

# HP Operations Orchestration

for the Windows and Linux operating systems

Software Version: 9.00.06

---

## HP Operations Manager i Integration Guide

Document Release Date: October 2011

Software Release Date: October 2011



## 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

© Copyright 2011 Hewlett-Packard Development Company, L.P.

### Trademark Notices

For information on open-source and third-party software acknowledgements, see *Open-Source and Third-Party Software Acknowledgements* (HPOO\_OpenSrc\_3rd-PartyAcks.pdf) in the documentation set for this release.

## Documentation Updates

The title page of this document 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:

**<http://h20230.www2.hp.com/selfsolve/manuals>**

This site requires that you register for an HP Passport and sign-in. To register for an HP Passport ID, go to:

**<http://h20229.www2.hp.com/passport-registration.html>**

Or click the **New users - please register** link on the HP Passport login page.

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

Visit the HP Software Support Web site at:

**[www.hp.com/go/hpsoftwaresupport](http://www.hp.com/go/hpsoftwaresupport)**

This Web site provides contact information and details about the products, services, and support that HP Software offers.

HP Software online support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the support Web 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 register for an HP Passport ID, go to:

**<http://h20229.www2.hp.com/passport-registration.html>**

To find more information about access levels, go to:

**[http://h20230.www2.hp.com/new\\_access\\_levels.jsp](http://h20230.www2.hp.com/new_access_levels.jsp)**

# Contents

- 1 Introduction .....7
  - Purpose of the HP Operations Manager i Integration .....8
  - Supported Versions .....8
  - Downloading OO Releases and Documents on HP Live Network.....8
- 2 Getting Started with the OMi Integration .....9
  - Installing and Configuring the Integration .....10
  - OMi — OO Integration Architecture.....10
  - OMi Use Cases .....10
- 3 Using the OMi – OO Integration .....12
  - Location of OMi Integration Operations and Flows in OO Studio.....13
  - Common Inputs in the Integration.....13
  - Descriptions of OMi Integration Operations and Flows .....14
    - Custom Attributes Sample .....14
    - Add Annotation .....14
    - Add Custom Attribute.....15
    - Add Symptom .....15
    - Delete Annotation .....16
    - Delete Custom Attributes .....17
    - Delete Symptom .....17
    - Get Annotations .....18
    - Get Event.....20
    - Get Symptoms .....22
    - Get User Action .....22
    - Launch Auto Action .....23
    - Launch User Action .....23
    - Stop Auto Action .....23
    - Stop User Action .....24
    - Update Annotation.....24
    - Update Custom Attribute .....25
    - Update Event.....25
- 4 Troubleshooting.....27
  - Troubleshooting Overview .....28
  - General Troubleshooting Procedures and Tools .....28

5 OO Tools .....	29
OO Tools You Can Use with the OMi – OO Integration.....	30

# 1 Introduction

This section includes the following topics:

- [Purpose of the HP Operations Manager i Integration](#)
- [Supported Versions](#)
- [Downloading OO Releases and Documents on HP Live Network](#)

## Purpose of the HP Operations Manager i Integration

With this integration, you can build HP Operations Orchestration (OO) flows that are integrated into the HP Operations Manager i (OMi).

The OMi integration operations interact with OMi through its REST-based Event Web service, which is described more completely in the BSM OMi Extensibility Guide. You cannot use the Event Web Service to create or modify events, so there are no operations in this integration to perform these actions. The OMi Event Web service can be reached via the HP Business Service Management Gateway Server, which is the host that should be supplied in all of the operations. The supported versions are 9.0, 9.01, and 9.10.

## Supported Versions

**Table 1 Supported Versions**

Operations Orchestration Version	OMi Version
9.00.06	9.0, 9.01, and 9.10

## Downloading OO Releases and Documents on HP Live Network

HP Live Network provides an **Operations Orchestration Community** page where you can find and download supported releases of OO and associated documents.

