HP Operations Orchestration Software

Software Version: 9.00.03

HP Universal Configuration Management Database Integration Guide

Document Release Date: November 2010

Software Release Date: November 2010

Legal Notices

Warranty

Restricted Rights Legend

Copyright Notices

Trademark Notices

For information on open-source and third-party software acknowledgements, see in the documentation set for this release, Open-Source and Third-Party Software Acknowledgements (3rdPartyOpenNotices.pdf).

On the Web: Finding OO support and documentation

There are two Web sites where you can find support and documentation, including updates to OO Help systems, guides, and tutorials:

- The OO Support site
- HP Live Network

Support

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.

HP Live Network

For support information, including patches, troubleshooting aids, support contract management, product manuals and more, visit the following site: *https://www.www2.hp.com/*.

This is the **HP Live Network** Web page. To sign in:

1. Click **Login**.
2. On the **HP Passport sign-in** page, enter your HP Passport user ID and password and then click **Sign-in**.
3. If you do not already have an HP Passport account, do the following:
   a. On the **HP Passport sign-in** page, click **New user registration**.
   b. On the **HP Passport new user registration** page, enter the required information and then click **Continue**.
   c. On the confirmation page that opens, check your information and then click **Register**.
   d. On the **Terms of Service** page, read the Terms of use and legal restrictions, select the **Agree** button, and then click **Submit**.

4. On the **HP Live Network** page, click **Operations Orchestration Community.**

   The Operations Orchestration Community page contains links to announcements, discussions, downloads, documentation, help, and support.

Note: Contact your OO contact if you have any difficulties with this process.

In OO: How to find Help, PDFs, and tutorials

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.

Table of Contents

Overview of the HP UCMDB integration

With the HP Universal Configuration Management Database (UCMDB) integration, administrators can build HP Operations Orchestration (OO) flows that are integrated with HP UCMDB.

This document explains how this integration has been implemented and how the integration OO operations communicate between OO and UCMDB.

Use cases and scenarios

1. Execute a Topology Query and obtain a Topology Map:
    - Get Topology Map by Query Name
    - Get Topology Map by Query Name with Parameter
    - Get Topology Map by View Name
    - Get Query Name of View
2. Manage Configuration Items and relationships:
    - Add Object
    - Delete Object
    - Add Relationship
    - Delete Relation
    - Get Neighboring Objects
    - Get Filtered Object List by Type
    - Get Object List by Type
    - Get Object Attributes by Id
    - Modify Object Attribute(s)
3. UCMDB IP validation:
    - Validate IP in UCMDB
    - Get Object List By Host Name
    - Is Object Related To

Installation and configuration instructions

There is no special integration installation and configuration needed for the HP UCMDB integration. Just make sure that the system that has the RSJRAS service running on it can access the UCMDB server.

The default UCMDB access URL is:

`http://<ucmdbserver>:8080/ucmdb`

using the default username **admin** and the default password **admin**. However, we recommend creating an integration user on the UCMDB server. To learn how to create an integration user, refer to the UCMDB documentation.

Versions

Operations Orchestration Version	HP Universal CMDB Version
9.00.03	7.0, 8.0, 9.0

Architecture

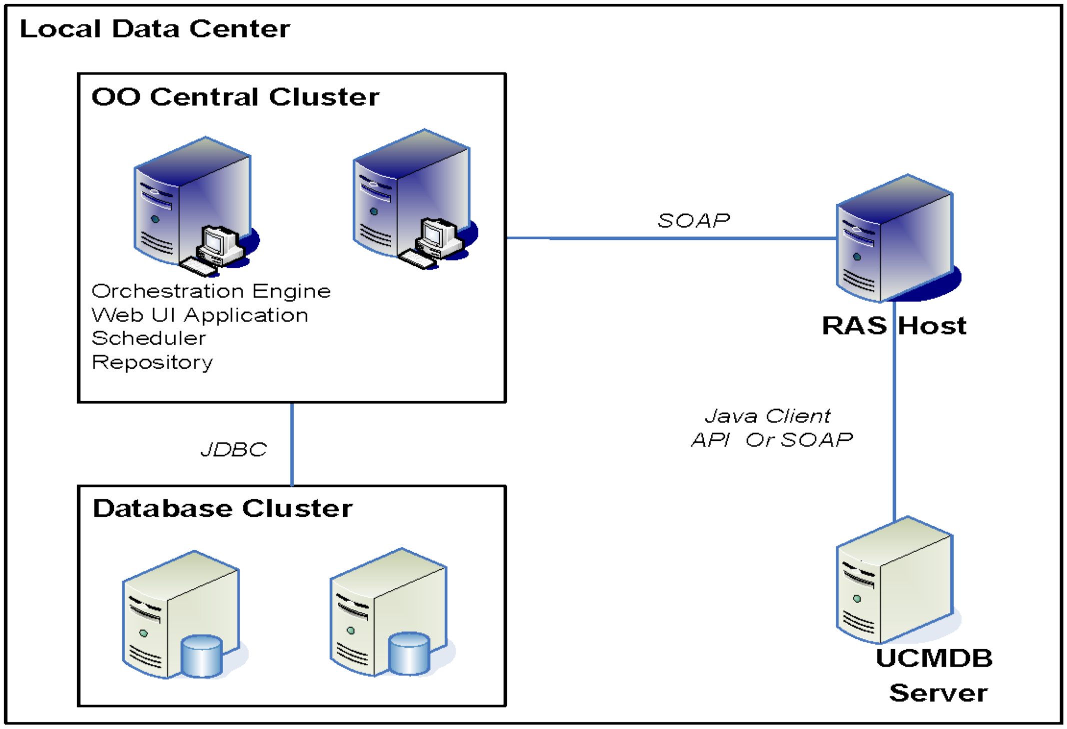
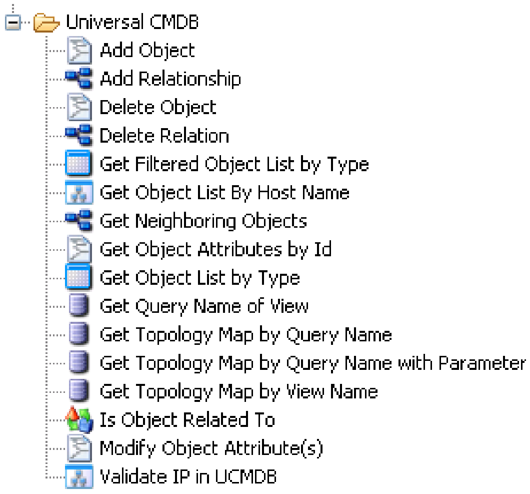


