

IBM
Version 9 Release 1

*Programming for Optim Service
Interface*



IBM
Version 9 Release 1

*Programming for Optim Service
Interface*



Note

Before using this information and the product it supports, read the information in "Notices" on page 23.

Version 9 Release 1

This edition applies to version 9, release 1, modification 0 of IBM InfoSphere Optim solution components and to all subsequent releases and modifications until otherwise indicated in new editions.

© **Copyright IBM Corporation 2012.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About this publication v

Programming for the service interface . 1

Starting the service interface on an application server 1

Service interface methods 2

 Service methods 2

 Service execution methods. 4

 Service schedule methods 5

 Service instance methods 10

 Repository methods 15

Command-line utility for service interface 16

 Get service (optimcmd -service) 16

Get proxies/servers (optimcmd -proxy) 18

Run service (optimcmd -run) 18

Monitor service instances (optimcmd -monitor) 19

Display repository databases (optimcmd
-databases) 20

Import repository data (optimcmd -import). . . 21

Export repository data (optimcmd -export) . . . 22

Notices 23

Trademarks 25

Index 27

About this publication

This document describes the functions of Optim™ Service Interface. Use this information to create applications that use InfoSphere® Optim solution components to run and manage data management services.

Programming for the service interface

Optim Service Interface is a web application that provides a public interface into the IBM® InfoSphere Optim data management solution environment. (Optim Service Interface is also known as the *service interface*.) The service interface can be used by other applications to run, monitor, and manage services.

How the service interface works

The service interface accepts HTTP requests from other applications. Each HTTP request must follow a method to accomplish a specific task. Some methods require the submission of XML request payloads. The service interface processes each request that the service interface receives. When the request is complete, the service interface returns an HTTP response code and output document where applicable to the other application.

Prerequisites

To use the service interface, you must first install the service interface on a computer. When you install the manager on a computer, the service interface is also installed by default. You must then deploy the service interface Web archive (WAR) file to an application server and start the application within the WAR file. For information about how to install and deploy the service interface, see the manager installation information and the component configuration information for InfoSphere Optim solution components.

Security

To secure the service interface from unauthorized use, use a firewall to control access to the application server on which the service interface is deployed. Also, ensure that any applications that use the service interface can authenticate users and log requests that are sent to the service interface.

Starting the service interface on an application server

To use the service interface to run and manage test- or production-level services, you must first start the service interface on its application server. After the service interface is started on the application server, any application can access the service interface at any time.

Before you can start the service interface, you must install the service interface. By default, the service interface is installed with the manager. You must also configure the service interface and the components that the service interface uses to run services. For example, you must deploy the service interface WAR file to the application server.

To start the service interface on an application server:

1. Start the application server. If the application server is set to start the service interface web application automatically, then the service interface is started immediately after the application server. If you deployed the service interface to the version of WebSphere® Application Server Community Edition that is delivered with the manager, then complete the following step. In this step, *shared_installation_directory* is the installation directory that you specified for the manager.
 - Microsoft Windows computers: Click **Start > All Programs > IBM InfoSphere > Optim > Start WAS-CE**, or run the script *shared_installation_directory\WebSphere\AppServerCommunityEdition\bin\startup.bat*.
 - Linux or UNIX computers: Run the script *shared_installation_directory/WebSphere/AppServerCommunityEdition/bin/startup.sh*.

2. If the service interface is not started after a few minutes, start the service interface web application by using the application server console. If you deployed the service interface to the version of WebSphere Application Server Community Edition that is delivered with the manager, then complete the following steps:
 - a. Use a web browser to access and sign into the Administrative Console. The default location is at `http://hostname:port/console/`, where *hostname* is the host name or IP address of the WebSphere Application Server Community Edition computer and *port* is the port number. The default port number is 8080. Use user ID system and password manager to access the Administrative Console.
 - b. Click **Web App WARs**.
 - c. Click **Start** for the component with an URL of `/server`.

To automate the starting of the service interface after you restart the computer, configure the application server as a Windows service or Linux or UNIX daemon.

Service interface methods

The service interface supports a set of methods that are implemented by using HTTP (Hypertext Transfer Protocol) services. Each method uses a distinct HTTP resource URL that indicates the method to use and any parameters to use with the method.

Resource URLs

The format of each resource URL is `http://hostname:port/server/method_URI`.

- *hostname* is the host name or IP address of the application server.
- *port* is the port number of the application server.
- *method_URI* is the URI that is used by the method. Depending on the method that is used, the method URI can contain user-specified parameters.

For example, if the application server uses port 8080 on a computer with host name `appserver`, and a method uses the URI `/monitor/`, the method resource URL is `http://appserver:8080/server/monitor/`.

Supported encoding

Use UTF-8 encoding in any parameters resource URL and in all request payloads.

Service methods

Use service methods to get a list of the services in the repository and to get information about these services. Service methods are a prerequisite for most other tasks in the service interface.

Get All Services method

Use the Get All Services method to receive an XML file that contains a list of all services that are in the repository. You can then get more detailed information about a service by using the Get Service method.