To download OO releases and documents, visit the following site:

**<https://www2.hp.com/>**

This site requires that you register for an HP Passport and sign-in. To register for an HP Passport ID, go to:

**<http://h20229.www2.hp.com/passport-registration.html>**

Or click the **New users - please register** link on the HP Passport login page.

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.

## 2 Getting Started with the OMi Integration

This section includes the following topics:

- [Installing and Configuring the Integration](#)
- [OMi – OO Integration Architecture](#)
- [OMi Use Cases](#)

# Installing and Configuring the Integration

There are no special installation or configuration steps.

## OMi — OO Integration Architecture

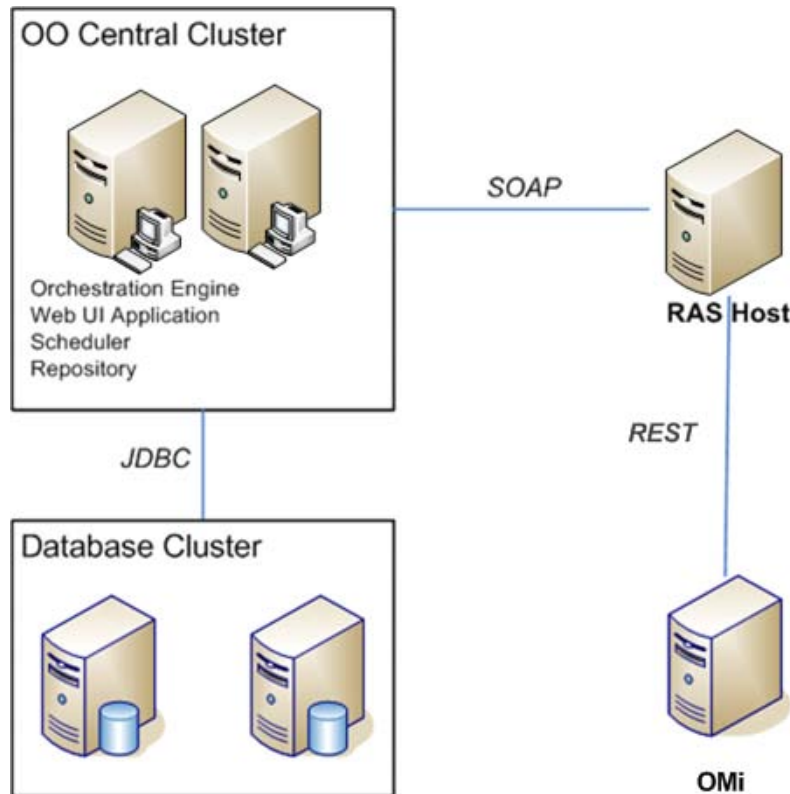


Figure 1 - OMi - OO Integration Architecture

## OMi Use Cases

The following are the major use cases for the HP OMi integration, and the operations and flows that you can use to implement them.

- 1 Samples
  - Custom Attributes Sample
- 2 Manage custom attributes:
  - Add Custom Attribute
  - Custom Attributes Sample

- Delete Custom Attribute
  - Get Custom Attributes
  - Update Custom Attribute
- 3 Manage annotations:
- Add Annotation
  - Delete Annotation
  - Get Annotations
  - Update Annotation
- 4 Manage symptoms:
- Add Symptom
  - Delete Symptom
  - Get Symptoms
- 5 Manage actions:
- Get Auto Action
  - Get User Action
  - Launch Auto Action
  - Launch User Action
  - Stop Auto Action
  - Stop User Action
- 6 Manage events:
- Update Event
  - Get Event