Figure 1 – Universal CMDB architecture

HP UCMDB Integration operation infrastructure

The HP UCMDB integration includes the following operations in the Studio Library, in the Library/Integrations/Hewlett-Packard/Universal CMDB/ folder.



Universal CMDB
- Add Object
- Add Relationship
- Delete Object
- Delete Relation
- Get Filtered Object List by Type
- Get Object List By Host Name
- Get Neighboring Objects
- Get Object Attributes by Id
- Get Object List by Type
- Get Query Name of View
- Get Topology Map by Query Name
- Get Topology Map by Query Name with Parameter
- Get Topology Map by View Name
- Is Object Related To
- Modify Object Attribute(s)
- Validate IP in UCMDB

Figure 2 - Universal CMDB flows and operations in Studio Library

The operations support the following types of tasks:

- Add, remove, and update Configuration Items in the UCMDB.
- Add, remove, and update Relations in the UCMDB.
- Retrieve information about Configuration Items and Relationships in the UCMDB.
- Validate an IP or a host name in the UCMDB.

Common inputs in the integration

The following inputs are used consistently throughout the HP UCMDB integration's operations.

cmdbHost

The host of the UCMDB server. You can specify it as an IP address or DNS name.

cmdbPort

The port of the UCMDB server on which the CMDB is running. The default port is **8080**.

username

The username to connect to the UCMDB server.

password

The password for the username.

cmdbVersion

The major version number of UCMDB; for example, 7 for UCMDB 7.0, 8 for UCMDB 8.0, or 9 for UCMDB 9.0. The default is **7**.

Operation specifics

This section describes the UCMDB integration's operations and flows, including any operation- or flow-specific inputs.

Add Object

The **Add Object** operation adds a Configuration Item to the UCMDB that has the key property or other properties specified in the **objectType** input. It returns the ID of the created Configuration Item.

All of the operation's inputs except the following are described in *Common inputs in the integration*.

objectType

> The Configuration Item Type (CIT) of the UCMDB. For example, you can specify an **objectType** of **nt**, which is the class name for the Configuration Item Type with the display name **Windows**. You can find the class name in the CI Type Manager by clicking the **Details** tab for the CI item. The **Name** field shows the class name that should be used for the **objectType** input. **Note:** the class names can be different in UCMDB 7.0, 8.0, and 9.0.

prop

> A property for the **objectType** to add (*name*=*value*). For the **Add Object** operation, the key property for the type of CI to add must be entered if there are keys for that type of CI. For example, if you integrate with UCMDB 7.0 or 8.0, you must enter a CI with CIT **ip, prop ip_address=19.35.55.34**. The **ip_address** has to be unique in UCMDB. If you integrate with UCMDB 9.0, you must enter a CI with CIT **ip_address**, **prop name=19.35.55.34**. The type of the property is determined automatically. You can add additional properties by creating extra **prop** inputs, each with a number appended to it, such as **prop1**. For example, if you integrate with UCMDB 7.0 or 8.0, enter **prop1 ip_domain=battleground.ad** or **prop1 isvirtual=false**. If you integrate with UCMDB 9.0, enter **prop1 routing_domain**= **battleground.ad** or **isvirtual=false**. You can find the property names in the CI Type Manager by clicking the **Attributes** tab for the CI item. The **Name** field shows the attribute name that should be used for **prop** input.