HTTP method: GET

URI: `/service/`

Request payload content type: None

Response payload content type: `application/xml`

Expected response: HTTP/1.1 200 OK

Response payload example

The following example shows a sample XML response for this method.

```
<?xml version="1.0" encoding="UTF-8"?>
<sim:services xmlns:atom="http://www.w3.org/2005/Atom"
  xmlns:sim="http://www.ibm.com/optim/xsd/sim/9.1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.ibm.com/optim/xsd/sim/9.1.0
  resource.xsd ">
  <link href="http://interface:8080/server/service/fbf1cdd5-bdf7-4682-96ad-722672af0001"/>
  <link href="http://interface:8080/server/service/9ed5e389-3e0c-434e-b76c-8dc0488856d9"/>
  <link href="http://interface:8080/server/service/0ffad96f-4e89-4347-aad1-610643aa174f"/>
</sim:services>
```

Response codes

The following response codes can be received when the method is complete.

Response code	Description
200	OK. The service list was successfully returned.
400	Bad Request. A malformed request was submitted.
404	Not Found. No services exist in the repository.

Get Service method

Use the Get Service method to receive an XML file that contains information about a service.

HTTP method: GET

URI: /service/service_id

Request payload content type: None

Response payload content type: application/xml

Expected response: HTTP/1.1 200 OK

Parameters

The method URI contains the following parameters.

Name	Description	Required
service_id	Enter the service ID that identifies the service whose information you want to view.	Y

Response payload example

The following example shows an XML file that contains information about the requested service. The information includes the service name, path, type, and platform.

```
<?xml version="1.0" encoding="UTF-8"?>
<srm:service xmlns:atom="http://www.w3.org/2005/Atom"
  xmlns:srm="http://www.ibm.com/optim/xsd/srm/9.1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.ibm.com/optim/xsd/srm/9.1.0
  resource.xsd ">
  <serviceId>fbf1cdd5-bdf7-4682-96ad-722672af0001</serviceId>
```

```

<serviceName>SMK_V910.ERGL23393T</serviceName>
<servicePath>DOR931KEQUSM/SMK_V910</servicePath>
<serviceType>Extract</serviceType>
<servicePlatform>Distributed</servicePlatform>
</srm:service>

```

Response codes

The following response codes can be received when the method is complete.

Response code	Description
200	OK. The service information was successfully returned.
400	Bad Request. A malformed request was submitted.
404	Not Found. The service ID does not exist in the repository.

Service execution methods

Use service execution methods to run services.

Run Service method

Use the Run Service method to run a service. The service can be run without any changes. Alternatively, you can run the service with input values that are different from the values that are saved with the service.

HTTP method: POST

URI: */execute/service_id*

Request payload content type: application/xml

Response payload content type: application/xml

Expected response: HTTP/1.1 201 Created

Parameters

The method URI contains the following parameters.

Name	Description	Required
<i>service_id</i>	Enter the service ID that identifies the service that you want to run.	Y

Request payload example

The following example shows a request to run a service.

```

<?xml version="1.0" encoding="UTF-8"?>
<sem:serviceRequestExecutionInput
  xmlns:sem="http://www.ibm.com/optim/xsd/sem/9.1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.ibm.com/optim/xsd/sem/9.1.0
  resource.xsd">
  <serviceId>783015a5-5ad4-43a8-b3be-a45acaba081a</serviceId>
  <serviceName>SMK_V910.ERGL23393T</serviceName>

```

```

    <servicePath>DOR931KEQUSM/SMK_V910</servicePath>
    <proxyURL>http://servercomputer:12000</proxyURL>
    <executedBy>jdoe</executedBy>
</sem:serviceRequestExecutionInput></p>

```

The following example shows a request to run the service from the first example with two changed input values (override values).

```

<p><?xml version="1.0" encoding="UTF-8"?>
<sem:serviceRequestExecutionInput
  xmlns:sem="http://www.ibm.com/optim/xsd/sem/9.1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.ibm.com/optim/xsd/sem/9.1.0
  resource.xsd">
  <serviceId>783015a5-5ad4-43a8-b3be-a45acaba081a</serviceId>
  <serviceName>SMK_V910.ERGL23393T</serviceName>
  <servicePath>DOR931KEQUSM/SMK_V910</servicePath>
  <proxyURL>http://servercomputer:12000</proxyURL>
  <executedBy>jdoe</executedBy>
  <overrides>
    <ns2:override>
      <id>override-id-000001</id>
      <type>TypeOne</type>
      <value>Value One</value>
    </ns2:override>
    <ns2:override>
      <id>override-id-000002</id>
      <type>TypeTwo</type>
      <value>Value Two</value>
    </ns2:override>
  </overrides>
</sem:serviceRequestExecutionInput></p>

```

Response payload example

The following example shows the response that confirms that the service was started.

```

<?xml version="1.0" encoding="UTF-8"?>
<sem:executionResults
  xmlns:sem="http://www.ibm.com/optim/xsd/sem/9.1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.ibm.com/optim/xsd/sem/9.1.0
  resource.xsd">
  <executionId>fbf1cdd5-bdf7-4682-96ad-722672af0001</executionId>
  <serviceId>783015a5-5ad4-43a8-b3be-a45acaba081a</serviceId>
  <serviceName>SMK_V910.ERGL23393T</serviceName>
  <servicePath>DOR931KEQUSM/SMK_V910</servicePath>
</sem:executionResults></p>

```

Response codes

The following response codes can be received when the method is complete.