---

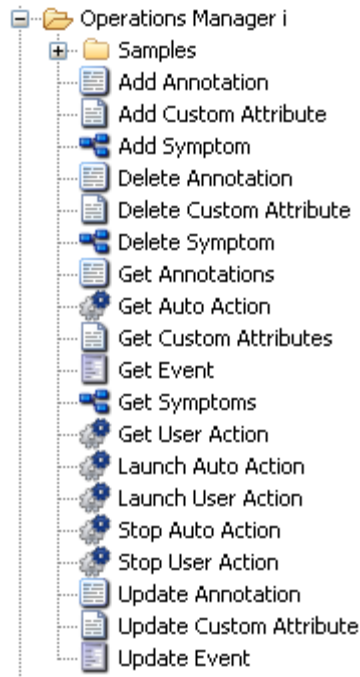
## 3 Using the OMi – OO Integration

This section includes the following topics:

- [Location of OMi Integration Operations and Flows in OO Studio](#)
- [Common Inputs in the Integration](#)
- [Descriptions of OMi Integration Operations and Flows](#)

# Location of OMi Integration Operations and Flows in OO Studio

The OMi integration includes the following operations in the OO Studio Library/Integrations/Hewlett-Packard/Operations Manager i/ folder.



**Figure 2 – Location of OMi Integration Operations and Flows**

## 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 OMi integration's operations and flows.

### host

The host on which the Event Web service runs, This should be the HP Business Service Management Gateway Server. The input value can be a host name or an IP address. If you specify a host name, it must be capable of being resolved to an IP address by a DNS server specified in your networking configuration. You can use the nslookup command, available in Windows and most UNIX systems, to verify that the given host name can be resolved.

### port

The port on which the Event Web service listens for requests. This is normally **80** for HTTP and **443** for HTTPS. If you do not specify a value for this input, it defaults to 80 or 443, depending on the value of the protocol input.

### protocol

The protocol to use to connect to the Event Web service. The valid values are **HTTP** and **HTTPS**. The default is **HTTPS**.

### username

In the OMi model, each user has a "user name" and a "login name" that may be different. The value for this input should be the OMi login name.

### password

The password to use to connect to the Event Web service.

### timeout

The length of time, in milliseconds, that can elapse before the connection times out. If the connection is not successfully made in this amount of time, the service gives up and the operation fails. If the connection is made within this time, this timeout value is the maximum amount of time that the operation waits to receive a reply. If you do not set a value for this input or set it to **0**, the default value of **60000** (one minute) is used.

### id

The event identifier, usually in UUID format—a string of hexadecimal digits and dashes (for example, **550e8400-e29b-41d4-a716-446655440000**).

## Descriptions of OMi Integration Operations and Flows

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

The sample flow **Custom Attributes Sample** in the OO Library/Integrations/Hewlett-Packard/Operations Manager i/Samples/ folder perform some of the most common tasks that need to be automated when using OMi.

### Custom Attributes Sample

The **Custom Attributes Sample** flow (in the Library/Integrations/Hewlett-Packard/Operations Manager i/Samples/ folder) illustrates how to use all of the operations that manipulate custom attributes. This flow performs the following steps:

- Adds a new custom attribute to an event.
- Retrieves all custom attributes for the event.
- Builds a list of results, and displays it to the user.
- Updates the value of the newly created custom attribute to a constant value.
- Deletes the newly created custom attribute.

### Add Annotation

The **Add Annotation** operation adds an annotation to an event.

## Inputs

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

`annotationText`

The annotation text to add to the event.

## Results

The operation returns the following:

`returnResult`

If the operation is successful, this output contains a brief message stating that it was successful. If the operation fails, this output contains details about the error.

## Add Custom Attribute

The **Add Custom Attribute** operation adds a custom attribute to an event with the specified custom attribute name and value. If the event already has a custom attribute with the specified name, its value is updated with the new value.

## Inputs

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

`caName`

The custom Attribute name.

`caValue`

The custom attribute value.

## Results

The operation returns the following:

`returnResult`

If the operation is successful, this output contains a brief message stating that it was successful. If the operation fails, this output contains details about the error.

## Add Symptom

The **Add Symptom** operation adds a symptom to the specified event. Symptoms are part of OMi's event correlation, where a symptom is the inverse of a cause. This operation triggers the following changes in OMi:

- The given event is marked as a "cause."
- The symptom event is marked as a "symptom."