The operation returns the following:

returnResult

> The ID of the created object.

Add Relationship

The **Add Relationship** operation adds a Relation between two Configuration Items to the UCMDB based on the type of relation specified in the **relationType** input. It returns the ID of the created relationship.

All of the operation's inputs except the following are described in *Common inputs in the integration*.

relationType

> The Configuration Item Type of the relation in the UCMDB. For example, if you integrate with UCMDB 7.0 or 8.0, **relationType contained** is the class name for the relation type with the display name **Contained**. If you integrate with UCMDB 9.0, **relationType containment** is the class name for the relation type with the display name **Containment**. You can find the

relationship class names in the CI Type Manager by clicking the **Details** tab for the Relationships item. The **Name** field shows the class name that should be used for **relationType** input.

fromId

The ID of the Configuration Item the relationship is from in the UCMDB.

fromType

The Configuration Item Type the relationship is from in the UCMDB.

toId

The ID of the Configuration Item the relationship is to in the UCMDB.

toType

The Configuration Item Type the relationship is to in the UCMDB.

prop

A property for **relationType** to add (*name*=*value*). For example, if you integrate with UCMDB 7.0 or 8.0, you can specify **display_label=My Contained** for **relationType contained.** If you integrate with UCMDB 9.0, you can specify **display_label=My Containment** for **relationType containment**. The type of the property is determined automatically. You can add additional properties by creating extra **prop** inputs, each with a number appended to it, such as **prop1**. You can find the property names in the CI Type Manager by clicking the **Attributes** tab for the Relationships item. The **Name** field shows the attribute name that should be used for **prop** input.

The operation returns the following:

returnResult

The ID of the created relation.

Delete Object

The **Delete Object** operation deletes a Configuration Item in the UCMDB based on the **objectId** and the **objectType** specified in the inputs.

All of the operation's inputs except the following are described in *Common inputs in the integration*.

objectId

The Configuration Item ID in the UCMDB.

objectType

The Configuration Item Type of the UCMDB. For example, you can specify an **objectType** of **nt** which is the class name for the CIT with the display name **Windows**. You can find the class name in the CI Type Manager by clicking the **Details** tab for the CI item. The **Name** field shows the class name that should be used for the **objectType** input. **Note:** The class names can be different in UCMDB 7.0, 8.0, and 9.0.

Delete Relation

The **Delete Relation** operation deletes a Relation between two Configuration Items in the UCMDB.

All of the operation's inputs except the following are described in *Common inputs in the integration*.

relationId

The Configuration Item ID for the Relation in the UCMDB.

relationType

The Configuration Item Type of the relation in the UCMDB. For example, if you integrate with UCMDB 7.0 or 8.0, **relationType contained** is the class name for the relation type with the display name **Contained**. If you integrate with UCMDB 9.0, **relationType containment** is the class name for the relation type with the display name **Containment**. You can find the relationship class names in the CI Type Manager by clicking the **Details** tab for the Relationships item. The **Name** field shows the class name that should be used for **relationType** input.

fromId

The ID of the Configuration Item the relationship is from in the UCMDB.

toId

The ID of Configuration Item the relationship is to in the UCMDB.

Get Filtered Object List by Type

The **Get Filtered Object List by Type** operation retrieves all of the Configuration Items that match the conditions specified in the UCMDB based on the **objectType**. It returns a list of the IDs of the Configuration Items.

All of the operation's inputs except the following are described in *Common inputs in the integration*.

objectType

The Configuration Item Type of the UCMDB. For example, you can specify an **objectType** of **nt** which is the class name for the CIT with the display name **Windows**. You can find the class name in the CI Type Manager by clicking the **Details** tab for the CI item. The **Name** field shows the class name that should be used for **objectType** input. **Note:** The class names can be different in UCMDB 7.0, 8.0 and 9.0.

logicalOperator

Specify **AND** if all conditions must be matched or **OR** if only one or more of conditions must match.

conditionAttributes

A comma-delimited list of the attribute names of the Configuration Item that the conditions test. For example, **ip_ismanaged, isvirtual** are the attributes for the Configuration Item Type **IP** (**ip** as class name) in UCMDB 7.0 and 8.0, or **IpAddress** (**ip_address** as class name) in UCMDB 9.0 with the display attribute names **IP Is Managed** and **Is Virtual**.

conditionOperators

A comma-delimited list of the comparisons to do, corresponding to the input **conditionAttributes**. For example, **==,!=,like**.

conditionValues

A comma-delimited list of the values to compare the property to, corresponding to the input **conditionOperators**. For example, **true,true**.

The operation returns the following:

CIs

The objects of that type that match the conditions.

Get Neighboring Objects

The **Get Neighboring Objects** operation retrieves all the related Configuration Items based on **objectId** and the **objectType** in the UCMDB. It returns all the related Configuration Item IDs, relation types, and a topology map in XML format of the related CIs.

