HP ServiceCenter

For the UNIX and Windows operating systems

Software Version: 1.0

HP OpenView Operations for UNIX
Node Bank and Outage Integration Guide

Manufacturing Part Number: none

Document Release Date: May 2007

Software Release Date: May 2007

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

Java™ and all Java based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

Microsoft® is a U.S. registered trademark of Microsoft Corporation.

Oracle® is a registered U.S. trademark of Oracle Corporation, Redwood City, California.

UNIX® is a registered trademark of The Open Group.

Windows® is a U.S. registered trademark of Microsoft Corporation.

Documentation Updates

This guide's title page 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, visit the following URL:

ovweb.external.hp.com/lpe/doc_serv/

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

You can visit the HP Software support web site at:

http://support.openview.hp.com/support.jsp

HP Software online support provides an efficient way to access interactive technical support tools. As a valued support customer, you can benefit by using the support 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 find more information about access levels and HP Passport, go to:

http://support.openview.hp.com/new_access_levels.jsp

Table of Contents

1 Introduction to this Guide

Introduction

This guide explains how to install and configure Node Bank and Outage integrations between HP OpenView ServiceCenter version 6.1 and 6.2 and HP OpenView Operations for UNIX version 8.x. The aim of the guide is to provide you with enough information to install, configure, and troubleshoot these integrations.

Audience

This guide is written for ServiceCenter (SC) consultants who wish to integrate SC with OVO for UNIX in order to arrange for outages that are planned in the context of a change process.

Other potential readers include:

- OpenView Architects
- OpenView Designers
- OpenView Developers
- OpenView Project Managers
- OpenView Program Managers
- OpenView General R&D Managers
- OpenView Information Engineers

What you should know before reading this guide

To install and configure the integration you should have a good knowledge of the following:

- ServiceCenter
- HP OpenView Operations

A working knowledge of the following is needed:

- Oracle RDBMS
- UNIX Operating System
- Connect-It, a tool that provides capabilities to exchange data between two data stores.

2 Concepts – Node Bank and Outage Integration

Overview

This HP OpenView Operations for UNIX (OVO for UNIX) – ServiceCenter (SC) integration can be divided into two areas, Node Bank Integration and Outage Integration.

The Integration can be used in environments with one Management Server but also in Manager-of-Manager (MoM) environments with multiple Management Servers.

Node Bank Integration

The Node Bank Integration synchronizes data in the ServiceCenter Configuration Manager database (SC CMDB) with the Node Bank information kept on the available OVO for UNIX Management Servers. This synchronization is designed to take place in a short time span on a regular basis. Synchronization starts with wiping the registration of Management Servers related to Nodes in the CMDB. After that, Node Bank information from each Management Server is transferred into the CMDB. At the end of a complete cycle, all information in the CMDB is up-to-date..

The synchronization is executed by means of Connect-It scenarios. The first scenario to run in a synchronization cycle removes the registration of Management Servers with Nodes in the CMDB, using ServiceCenter Web Services. After that, a scenario is run for each Management Server, connecting to the underlying database and transfering the relevant data into the CMDB by means of the ServiceCenter Web Services.

Additionally, the Node Bank Integration scenario copies Message Group data from the OVO for UNIX Management Servers to the SC Database, so they can be used by the Outage integration as filtering conditions.

Outage Integration

OVO for UNIX must be made aware of outages planned in the context of a change process. During the specified outage periods, OVO for UNIX will interpret events detected on systems and will accordingly suppress or only log the corresponding messages. Messages that comply to the outage descriptions sent to OVO will not be escalated to the incident management process.

The Outage Integration uses the Node Bank information to identify Configuration Items (CIs) as nodes managed by OVO for UNIX, and uses a wizard to configure OVO for UNIX outages that are then sent to and uploaded onto the appropriate OVO for UNIX Management Server.

Node Bank Integration

The first step in a ServiceCenter – OVO for UNIX integration is to synchronize the SC CMDB with the OVO for UNIX node bank. Once synchronization is complete, SC can identify CIs as nodes managed by OVO for UNIX, and validate that the CIs are suitable for sending outage notifications to OVO for UNIX. This synchronization is executed on a regular basis (by default) in order to keep the CMDB in line with the changing IT environment. The logical architecture of this process is shown in the diagram below.

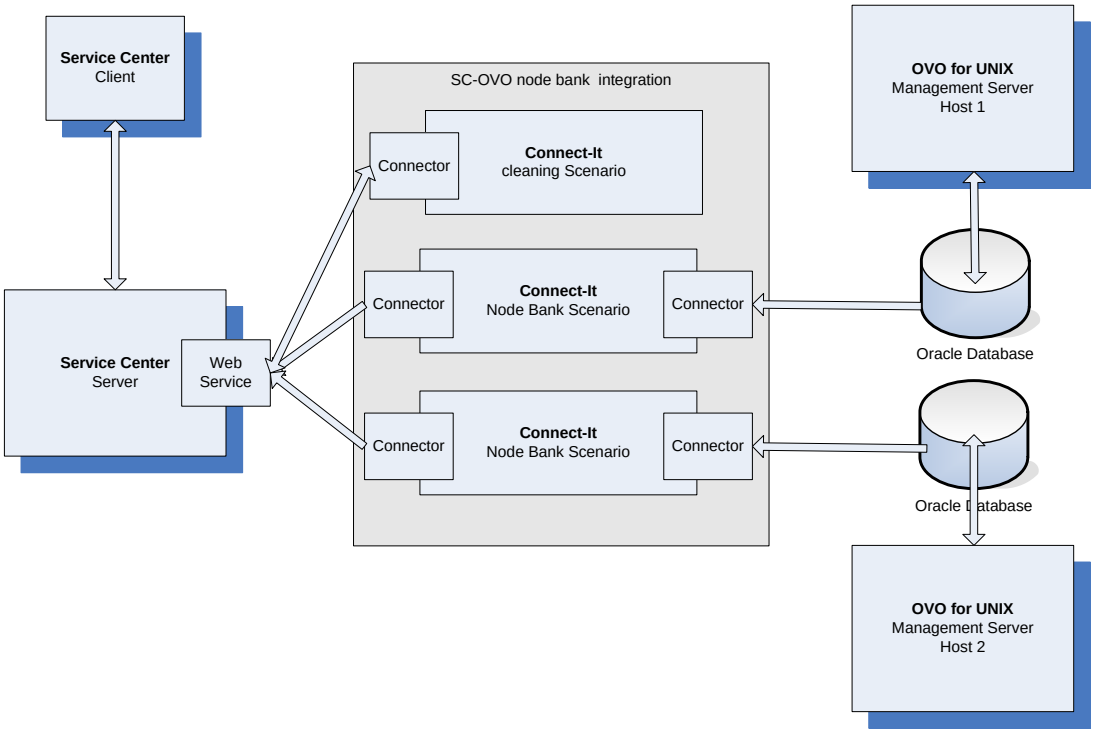
Each complete run of the Node Bank integration starts with a Connect-It scenario that:

- Wipes the information on OVO Management Servers stored with the Node data in the SC CMDB.
- Changes the Node type to 'OVO node being synchronized' for each 'device' that is managed by OVO.

Next, a Connect–It scenario is run for each Management Server. This scenario connects to the Oracle database underlying the OVO Management Server and to SC in order to complete the following tasks:

- Collect Node Bank information and pass this data via a SC Web Services Connector to the "device" table in SC. Some additional fields need to be added to the device table; this procedure is described in the *Installing Node Bank and Outage Integration*.
- Copy Message Group information from the OVO for UNIX Management Server into "SC Global Lists" for use with the OVO-Outage integration.

The user is supplied with two default Connect-It Node Bank Scenarios, one for the preliminary actions, and one that passes data into SC. The second scenario can be copied and configured with the appropriate SC and OVO for UNIX Management Server information. This should be done for each OVO for UNIX Management Server involved in the Node Bank integration, resulting in a scenario being defined for each Management Server. These procedures are described in detail in *Installing Node Bank and Outage Integration*.



Node Bank Integration – Logical Architecture

Fields imported into the ServiceCenter CMDB

By default, the following fields will be imported into the SC CMDB.

Managed node data

The following data is extracted from the OVO for UNIX database:

- Fully Qualified Domain Name (FQDN) of the managed node
- IP address of the manged node
- Management servers of the managed node
- Node type of the managed node

You can add additional fields by editing the mapping Connectors.

Message group data

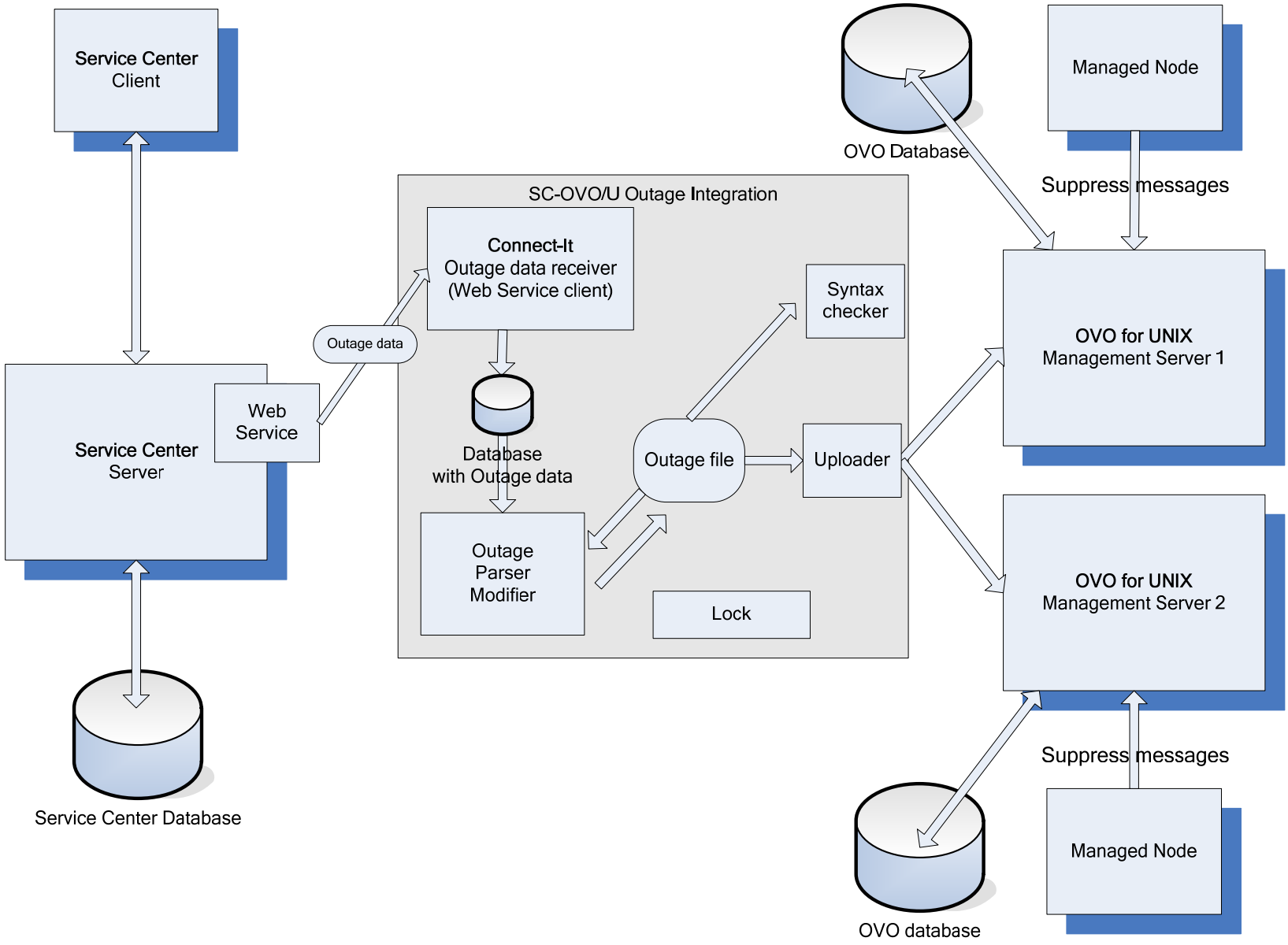
The values of the message group attribute are extracted from the OVO for UNIX database. Customers can add or modify message groups in OVO. The Node Bank Integration copies the new message groups to ServiceCenter.. The default attributes are:

Backup	Certificate	Database	HA	Hardware	Job
Misc	NetWare	Network	OS	OpC	Output
Performance	SNMP	SSP	Security		

Outage Integration

Similar to the Node Bank Integration process, Outage Integration also uses a Connect-It scenario to copy data. In this case the scenario copies outage information, contained in the `ovooutage` table in ServiceCenter to an intermediate table stored in an Oracle database accessible by the Connect-It Server and all OVO for UNIX Management Servers (usually but not necessarily the Oracle database of (one of) the OVO Management Servers is used). When this is completed, a Java program, running on all the OVO for UNIX Management Servers, connects to the intermediate table, copies the outage information into an “outage” file on the Management Servers, checks if its syntax is correct for processing by the OVO for UNIX Management Server, and uploads it onto the OVO for UNIX Management Server.