- The correlation rule is marked as "Manually related."

## Inputs

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

### symptomTargetId

The identifier of the event which is considered to be a symptom of the event in the id input. The symptomTargetId input value is usually in UUID format, a string of hexadecimal digits and dashes (for example, 550e8400-e29b-41d4-a716-446655440000).

## Results

The operation returns the following:

### returnResult

If the operation is successful, this output contains a brief message stating that it was successful. If the operation fails, this output contains details about the error.

## Example

Assume that OMi has received the following two events:

11111111-1111-1111-1111-111111111111 - Bad sector detected on disk 1

22222222-2222-2222-2222-222222222222 - Unable to save file 'myinfo.txt'

After determining that the event 11111111-1111-1111-1111-111111111111 is responsible for 00000000-0000-0000-0000-000000000000, this operation could be called with the following inputs:

id: 11111111-1111-1111-1111-111111111111

symptomTargetId: 22222222-2222-2222-2222-222222222222

indicating that the second event is a symptom of the first. Event 11111111-1111-1111-1111-111111111111 is marked as a cause and event 22222222-2222-2222-2222-222222222222 is marked as a symptom.

## Delete Annotation

The **Delete Annotation** operation deletes an annotation from an event. Annotation IDs can be obtained using the **Get Annotations** operation.

## Inputs

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

annotationId

The annotation ID to delete from the event. This ID is assigned by OMi when the annotation is created, and it may not be accessible via the OMi user interface. Annotation IDs are usually in UUID format.

## Results

The operation returns the following:

returnResult

If the operation is successful, this output contains a brief message stating that it was successful. If the operation fails, this output contains details about the error.

## Delete Custom Attributes

The **Delete Custom Attributes** operation deletes a custom attribute from an event with the specified attribute name.

## Inputs

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

caName

The name of the custom attribute to delete.

## Results

The operation returns the following:

returnResult

If the operation is successful, this output contains a brief message stating that it was successful. If the operation fails, this output contains details about the error.

## Delete Symptom

The **Delete Symptom** operation deletes a symptom from the specified event. Symptoms are part of OMi's event correlation, where a symptom is the inverse of a cause. This operation triggers changes in OMi, removing the "cause" from the given event and the "symptom" from the symptom event.

## Inputs

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

symptomTargetId

The identifier of the event that is a symptom of the event in the **id** input. The **symptomTargetId** is usually in UUID format, a string of hexadecimal digits and dashes (for example, **550e8400-e29b-41d4-a716-446655440000**).

## Results

The operation returns the following:

`returnResult`

If the operation is successful, this output contains a brief message stating that it was successful. If the operation fails, this output contains details about the error.

## Get Annotations

The **Get Annotations** operation gets the annotations for an event.

### Inputs

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

## Results

The operation returns the following:

`returnResult`

If the operation is successful, this output contains a brief message stating that it was successful. If the operation fails, this output contains details about the error.

`jsAnnotations`

A JavaScript Object string that is an array of annotation objects. The folder Library/Utility Operations/Containers/JavaScript Objects/ contains operations to manipulate these objects, and the description of that folder has a more complete discussion of the JavaScript Object format.

Each object in the array contains the following information about an annotation:

`id`

The annotation ID. This ID is used as an input to other operations such as Delete Annotation.

`author`

The login name of the user that last modified this annotation.

`text`

The annotation text.

`date`

The date and time of the last modification of this annotation, in ISO 8601 format.

## Example

If there are two annotations on the event, then **jsAnnotations** might contain:

```
[ { "id"      : "ad3d404e-0c23-463c-8910-ffd4466a785a",
    "author" : "admin",
    "text"   : "This event was painful to track down",
    "date"   : "2010-12-07T21:37:14-0800" },
  { "id"      : "b345a3d7-302a-42e1-af47-292d8837dc2d",
    "author" : "SYSTEM",
    "text"   : "automatic action\n started.\nOperator : SYSTEM",
    "date"   : "2010-12-02T06:35:24-0800" } ]
```

(Extra whitespace inserted for readability.)