Response code	Description
202	Accepted. The service request was processed and accepted. This response code does not mean that the service request completed successfully.
400	Bad Request. A malformed request was submitted.

Service schedule methods

Use service schedule methods to view, create, change, and delete service schedules. Each service in a repository can have one schedule at a time.

Add Schedule method

Use the Add Schedule method to add a schedule to run a service.

HTTP method: POST

URI: `/scheduler/schedule_id`

Request payload content type: application/xml

Response payload content type: None

Expected response: HTTP/1.1 201 Created

Parameters

The method URI contains the following parameters.

Name	Description	Required
<code>schedule_id</code>	Enter the schedule ID that identifies the schedule to be added.	Y

Request payload example

The following example shows a request to schedule a service to run on Thursday, August 1, 2013 at 17:30:00 local time (epoch time 1375378200000 in milliseconds).

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:schedule xmlns:ns2="http://www.ibm.com/optim/xsd/scheduler/9.1.0">
  <id>fbf1cdd5-bdf7-4682-96ad-722672af0001</id>
  <serviceId>783015a5-5ad4-43a8-b3be-a45acaba081a</serviceId>
  <startDate>1375378200000</startDate>
  <endDate>0</endDate>
  <repeatCount>0</repeatCount>
  <repeatInterval>0</repeatInterval>
  <schedulerTaskType>SOA_SERVICE</schedulerTaskType>
  <proxyURL>http://servercomputer:12000</proxyURL>
  <creatorId>jdoe</creatorId>
</ns2:schedule>
```

The following example shows a request to schedule the service to run on Thursday, August 1, 2013 at 17:30:00 local time. The cron expression sets the schedule to repeat every Thursday afterward at 17:30:00 local time.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:schedule xmlns:ns2="http://www.ibm.com/optim/xsd/scheduler/9.1.0">
  <id>fbf1cdd5-bdf7-4682-96ad-722672af0002</id>
  <serviceId>783015a5-5ad4-43a8-b3be-a45acaba081a</serviceId>
  <startDate>1375378200000</startDate>
  <endDate>0</endDate>
  <cronExpression>0 30 17 ? * THU</cronExpression>
  <schedulerTaskType>SOA_SERVICE</schedulerTaskType>
  <proxyURL>http://servercomputer:12000</proxyURL>
  <creatorId>jdoe</creatorId>
</ns2:schedule>
```

Response codes

The following response codes can be received when the method is complete.

Response code	Description
201	Created. The service schedule was created in the repository and scheduled.
400	Bad Request. A malformed request was submitted.
409	Conflict. The service ID is already associated with a schedule in the repository and the schedule cannot be added, or the schedule ID is already defined in the scheduler.

Update Schedule method

Use the Update Schedule method to update an existing schedule to run a service.

HTTP method: PUT

URI: */scheduler/schedule_id*

Request payload content type: application/xml

Response payload content type: None

Expected response: HTTP/1.1 200 OK

Parameters

The method URI contains the following parameters.

Name	Description	Required
<i>schedule_id</i>	Enter the schedule ID that identifies the schedule to be updated.	Y

Request payload example

The following example shows a request to schedule a service to run on Thursday, August 1, 2013 at 17:30:00 local time (epoch time 1375378200000 in milliseconds).

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:schedule xmlns:ns2="http://www.ibm.com/optim/xsd/scheduler/9.1.0">
  <id>fbf1cdd5-bdf7-4682-96ad-722672af0001</id>
  <serviceId>783015a5-5ad4-43a8-b3be-a45acaba081a</serviceId>
  <startDate>1375378200000</startDate>
  <endDate>0</endDate>
  <repeatCount>0</repeatCount>
  <repeatInterval>0</repeatInterval>
  <schedulerTaskType>SOA_SERVICE</schedulerTaskType>
  <proxyURL>http://servercomputer:12000/</proxyURL>
  <creatorId>jdoe</creatorId>
</ns2:schedule>
```

The following example shows a request to schedule the service to run on Thursday, August 1, 2013 at 17:30:00 local time. The cron expression sets the schedule to repeat every Thursday afterward at 17:30:00 local time.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:schedule xmlns:ns2="http://www.ibm.com/optim/xsd/scheduler/9.1.0">
  <id>fbf1cdd5-bdf7-4682-96ad-722672af0002</id>
  <serviceId>783015a5-5ad4-43a8-b3be-a45acaba081a</serviceId>
  <startDate>1375378200000</startDate>
  <endDate>0</endDate>
  <cronExpression>0 30 17 ? * THU</cronExpression>
  <schedulerTaskType>SOA_SERVICE</schedulerTaskType>
  <proxyURL>http://servercomputer:12000/</proxyURL>
  <creatorId>jdoe</creatorId>
</ns2:schedule>
```

Response codes

The following response codes can be received when the method is complete.

Response code	Description
200	OK. The service schedule was updated in the repository and scheduled.
400	Bad Request. A malformed request was submitted.
404	Not Found. The schedule ID is not found in the repository.
409	Conflict. The service ID is already associated with a different schedule in the repository.

