HP Operations Orchestration Software

Software Version: 7.60

VMware vCenter Orchestrator Integration Guide

Document Release Date: January 2010

Software Release Date: January 2010

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

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
- BSA Essentials 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.

BSA Essentials Network

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

This is the **BSA Essentials Network** Web page. To sign in:

1. Click **Login Now**.
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 **BSA Essentials 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 VMware vCenter Orchestrator integration

With this integration, administrators can build HP Operations Orchestration (OO) flows that are integrated into VMware vCenter Orchestrator (vCO).

The vCO integration uses the vCO Web service API to integrate with OO. To use this integration successfully, you should have administrator-level knowledge of vCO.

This document explains how this integration has been implemented, and how the integration's operations and flows communicate between OO and vCO.

Note: vCO workflows automate tasks for VMware vSphere. In this guide, the term *workflow* refers to a vCO workflow and the term *flow* refers to an OO flow.

Use cases and scenarios

This section defines the major use cases for the vCO integration, and lists the operations and flows that you can use to implement them.

1. Manage vCO workflows:
   - CancelWorkflow operation
   - ExecuteWorkflow operation
   - LaunchWorkflow operation
   - RetrieveResults operation
2. Obtain information about objects, workflow details, and connectivity:
   - FindObjectsByProperty operation
   - GetWorkflowsDetails operation
   - Ping operation

Installation and configuration instructions

The vCO integration requires no special installation and configuration. Just make sure that the system that has the RSJRAS service running on it can access the vCO server.

The default vCO Web service access URL is:

`https://<vcoserver>:8281/vmware-vmo-webcontrol/webservice`

Versions

Operations Orchestration Version	VMware vCenter Orchestrator Version
7.60	vCenter Orchestrator 4.0 vCenter Orchestrator vSphere plugin 3.5 or 4.0

Architecture

Operations Orchestration communicates with the vCO host via SOAP. You must supply the host name, username, and password to operation and flow inputs. The connection protocol defaults to HTTPS, and the default port is 8281. These can vary per installation of vCO.

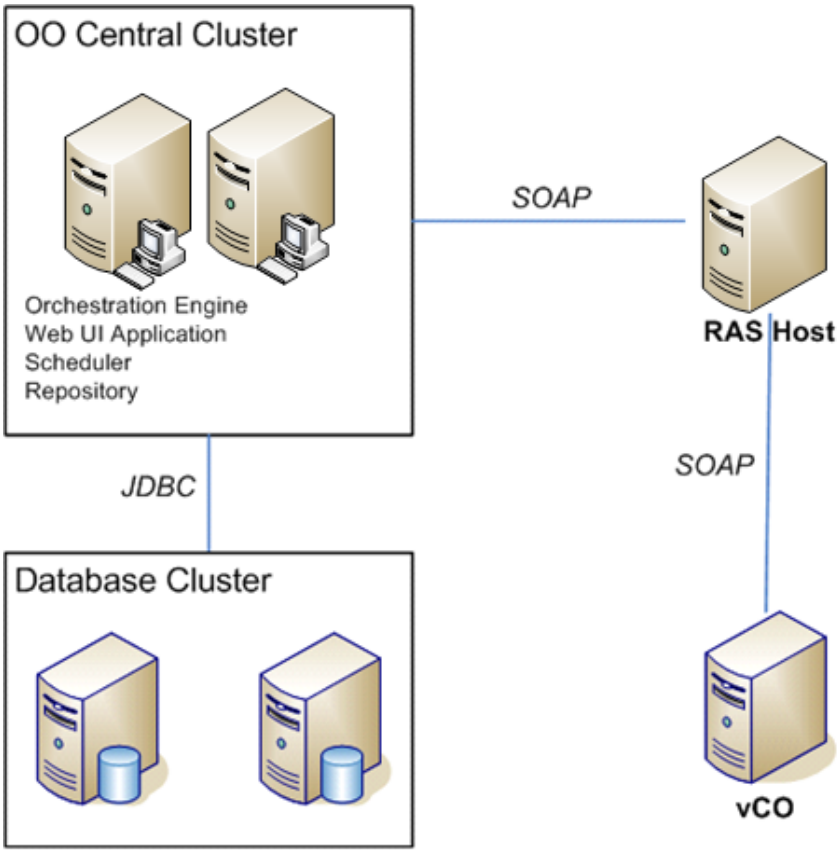

Figure 1 – vCO architecture

vCO integration operation infrastructure

The vCO integration includes the following operations in the OO Studio Library/Integrations/VMware/VMware vCenter Orchestrator/ folder.