All of the operation's inputs except the following are described in *Common inputs in the integration*.

objectId

The Configuration Item ID in the UCMDB.

objectType

The Configuration Item Type (CIT) of the UCMDB. For example, you can specify an **objectType** of **nt** which is the class name for the CIT with the display name **Windows**. You can find the class name in the CI Type Manager by clicking the **Details** tab for the CI item. The **Name** field shows the class name that should be used for **objectType** input. **Note:** The class names can be different in UCMDB 7.0, 8.0, and 9.0.

The operation returns the following:

objectIds

The IDs of the neighboring objects of that type.

objectTypes

The types of neighboring objects.

relationIds

The IDs of neighboring relations.

relationTypes

The types of neighboring relations.

topologyXML

The XML of the local topology (objects and types).

Get Object Attributes by Id

The **Get Object Attributes by Id** operation retrieves the attributes specified in **attributeList** for the Configuration Item (CI) based on the **objectId** and the **objectType** specified in the inputs in the UCMDB. It returns the CI attributes in the format of <*name*=*value*> list.

All of the operation's inputs except the following are described in *Common inputs in the integration*.

objectId

The Configuration Item ID in the UCMDB.

objectType

The Configuration Item Type of the UCMDB. For example, you can specify an **objectType** of **nt** which is the class name for the CIT with the display name **Windows**. You can find the class name in the CI Type Manager by clicking the **Details** tab for the CI item. The **Name** field shows the class name that should be used for **objectType** input. **Note:** The class names can be different in UCMDB 7.0, 8.0, and 9.0.

attributeList

A comma-delimited list of attributes to retrieve in the UCMDB. For example, **attributeList host_key,root_lastaccesstime** contains the attributes for the Configuration Item Type **Windows** (or **nt** as class name) with display attribute names **Host Key** and **Last Access Time**. If you specify **none** for **attributeList**, no attributes are retrieved. If you leave **attributeList** blank, all of the attributes of that CIT are retrieved. You can find the attribute names in the CI Type Manager by clicking the **Attributes** tab for the CI item. The **Name** field shows the attribute name that should be used for **attributeList** input.

The operation returns the following:

Attributes

The attributes and their values.

Get Object List By Host Name

The **Get Object List by Host Name** flow checks if the Configuration Item (CI) with **hostname** exists in UCMDB. If you integrate with UCMDB 7.0 or 8.0, it returns the IDs of the CIs in UCMDB with the attribute **host_hostname,host_key**, **host_dnsname**, or **display_label** that is equal to the input **hostname** and the CI type of the returned CIs. If you integrate with UCMDB 9.0, it returns the IDs of the CIs in UCMDB with **attribute name**, **primary_dns_name**, **host_key**, or **display_label** that is equal to the input **hostname** and the CI type of the returned CIs. The flow won't return all the CIs that have the same hostname but belong to different CI types. This flow is intended to be used as a subflow for the flow **Validate IP** in UCMDB.

All of the flow's inputs except the following are described in *Common inputs in the integration*.

hostname

The host name to check in UCMDB. For example, myserver.rose.hp.com or myserver.

cmdbHostTypeList

The list needed to include UCMDB CI types that belong to **host** CI types if you integrate with UCMDB 7.0 and 8.0, or **node** CI types if integrate with UCMDB 9.0. Start with the host or node type that is more specific. For example, nt,unix. If not specified, the following default list will be used: nt,unix,vmware_esx_server, cluster_resource_group, clusteredservice,terminalserver,vax,mainframe,lpar,host_node,node,host. node and cluster_resource_group are the class names for UCMDB 9.0. We recommended that you not leave this input empty.

The flow returns the following:

returnHostCIResult

The IDs of the CIs in UCMDB with the attribute **host_key** or **host_hostname**, or **display_label** that is equal to the input **hostname**.

returnHostCITypeResult

The CI type of the **returnHostCIResult**.

Get Object List by Type

The **Get Object List by Type** operation retrieves all the Configuration Items in the UCMDB based on the **objectType** specified in the input. It returns a list of CI IDs for the given Configuration Item Type.

All of the operation's inputs except the following are described in *Common inputs in the integration*.

objectType

The Configuration Item Type of the UCMDB. For example, you can specify an **objectType** of **nt** which is the class name for the CIT with the display name **Windows**. You can find the class name in the CI Type Manager by clicking the **Details** tab for the CI item. The **Name** field shows the class name that should be used for **objectType** input. **Note:** The class names can be different in UCMDB 7.0, 8.0, and 9.0.

The operation returns the following:

CIs

The objects of that type.

Get Query Name of View

The **Get Query Name of View** operation retrieves the query name in the UCMDB for the given **viewName**.

All of the operation's inputs except the following are described in *Common inputs in the integration*.

viewName

The view name in the UCMDB. For example, **NetworkTopoloy** is one of the default views in the UCMDB.

The operation returns the following:

returnResult

The name of the query.

Get Topology Map by Query Name

The **Get Topology Map by Query Name** operation retrieves the Topology Map in XML format for the given query name in the UCMDB.

All of the operation's inputs except the following are described in *Common inputs in the integration*.

queryName

The query name in the UCMDB. For example, **NetworkTopoloy** is one of the default queries in the UCMDB.