Get All Schedules method

Use the Get All Schedules method to list the services for which schedules exist in the repository. You can then use the Get Schedule method to get the schedule for a specified service.

HTTP method: GET

URI: /scheduler/

Request payload content type: None

Response payload content type: application/xml

Expected response: HTTP/1.1 200 OK

Response payload example

The following example shows a sample XML response for this method.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:schedules xmlns="http://www.w3.org/2005/Atom"
  xmlns:ns2="http://www.ibm.com/optim/xsd/scheduler/9.1.0">
  <link href="http://interface:8080/server/scheduler/9ed5e389-3e0c-434e-b76c-8dc0488856d9"/>
  <link href="http://interface:8080/server/scheduler/0ffad96f-4e89-4347-aad1-610643aa174f"/>
</ns2:schedules>
```

Response codes

The following response codes can be received when the method is complete.

Response code	Description
202	Accepted. The service request was processed and accepted. This response code does not mean that the service request completed successfully.

Response code	Description
400	Bad Request. A malformed request was submitted.
404	Not Found. No schedules were found in the repository.

Get Schedule method

Use the Get Schedule method to get the schedule for a specified service.

HTTP method: GET

URI: `/scheduler/schedule_id` or `/scheduler/?serviceId=service_id`

Request payload content type: None

Response payload content type: application/xml

Expected response: HTTP/1.1 200 OK

Parameters

The method URI contains the following parameters.

Name	Description	Required
<i>schedule_id</i>	Enter the schedule ID that identifies the schedule you want to see.	Y if you use <code>/scheduler/schedule_id</code>
<i>service_id</i>	Enter the service ID that identifies the service whose schedule you want to see.	Y if you use <code>/scheduler/?serviceId=service_id</code>

Response payload example

The following example shows a schedule to run a service on Thursday, August 1, 2013 at 17:30:00 local time (epoch time 1375378200000 in milliseconds). The format of the response payload for the Get Schedule method is similar to the request payload for the Add Schedule or Update Schedule methods.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:schedule xmlns:ns2="http://www.ibm.com/optim/xsd/scheduler/9.1.0">
  <id>fbf1cdd5-bdf7-4682-96ad-722672af0001</id>
  <serviceId>783015a5-5ad4-43a8-b3be-a45acaba081a</serviceId>
  <startDate>1375378200000</startDate>
  <endDate>0</endDate>
  <repeatCount>0</repeatCount>
  <repeatInterval>0</repeatInterval>
  <schedulerTaskType>SOA_SERVICE</schedulerTaskType>
  <proxyURL>http://servercomputer:12000</proxyURL>
</ns2:schedule>
```

The following example shows a request to schedule the service to run on Thursday, August 1, 2013 at 17:30:00 local time. The cron expression sets the schedule to repeat every Thursday afterward at 17:30:00 local time.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:schedule xmlns:ns2="http://www.ibm.com/optim/xsd/scheduler/9.1.0">
  <id>fbf1cdd5-bdf7-4682-96ad-722672af0002</id>
  <serviceId>783015a5-5ad4-43a8-b3be-a45acaba081a</serviceId>
  <startDate>1375378200000</startDate>
  <endDate>0</endDate>
```

```
<cronExpression>0 30 17 ? * THU</cronExpression>
<schedulerTaskType>SOA_SERVICE</schedulerTaskType>
<proxyURL>http://servercomputer:12000/</proxyURL>
</ns2:schedule>
```

Response codes

The following response codes can be received when the method is complete.

Response code	Description
200	OK. The schedule was successfully returned.
400	Bad Request. A malformed request was submitted.
404	Not Found. The schedule does not exist in the repository.

Delete Schedule method

Use the Delete Schedule method to delete the schedule for a service.

HTTP method: DELETE

URI: */scheduler/schedule_id*

Request payload content type: None

Response payload content type: None

Expected response: HTTP/1.1 204 No Content

Parameters

The method URI contains the following parameters.

Name	Description	Required
<i>schedule_id</i>	Enter the schedule ID whose schedule is to be deleted.	Y

Response codes

The following response codes can be received when the method is complete.

Response code	Description
204	No Content. The service schedule was removed from the repository.
400	Bad Request. A malformed request was submitted.
404	Not Found. The schedule ID does not exist in the repository.

Service instance methods

Use service instance methods to view the service instances in your repository. When you run a service, a service instance record is created in the repository to store information about how the service was run. Service instance information includes the start and end times of the service, the return code, and artifacts that contain additional information.

Get All Service Instances method

Use the Get All Service Instances method to receive an XML file that contains a list of all service instances that are in the repository. You can then get more detailed information about a service instance by using the Get Service Instance method.

HTTP method: GET

URI: /monitor

Request payload content type: None

Response payload content type: application/xml

Expected response: HTTP/1.1 200 OK

Response payload example

The following example shows a sample XML response for this method.