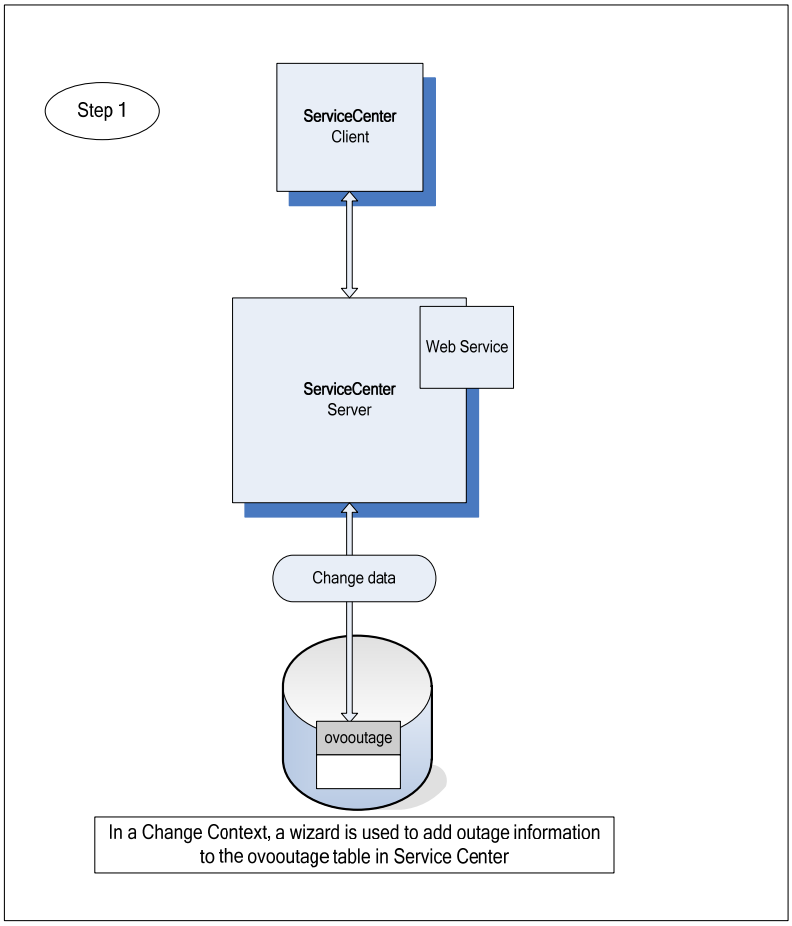
The logical architecture of this process is shown in the diagram below. Steps 1, 2 and 3 below provide more detail on sections of the process.



Outage Integration – Logical Architecture

Outage Integration Process

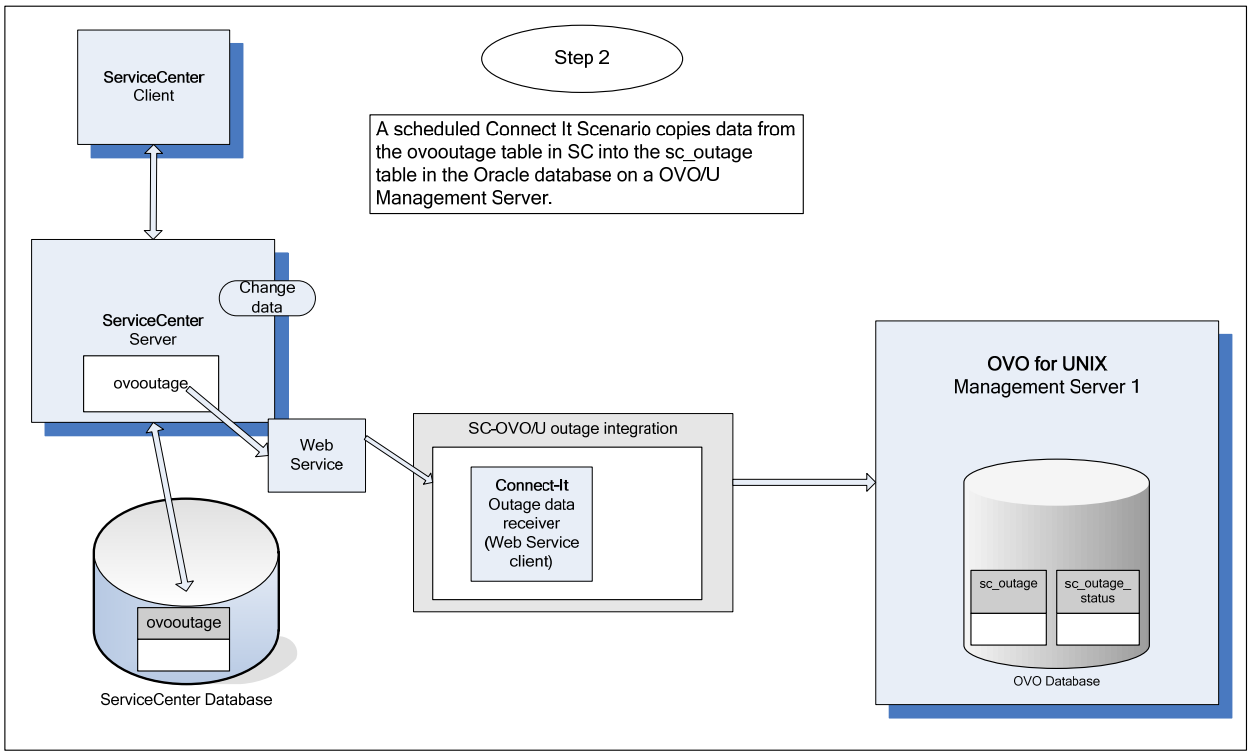
Step 1: Create an Outage Table Record in ServiceCenter



Adding outage information in ServiceCenter

The Outage Integration process is designed to allow users add outages to change processes. A wizard that can be run from within a change creates outages on CIs associated with the change. The outage information is stored in the `ovooutage` table in the SC database.

Step 2: Copy Outage Table Contents to Intermediate Table

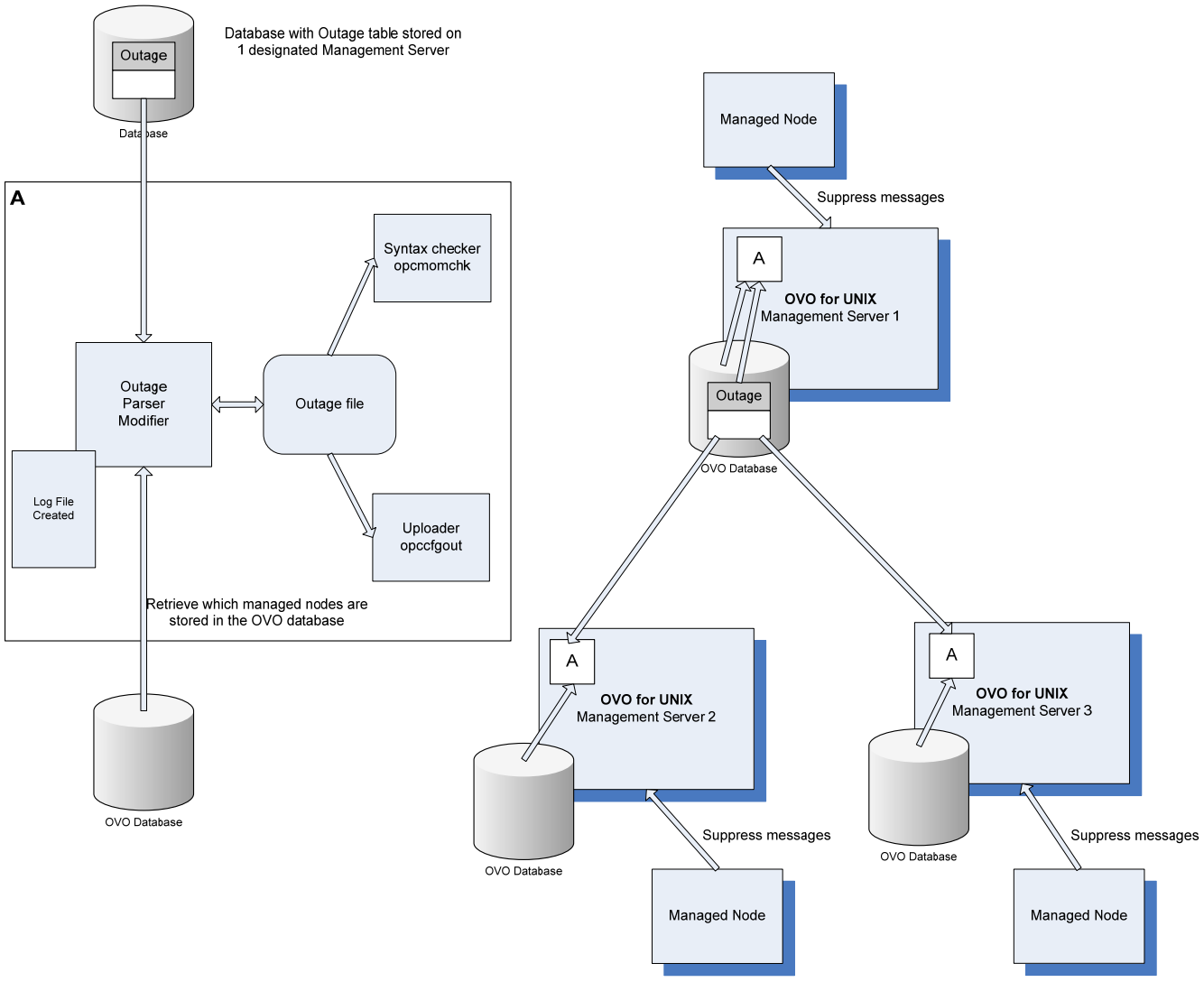


Copying outage information to an intermediate table

A Connect-It scenario is scheduled to connect via a SC Web Service connector to the `ovooutage` table and copy the contents of this table to the intermediate `sc_outage` table. You can configure the location of this intermediate table during installation and configuration (in the diagram it is located in a database on the OVO Management server but it could be located in any arbitrary Oracle database). On completion of the copy process, the Connect-It scenario changes the status of the Connect-It record in the `sc_outage_status` table. This table plays a part in synchronizing the Java process on the OVO for UNIX Management Servers with the Connect-It process on the Connect-It server. See *Schedule the Extraction and Upload Processes*

A predefined ServiceCenter Operator named “ovo” is supplied, which is set to use the Universal Time Coordinated (UTC) time zone. This helps to ensure that the outage date and time information conforms to the UTC time zone.

Step 3: Process Intermediate Table on OVO for UNIX Management Server



Processing the intermediate table in the OVO/U Management Server

On the individual OVO for UNIX Management Servers, the outage shell script is scheduled to run using the UNIX crontab scheduler. The script, shown as 'A' in the diagram, first checks if the Connect-It scenario has run – if its status is `finished` the script will run. The Java program called by the script makes a list of all nodes managed by the OVO for UNIX Management Server it is running on, and then extracts information on all relevant outages from the sc_outage (intermediate) table, in order to write it into the outage file on the OVO for UNIX Management Server. The "`opcmomchk -outage`" command is then used to check the syntax of the newly updated outage file; the "`opccfgout -update`" command is called to force the message manager to upload the new file.

Modifying the OVO Outage file

OVO for UNIX keeps all information regarding outage periods in one file, which contains the entire configuration. When OVO is instructed to read its outage template, it regards this as the whole set of outage information and removes any and all outage data it contained before. Therefore, the content of the outage template file is not under the exclusive control of the

Outage Integration, as the administrator of OVO may need to configure outage information that falls outside the scope of what the integration with SC offers.

To prevent the outage integration from disturbing sections of the outage template file that it is not responsible for, it delimits the sections it considers under its control with special comment codes. It is important that these delimiters are not removed when the outage template is edited manually.

Over time, the outage information configured in OVO's template will keep growing if outdated information is never cleaned up. Therefore, each time that the Outage integration runs, the first step it takes after reading the outage template file is to look for sections that contain information that has become useless. When the Outage integration rewrites the outage template file, these so-called obsoleted sections will be excluded from the updated template.

The Outage integration considers a subsection obsolete when one of the following rules apply:

1. The section contains an end-time that is more than 24 hours in the past compared to the current time on the OVO/U management server.
2. The section contains information on the current change that is being processed.

The second rule prevents that updates to outage information in SC creates multiple entries in the outage template file. In other words, simply deleting any template information that is related to the current update from the original outage template will ensure that the data in the outage template file has a one-to-one relation to the changes in SC.

3 System Specifications and Requirements

Supported Platforms

The following platforms are supported:

Windows

Solaris

HP-UX

The Connect-It user interfaces run on a Windows platform only. If you want to configure the scenarios and/or modify the mapping you need to do this on a Windows platform.

Supported versions of ServiceCenter and OpenView Operations

The Node Bank and Outage Integrations support ServiceCenter 6.1.3 and 6.2 on any platform that is supported by ServiceCenter (Windows, Linux, HP-UX, and Solaris). ServiceCenter running on the P4 database or on an Oracle database is supported.

The Node Bank and Outage Integrations support OVO 8.x on any platform that is supported by OVO (HP-UX and Solaris).

Prerequisites

Outage Integration

- To install and configure the Outage Integration, you must first install and configure the Node Bank Integration.
- The UNIX part of the Outage Integration needs a Java Runtime Environment (JRE) version 1.4 or higher.
- The UNIX part of the Outage Integration needs the Oracle utility `sqlplus`. Usually `sqlplus` is installed on the OVO for UNIX management server. If not, then install the Oracle Client software as described in *Appendix A*.

General Prerequisites

- Connect-It software should be installed before scenarios can be configured and run. Refer to the relevant Connect-It documentation for information on installing the product. The machine running Connect-It is referred to as the Connect-It server.
- Within Connect-It, the integrations use the ServiceCenter Web Service connector, the Oracle database connector, the XML connector, and the Data generator connector.
- For the Oracle database connector, the Oracle client software must be first installed on the Connect-It server. The Oracle client installation procedure is described in *Appendix A*. Connect-It must use the native Oracle database driver to connect to the Oracle database.

4 Installing Node Bank and Outage Integration

To install the Node Bank and Outage integration, you must have administrator rights on Windows platforms, or be a root user on UNIX platforms.

Node Bank Integration

Packaging

The files that constitute the Node Bank Integration are packaged in a self-extracting zip file for Windows.

The package naming convention is HpScNodeBank-MM.mm.bbb-Windows.exe. In this convention:

- MM refers to the major version number
- mm refers to the minor version number
- bbb refers to the build number

For example:

`HpScNodeBank-01.00.016-Windows.exe` for the Windows platform. The executable is a self-extracting zip file.

Installation

1. Copy the install package to the Connect-It server where you want to install the integration.
2. Unzip or extract the package.