The operation returns the following:

topologyXML

The XML of the local topology (objects and types).

Get Topology Map by Query Name with Parameter

The **Get Topology Map by Query Name with Parameter** operation retrieves a topology for a query you set up with a parameterized value. For example, if want to restrict the query to list only **display_label=10.22.0.0** for the network node, create a query in UCMDB with the Network Configuration Item Type that has an attribute of **Display Label** with a parameterized value type.

All of the operation's inputs except the following are described in *Common inputs in the integration*.

queryName

The query to run. For example, you can set up a query called **NetworkTopoloyWithParameters** which allows filtering parameters in the UCMDB.

parameterName

The parameter name to filer on the attributes of in the UCMDB.

stringProp

The parameter to pass, which must be of the type **string**. For example, **display_label=10.22.0.0**. If you do not specify a value, the query returns everything without restriction. Additional properties can be added by adding extra **stringProp** inputs each with a sequential number appended to it, such as **stringProp1**. The same rule applies to the props of different data types.

floatProp

A parameter to pass of type **float**.

booleanProp

A parameter to pass of type **boolean**.

integer_listProp

A parameter to pass that is a comma-delimited list of integers.

longProp

A parameter to pass of type **long**.

xmlProp

A parameter to pass of type **xml**.

string_listProp

A parameter to pass that is a comma-delimited list of strings.

integerProp

A parameter to pass of type **int**.

dateProp

A parameter to pass of type **date**.

bytesProp

A parameter to pass, as a hex number.

doubleProp

A parameter to pass of type **double**.

parameterName

The name of the parameter to filter the attributes on. This is for internal use only. It doesn't affect the query.

The operation returns the following:

topologyXML

The XML of the local topology (objects and types).

Get Topology Map by View Name

The **Get Topology Map by View Name** operation retrieves the Topology Map in XML format for the specified view name from the UCMDB.

All of the operation's inputs except the following are described in *Common inputs in the integration*.

viewName

The view name in the UCMDB. For example, **NetworkTopoloy** is one of the default views in the UCMDB.

The operation returns the following:

topologyXML

The XML of the local topology (objects and types).

Is Object Related To

The **Is Object Related To** flow checks to see if one Configuration Item (CI) ID is related to another CI ID in UDMCB. It returns **Success** if they are related. This flow is intended to be used as a subflow for the flow **Validate IP** in UCMDB.

All of the flow's inputs except the following are described in *Common inputs in the integration*.

objectIdToRelate

The CI ID in UCMDB to check if it is related to the input **objectIdRelateTo**.

objectIdRelateTo

The CI ID in UCMDB to check if the input **objectIdToRelate** has a relationship with this id. This CI ID has to be from CI in the relationship in order to get neighboring objects.

objectTypeOfObjectIdRelateTo

The CI type in UCMDB for the input **objectIdRelateTo**. For example, you can specify an **objectType** of **nt or unix**, which is the class name for the CIT with the display name **Windows or Unix**.

Modify Object Attribute(s)

The **Modify Object Attribute(s)** operation modifies the attributes of the Configuration Item in the UCMDB based on the **objectId** and the **objectType** specified in the inputs.

All of the operation's inputs except the following are described in *Common inputs in the integration*.

objectId

The Configuration Item ID in the UCMDB.

objectType

The Configuration Item Type of the UCMDB. For example, you can specify an **objectType** of **nt** which is the class name for the Configuration Item Type with the display name **Windows**. You can find the class name in the CI Type Manager by clicking the **Details** tab for the CI item. The **Name** field shows the class name that should be used for **objectType** input. **Note:** The class names can be different in UCMDB 7.0, 8.0, and 9.0.

prop

A property for the **objectType** to modify (*name=value*). The type of the property is determined automatically. You can add additional properties by creating extra **prop** inputs, each with a number appended to it, such as **prop1**. For example, if you integrate with UCMDB 7.0 or 8.0, **prop ip_domain=battleground.ad** or **prop1 isvirtual=false**. If you integrate with UCMD 9.0 **prop routing_domain=battleground.ad** or **prop1 isvirtual=false**. You can find the attribute names in the CI Type Manager by clicking the **Attributes** tab for the CI item. The **Name** field shows the attribute name that should be used for **prop** input.

Validate IP in UCMDB

The **Validate IP in UCMDB** flow looks up the IP for **hostname** from DNS server. If it finds the IP, it checks if the Configuration Item (CI) with that IP exists in UCMDB. If the IP exists, it checks to see if the CI with **hostname** exists in UCMDB. If **hostname** exists it checks to see if the **host** CI is related to the **ip** CI. The operation returns the found host CI type, host CI ID, and ip CI ID.