```
<?xml version="1.0" encoding="UTF-8"?>
<sim:serviceInstances xmlns:atom="http://www.w3.org/2005/Atom"
  xmlns:sim="http://www.ibm.com/optim/xsd/sim/9.1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.ibm.com/optim/xsd/sim/9.1.0 resource.xsd ">
  <link href="http://interface:8080/server/monitor/fbf1cdd5-bdf7-4682-96ad-722672af0001"/>
  <link href="http://interface:8080/server/monitor/9ed5e389-3e0c-434e-b76c-8dc0488856d9"/>
  <link href="http://interface:8080/server/monitor/0ffad96f-4e89-4347-aad1-610643aa174f"/>
</sim:serviceInstances>
```

Response codes

The following response codes can be received when the method is complete.

Response code	Description
200	OK. The service instance list was successfully returned.
400	Bad Request. A malformed request was submitted.
404	Not Found. No service instances exist in the repository.

Get Service Instance method

Use the Get Service Instance method to receive an XML file that contains information about a service instance. The file includes the service ID and the service instance ID, start time, end time, and return code. The file also lists any associated artifacts, which can be retrieved by using the Get Service Instance Artifact method.

HTTP method: GET

URI: /monitor/*execution_id*

Request payload content type: None

Response payload content type: application/xml

Expected response: HTTP/1.1 200 OK

Parameters

The method URI contains the following parameters.

Name	Description	Required
<i>execution_id</i>	Enter the execution ID that identifies the service instance whose information you want to view.	Y

Response payload example

The following example shows a service instance with execution ID 34770e5c-e282-47bf-9467-160cda2a1e06. Service 783015a5-5ad4-43a8-b3be-a45acaba081a was started on Thursday, August 1, 2013 at 17:30:00 local time (epoch time 1375378200230 in milliseconds). The service completed successfully at 17:33:07 local time (epoch time 1375378387206 in milliseconds). The service instance contains the artifacts execution.properties, overrides.txt, run.log, and svc_request.xml.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<ns2:serviceInstance
  xmlns:ns2="http://www.ibm.com/optim/xsd/sim/9.1.0">
  <id>34770e5c-e282-47bf-9467-160cda2a1e06</id>
  <serviceId>783015a5-5ad4-43a8-b3be-a45acaba081a</serviceId>
  <startTime>1375378200230</startTime>
  <endTime>1375378387206</endTime>
  <returnCode>0</returnCode>
  <artifactList>
    <artifactName>execution.properties</artifactName>
    <artifactName>overrides.txt</artifactName>
    <artifactName>run.log</artifactName>
    <artifactName>svc_request.xml</artifactName>
  </artifactList>
</ns2:serviceInstance>
```

Response codes

The following response codes can be received when the method is complete.

Response code	Description
200	OK. The service instance information was successfully returned.
400	Bad Request. A malformed request was submitted.
404	Not Found. The execution ID does not exist in the repository.

Get Service Instance Artifact Names method

Use the Get Service Instance Artifact Names method to receive an XML file that contains a list of artifact names for a service instance. For example, you can determine whether a service instance has an override file. You can then use the Get Service Instance Artifact method to receive the content that is contained in each artifact.

HTTP method: GET

URI: /monitor/artifacts/*execution_id*

Request payload content type: None

Response payload content type: application/xml

Expected response: HTTP/1.1 200 OK

Parameters

The method URI contains the following parameters.

Name	Description	Required
<i>execution_id</i>	Enter the execution ID that identifies the service instance whose artifacts you want to view.	Y

Response payload example

The following example shows a list of artifacts for a service instance with the execution ID fbflcdd5-bdf7-4682-96ad-722672af0001. The artifacts are named execution.properties, overrides.txt, run.log, and svc_request.xml.

```
<?xml version="1.0" encoding="UTF-8" ?>
<ns2:serviceInstanceArtifacts
  xmlns="http://www.w3.org/2005/Atom"
  xmlns:ns2="http://www.ibm.com/optim/xsd/sim/9.1.0">
  <link href="http://interface:8080/server/monitor/fbflcdd5-bdf7-4682-96ad-722672af0001/execution.properties"/>
  <link href="http://interface:8080/server/monitor/fbflcdd5-bdf7-4682-96ad-722672af0001/overrides.txt"/>
  <link href="http://interface:8080/server/monitor/fbflcdd5-bdf7-4682-96ad-722672af0001/run.log"/>
  <link href="http://interface:8080/server/monitor/fbflcdd5-bdf7-4682-96ad-722672af0001/svc_request.xml"/>
</ns2:serviceInstanceArtifacts>
```

Response codes

The following response codes can be received when the method is complete.

Response code	Description
200	OK. The artifact list was successfully returned.
400	Bad Request. A malformed request was submitted.
404	Not Found. The execution ID does not exist in the repository.

Get Service Instance Artifact method

Use the Get Service Instance Artifact method to receive a service instance artifact. For example, you can use this method to request the process report for a service instance.

HTTP method: GET

URI: /monitor/*execution_id*/*artifact_name*

Request payload content type: None

Response payload content type: application/xml

Expected response: HTTP/1.1 200 OK

Parameters

The method URI contains the following parameters.