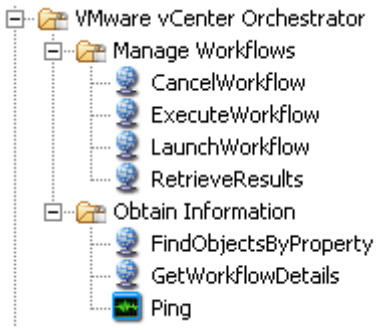

Figure 2 - vCO operations

Common inputs in the integration

OO flows and operations use inputs to specify how they obtain the data that they need and when the data is obtained. The following inputs are used consistently throughout the vCO integration's operations and flows.

host

The vCO Web service host. You can specify the host by using its IP address (for example, 10.2.255.116) or its DNS name (for example, myvsphere.company.com).

port

The server port on which the vCO Web service is running. The default port is **8281**.

protocol

The protocol used to communicate with the vCO server, either **http** or **https**. The default protocol is **https**.

username

The username to use to connect to vCO.

password

The password for the username.

vCO workflow input parameter types

The vCO workflows have several different types of inputs. It is important to understand how these input types map to Operations Orchestration inputs. To see some examples of vCO workflow input types, look in the **Workflows** section in the vCO client, navigate to Library/vCenter/Virtual Machine management/Clone/Linux Customization/. Open the **Properties** view of the workflow **Clone VM (Linux customization, 1 Nic)**, and click the **Input** tab.

There are five general categories of vCO workflow input types:

- **Simple types**

  These are types such as **string**, **boolean**, **number**, and **date**. These values are passed verbatim from the corresponding OO variables. vCO expects values of **true** or **false** for booleans. VCO expects dates to be in the format `yyyyMMddHHmmssZ`, which can be formatted with the **Date Parser** operation located in the Library/Utility Operations/Date and Time/ folder.

- **Plugin types**

  vCO plugins may define their own sets of custom types, whose names are normally prefixed with the plugin name. For example, the **vCenter** (VC) plugin defines a custom type `VC:VirtualMachine` to hold a virtual machine object. vCO requires that these inputs be populated with a special URI (uniform resource identifier) value, called a `dunesURI`.

  There are two methods you can use to pass a plugin value from Operations Orchestration:

  - Send a URI
  - Send an object property such as **name**

  Following is an example that shows how to populate a parameter **vm** of type `VC:VirtualMachine` with the value that corresponds to the virtual machine named **myVM** using the above methods:

  - **URI method**

    The URI is the identifier used by vCO to directly refer to the object. Using URI values offers the best performance since no additional lookups are required to resolve the object. In the following example, the URI for **myVM** is:

    ```
    dunes://service.dunes.ch/CustomSDKObject?id='myvsphere.company.com/vm-140'&dunesName='VC:VirtualMachine'.
    ```

    Since `dunesURI` values are not visible in the vCO client nor in any other VMware product, they are generally obtained by using the **FindObjectsByProperty** operation.

  - **Property method**

    Passing a property of the object, such as its name, is the more obvious method, but it takes more time as it requires a full search of the vCO inventory to resolve the property to a URI. Since vCO has no interface to perform restricted searches by property, this method requires retrieving all object names and searching through the list to find the indicated property. In environments with large numbers of objects of a particular type (for instance, virtual machines), where the RAS is separated from the vCO server by a slow network link, this may require large amount of time.

    To use this method, the input parameter should be in the format `property:value`. For example, to refer to the virtual machine whose name property is **MyVM**, specify the input as `name:MyVM`. If you do not specify a property name, it assumes a default of `name`, which is a valid property of most vCO objects.

    The **Inventory** section in vCO displays all objects, and each object has a **General** tab that displays pairs of properties and their values. Unfortunately, the property name displayed on the **General** tab is a user-friendly name that is not necessarily the same as the underlying property name. You may need to use the vCO API explorer to identify the actual property name. For example, a resource pool on the **Inventory** tab shows a property called `CPU Reservation`, whereas the API explorer shows that the property for the resource pool is actually named `cpuReservation`.

    If more than one object of the specified type has the same value for the specified property, the step in the OO workflow will return an error indicating that it cannot uniquely identify the object in the vCO inventory. Note that property names and values are matched without regard to case.

    When a workflow output is a plugin type, the URI is returned.