All of the flow's inputs except the following are described in *Common inputs in the integration*.

hostname

The fully-qualified domain name. For example, myserver.rose.hp.com.

dnsServer

This is an optional input. Specifies the DNS server name or ip to look up the IP for the input **hostname**.

cmdbHostTypeList

A list that includes UCMDB CI types that belong to **host** CI types if you integrate with UCMDB 7.0 and 8.0, or **node** CI types if you integrate with UCMDB 9.0. Start with the host or node type that is more specific. For example, **nt,unix**. If not specified, the following default list is used: nt,unix,vmware_esx_server, cluster_resource_group, clusteredservice,terminalserver,vax,mainframe,lpar,host_node,node,host. node and cluster_resource_group are the class names for UCMDB 9.0. We recommend that you not leave this input empty.

The flow returns the following:

returnIpCIResult

The ID of the CI in UCMDB with an **ip_address** that is equal to the IP of the input hostname.

returnHostCITypeResult

The CI type of the host to which the **returnIpCIResult** is related.

returnHostCIResult

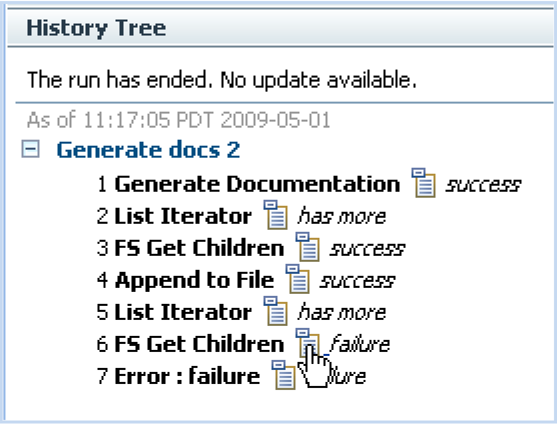
The ID of the host CI in UCMDB to which the **returnIpCIResult** is related.

Troubleshooting

General troubleshooting procedures and tools

If a run of a flow in OO Central fails

1. In Central, expand the **History Tree** of the flow execution and click the **expand details** () icon for a failed step.



History Tree

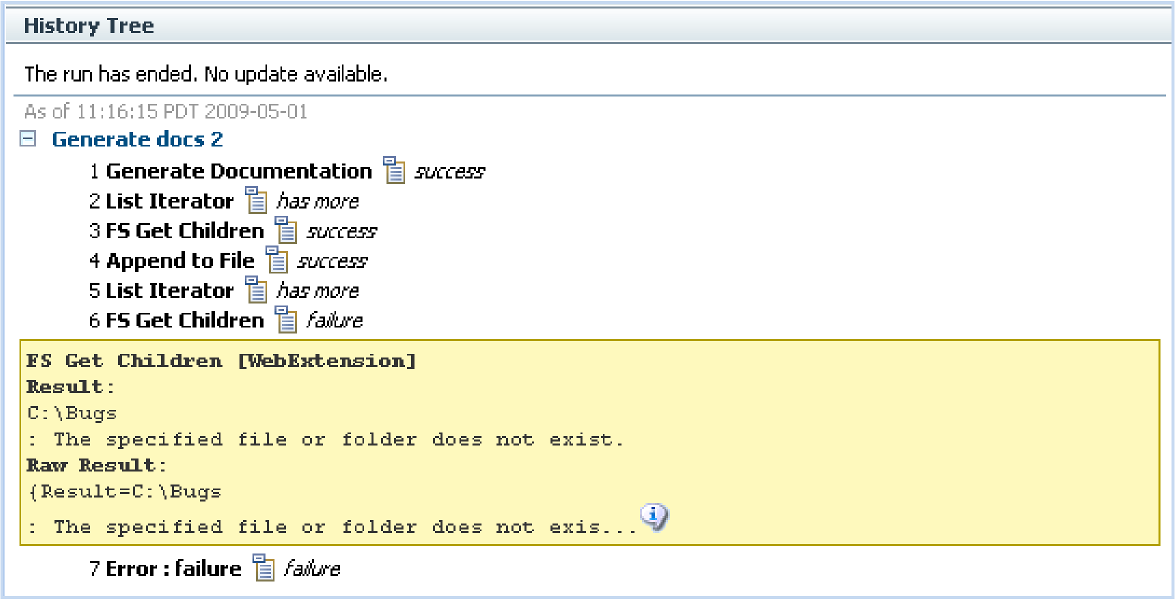
The run has ended. No update available.

As of 11:17:05 PDT 2009-05-01

⊟ Generate docs 2

1 **Generate Documentation** *success*
2 **List Iterator** *has more*
3 **FS Get Children** *success*
4 **Append to File** *success*
5 **List Iterator** *has more*
6 **FS Get Children** *failure*
7 **Error : failure** *failure*

A yellow text box opens to show the execution result of the failed step.



History Tree

The run has ended. No update available.