   For a Windows platform, run the self-extracting zip file. By default the files are installed in the C:\Program Files\HP OpenView directory. If OpenView products are already installed on the same server but in another directory, then install the Node Bank Integration in this directory. If Node Bank integration is the first HP OpenView product on your install machine, you can select the install folder.
3. A ServiceCenter unload file forms part of the install package and must be imported into ServiceCenter using a ServiceCenter client. If a ServiceCenter client is not installed on the Connect-It server, copy the unload file `sc-nb-messagegroup.unl` to a host running a ServiceCenter client, or copy and unzip the installation package on the ServiceCenter client machine.

List of Installation Files

For a list of files included in the node bank installation package, and the location of the installed files, see *Appendix B*.

Outage Integration

Packaging

The Outage Integration consists of a Windows part and a UNIX part. The files that constitute the Outage Integration are packaged in a zip file for Windows and a GNU zipped tar file for UNIX.

The package naming convention is `HpScOutage-MM.mm.bbb-PP.zz` . In this convention:

- MM refers to the major version number
- mm refers to the minor version number
- bbb refers to the build number
- PP refers to the platform
- zz refers to the extension of the zipped file.

For example:

`HpScOutage-01.00.016-Windows.exe` for the Windows platform. The executable is a self-extracting zip file.

`HpScOutage-01.00.16-UNIX.tar.gz` for all UNIX-based platforms.

Installation

1. Copy the Windows install package to the Connect-It server where you want to install the integration.
2. Copy the UNIX install package to each OVO for UNIX management server where you want to install the integration.
3. Unzip or extract the packages.

   For a Windows platform, run the self-extracting zip file. By default the files are installed in the C:\Program Files\HP OpenView directory. If OpenView products are already installed on the same server but in another directory, then install the Node Bank Integration in this directory. If Node Bank integration is the first HP OpenView product on your install machine, you can select the install folder.

   For HP-UX or Solaris platforms, uncompress and extract the gz file using the following command:

   ```
   # gunzip -c HpScOutage-01.00.16-UNIX.tar.gz | tar -xvf -
   ```

   The files are installed in the /opt/OV and /var/opt/OV directories. You cannot select another install directory.

4. A ServiceCenter unload file forms part of the install package and must be imported into ServiceCenter using a ServiceCenter client. The unload file is shipped with the Windows package. If a ServiceCenter client is not installed on the Connect-It server, copy the unload file sc-outage6.1.unl or sc-outage6.2.unl depending on the version of ServiceCenter you are using to a host running a ServiceCenter client, or copy and unzip the installation package on the ServiceCenter client machine.

List of Installation Files

For a list of files included in the outage installation package, and the location of the installed files, see *Appendix C*.

5 Configuring Node Bank and Outage Integration

Configure Node Bank Integration

This consists of the following activities:

1. Importing the ServiceCenter unload file into ServiceCenter
2. Adding two fields to the ServiceCenter `device` file
3. Modifying the existing network address attribute
4. Configuring the Connect-It scenarios

Each of these activities is discussed below.

Import the unload file into ServiceCenter

1. Ensure that the ServiceCenter unload file, which is included in the installation package (see *Appendix B*), is accessible from the ServiceCenter client.
2. Start the ServiceCenter client.
3. Go to **Database Manager**.
4. Select **Import/Load**.
5. Navigate to the <InstallDir>/newconfig/conf/sc directory.
6. Select the `sc-nb-messagegroup.unl` file.
7. Start the import by clicking Load Foreground (F1)

Add ServiceCenter attributes

Data on the OVO managed nodes are stored in the device `dbdict` file. This file contains attributes of generic CIs. You must create two new attributes, `ovo.management.server` and `ovo.node.type`. The first attribute, `ovo.management.server`, will contain a list of OVO management servers that manage this node. The attribute `ovo.node.type` will be used to indicate if the node is a managed node or a management server. Note that the attribute names contain the leading string "`ovo`" to indicate that they are used for the OVO for UNIX Integration. The details are shown in the tables below.

OVO Management Server Array

Name	ovo.management.server
Type	Array
Caption	ovo.management.server
Include in API	true
Field name API	ManagementServer

OVO Node Type

Name	ovo.node.type
Type	Text
Caption	OVO Node Type
Include in API	true
Field name API	NodeType

The permitted values for this attribute are: “OVO Managed Node” and “OVO Management Server”.

When the Connect-It scenarios are running, the value of the field is first set to “OVO node being synchronized”

Editing the existing network address attribute

The integration uses **network.address**, an existing attribute of the device table, to store the IP address of the managed node. The network address must be published as a Web Service. The field name in the API must be `IPAddress`. The details are shown in the table below.

Name	network.address
Type	Text
Caption	network.address
Include in API	true
Field name API	IPAddress

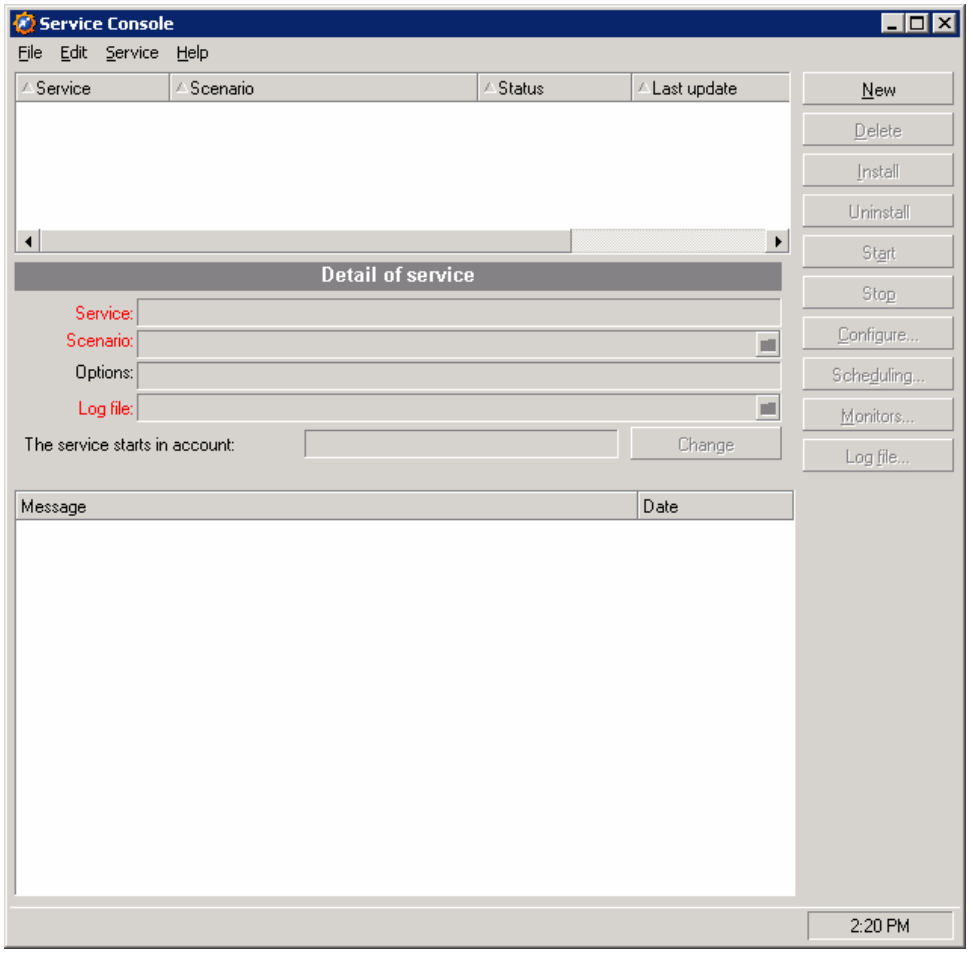
Configure Connect-It for Node Bank Integration

This section describes the following topics related to Connect-It components for Node Bank integration:

- Configuring the standard scenarios **sc-nb-clean** and **sc-nb** from the Service Console
- How to configure the Connect-It schedule from the Service Console
- How to create a Windows service for executing the scenarios from the Service Console
- Handling multiple OVO management servers
- Information on Logging – what and where

About the Service Console

The Service Console is a graphical tool for managing services that implement a Connect-It scenario. The console is installed as part of the Connect-It installation on a Windows machine. Using the console you can define, configure, and schedule a scenario, and then install and start a Windows service to implement the scenario. This section discusses how to use the console to configure, schedule and run the Node Bank integration.



Configure the Standard Scenarios

The Node Bank integration consists of two scenarios, **sc-nb-clean** and **sc-nb**. This section describes how to configure, schedule and start sc-nb-clean and sc-nb.

sc-nb-clean

This scenario clears the management server information and sets the node type to `OVO node being synchronized` for all OVO managed nodes in the ServiceCenter CMDB. In a set of scenarios, this scenario is always the first to run. It prepares the Service Center data for updates that will reflect the current Node Bank situation.

sc-nb

This scenario synchronizes the devices registered in the Service Center database with information from the OVO Node Bank. More than one version of this scenario can be run, generally one for each OVO Management Server.

Running the scenarios

The scenarios should be executed within a relatively short time interval. During this interval, the data in ServiceCenter are not complete, so it is important to keep it as short as possible.

The sc-nb-clean scenario must be run before running the other scenarios. The sc-nb-clean scenario deletes the data used to indicate to which management server each node is linked. Then, for each OVO Management Server in turn, a copy of the sc-nb scenario is run. The scenario synchronizes other data on the nodes,and adds the name of the management server to the list of management servers for each node known to the server. After all copies of the sc-nb scenario have run, the data are synchronized and all relations between nodes and management servers are registered in ServiceCenter as they currently exist.

Both scenarios are placed in the <DataDir>\conf\sc directory, where DataDir usually refers to C:\Program Files\HP OpenView\data.

Configure the sc-nb-clean scenario

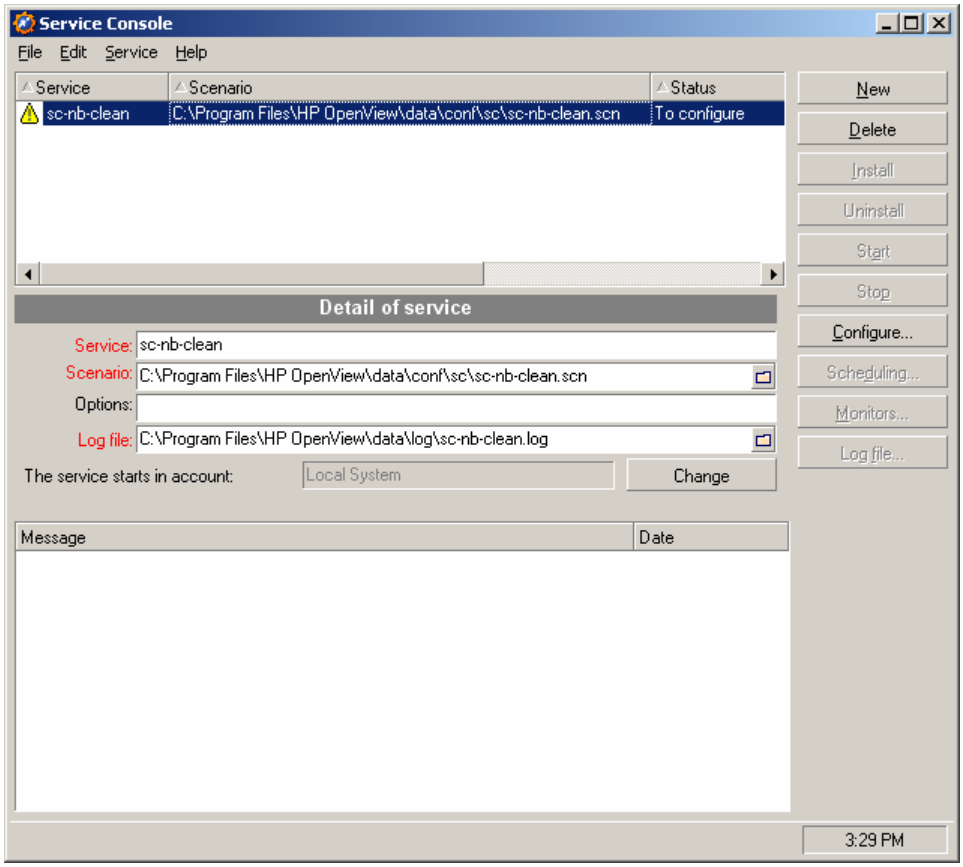
1. First create a new service for the Service Console: open the Service Console and click the **New** button.

   In the Service Console screen that appears, enter values for the fields shown below.

Service	Enter a name for the Windows service you want to create, for example sc-nb-clean. This will generate a service called **Connect-IT service (sc-nb-clean**).
Scenario	Enter the scenario file to use, normally C:\Program Files\HP OpenView\data\conf\sc\sc-nb-clean.scn

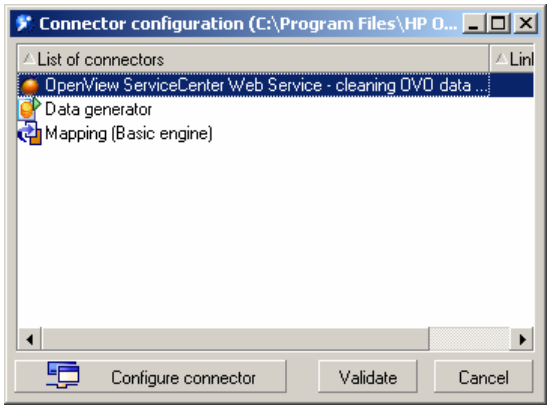
Log file	Enter a location for the Connect-It log. The suggested location is C: \Program Files\HP OpenView\data\log\sc-nb-clean.log. This is not the value that appears automatically when you enter a value in the Service field.