Name	Description	Required
<i>execution_id</i>	Enter the execution ID that identifies the service instance whose artifacts you want to view.	Y
<i>artifact_name</i>	Enter the name of the artifact whose content you want to view.	Y

Response payload example

The following example shows possible contents of an artifact. Artifacts are text-based files that are either in XML format or text format. Use artifacts to troubleshoot issues with a service.

```
/OUTPUT PSTDIR=OPTDIRORA TYPE=Extract REQUEST=TESTDATA.ERCUST11K STOP=None ERRORLEVEL=0
Extract Process Report
```

```
Request Name          TESTDATA.ERCUST11K
Server Name           (Local)
Extract File          C:\IBM\InfoSphere\Optim\data\ERCUST11K.xf
Access Definition     TESTDATA.CUST11K
File Attachments      Processed
Client User ID        optadmin
Server User ID        optadmin
Teradata Character Set WE8MSWIN1252
Time Started          5/16/2013 13:03:52
Time Finished         5/16/2013 13:03:54
Elapsed Time          00:00:02
Extract File Data Byte Count 0.001 MB
Process Status no errors, no warnings
```

Response codes

The following response codes can be received when the method is complete.

Response code	Description
200	OK. The artifact was successfully returned.
400	Bad Request. A malformed request was submitted.
404	Not Found. The execution ID does not exist in the repository, or the artifact name does not exist for the execution ID.

Delete Service Instance method

Use the Delete Service Instance method to purge or delete a service instance from the repository.

HTTP method: DELETE

URI: */monitor/execution_id*

Request payload content type: None

Response payload content type: application/xml

Expected response: HTTP/1.1 204 No Content

Parameters

The method URI contains the following parameters.

Name	Description	Required
<i>execution_id</i>	Enter the execution ID that identifies the service instance to be deleted.	Y

Response codes

The following response codes can be received when the method is complete.

Response code	Description
204	No Content. The service instance was successfully purged from the repository.
400	Bad Request. A malformed request was submitted.
404	Not Found. The execution ID does not exist in the repository.

Repository methods

Use repository methods to view and interact directly with components that are registered in the repository. For example, you can use a repository method to get a list of proxies (servers) that are registered in the repository.

Get Server List method

Use the Get Server List method to receive an XML file that contains a list of proxies (servers) that are registered in the repository.

HTTP method: GET

URI: /registry/proxy

Request payload content type: None

Response payload content type: application/xml

Expected response: HTTP/1.1 200 OK

Response payload example

The following example shows a list of three servers (proxies) with host names server1, server2, and server3.

```
<?xml version="1.0" encoding="UTF-8"?>
<rr:proxies xmlns:rr="http://www.ibm.com/optim/xsd/sem/9.1.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.ibm.com/optim/xsd/sem/9.1.0 resource.xsd">
<proxy>http://server1:12000/</proxy>
<proxy>http://server2:12000/</proxy>
<proxy>http://server3:12000/</proxy>
</rr:proxies>
```

Response codes

The following response codes can be received when the method is complete.

Response code	Description
200	OK. The proxy list was successfully returned.
400	Bad Request. A malformed request was submitted.
404	Not Found. No proxies were found.

Command-line utility for service interface

By default, the service interface is installed with a command-line utility. The command-line utility is an example of an application that uses the service interface to run and manage services.

Command-line utility location

The command-line utility uses the **optimcmd** tool. The location of the **optimcmd** tool depends on the operating system. *shared_installation_directory* is the installation directory that you specified for the service interface.

- Microsoft Windows computers: *shared_installation_directory*\tools\optimcmd\optimcmd.bat.
- Linux or UNIX computers: *shared_installation_directory*/tools/optimcmd/optimcmd.sh.

Configuration file location

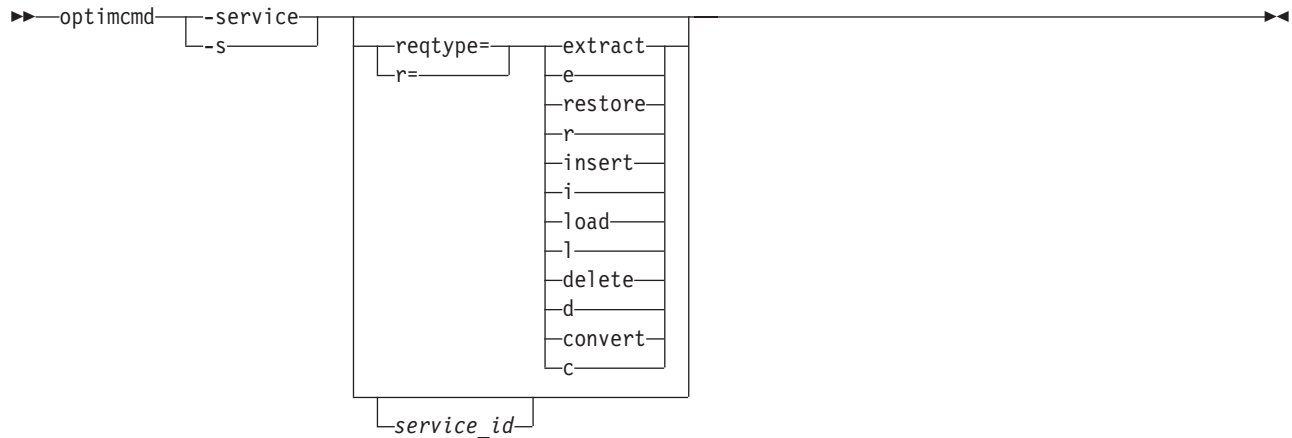
The `optimcmd.properties` file contains the URLs that the command-line utility uses to locate the service interface, the proxy, and the repository manager. Before you use the command-line utility, confirm that the `optimcmd.properties` file contains the URLs that you want the command-line utility to use. The `optimcmd.properties` file is in the *shared_installation_directory*/tools/optimcmd/ directory, where *shared_installation_directory* is the installation directory that you specified for the service interface.