As of 11:16:15 PDT 2009-05-01

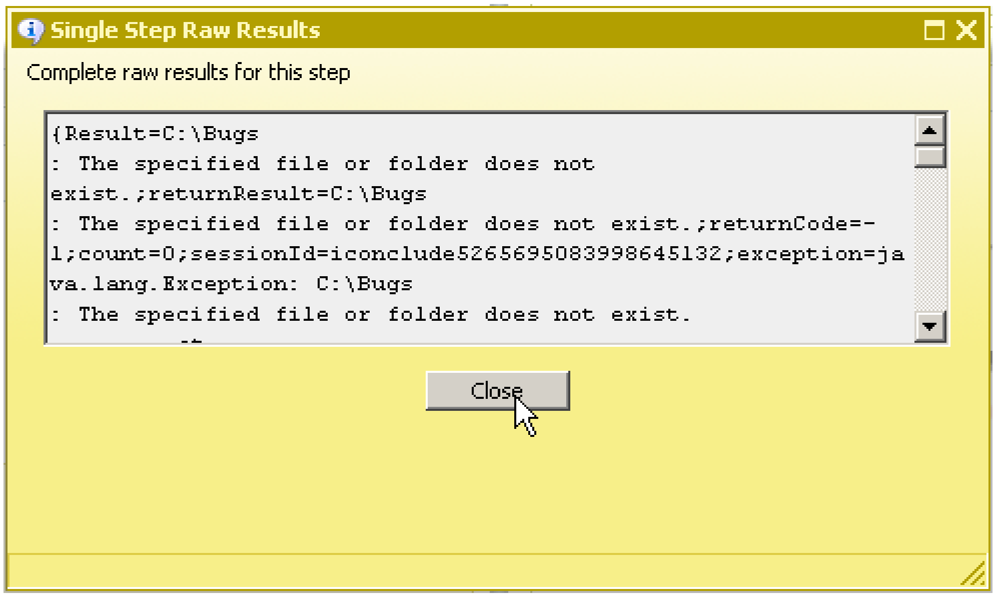
⊟ Generate docs 2

1 **Generate Documentation** *success*
2 **List Iterator** *has more*
3 **FS Get Children** *success*
4 **Append to File** *success*
5 **List Iterator** *has more*
6 **FS Get Children** *failure*

```
FS Get Children [WebExtension]
Result:
C:\Bugs
: The specified file or folder does not exist.
Raw Result:
{Result=C:\Bugs
: The specified file or folder does not exis...
```

7 **Error : failure** *failure*

2. Click the **more...** () icon to open the **Single Step Raw Results** window.



Complete raw results for this step

```
{Result=C:\Bugs
: The specified file or folder does not
exist.;returnResult=C:\Bugs
: The specified file or folder does not exist.;returnCode=-
1;count=0;sessionId=iconclude5265695083998645132;exception=ja
va.lang.Exception: C:\Bugs
: The specified file or folder does not exist.
```

If you run a flow in OO Studio using **Debug Flow in Central**, you can use the Step Result Inspector to check the errors and exceptions in detail.

Similar information can also be found in the OO Central log %OO_home%/Central/logs/Central_wrapper.log. For information about the RAS service, check the log %OO_home%/RAS/Java/Default/webapp/logs/wrapper.log.

Error messages

org.apache.axis2.AxisFault Transport error: 404 Error: Path+Not+Found

This message means that the logon information for UCMDB 7.0 may be incorrect. Check the **cmdbHost** and **cmdbPort** inputs to make sure that they are correct.

com.hp.ucmdb.api.CommunicationException: Got response code 404 for URL http://10.51.0.198:8080/ucmdb-api/connect?VERSION=8.0

This means that the logon information for UCMDB 8.0 may be incorrect. Check the **cmdbHost** and **cmdbPort** inputs to make sure that they are correct.

org.apache.axis2.AxisFault Transport error: 401 Error: Unauthorized

This means that the logon information for UCMDB 7.0 may be incorrect. Check the **username** and **password** inputs to make sure that they are correct.

Invalid credentials

This means that the logon information for UCMDB 8.0 may be incorrect. Check the **username** and **password** to make sure that they are correct.

No enum const class com.hp.oo.content.connectors.hp.ucmdb.AbstractUCMDBConnector$versions.v10

The **cmdbVersion** string is incorrect. The valid strings are **7**, **8**, and **9**.

class [Windows] not found in CMDB Class Model; nested exception is: java.lang.Exception:

The **objectType** string is not defined in UCMDB 7.0. Make sure the type is defined as the **class_name** in the **dbo.ccm_classes** table of <CMDB Schema>. In the UCMDB GUI, the Configuration Item Type (CIT) shows the type's display name. For example, for the CIT display name **Windows**, the actual class name is **nt**.

com.hp.ucmdb.api.classmodel.ClassDoesNotExistException: Class "Windows" is not defined in the uCMDB class model

The **objectType** string is not defined in UCMDB 8.0. Make sure the type is defined as the **class_name** in the **dbo.ccm_classes** table of <CMDB Schema>. In the UCMDB GUI, the Configuration Item Type (CIT) shows the type's display name. For example, for the CIT display name **Windows**, the actual class name is **nt**.

The received stringec928b8e1398f73b5011067b404088 representation is inconsistent; nested exception is: java.lang.Exception:

The **objectId** doesn't exist in UCMDB 7.0. Make sure that the ID is valid.