- **Array types**

  Some workflow parameters accept a list of values. The parameters have type `Array/itemtype`, where `itemtype` is the type of each value in the list. For example, the **dnsServerList** parameter in the workflows for cloning VMs is of type `Array/string`, meaning that it expects a list of string values. Note that `itemtype` can be any of the other parameter types including simple types and plugin types.

  Array parameters can be populated from OO by supplying a comma-separated (or other delimiter, if specified) list of values.

- **Properties**

  Workflows with a parameter whose type is `Properties` are not currently supported.

- **Native Java objects**

  A parameter with a type of `Any` is a native Java object. Workflows with native Java object parameters are not currently supported.

Operation and flow specifics

This section describes the vCO integration's operations and flows, including any operation- or flow-specific inputs. The flows and operations are grouped by their basic functionality:

- *Managing vCO workflows*
- *Obtaining information* about objects, workflow details, and connectivity

Managing vCO workflows

CancelWorkflow

The **CancelWorkflow** operation cancels the running workflow. The behavior of the workflow in vCO depends on the workflow being cancelled; even though the workflow enters the canceled state, it may continue to run. The operation produces a **Success** response if the workflow is successfully cancelled or has already completed. It produces a **Failure** response if the workflow could not be cancelled.

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

workflowToken

The workflow token returned from the **LaunchWorkflow** operation.

The operation returns the following:

returnResult

The main string containing the entire output.

ExecuteWorkflow

The **ExecuteWorkflow** operation combines the functionality of the **LaunchWorkflow** and **RetrieveResults** operations. This provides the ability to launch a workflow and wait for its results as a single step.

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

workflowID

The ID of the workflow to launch. Since workflow IDs are not visible in the vCO client, you may need to use the **GetWorkflowDetails** operation to obtain an ID.

workflowName

The name of the workflow to launch. If there is more than one workflow with the specified name, the workflow whose list of input parameters contains all of the **workflowInput_** inputs is used.

workflowInput_

The prefix to use for parameters to pass as an input to the vCO workflow. For example, an input named **workflowInput_cmd** containing the value `dir`, causes a value of `dir` to be passed to the vCO workflow input named **cmd**. You can pass more than one parameter.

retryInterval

The number of milliseconds to wait between retries. The default is **5000** (milliseconds).

retryLimit

The maximum number of retries to attempt. A value of **0** indicates that the operation tries to obtain the results once, but will not retry if the workflow has not completed. The default is **0**.

delimiter

For workflow inputs and outputs that contain a array list of values, the delimiter character will be used to separate the individual items. The default is a comma (,).

The operation returns the following:

returnResult

The main string containing the values of the output parameters, one per line. Each line contains the parameter name and value separated by a colon.

workflowToken

The token that can be used to retrieve results from the operation. Normally this operation returns the results directly, but if the **retryInterval** and **retryLimit** do not permit enough time for the flow to complete, it may be necessary to retrieve the results later.

workflowStatus

The status of the workflow. The valid values are **running**, **completed**, **canceled**, or **failed**.

LaunchWorkflow

The **LaunchWorkflow** operation launches a vCO workflow and returns without waiting for it to complete. The workflow is specified by the **workflowID** or **workflowName** parameter. WorkflowIDs can be obtained by using the **GetWorkflowDetails** operation. The operation produces a **Success** response if the workflow is successfully launched. It produces a **Failure** response if the workflow could not be launched.

This is a generic operation that permits launching any workflow in vCO; each vCO workflow has a different set of inputs. Any input to this operation whose name begins with **workflowInput_** is mapped to a vCO input of the same name but without that prefix. For example, if you create a step input named **workflowInput_vm**, the value will be passed to the **vm** input in the vCO workflow.

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

workflowID

The ID of the workflow to launch. Since workflow IDs are not visible in the vCO client, you may need to use the **GetWorkflowDetails** operation to obtain an ID.

workflowName

The name of the workflow to launch. If there is more than one workflow with the specified name, the workflow whose list of input parameters contains all of the **workflowInput_** inputs is used. We strongly recommend that you create workflows with unique names in vCO. If that is not possible, launch a workflow by specifying a **workflowID** that is guaranteed to be unique.

workflowInput_

A prefix to use for parameters to pass to the vCO workflow. For example, an input named **workflowInput_cmd** containing the value `dir` causes a value of `dir` to be passed to the vCO workflow input named **cmd**. You can pass more than one parameter.

delimiter

For workflow inputs that contain an array list of values, the delimiter character separates the individual items in the list. The default delimiter is a comma (,).