2. Click the **Create** button. You can now configure, schedule, install and start the service, in this order.



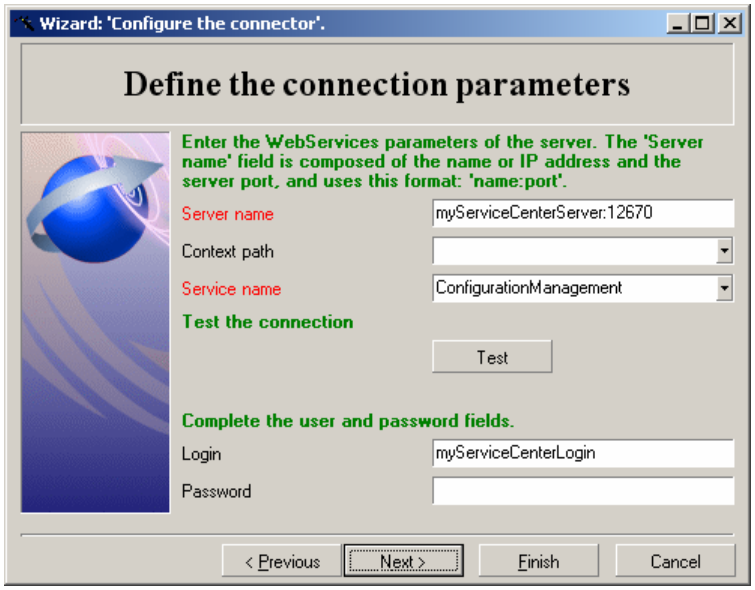
Configure the Connector

3. Click the **Configure** button to display a list of scenario connectors, as shown in the screen below.
4. You only need to configure the **OpenView ServiceCenter Web Service – clearing OVO data** connector. Place your cursor on this connector and click the **Configure connector** button.



5. In the first screen (**Name and Describe the connector**), enter a name and an optional description and click **Next**.
6. In the second screen (**Define the connection parameters)**) shown below, enter the following values:

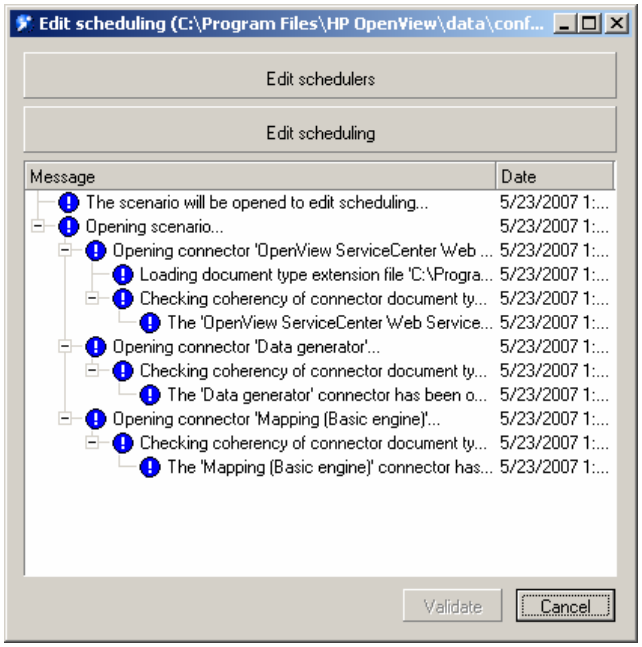
Server name	Replace the string myServiceCenterServer with the fully qualified domain name of the ServiceCenter server. If this server does not use the default port 12670 to run the ServiceCenter Web Services, enter the appropriate number.
Service Name	'Configuration Management'
Login	Enter a valid ServiceCenter account name to replace the myServiceCenterLogin string.
Password	Enter the password for the ServiceCenter account (if any).



7. Click the **Test** button to test the connection. If the test is successful, click the **Finish** button to finalize it. In the Connector configuration screen that reappears, click the **Validate** button to conclude the scenario configuration.

Scheduling

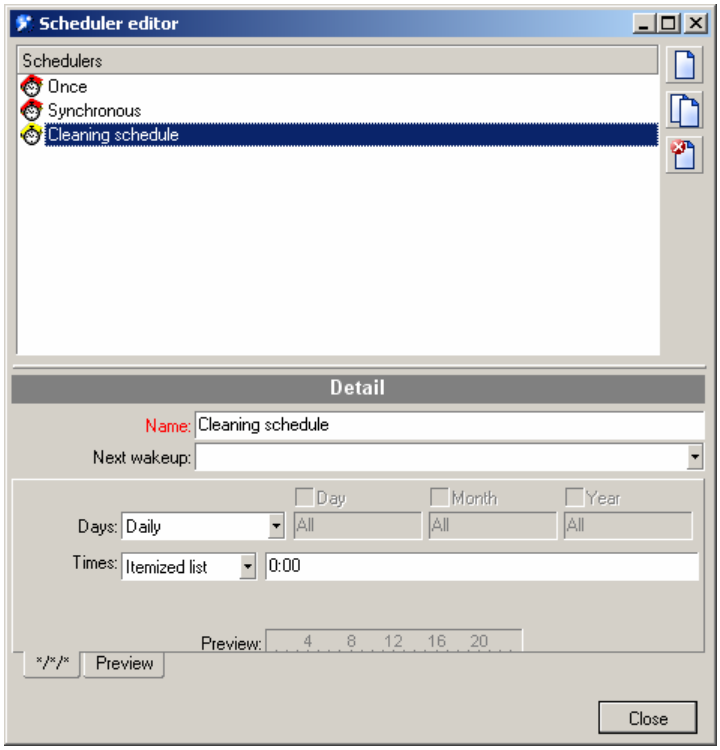
8. Once you have configured the connector, the **Scheduling** button in the Service Console screen is available. Click this button to display the **Edit Scheduling** screen. Click the **Edit Schedulers** button.



9. In the **Scheduler editor** screen, select the Cleaning schedule. This is pre-set to run every night at midnight, which can be adjusted. Ensure that you save any changes you make.

10. Click the **Close** button, and click the **Validate** to check the editor changes.

11. Close the **Edit scheduling** window.



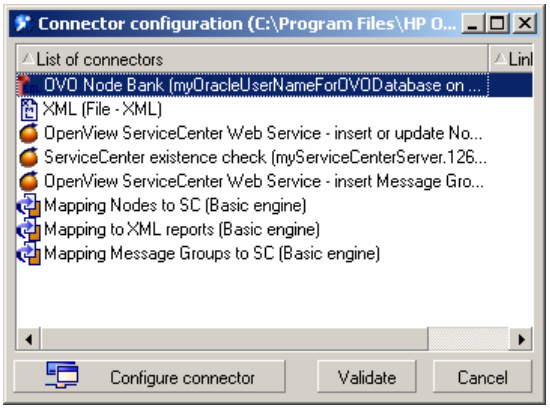
Configure the sc-nb scenario

You define this scenario in the same way the sc-nb-clean scenario was defined.

1. Click the **New** button and enter data in the fields listed below.

Service	Enter a name for the Windows service you want to create, for example sc-nb. This will generate a service called **Connect-IT service (sc-nb)**.
Scenario	Enter the scenario file to use, normally C:\Program Files\HP OpenView\data\conf\sc\sc-nb.scn
Log file	Enter a location for the Connect-It log. The suggested location is C: \Program Files\HP OpenView\data\log\sc-nb.log. This is not the value that appears automatically when you enter a value in the Service field.

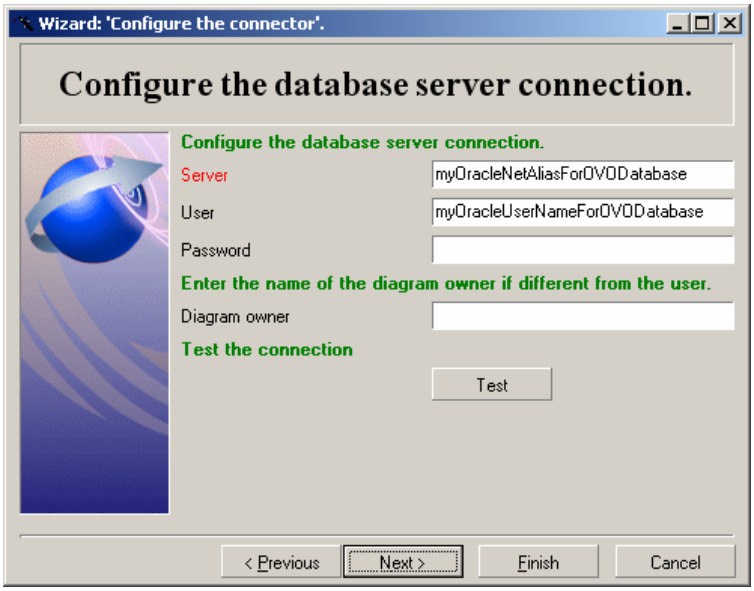
2. When you click **Configure**, a longer list of connectors appears. Four of these need to be configured.



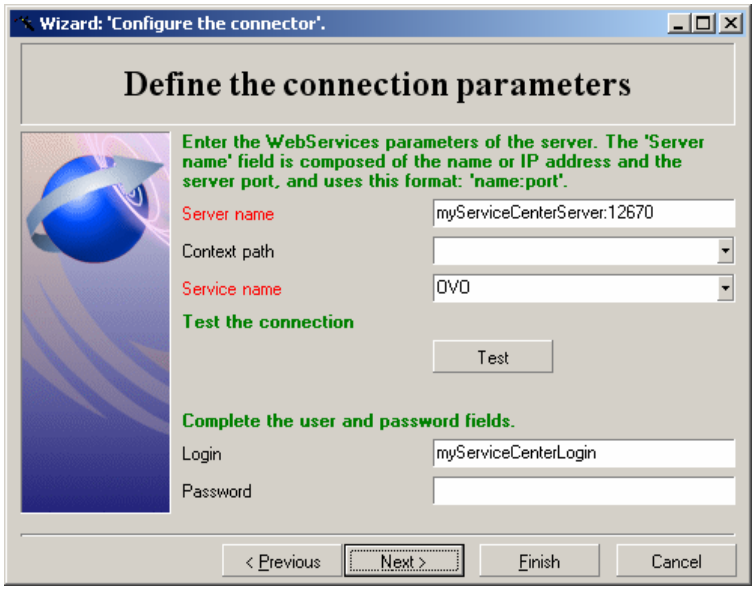
3. The **OVO Node Bank** connector is configured on the third screen of its wizard. Enter values according to the following table.

Server	Replace the placeholder 'myOracleNetAliasFor OVODatabase' with the Net Service Name for the OVO database server. This is (one of) the Net Service Name(s) defined in the *Net Configuration Assistant*.
User	Replace the placeholder 'myOracleUserNameFor OVODatabase' with the user name for the Oracle account holding the OVO data (generally **opc_op**)
Password	Enter the password related to the user name given above.

4. Click the **Test** button to test connection, and when successful, finalize it with a click on the **Finish** button.



5. Configure the **HP OpenView ServiceCenter Web Services – insert or update Nodes** connector. The procedure is exactly identical to the connection described for the ***OpenView ServiceCenter Web Service – clearing OVO data*** connector..



6. Configure the **HP OpenView ServiceCenter Web Services – insert Message Groups**.. Use values as given below (the only difference with the previous connector is in the **Service Name**).

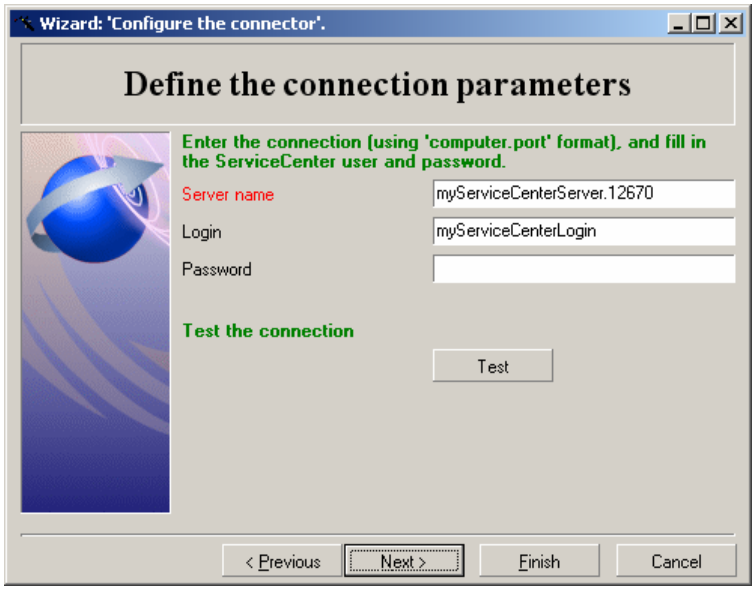
Server name	Replace the string myServiceCenterServer with the fully qualified domain name of the ServiceCenter server. If this server does not use the default port 12670 to run the ServiceCenter Web Services, enter the appropriate number.
Service Name	'OVO'
Login	Enter a valid ServiceCenter account name to replace the myServiceCenterLogin string.
Password	Enter the password for the ServiceCenter account (if any).

7. Finally, you configure the **ServiceCenter existence check** connector on the second screen of the wizard. Use values as speciefied in the table below. Use values as given below.

Server name	Replace the string myServiceCenterServer with the fully qualified domain name of the ServiceCenter server. If this server does not use the default port 12670 to run the ServiceCenter Web Services, enter the appropriate number.
Login	Enter a valid ServiceCenter account name to replace the myServiceCenterLogin string.

Password	Enter the password for the ServiceCenter account (if any).

8. **Test** this connection, **Validate** the connector configuration, and return to the Service Console's main screen.



