothing*
Caller.Email	*null / nothing*

Corresponds roughly to:

```
IServicecall servicecall = IServicecallHome.open(servicecallID);
IPerson requestor = servicecall.getRequestor();
ILocation location = requestor.getLocation();
String remark = location.getRemark();
IPerson caller = servicecall.getCaller();
// save name in output
```

```
String name = caller.getName();
// save caller's birthdate in output as string representation
// Note: dates in Service Desk are stored as double
// use the Convert Date operation to convert to and from Service Desk's
// system to the date format of your choice
Double birthdate = caller.getBirthdate();
// save caller's e-mail in output
String email = caller.getEmail();
```

This specification is different when *setting* values in an object.

Because it is preferable not to create new objects in a set operation, we use the "where" methods of objects to search for existing objects to use as references. More specifically, any HP Service Desk object that will be assigned into another object must be created prior to executing the set operation. For example, if you want to assign a caller to a service call, the person must exist before we execute the operation to assign that person as the caller. This may require you to use additional Create operations prior to your flow.

To set a complex (non-Java) property, the most you can do, even as a developer is to find an existing object and set the property to a reference of that object. This means that you can only set basic Java values to your objects and that you must specify the full path to the object property each time.

Refer to the HP Service Desk Javadoc API documentation for more information on the properties and behaviors that are available per object. This will differ from schema to schema (if modified under 5.0, for example), so the documentation might be invalid for the installation. The **Get Object Properties** operation in the \Library\Integrations\Hewlett-Packard\Service Desk\Utilities\ folder is provided to help you retrieve the properties available to a particular object and tell you which ones are required.

Examples

Example 1: Retrieving a Service Call

We add the following extra inputs to the **Get Servicecall** operation:

```
Status.State.IsClosed : ""
Description : ""
Solution : ""
Caller.Name: “”
```

Returned data looks like the following:

```
{
"Status.State.IsClosed" : "true",
"Description" : "Workstation go boom",
"Solution" : "Deferred",
“Caller.Name” : “Robert, Paulson”
}
```

Note that non-existent properties or properties that are not set are returned as blank strings.

Example 2: Creating / modifying a Service Call:

Setting data on an object is different from retrieving it although it may look similar. In this example, the user creates “where” objects and finds them to set them as parameters to set (or add, in the case of some arrays) calls:

```
{
"Caller.Name" : "Paulson, Robert",
"Impact.Text" : "Low",
"Description" : "My computer is on fire.",
"Workaround" : "Don't look at it, it will go away."
}
```

The first word in the text is the "set" method to call on the object, and the rest is the “where” clause criterion.

This roughly calls the Java equivalent of the following code (assuming sd_object is "Servicecall"):

```
// serviceCall is reference to a specified Servicecall
IPersonHome personHome = session.getPersonHome();
IPersonWhere personWhere = personHome.createPersonWhere();
personWhere.addContainCriteriumOnName("Paulson, Robert");
IPerson persons[] = personHome.findPerson(personWhere);
serviceCall.setCaller(persons[0]);
IImpactHome impactHome = session.getImpactHome();
IImpactWhere impactWhere = impactHome.createImpactWhere();
impactWhere.addContainCriteriumOnText("Low");
IImpact impacts[] = impactHome.findImpact(impactWhere);
serviceCall.setImpact(impacts[0]);
serviceCall.setDescription("My computer is on fire");
serviceCall.setWorkaround(“Don't look at it, it will go away.");
serviceCall.save();
```

Alternative Integration Method

If you are familiar with Java and would rather write code to integrate HP OO operations with HP Service Desk, you can use the IAction SDK and not use any of the provided operations. Doing so entails linking to the .jar file of the RAS SDK while developing your operations. Using this method, you still need to create a web-api.jar file.

If the integration you are attempting exceeds the scope of the out-of-box operations (see “Limitations,” below) you will need to write your own Java-enabled RAS operation. See the IAction SDK documentation for a procedure for doing so.

Lastly, if you are using Service Desk 5.0, currently the only HP OO operations that will work are the deprecated operations, which only work for a stock configuration. Most likely, you will need to use the alternative method.

Limitations

Get and set operations and searching are the only supported operations. HP Service Desk has a class of operations prefixed with “unrelate” that remove references between objects. This is the equivalent of removing a pointer from a list – the original value is not destroyed. An “unrelated” operation is not supported and is not planned at the moment. Therefore, if a get / set operation you use needs to unrelate objects, then you will need to create the operation yourself via Java by writing your own custom IAction and putting it on the RAS.

Views and templates are not supported, because in almost all scenarios, the integration will be done with a system administrator account that has access to all objects, thus removing the need for views.

You may encounter problems related to the handling of arrays. If you try to retrieve an array, for example, you will not get the contents of the array but a textual (Java class) representation of the array object.

At present, you cannot set an array of values unless an array of matching object types can be returned by a where query for that object type / class. This can only be truly handled by an intermediate object format like JSON or XML, but it would need to be heavily customized and would result in a slow implementation of an ORM. This is because of HP OO’s enforcement of type checking.

The API may be tedious while looking for:

- The mapping of SD fields to properties.
- What is and isn’t required.

The **Get Object Properties** and **List Objects** utility operations may be helpful in overcoming this tendency.

Hidden Inputs

delimiter

A number of operations allow you to specify the delimiter in list outputs. This is helpful if you get back serialized results that are incorrectly formatted due to the use of the default comma.

outFormat

You can change the output date format of the **Convert Date** operation.

To change the output date format of the Convert Date operation

- Add the input outFormat and specify the output format.

The default output date format is “EEE MMM dd HH:mm:ss zzz yyyy”

The recommended method is to use the convert date operation in a flow.

Troubleshooting Tips

Error exception - No matching objects found for subentity "foo" or couldn't find a matching object under it

This may happen when there is an attempt to assign an invalid item to a property. For example, you might try to input a ClosureCode of "Invalid" when no such code exists in the schema.

This error can also occur on an attempt to specify an invalid property name.

Error exception – "foo" does not seem to support find operation

This error can result from an attempt to access a property or SD object that cannot be found with a findAll*Foo* method.

Error exception – HP Service Desk object "foo" not found

This can occur if you try to access a property or SD object that cannot be instantiated (e.g., static objects, meta-data, etc.) or does not exist.