## Get Auto Action

The **Get Auto Action** operation gets information about the auto action assigned to an event.

### Inputs

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

### Results

The operation returns the following:

#### returnResult

If the operation is successful, this output contains a brief message stating that it was successful. If the operation fails, this output contains details about the error.

#### actionStatus

The state of the auto action. Values include **available** and **succeeded**.

#### actionCall

The command that is called when the action is run.

## Get Custom Attributes

The **Get Custom Attributes** operation gets the custom attributes of an event.

### Inputs

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

### Results

The operation returns the following:

#### returnResult

If the operation is successful, this output contains a brief message stating that it was successful. If the operation fails, this output contains details about the error.

### jsCustomAttributes

A JavaScript Object string that is an array of annotation objects. The folder Library/Utility Operations/Containers/JavaScript Objects/ contains operations to manipulate these objects, and the description of that folder has a more complete discussion of the JavaScript Object format.

Each object in the array contains the following pieces of information about a custom attribute:

#### caName

The custom attribute name.

#### caValue

The custom attribute value.

## Example

If there are two custom attributes on the event, then **jsCustomAttributes** might contain:

```
[ { "caValue" : "Reviewed by",  
    "caName"  : "operator2" },  
  { "caValue" : "Region",  
    "caName"  : "43" } ]
```

(Extra whitespace inserted for readability.)

## Get Event

The **Get Event** operation gets the details of an event.

### Inputs

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

### Results

The operation returns the following:

#### returnResult

If the operation is successful, this output contains a brief message stating that it was successful. If the operation fails, this output contains details about the error.

#### application

The application that triggered the event.

#### assignedGroup

The assigned group of the event.

#### assignedUser

The login name of the user assigned to the event.

#### category

The category of the event.

#### description

The description of the event.

#### duplicates

The count of duplicates of the event.

#### eti

The Event Type Indicator.

#### id

The ID of the event.

#### node

The node on which the event occurred.

#### object

The object of the event.

#### priority

The priority of the event. Possible values are **none**, **low**, **medium**, **high**, and **highest**.

#### severity

The severity of the event. Possible values are **unknown**, **normal**, **warning**, **minor**, **major**, and **critical**.

#### sourceCI

The source CI.

#### state

The state of the event. Possible values are **open**, **in\_progress**, **resolved**, and **closed**.

#### subcategory

The subcategory of the event.

#### timeCreated

The time the event was created, in ISO 8601 format.

#### timeReceived

The time the event was received by OMi, in ISO 8601 format.

#### title

The title of the event.

## Get Symptoms

The **Get Symptoms** operation retrieves symptoms for an event. Symptoms are part of OMi's event correlation, where a symptom is the inverse of a cause.

### Inputs

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

### Results

The operation returns the following:

#### `returnResult`

If the operation is successful, this output contains a brief message stating that it was successful. If the operation fails, this output contains details about the error.

#### `jsSymptoms`

This is a JavaScript Object string that is an array of event IDs. The folder Library/Utility Operations/Containers/JavaScript Objects/ contains operations to manipulate these arrays, and the description of that folder has a more complete discussion of the JavaScript Object format.

### Example

If the event has two symptoms, **jsSymptoms** might contain:

```
[ "cac7516a-b58b-48ff-b60e-0e23bc9886b5" ,  
  "b5341300-fe28-71df-07f6-1039417e0000" ]
```

## Get User Action

The **Get User Action** operation gets information about the user action assigned to an event.

### Inputs

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

### Results

The operation returns the following:

#### `returnResult`

If the operation is successful, this output contains a brief message stating that it was successful. If the operation fails, this output contains details about the error.

#### `actionStatus`

The state of the user action. Values include **available** and **succeeded**.

`actionCall`

The command that is called when the action is run.

## Launch Auto Action

The **Launch Auto Action** operation launches the auto action assigned to an event. Note that this operation does not wait for the action to complete.