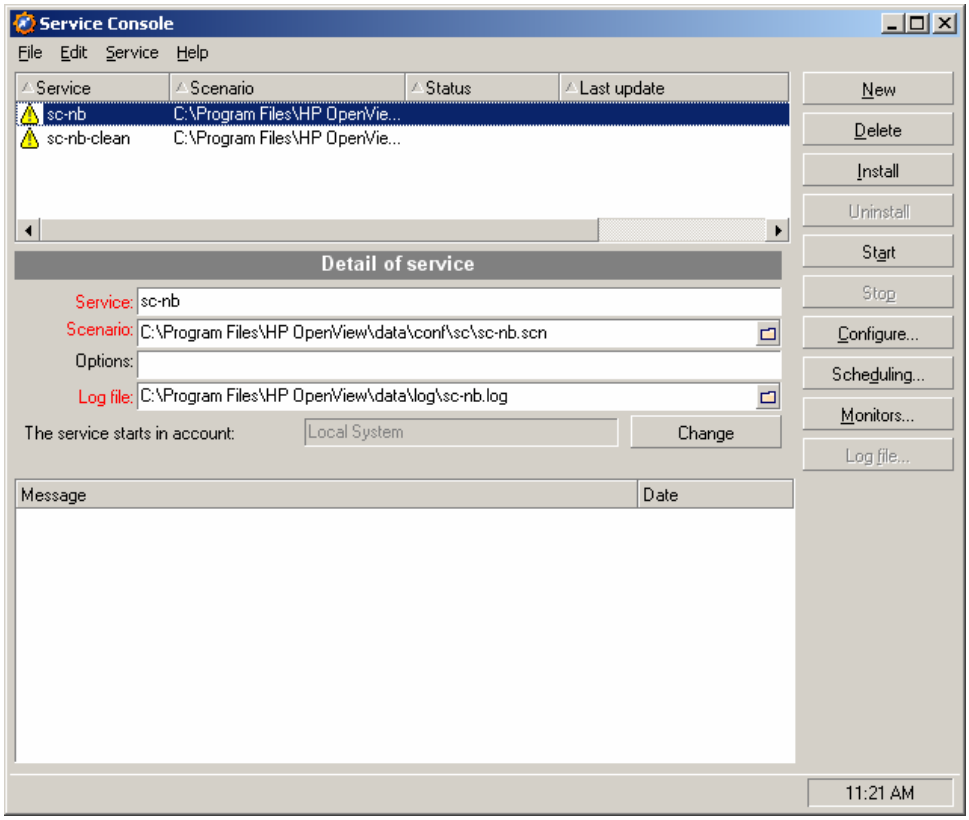
Scheduling

From a technical perspective, scheduling for this scenario is identical to scheduling the **sc-nb-clean** scenario. When installed, the scenario is scheduled to run every night at five minutes past midnight. When configuring this scenario, make sure the sc-nb-clean scenario will have finished before the sc-nb scenario starts.

Create and start services

The Service Console now looks similar to that shown below. Two services have been defined, but these have not yet been created as Windows services, nor have they started executing.

To create a Windows service, select the service and click the **Install** button. Connect-It creates a service for the selected Service. It will be named Connect-It service (<service name>, so clicking the **Install** button in this example will create the service **Connect-It service (sc-nb)**. The services you create are Windows services that can be monitored and managed through the services interface of the Windows operating system.



Once the service has been created, ensure that it starts automatically when your machine the Connect-It server boots up. Do this by making it an Automatic service in the Windows services interface. Once installed, the service can also be started and stopped from the Service Console by clicking the **Start** button or **Stop** button. To remove the service, click the **Uninstall** button.

Configure Multiple OVO Management Servers

When more than one OVO Management Server is involved in an integration, a separate copy of the sc-nb scenario must be run for each Management Server. To do this, configure one scenario completely and then make copies of this scenario. In doing so, a major part of the configuration work (configuration of three ServiceCenter connectors) can be reused.

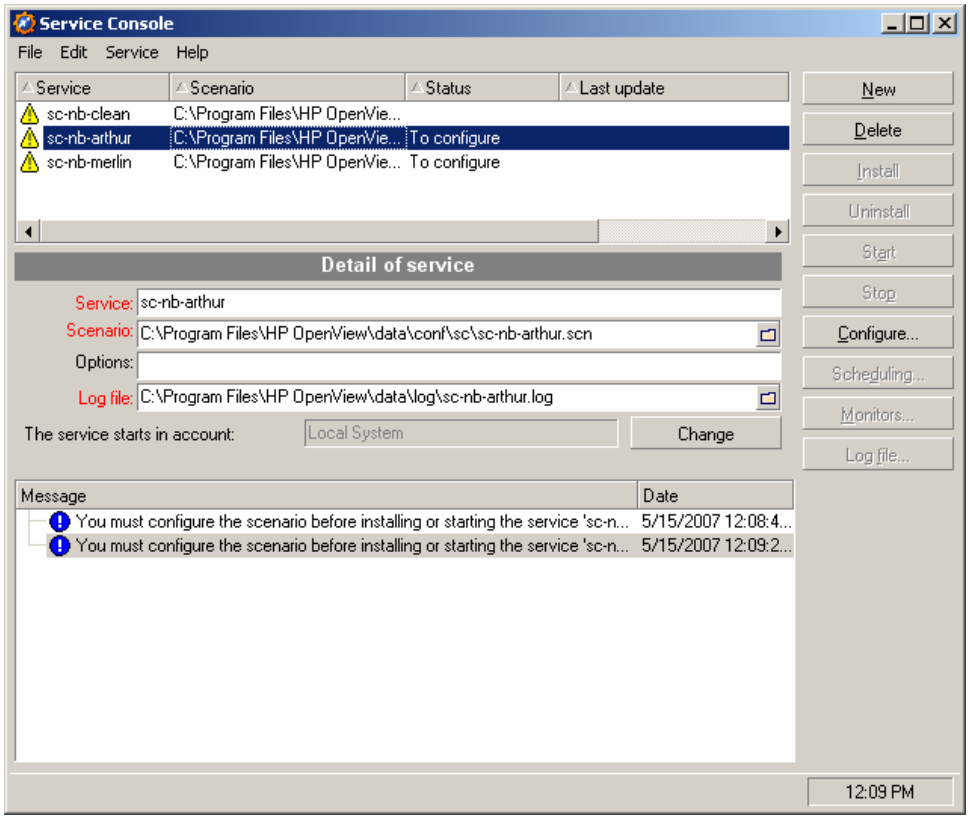
A working integration consists of several scenarios running in close succession. First, the sc-nb-clean scenario is run. This deletes the data that indicate to which management servers each node is linked from the ServiceCenter CMDB. This scenario also changes the Node Type field in Service Center to read “OVO node being synchronized”, thus indicating that integration is running. Then, copies of the sc-nb scenario, created for each management server, are run in succession. These scenarios update the information in ServiceCenter on the nodes known to the management server, including the management server name. After all copies have run, all management servers related to a given node are registered in ServiceCenter as they are at that point in time.

It is advised to make sure that the different scenarios do not run at the same time. There is a distinct possibility that one node will be affected by more than one scenario, as it may be known to multiple management servers. All nodes managed by OVO will certainly be affected by the sc-nb-clean scenario. This is typically a situation in which concurrency problems may occur, leading to lost updates, for example.

To avoid scenarios running simultaneously, one should be aware of the time needed to run each scenario. The scenarios should then be configured to run consecutively.

Copy scenario files

For each OVO Management Server, copy the original scenario file and its accompanying initialization file. These files are located in the <DataDir>/conf/sc directory, normally C:\Program Files\HP OpenView\data\conf\cs. Note that the `sc-nb.cfg` and `sc-nb.bas` files must remain in their original location, or the new scenario will not work. These files are shared by all scenarios.



For convenience and to avoid errors, you should rename the files by adding the name of the OVO management server to the file name. For example, if the hostnames of your OVO Management Servers are `merlin.mydomain.com` and `arthur.mydomain.com`, assign the names `sc-nb-merlin.scn` and `sc-nb-arthur.scn` to the scenario files, and `sc-nb-merlin.ini` and `sc-nb-arthur.ini` to the initialization files,

This is also a good moment to create additional directories for each scenario's log files. The scenarios can thus write their log files to separate directories.

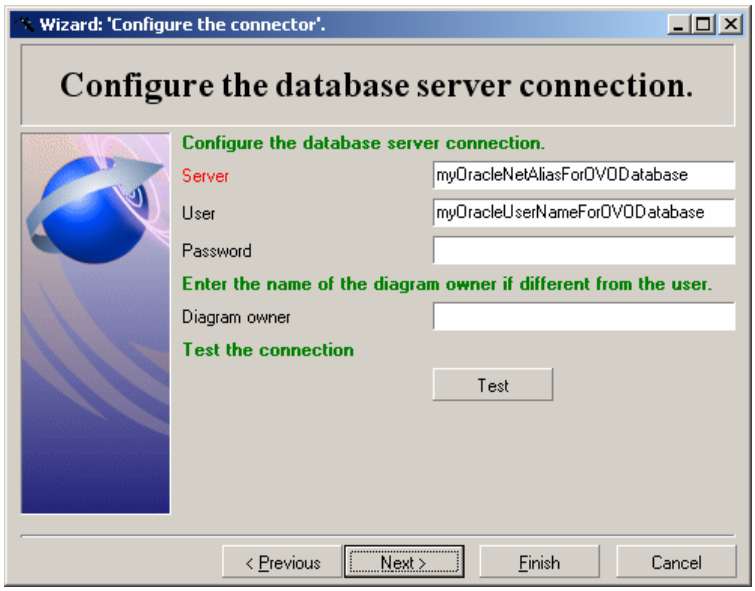
This avoids potential confusion as to the origin of the log files. The preferred naming of these directories is the name of the OVO management server. So in the above example the directories C:\Program Files\HP OpenView\data\log\arthur and C:\Program Files\HP OpenView\data\log\merlin would be created.

Configure the copied scenarios

The screenshot of the Service Console above shows the two example scenarios already added. Services have been added in the same way as they were defined for the standard scenarios. Assuming the original scenario had already been configured, the copied scenarios need only minor adjustments.

OVO Node Bank

Make sure the **OVO Node Bank** database connector is *configured* to use the database associated with the proper OVO management server. In the third screen of its wizard, type the Net Service Name for the OVO database server in the Server field, and the user name and password for the Oracle account holding the OVO data in the 'User' and 'Password' fields.

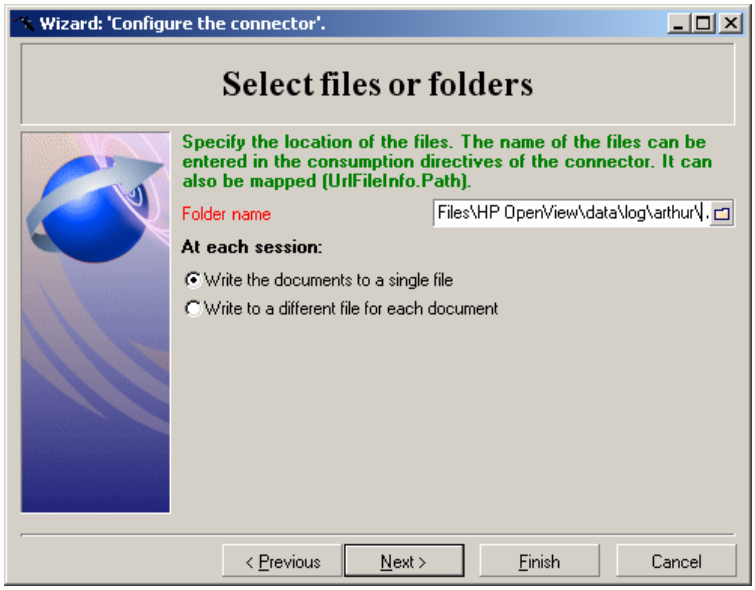


Click the **Test** button to ensure that the connection parameters are correct. Finalize the configuration by clicking the **Finish** button.

XML

The **XML** connector must be configured to write the log files for each connector to the separate directory for which it was created. This directory must be specified on the fourth screen of this connector's wizard.

After these two changes, **Validate** the connector configuration, and return to the Service Console main screen.



Scheduling

Scheduling these scenarios is done in the same way as for the other scenarios. Make sure that each scenario runs in its own timeframe, thus avoiding possible concurrency issues. A close inspection of the Connect-It logs for the scenarios makes clear how much time is required by each scenario for a full run. Scheduling can be based on this data.

If we assume the following times (minutes, seconds) for each scenario:

sc-nb-clean takes 2‘ 25“ to run

sc-nb-arthur takes 4‘ 33“ to run

sc-nb-merlin takes 6‘ 13“ to run

Obviously, sc-nb-clean can remain scheduled to run at midnight. If we allow for an adequate time-interval, sc-nb-arthur can be started at 00.05 a.m. and sc-nb-merlin at 00.15 a.m. This allows for a generous measure of change before concurrency would become a problem, and at the same time it ensures that the synchronization process will be finished by 00.25 AM. Based on this schedule, ServiceCenter operators can be warned that the information on OVO management servers listed with a device may not be entirely reliable between midnight and 00.30 AM.

Create and start services

Once all services are properly scheduled, these can be installed as Windows services and started as described above.

Log files

Service log

As each service executes, status and error messages are written to the Connect-It log file. This file is created in the location defined in the Service

Console for each service. The logs contains general information about each execution of the Connect-It scenario, including whether it has run successfully or not. The log file includes information on the following types of errors:

- Mandatory files that are missing or damaged
- Incorrectly configured connectors
- Mapping errors
- Document Type Definition errors

A

```
2007/04/11 11:05:0.164  3     2     Document(s) produced ignored: 14
2007/04/11 11:05:0.164  4     4     CreateDevice: 7
2007/04/11 11:05:0.180  4     4     UpdateDevice: 7
```

f

Apart from these general points, the Connect-It log will also report on any information retrieved from the OVO database and not transferred to ServiceCenter, because it already exists or because it has not changed. These nodes and message groups will be recognized in entries similar to the extract shown below.

Note that the number of node-related documents ignored may be higher than the number of documents produced by the OVO Node Bank connector. This is because each document is evaluated twice, once to see whether an insert should be done, and a second time to see whether existing data should be updated. As message group data is only inserted, and never updated, these are only evaluated once.

Insert/update xml logs

Each time a scenario is executed, information on the inserts and updates that were performed is logged in XML format. By default these log files are located in C:\Program Files\HP OpenView\data\log, but a different location can be specified with the configuration for the XML connector. These log files are named as follows:

`sc-nb-NewNodeResultxxx.xml` (node inserts)

`sc-nb-UpdateNodeResultxxx.xml` (node updates)

`sc-nb-MessageGroupResultxxx.xml` (message groups).