ID 5bf303466b8a982601dd3a9e88ad75 being restored has incorrect length (not 32 characters)

The **objectId** entered does not exist in UCMDB 8.0. Make sure that the ID is valid.

Object with ID: 5bf303466b8a982601dd3a9e88ad7557 was not found

The **objectId** entered does not exist in UCMDB 8.0. Make sure that the ID is valid.

Element Number: 1 doesn't exist in the result.; nested exception is: java.lang.Exception:

When you executed the operation **Get Neighboring Objects**, the **objectId** was not correct in UCMDB 7.0. Make sure that the ID is valid.

CMDB Operation Internal Error:... class name: Windows doesn't exist in class model

When you executed the operation **Get Neighboring Objects**, the **objectId** was not correct in UCMDB 7.0. Make sure that the ID is valid.

No Error Messages, but Got Empty Topology Map of Neighbors CIs.

When running the OO operation **Get Neighboring Objects**, the **objectId** was not correct in UCMDB 8.0. Make sure that the ID is valid.

Unable to determine the type of attribute: ip_isManaged because it is not a member of the class: ip

Make sure that you enter the **conditionAttributes** with valid attribute names. The attribute names can be found in the **attribute_name** in the **dbo.ccm_attributes** table of <CMDB Schema>.

Unknown Comparison type: pp

Make sure that you enter the **conditionOperators** with valid condition operators. Valid condition operators include **==**, **!=**, **<>**, **>**, and **<**.

java.lang.Exception: The specified condition type is not supported for: BooleanCondition, Supported Types are: EqualNotEqual

Make sure that the **conditionOperators** match with **conditionAttributes** type. For example, **conditionAttributes ip_ismanaged**, should have a **conditionOperator** of **==** or **!=**.

The lists of conditional types, names, operators and values are not of equal length

Make sure that the **conditionOperators**, **conditionAttributes**, and **conditionValues** are all the same length.

Unable to locate view badViewName

Make sure to enter a valid view name of the UCMDB in **viewName**.

Pattern [badQueryName] doesn't exist

Make sure to enter a valid query name of the UCMDB in **queryName**.

No Error Messages, but Got Empty Topology Map

If you get an empty topology map based on the **viewName** or **queryName**, the view or query might be empty in UCMDB.

Error occurred while trying to execute operation Data In - Add Data Strict : operation Data In - Add Data Strict….Couldn't append existing object:

When running the OO operation **Add Object** to create a Configuration Item, the Configuration Item is already in UCMDB 8.0. Make sure that the CI is not an existing CI.

the ID property [attribute name='ip_address'] doesn't exist !!!; nested exception is: java.lang.Exception:

When running the OO operation **Add Object** to create a Configuration Item (CI) in UCMDB 7.0, the **prop** field should be the ID attribute. For example, if you create a type **ip** CI, the ID attribute is **ip_address;** if you create a **host** CI, the ID attribute is **host_key**.

Error occurred while trying to execute operation Data In - Add Data Strict : operation Data In - Add Data Strict…. Can't achieve ID properties: not all the requested ID properties exist

When running the OO operation **Add Object** to create a Configuration Item (CI) in UCMDB 8.0, the **prop** field should be the ID attribute. For example, if you create a type **ip** CI, the ID attribute is **ip_address;** if you create a **host** CI, the ID attribute is **host_key**.

Error occurred while trying to execute operation Data In - Add Or Ignore Data : operation Data In - Add Or Ignore Data … Caused by: java.lang.NullPointerException

When running the OO operation **Add Relationship** to create a Relation in UCMDB 8.0, the relationship type might be not correct.

Security

UCMDB 7.0 servers are accessed via SOAP over HTTP (or HTTPS, if enabled on the host). UCMDB 8.0 servers are accessed via JAVA Client API. The UCMDB administrator provides logon credentials for connecting with the SOAP or API. The SOAP client or API client needs the username and password of an integration user defined in the UCMDB. These users do not represent actual users of the UCMDB, but rather applications that connect to the UCMDB.

Tools

Following are OO tools that you can use with the HP UCMDB integration:

- **RSFlowInvoke.exe** and **JRSFlowInvoke.jar**

  RSFlowInvoke (RSFlowInvoke.exe or the Java version, JRSFlowInvoke.jar) is a command-line utility that allows you to start a flow without using Central (although the Central service must be running). RSFlowInvoke is useful when you want to start a flow from an external system, such as a monitoring application that can use a command line to start a flow.

- **Web Services Wizard (wswizard.exe)**

  When you run the Web Services Wizard, you provide it with the WSDL for a given Web service. The WSDL string you provide as a pointer can be a file's location and name or a URL. The Web Services Wizard displays a list of the methods in the API of the Web service that you specify. When you run the wizard, pick the methods you want to use, and with one click for each method you have selected, the wizard creates an HP OO operation that can execute the method. This allows you to use the Web Services Wizard to create operations from your monitoring tool's API.

These tools are available in the %OO_home%/Studio/tools/ folder.