The operation returns the following:

returnResult

A brief message stating that the workflow started or an error message if the workflow was unable to start.

workflowStatus

The status of the workflow. The status is returned immediately after attempting to launch the workflow, and does not reflect failures in the execution of the workflow.

workflowToken

This is an ID that can be used to retrieve results from the operation at a later time.

RetrieveResults

The **RetrieveResults** operation retrieves the results of the specified workflow. If the workflow has not yet completed, this operation may optionally wait and retry several times. A workflow is considered to be complete if it is one of the following states: **completed**, **failed**, or **canceled**. The operation produces a **Success** response if the workflow has completed and the results could be retrieved; otherwise it produces a **Failure** response.

This is a generic operation that retrieves workflow outputs for any workflow in vCO; each vCO workflow has a different set of outputs. All outputs are concatenated, one per line, in the **returnResult** in the format `name:value`. You can use step filters to extract any outputs of interest.

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

workflowToken

The workflow token returned from the **LaunchWorkflow** operation.

retryInterval

The number of milliseconds to wait between retries. The default is **5000** (milliseconds).

retryLimit

The maximum number of retries to attempt. A value of **0** indicates that the operation will try to obtain the results once, but will not retry if the workflow has not completed. The default is **0**.

delimiter

For workflow outputs that contain an array list of values, the delimiter character that is used to separate the individual items in the list. The default is a comma (,).

The operation returns the following:

returnResult

Contains the values of the output parameters, one per line. Each line contains the parameter name and value separated by a colon.

workflowStatus

The status of the workflow.

Obtaining information

FindObjectsByProperty

The **FindObjectsByProperty** operation finds the object that corresponds to the specified name, returning its URI and ID. vCO has no interface to perform restricted searches by name. It retrieves all object names in a list and searches through the list to find the indicated name, so the performance of this operation may be poor in environments with large numbers of objects of a particular type (for instance, virtual machines) where the RAS is separated from the vCO server by a slow network link.

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

name

The object with the specified name. For example, `myVM`.

type

The input type. For example, `VC:VirtualMachine`.

The operation returns the following:

uri

Returns the URI corresponding to this object. If more than one object exists with the specified name (which is not possible with many object types), then a list of matching URIs is returned.

id

Returns the ID corresponding to this object. If more than one object exists with the specified name (which is not possible with many object types), then a list of matching IDs is returned.

matches

Returns the number of matching objects found. In most cases this will be `1`, but if more than one object exists with the specified name (which is not possible with many object types), then a count of matching objects is returned.

GetWorkflowDetails

The **GetWorkflowDetails** operation retrieves the detailed information about one or more vCO workflows. Only workflows that can be both read and executed by the user are returned.

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

workflowID

Finds the workflow with the specified ID. A workflow ID normally consists of 65 hexadecimal digits (0-9 and a-f).

workflowName

Finds the workflow with the specified name. The name can be a literal name, or it can contain a wildcard (*). For example, `Clone*` finds all workflows whose names begin with the word `Clone`. If both **workflowID** and **workflowName** are specified, only **workflowID** is used in the search. If neither is specified, all workflows are returned (subject to the constraints that they be readable and executable by the user).

The operation returns the following:

returnResult

Creates a detailed list of the flows that match the input criteria. For each flow, the list contains the workflow name, ID, and description, as well as the name and type of its parameters.

Ping

The **Ping** operation performs a simple test to verify connectivity to the vCO Web service. The operation produces a **Success** response if the Web service can be contacted and responds; otherwise it produces a **Failure** response.

Launching flows

You can use the REST service to launch flows from vCO using the following URL syntaxes to interact with HP OO Central:

Note: Synchronous flow execution means that Central does not return a result until the flow run has completed. In asynchronous flow execution, the flow result is returned immediately after the flow is launched.

- The following example shows a URL that synchronously launches a flow identified by its name and location in the OO Studio Library or Central repository:
  `https://localhost:8443/PAS/services/rest/run/Library/MyFolder/TestFlow`
- The following example shows a URL that synchronously launches a flow identified by its UUID:
  `https:// localhost:8443/PAS/services/rest/run/503c2500-7aae-11dd-a3b5-0002a5d5c51b`
- The following example shows a URL that asynchronously launches a flow by its name:
  `https://localhost:8443/PAS/services/rest/run_async/Library/MyFolder/TestFlow`
- The following example shows a URL that asynchronously launches a flow by its UUID:
  `https:// localhost:8443/PAS/services/rest/run_async/503c2500-7aae-11dd-a3b5-0002a5d5c51b`

Tools

Following are OO tools that you can use with the vCO 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 specified 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.