Get service (optimcmd -service)

Use the Get service command to get a list of services in the repository or to get detailed information about a specific service. A list of services contains the service ID, the service name, the path of the service in the repository, and the service type.

Command structure

The following diagram indicates the structure of the command and its parameters.



Input

The following information can be entered with the command.

optimcmd

The `optimcmd` tool.

-service or -s

The Get service command.

reqtype= or r=

The service type whose services you want to list. Use any of the following values:

- `extract` or `e`: Extract
- `restore` or `r`: Restore
- `insert` or `i`: Insert
- `load` or `l`: Load
- `delete` or `d`: Delete
- `convert` or `c`: Convert

service_id

The ID for the service that you want to view.

Output

The output depends on whether you specify a service ID.

- If you specify a service ID, the command returns detailed information about the service.
- If you do not specify a service ID, the command returns a list of services in the repository. Each line contains information for a service (the service ID, the service name, the path of the service in the repository, and the service type). You can optionally limit the services in this list to services with a specified service type.

Return codes

The following return codes can be received when you enter the command.

Return code	Description
0	No errors.
4	No services match the specified criteria.
12	The command cannot connect with the repository.

Get proxies/servers (optimcmd -proxy)

Use the Get proxies/servers command to get a list of proxies that are registered in the repository. The output of this command contains the full URL of the proxy with the host name or IP address and the port (for example, <http://proxy:12000/>).

Command structure

The following diagram indicates the structure of the command and its parameters.



Input

The following information can be entered with the command.

optimcmd

The optimcmd tool.

-proxy or -p

The Get proxies/servers command.

Output

The command returns a list of proxies that are registered in the repository.

Return codes

The following return codes can be received when you enter the command.

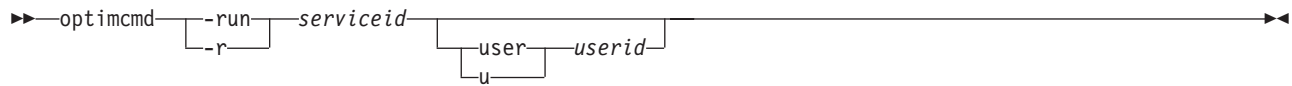
Return code	Description
0	No errors.
4	No proxies were found in the repository.
12	The command cannot connect with the repository.

Run service (optimcmd -run)

Use the Run service command to run a service by using the server to which the service is assigned.

Command structure

The following diagram indicates the structure of the command and its parameters.



Input

The following information can be entered with the command.

optimcmd

The optimcmd tool.

-run or -r

The Run service command.

serviceid

The service ID for the service that you want to run.

-user or -u

The user parameter for the Run service command.

userid The user ID to use to run the service.

Output

When successful, the command returns the execution ID of the service instance that is started when you run the service.

Return codes

The following return codes can be received when you enter the command.

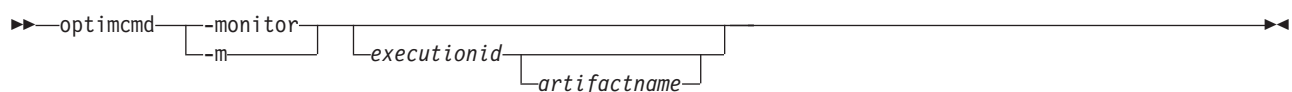
Return code	Description
0	No errors.
8	The specified service failed to run.
12	The command cannot connect with the repository.

Monitor service instances (optimcmd -monitor)

Use the Monitor service instances command to view a list of service instances in the repository or to view detailed information about a specific service instance.

Command structure

The following diagram indicates the structure of the command and its parameters.



Input

The following information can be entered with the command.

optimcmd

The optimcmd tool.

-monitor or -m

The Monitor service command.

executionid

The execution ID for the service instance that you want to view.

artifactname

The name of the artifact that you want to view. Artifacts are XML or text files that contain additional information about the service instance.

Output

The output depends on whether you specify an execution ID and an artifact name.

- If you do not specify an execution ID, the command returns a list of service instances in the repository. Each line contains information for a service instance (the execution ID, the service ID, and the return code).
- If you specify an execution ID without an artifact name, the command returns detailed information about the service instance. The information includes the execution ID, the service ID, the start and end date, the return code, and a list of artifacts that contain detailed information about the service instance.
- If you specify an execution ID with an artifact name, the command returns the contents of the artifact. The artifact can be text output or XML code.

Return codes

The following return codes can be received when you enter the command.