Here, 'xxx' refers to a sequence number that will be incremented each time a file is generated.

Be aware that new instances of these log files are not created every time a scenario is run – this only happens when inserts and/or updates are performed, in other words when at least one document is actually processed. As the Connect-It log shows the numbers of documents that were processed, it indicates whether the log files exist.

An example of the xml files is shown here. In this file, each `<DeviceResponse>` tag contains data on one OVO node being transferred. The values are:

Name	The machine name or the message group name, which is used as a key
Status	SUCCESS or FAILURE for each node
Message	May provide additional information
Zero, one or more DetailMessage entries	Provides more detailed error information.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<DeviceResponseList>
  <DeviceResponse>
    <Message>Success</Message>
    <Status>SUCCESS</Status>
    <Name>merlin.mydomain.com</Name>
    <Messages>
      <DetailMessage>
        <Message>CI record updated.</Message>
      </DetailMessage>
    </Messages>
  </DeviceResponse>
</DeviceResponseList>
```

Configure Outage Integration

This section describes the activities listed below.

1. Import the ServiceCenter unload file file into ServiceCenter.
2. Manually configure existing files and objects in Service Center. This consists of the following tasks:
   - Add two new outage fields to the `dbdict` file.
   - Create new outage fields in any required device formats or subformats.
   - Add alias fields in the `cm3r` and `cm3t` files.
   - Add new link lines for Change and Change Task.
   - Amend the `cm.view` State record.
   - Add new Tab and Sub-form to Change and Change Task formats.
   - Add a new tab to the related records Tab.
   - Add a subform to the Change or Change Task formats.

   In addition, the following administrative tasks are described in *Appendix D* and *Appendix E*:
   - How to configure the OVO Outage table clean-up schedule
   - How to configure the `slamodulecontrol` table
3. Configure the upload process on each OVO Management Server.
4. Schedule the Extraction and Upload Processes.
5. Configure the Connect-It scenario for outage integration.

Import the unload file into ServiceCenter

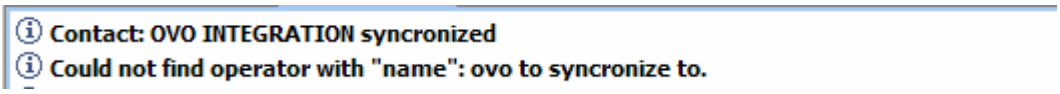
The procedure is as follows:

1. Ensure that the ServiceCenter unload file included in the installation package for Windows is accessible from the ServiceCenter client.
2. Start the ServiceCenter client.
3. Go to **Database Manager**.
4. Select **Import/Load**.
5. Navigate to the <InstallDir>/newconf/conf/sc directory.
6. Select the correct unload file for the ServiceCenter version you are using. Select `unload file sc-outage6.1.unl` for SC 6.1 or `sc-outage6.2.unl` for SC 6.2.
7. Start the import by clicking Load Foreground (F1)

   The error messages shown in the screen below will appear when you load the unload file, but can be ignored. The error messages result from related information between certain objects being loaded in a different order, that is,

the message appears before the display screen is loaded. The message is displayed because the display options are loaded before the display screen, and cannot therefore reference the display screen.

The following error message relates to SC v6.2 unload only



In this case the contact record is loaded before the operator record, and at that point cannot relate to the operator record. Once the operator record loads, the two records are synchronized.

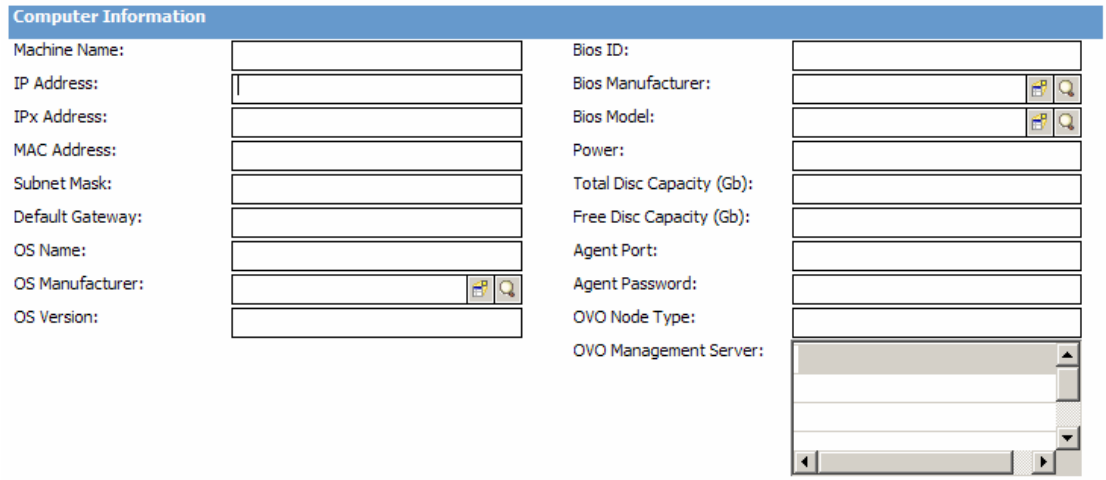
Manually configure existing ServiceCenter files and objects

Add Outage-related fields to Device File

Add two fields (attributes) `ovo.node.type` and `ovo.management.server` to the `dbdict` file. This procedure is described in the section on Node Bank integration. See *Adding Fields to device file*.

Add Outage-related fields to Formats or Subformats

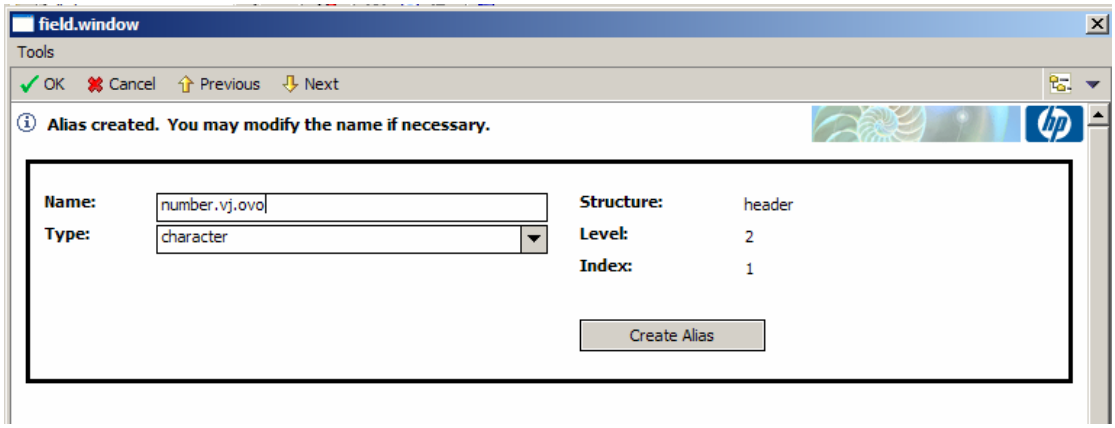
Add the new OVO Outage-related fields from the device table (described in *Adding Fields to device file*) to any required formats/subformats, for example to `device.computer.info`. This depends on the formats used by the client.



Add alias fields in the cm3r and cm3t files

These fields are required for linking the OVO Outage Sub-form to the Change and Task forms.

1. Type `dbdict` from the ServiceCenter command line.
2. Enter `cm3r` in the File Name field and click Search.
3. Position your cursor on the number field and click Edit.
4. Click Create Alias and enter `number.vj.ovo` as alias.
5. Repeat this procedure for `cm3t`. The entry details are shown below.



Change

Filename: `cm3r`

Add an alias for the number field with the following name and type:

Name: `number.vj.ovo`

Type: `Character`

Change Task

Filename: `cm3t`

Add an alias for the number field with the following name and type:

Name: `number.vj.ovo`

Type: `Character`

Add two link lines for Change and Change Task

Perform the procedure described below twice – for the Link records for Change (**Link: cm3r**) and for Change Task (**Link: cm3t**).

In the Change and Task formats, the OVO Outage records are displayed in the Sub-form on the Related Records Tab. This Link change makes it possible to open the OVO Outage record when it is double clicked from the Sub-form list

First new link line

1. Enter `link` from the command line.
2. Enter the name of the link (cm3r or cm3t) and click **Search**.
3. Scroll to the end of the Source Field Names column.
4. Click **Add**.
5. Add the first new link line. The input is shown below.

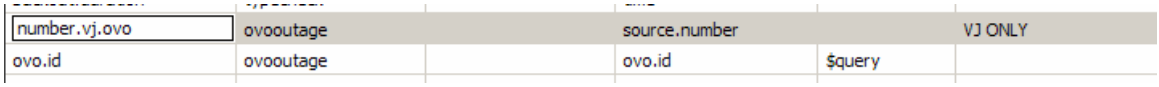
   Source Field Name: `number.vj.ovo`

   Target File: `ovooutage`

   Target Field Name: `source.number`

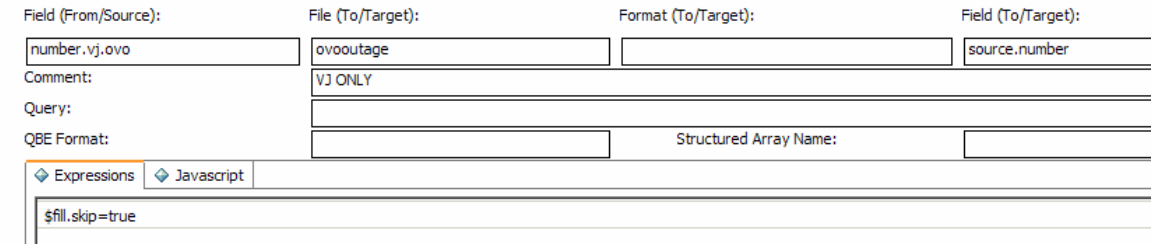
   Add query:

   Comments: VJ ONLY



6. Save the record and click **Select Line** for the link structure for the new line.
7. Add the following information and save.

   Expressions: `$fill.skip=true`



Second new link line:

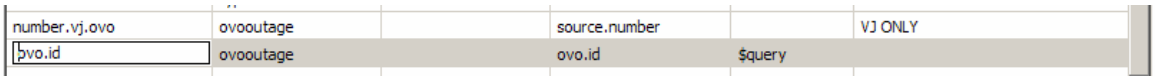
8. Add the second new link line. The input is shown below.

   Source Field Name: `ovo.id`

   Target File: `ovooutage`

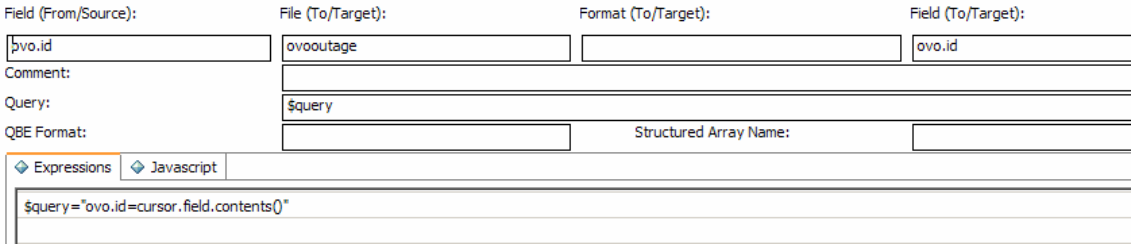
   Target Field Name: `ovo.id`

   Add query: `$query`



9. Save the record and click **Select Line** for the link structure for the new line.
10. Add the information below and save.

    Expressions: `$query= "ovo.id=cursor.field.contents()"`



Amend the cm.view State record

1. Enter `doc` from the ServiceCenter command line.
2. Select the States icon from the list displayed.
3. On the State Definition form, enter the values shown below.

   State: `cm.view`

   Display Action `ovooutageadd`

   Process: `ovo.outage.add`

   Condition: `true`

   Save First: `true`

State Definition

State: cm.view

Display Screen: cm.view.display

Initialization Process: cm.update

Format: view.default in $L.phase

Input Condition (view state only):

Non-base methods

Display Action	Process Name	Condition	Save First
approvedocumentint	cm.km.approveint	true	
retiredocument	cm.km.retire	true	
revertdocument	cm.km.revert	true	
transferdocownership	cm.km.transfer	true	
editdocument	cm.km.edit	true	true
km.addfeedback	cm.km.addfeedback	true	
km.delfeedback	cm.km.delfeedback	true	
km.loadfbtext	cm.km.loadfbtext	true	
approvedocumentext	cm.km.approveext	true	
kmsearch	kmmappedsearch	true	
nextkmphase	cm.km.next.phase	true	true
fillkmowner	cm.km.select.owner	true	
fillkmassignee	cm.km.select.assignee	true	
unowned	cm.km.unowned	true	
assignedtome	cm.km.assignedtome	true	
ovooutageadd	ovo.outage.add	true	true

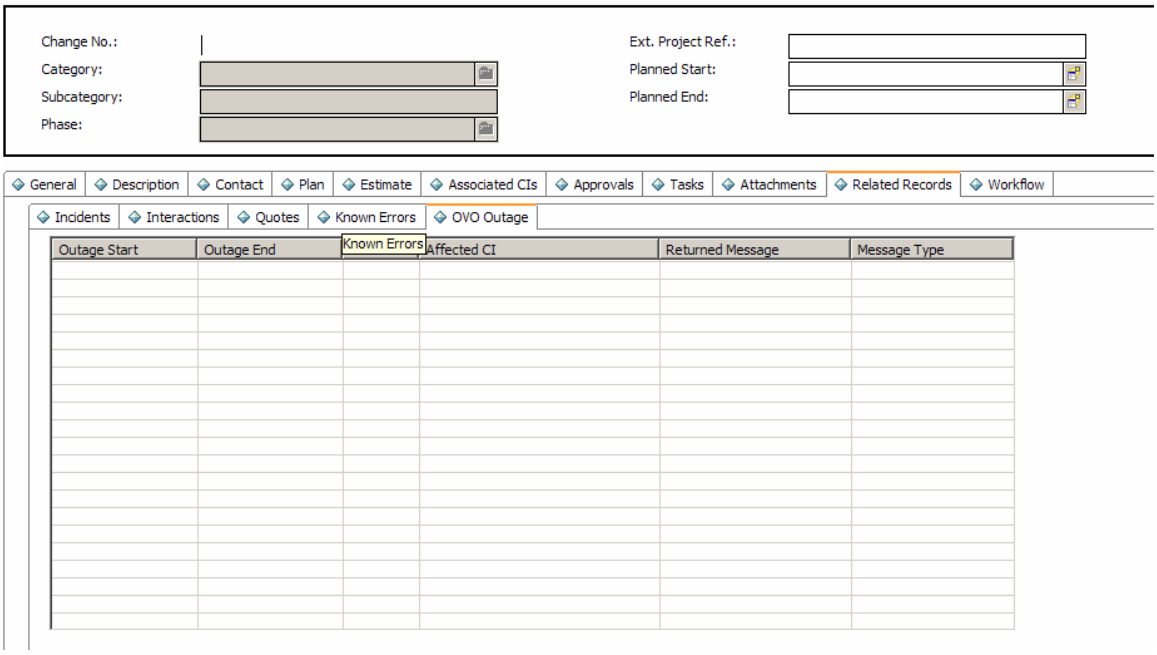
Add new Tab and Sub-form to Change and Change Task formats

The Change and Task Formats requiring this Sub-form are dependent on the format used by the client.

The unload file contains an example format (`cm3r.ovo.default.g`) on which the OVO Outage Tab is displayed.

On the required forms, for example `cm3r.plan.default.g`, add a new Tab to show related OVO Outage records.

Add a new tab to the related records Tab



Add a subform to the Change or Change Task formats

The properties are shown below.

Format:	`ovooutage.vj` (This is a Sub-form included in the unload file)
Virtual Join:	`true`
Display Blank:	`false`
Display Using Table:	`true`
Input:	`number.vj.ovo`

This concludes the manually configuration of ServiceCenter

Configure the upload process on each OVO Management Server

From the /opt/OV/bin directory, run the setup shell script `ovsc-setup.sh`. The script does the following:

- Creates the configuration files required by the SC-OVO/U Outage integration
- Creates the intermediate outage tables `sc_outage` and `sc_outage_status` in the Oracle database (usually but not necessarily the Oracle database of (one of) the OVO Management Server)
- Registers the OVO Management Server in the `sc_outage_status` table

The `ovsc-setup.sh` script prompts the use for input related to the following areas:

- The locally installed Oracle environment
- How to connect to the Oracle database that will hold the intermediate outage tables
- How to connect to the Oracle database used by the OVO Management Server

Run the `ovsc-setup.sh` script on each OVO Management Server participating in the Outage integration.

Schedule the Extraction and Upload Processes on each OVO Management Server

The extraction process on the Connect-It server is scheduled using the Connect-It built-in scheduler. For information on scheduling the outage scenario on the Connect-It server, see *Configuring the outage scenario*.

The upload processes on the OVO Management Servers are scheduled using the crontab utility supplied with all UNIX-based operating systems. Form more information on scheduling the upload processes, see the section on *Schedules on UNIX* . The crontab scheduler must be configured on each OVO Management Server.

The schedules for the extraction on the Connect-It server and for the upload processes on the OVO Management Servers must be accurately timed to ensure that they don't run simultaneously. Incorrectly scheduled processes will result in unnecessary server load.

The frequency of processes or the time interval between the processes depends on latency that is still acceptable between the definition of the outage in SC and the effect in OVO. For example, if changes or change tasks are defined well in advance, a frequency of synchronization of 1 per hour will be sufficient. One should schedule the extraction process on the Connect-It server to occur just before the upload processes on the OVO for UNIX management servers will run. The duration of the upload processes can be viewed using the `ovsc.sh` command line utility. Enter `ovsc.sh -status` from the command line.

An example of the output of the `ovsc.sh -status` command is shown below.