### Inputs

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

### Results

The operation returns the following:

`returnResult`

If the operation is successful, this output contains a brief message stating that it was successful. If the operation fails, this output contains details about the error.

## Launch User Action

The **Launch User Action** operation launches the user action assigned to an event. Note that this operation does not wait for the action to complete.

### Inputs

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

### Results

The operation returns the following:

`returnResult`

If the operation is successful, this output contains a brief message stating that it was successful. If the operation fails, this output contains details about the error.

## Stop Auto Action

The **Stop Auto Action** operation stops the auto action assigned to an event.

### Inputs

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

### Results

The operation returns the following:

`returnResult`

If the operation is successful, this output contains a brief message stating that it was successful. If the operation fails, this output contains details about the error.

## Stop User Action

The **Stop User Action** operation stops the user action assigned to an event.

### Inputs

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

### Results

The operation returns the following:

`returnResult`

If the operation is successful, this output contains a brief message stating that it was successful. If the operation fails, this output contains details about the error.

## Update Annotation

The **Update Annotation** operation updates an annotation on an event. The specified annotation ID must already exist. Annotation IDs can be obtained using the **Get Annotations** operation.

### Inputs

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

`annotationId`

The annotation ID to update on the event. This ID is assigned by OMi when the annotation is created, and it may not be accessible via the OMi user interface. Annotation IDs are usually in UUID format.

`annotationText`

The annotation text to update on the event.

### Results

The operation returns the following:

`returnResult`

If the operation is successful, this output contains a brief message stating that it was successful. If the operation fails, this output contains details about the error.



OMi 9.0 has a defect that causes this operation to always fail. This defect has been fixed in OMi 9.01.

## Update Custom Attribute

The **Update Custom Attribute** operation updates the value of an existing custom attribute of an event.

### Inputs

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

`caName`

The custom attribute name to update.

`caValue`

The new custom attribute value.

### Results

The operation returns the following:

`returnResult`

If the operation is successful, this output contains a brief message stating that it was successful. If the operation fails, this output contains details about the error.

## Update Event

The **Update Event** operation updates the details of an event. Note that only inputs that contain values are updated; others remain unchanged.

### Inputs

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

`assignedGroup`

The assigned group of the event.

`assignedUser`

The login name of the user to assign to the event. OMi expects this to be a login name (the one used to log on), which may be different from the display username. If the login name is unknown, OMi silently ignores this change.

`description`

The description of the event.

#### priority

The priority of the event. This value is case-insensitive. The valid values are **none**, **low**, **medium**, **high**, and **highest**.

#### severity

The severity of the event. This value is case-insensitive. The valid values are **unknown**, **normal**, **warning**, **minor**, **major**, and **critical**.

#### solution

The solution of the event.

#### state

The state of the event. This value is case-insensitive. The valid values are **open**, **in\_progress**, **resolved**, and **closed**.

#### title

The title of the event.

## Results

The operation returns the following:

#### returnResult

If the operation is successful, this output contains a brief message stating that it was successful. If the operation fails, this output contains details about the error.

---

## 4 Troubleshooting

This section includes the following topics:

- [Troubleshooting Overview](#)
- [General Troubleshooting Procedures and Tools](#)

## Troubleshooting Overview

This section provides troubleshooting procedures and tools you can use to solve problems you may encounter while using this integration. It also includes a list of the error messages you may receive while using the integration and offers descriptions and possible fixes for the errors.

## General Troubleshooting Procedures and Tools

Section VI of the *BSM OMi Extensibility Guide*, “Automating Operator Functions and Event Change Detection,” contains some useful tips for using a third-party REST client. This can be very helpful in troubleshooting connectivity or other problems.

---

## 5 OO Tools

This section includes the following topic:

- [OO Tools You Can Use with the OMi - OO Integration](#)

## OO Tools You Can Use with the OMi – OO Integration

Following are OO tools that you can use with the OMi 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 Operations Orchestration home folder in /Studio/tools/.