Return code	Description
0	No errors.
4	The specified service instance was not found.
8	The specified artifact was not found.
12	The command cannot connect with the repository.

Display repository databases (optimcmd -databases)

Use the Display repository databases command to list the databases that are contained in the repository.

Command structure

The following diagram indicates the structure of the command and its parameters.



Input

The following information can be entered with the command.

optimcmd

The optimcmd tool.

-databases or -d

The Display repository databases command.

Output

If successful, the output contains a list of the databases in the repository, with each database on a separate line.

Return codes

The following return codes can be received when you enter the command.

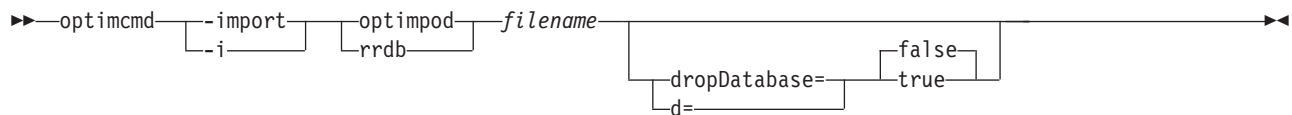
Return code	Description
0	No errors.
8	The repository did not return any databases because of an internal error.
12	The command cannot connect with the repository.

Import repository data (optimcmd -import)

Use the Import repository data command to overwrite the contents of a repository database with the data from an export file (such as rrdb.zip or optimpod.zip). To copy a repository, use the Export repository data command to export the data from every database in the repository. Then, use the Import repository data command to import each of the resulting export files into the corresponding database in another repository.

Command structure

The following diagram indicates the structure of the command and its parameters.



Input

The following information can be entered with the command.

optimcmd

The optimcmd tool.

-import or -i

The Import repository data command.

optimpod or rrdb

The databases that can be imported.

filename

The file name and path of the export file that you want to import.

-dropDatabase or -d

The drop database parameter (which is used to specify whether the specified database is to be dropped before you import the database).

true or false

Specify true to drop the specified database or false.

Return codes

The following return codes can be received when you enter the command.

Return code	Description
0	No errors.
8	The specified export file was not found, or there was an error in importing the database.
12	The command cannot connect with the repository.

Export repository data (optimcmd -export)

Use the Export repository data command to export the data in a repository database to an export file (such as rrdb.zip or optimpod.zip). To copy a repository, use the Export repository data command to export the data from every database in the repository. Then, use the Import repository data command to import each of the resulting export files into the corresponding database in another repository.

Command structure

The following diagram indicates the structure of the command and its parameters.



Input

The following information can be entered with the command.

optimcmd

The optimcmd tool.

-export or -e

The Export repository data command.

optimpod or rrdb

The databases that can be exported.

filename

The file name and path to use for the resulting export file.

Return codes

The following return codes can be received when you enter the command.

Return code	Description
0	No errors.
8	The export file cannot be written to the specified location, or there was an error in exporting the database.
12	The command cannot connect with the repository.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Software Interoperability Coordinator
Director of Engineering, Information Management (Office 16)
111 Campus Drive
Princeton, NJ 08540
USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information in softcopy, the photographs and color illustrations may not appear.

Trademarks

IBM, the IBM logo, and ibm.com[®] are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Index

A

Add Schedule
 method 6
 service schedule methods 6
Add Schedule method 6
artifact 19

C

command-line utility 16
 configuring 16
 location 16
configuring
 command-line utility 16
copy a repository 21, 22

D

Delete Schedule
 method 10
 service schedule methods 10
Delete Schedule method 10
Delete Service Instance
 method 14
 service instance methods 14
Delete Service Instance method 14
Display repository databases
 command 20

E

encoding 2
Export repository data command 22

F

format
 URLs 2
format of resource URLs 2

G

Get All Schedules
 method 8
 service schedule methods 8
Get All Schedules method 8
Get All Service Instances
 method 11
 service instance methods 11
Get All Service Instances method 11
Get All Services
 method 2
 service methods 2
Get All Services method 2
Get proxies/servers command 18
Get Schedule
 method 9
 service schedule methods 9

Get Schedule method 9
Get Server List
 method 15
 repository methods 15
Get Server List method 15
Get Service
 method 3
 service methods 3
Get service command 17
Get Service Instance
 method 11
 service instance methods 11
Get Service Instance Artifact
 method 13
 service instance methods 13
Get Service Instance Artifact method 13
Get Service Instance Artifact Names
 method 12
 service instance methods 12
Get Service Instance Artifact Names
 method 12
Get Service Instance method 11
Get Service method 3

I

Import repository data command 21

M

Monitor service instances command 19

O

optimcmd
 -databases 20
 -export 22
 -import 21
 -monitor 19
 -proxy 18
 -run 19
 -service 17

P

prerequisites 1
proxy 18

R

repository
 copy a 21, 22
repository methods 15
Run Service
 method 4
 service execution methods 4
Run service command 19
Run Service method 4

S

server 18
service execution methods 4
service instance 19
service instance methods 11
service interface
 security 1
service interface methods 2
service interface security 1
service methods 2
service schedule methods 6

U

Update Schedule
 method 7
 service schedule methods 7
Update Schedule method 7



Printed in USA