```
Management server            Status       Last run            Duration
---------------------------- ------------ ------------------- --------
Connect-It server            Finished     01-06-2007 10:00:00 00:01:12
ovoserver1.mydomain.com      Finished     01-06-2007 10:04:14 00:00:13
ovoserver2.mydomain.com      Running      01-06-2007 10:05:27 00:00:00
ovoserver3.mydomain.com      Ready to run 01-06-2007 09:06:15 00:00:14
ovoserver4.mydomain.com      Ready to run 01-06-2007 09:07:09 00:00:08
```

The example shows the extraction process on the Connect-It server run at 10:00 and took 1 minute and 12 seconds. The upload proceses on the 4 OVO

management servers are scheduled to run every hour at 4, 5, 6 and 7 minutes past 10:00.

Background information about scheduling the extraction and upload processes

The extraction process of the Outage Integration runs ons the Connect-It server whereas the upload process runs on the OVO for UNIX Management Servers. All processes need to be synchronized, i.e. the extraction processes and the upload process are not allowed to run simultaneously. During the update of the intermediate `sc_outage` table by Connect-It the upload process is not allowed to read the table to prevent data inconsistency.

All processes run on separate hosts and must notifiy each other about their running or finished task. The communication is perfomed through the database that holds the `sc_outage_status` table. This table is used to keep both processes informed about the status of each other.

The table `sc_outage_status` contains the following columns:

Column	Type
MGMTSERVER	VARCHAR2(80)
STATUS	VARCHAR2(12)
LASTRUN	DATE
DURATION	VARCHAR2(8)

Each OVO for UNIX Management Server is represented by one record in the table. There is also a record for the host that runs the Connect-It process. The mgmtserver column contains the hostname of the OOV for UNIX management server or the Connect-It server. The value of the mgmtserver column of the record that represents the Connect-It server is ‘Connect-It server’.

The status column contains the current status of the process running on the management server. The values of the status column are:

- Ready to run – The process is ready to run
- Running – The process is currently running
- Finished – The process has finished
- Failed – The process has failed
- Stopped – The process doesn not participate in the process flow

The column lastrun contains the date and time the process has run for the last time. The column duration contains the time elapsed between the start and the end of the process.

The process flow is as follows. The extraction process runs at scheduled intervals on the Connect-It server. Next the upload processes run on each

OVO management server. Only when all upload processes have finished the extraction process runs again.

The process flow is implemented as follows. The extraction process is scheduled using the built-in scheduler of Connect-It. The upload processes are scheduled using the scheduler of the Operating System.

When the extraction process start to run it first checks if no upload processes are running on the OVO Management Servers. It does so by checking the status column of the record in the `sc-outage_status` table that represents the Connect-It server. If the status is 'Ready to run' then it changes the status to 'Running' and starts to run. If the status is anything else then it just exits and will start at the next scheduled run. Once the extraction process has finished it changes its status to 'Finished'.

When an upload process starts to run on an OVO management server it first checks the status of the extraction process by checking the status column of the Connect-It server record. If the status is 'Finished' then it changes the status of its record to 'Running' and starts to run. If the status is anything else then it just exits and will start at the next scheduled run. Once the extraction process has finished it changes its status column to 'Finished'. It then checks the status of the upload processes of the other OVO Management Servers. If all of them are 'Finished' then it changes the statuses of all management servers including the status of the Connect-It server to 'Ready to run'.

When the upload process on an OVO management server cannot run, for example when the server is down or disconnected from the network, then this could cause a standstill of the whole synchronization process. Therefore, this management server must be removed from the list of participating servers by setting the value of the status to 'Stop' for this management server. This is a manual action and can be performed on any OVO management server with the comand `ovsc.sh –stop <mgmtserver>`. To add the server to the list of participating servers the status must be set to 'Ready to run'. This can be done by issuing the command `ovsc.sh –start <mgmtserver>`.

When the upload process on an OVO/U management server fails, for example when the OVO database is not accessible, then the value of status is set to 'Failed'. However, the server will continue to participate in the synchronization process. With the next scheduled cycle of the failed last upload process the status of the failed server is set to 'Finished' upon succesful completion of the process.

Collisions of the upload process on the same OVO Management Server, that is when two or more processes are running simultaneously, are prevented by using a lock file. When the process starts it looks for the lock file. If the lock file exists the process exits immediately. If the lock file does not exist it is created. Just before the process has finished it removes the lock file. By default the lock file is `sc_outage.lck` and is located in the directory $DATADIR/tmp.

The command line interface `ovsc.sh` allows you to view or modify the status of the processes in the `sc_outage_status` table. Additionally `ovsc.sh` allows you to view the contents of the `sc_outage` table that holds the outage data extracted from ServiceCenter. `ovsc.sh` accepts the following parameters:

Parameters	Description
–help	Show detailed help
–version	Show the version of the ovsc.sh command
–status	Show the status of the processes on all servers
–start [<OVO Management Server>]	Set the status of the specified OVO Management Server to 'Ready to run'. When the OVO Management Server is omitted the local server is used.
–stop [<OVO Management Server>]	Set the status of the specified OVO Management Server to 'Stopped'. When the OVO Management Server is omitted the local server is used.
–register [<OVO Management Server>]	Insert a record for the specified OVO Management Server in the sc_outage_table. When the OVO Management Server is omitted the local server is used.
–unregister [<OVO Management Server>]	Delete a record for the specified OVO Management Server from the sc_outage_table. When the OVO Management Server is omitted the local server is used.
–outage	Show the outage information that is retrieved from ServiceCenter and stored in the intermediate sc-outage table.
–outagedetail <OVO id >	Show detailed information of a given outage. The OVO id of the outage record is mandatory and can be found using the outage command

Example output of the command `ovsc.sh –outage`:

```
OVO id  Node                    Start time       End time        Operation
------- ---------------------- ---------------- ---------------- --------
OVO1    mngdnodeA.mydomain.com  02-06-2007 02:00 03-06-2007 23:00 SUPPRESS
OVO2    mngdnodeA.mydomain.com  02-06-2007 17:00 02-06-2007 22:00 SUPPRESS
OVO3    mngdnodeBmydomain.com   02-06-2007 01:00 02-06-2007 03:00 LOGONLY
OVO4    mngdnodeB.mydomain.com  02-06-2007 01:00 02-06-2007 03:00 SUPPRESS
OVO5    mngdnodeC.mydomain.com  02-06-2007 00:00 09-06-2007 00:00 SUPPRESS
```

Example output of the command `ovsc.sh –outagedetail OVO1`:

```
OVO id          : OVO1
Table           : cm3r
```

```
Reference        : C14
Node             : mngdnodeA.mydomain.com
IP address       : 12.34.56.78
Down start time  : 02-06-2007 02:00
Down end time    : 03-06-2007 23:00
Message operation: SUPPRESS
Severity         :
Message group    : Database
Message type     :
Outage object    :
Status variable  :
Application      : Oracle
Service name     :
Outage text      :
```

Schedules on UNIX

This section describes how to configure the upload schedules using UNIX crontab.

The entries in a crontab file are lines, each of which contains six fields. The fields are separated by spaces or tabs. The format of each line is as follows:

```
minute  hour  monthday  month  weekday  command
```

The first five fields are integer patterns that specify when the sixth field, command, should be executed. The permitted range of values for each field is shown below.

Field	Used to represent...	Permitted Values
Minute	The minute of the hour	0-59
Hour	The hour of the day	0-23
Monthday	The day of the month	1-31
Month	The month of the year	1-12
Weekday	The day of the week	0-6 (0=Sunday)

More information about crontab can be found in the man pages.

Crontab Schedule Examples

Run the outage synchronization every hour on the hour:

```
0  *  *  *  *  /opt/OV/bin/ovsc-outage.sh
```

Run the outage synchronization every 6 hours:

```
0  0,6,12,18  *  *  *  /opt/OV/bin/ovsc-outage.sh
```

Run the outage synchronization once a day at 1 AM:

```
0  1  *  *  *  /opt/OV/bin/ovsc-outage.sh
```

The file `sc-outage.cronjob`, located in the /var/opt/OV/conf/sc directory, contains a number of examples. Comment out the schedule you want to use and then run the crontab command to activate the schedule. Do not overwrite existing schedules.

To check for existing schedules, enter `# crontab -l`. If no schedules exist then issue the following command:

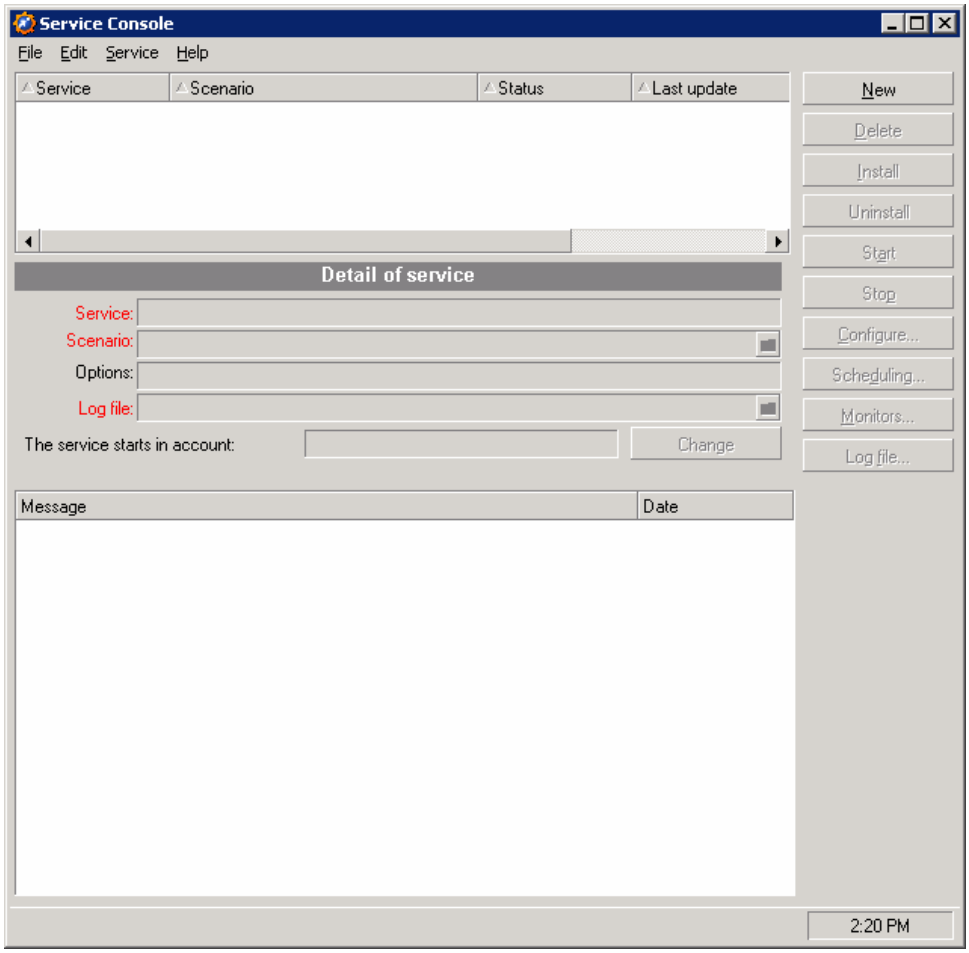
```
# crontab /var/opt/OV/conf/sc/sc-outage.cronjob
```

If schedules exist, then edit these by entering `# crontab -e`. Add the entry for the SC outage using one of the examples from the `sc-outage.cronjob` file.

Configure Connect-It for Outage Integration

The Service Console

The Service Console is a graphical tool for managing services that implement a Connect-It scenario. The console is installed as part of the Connect-It installation on a Windows machine. Using the console you can define, configure, and schedule a scenario, and then install and start a Windows service to implement the scenario. This section discusses how to use the console to configure, schedule, and run the Outage integration.

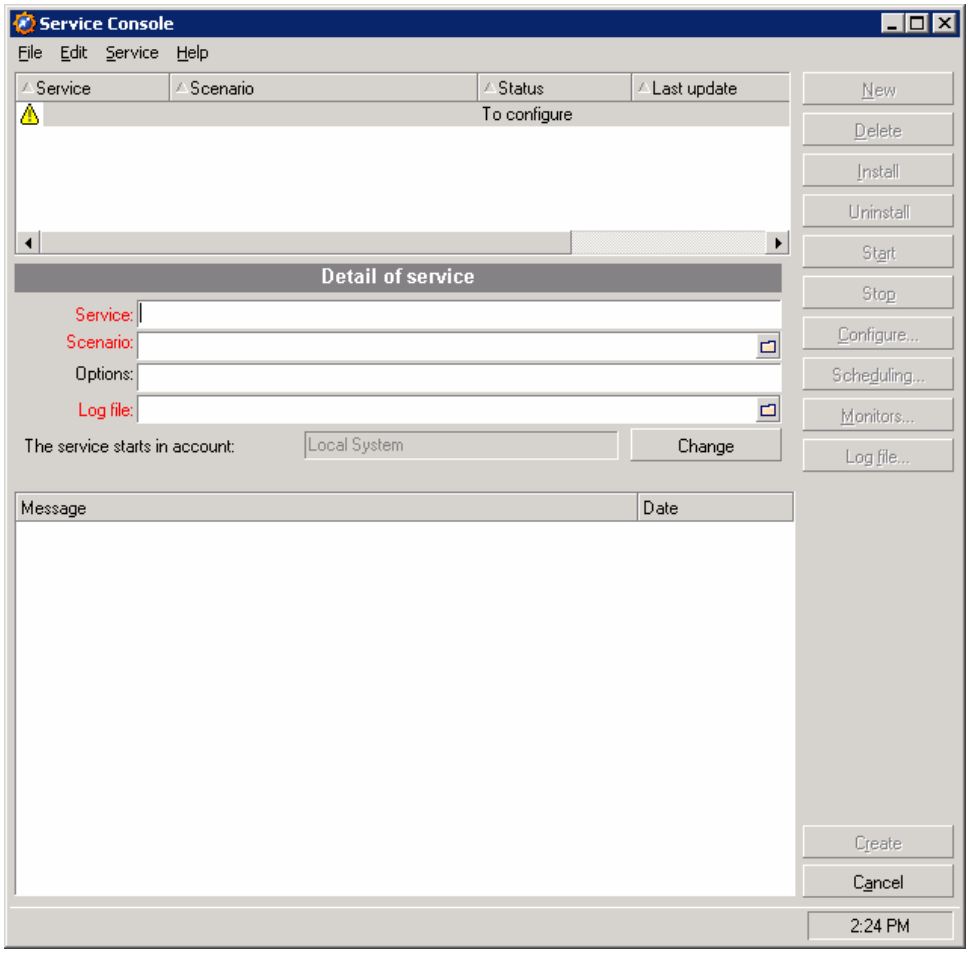


Configuring the outage scenario

This section describes how to configure, schedule and start the outage scenario.

The sc-outage scenario transfers data in the `ovooutage` table defined in ServiceCenter to an intermediate `sc_outage` table defined in the Oracle database on an OVO Management Server.

The scenario is located in the data\conf\sc directory relative to HP OpenView install directory. Normally the install directory is C:\Program Files\HP OpenView\data\conf\sc.

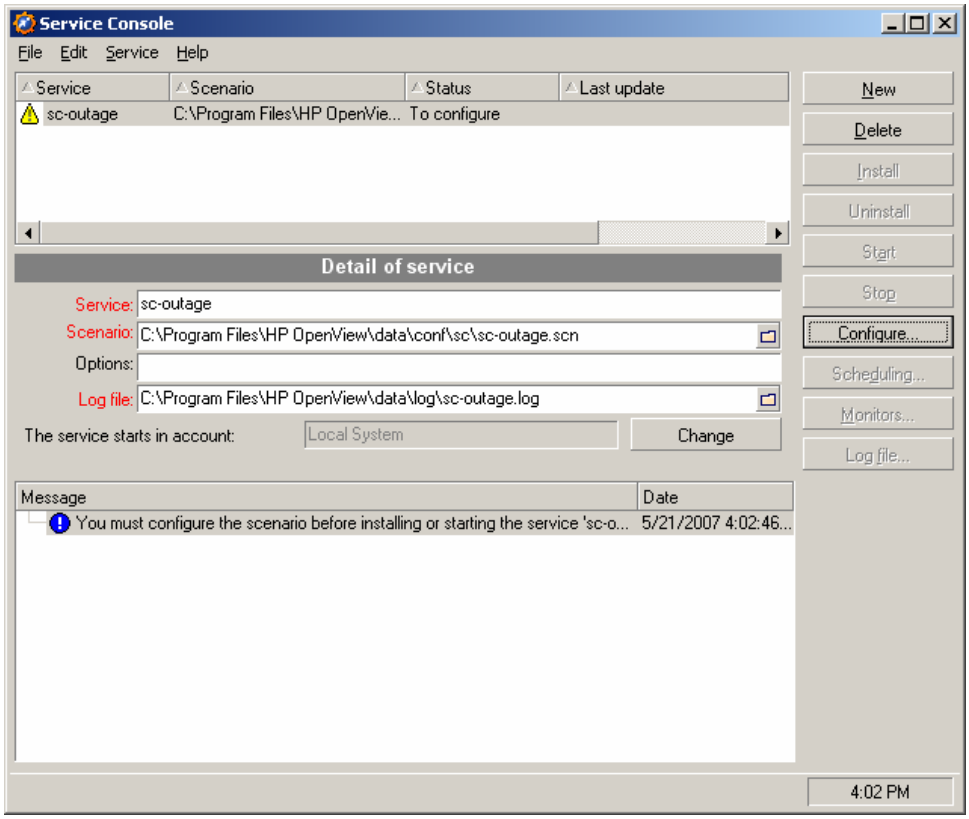


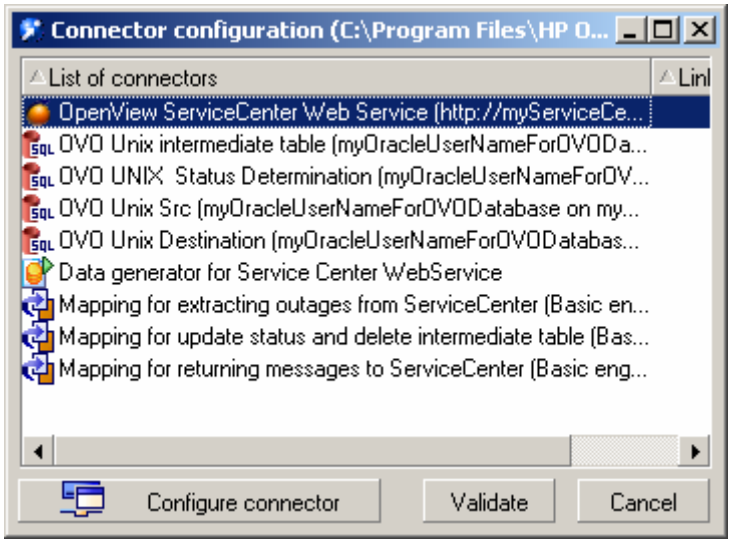
1. Create a new service for the Service Console. Open the Service Console and click the **New** button.
2. In the Service Console screen that appears, enter values for the fields shown below.

Service	Enter a name for the Windows service you want to create, for example sc-outage. This will generate a service called **Connect-IT service (sc-outage**).
Scenario	Enter the scenario file to use, normally C:\Program Files\HP OpenView\data\conf\sc\sc-outage.scn
Log file	Enter a location for the Connect-It log. The suggested location is C: \Program Files\HP OpenView\data\log\sc-outage.log. This is not the value that appears automatically when you enter a value in the Service field.

3. Click the **Create** button. You can now configure, schedule, install, and start the service, in this order.

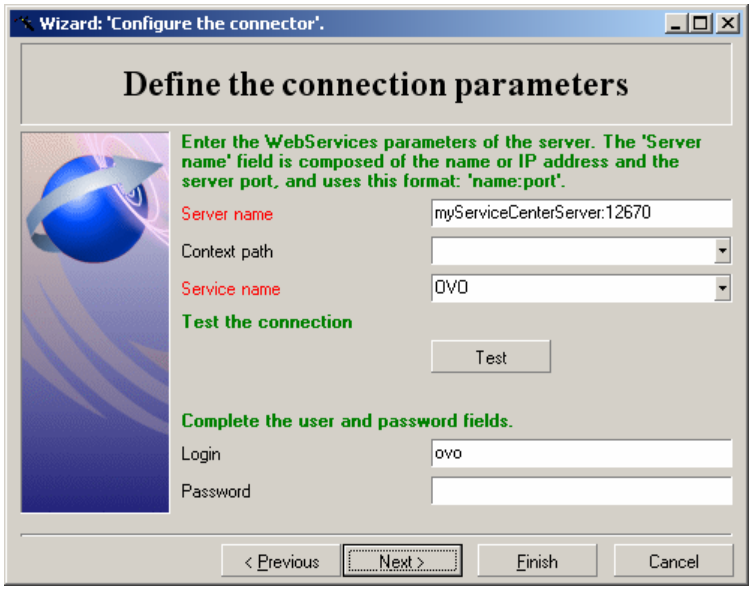
4. Click the **Configure** button to display a list of scenario connectors, as shown in the screen below. Five of these need to be configured; their configuration will be described here. In order to do the actual configuration, select the connector and click the **Configure connector** button.





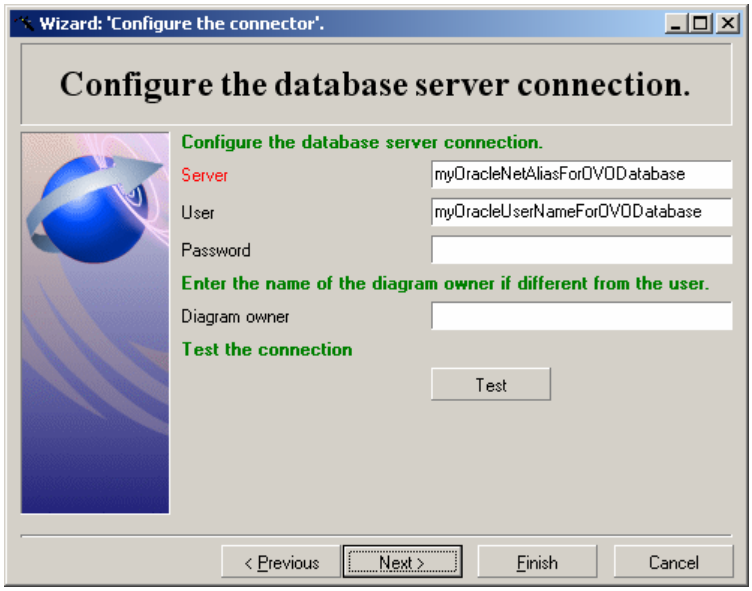
5. First configure the **OpenView ServiceCenter Web Service** connector. The values to be adjusted are on the second screen of the configuration wizard. In the second screen (**Define the connection parameters**) shown below, enter the following values:

Server name	Replace the string myServiceCenterServer with the fully qualified domain name of the ServiceCenter server. If this server does not use the default port 12670 to run the ServiceCenter Web Services, enter the appropriate number.
Service Name	'OVO'
Login	Use the default account 'ovo' or enter a valid ServiceCenter account. Ensure that you take the time zone issues associated with the ovo account into consideration.
Password	Enter the password for the ServiceCenter account (if any).



6. Click the **Test** button to test the connection. If the test is successful, click the **Finish** button to finalize it.
7. There are four connectors that are linked to the database underlying OVO Unix. These tables are named as follows:
   - **OVO Unix intermediate**
   - **OVO UNIX Status Determination**
   - **OVO Unix Src**
   - **OVO Unix Destination**.
   - The configuration for these connectors is identical, and is performed on the third screen of the wizard. Enter values according to the following table.

Server	Replace the placeholder 'myOracleNetAliasFor OVODatabase' with the Net Service Name for the OVO database server. This is (one of) the Net Service Name(s) defined in the *Net Configuration Assistant*.
User	Replace the placeholder 'myOracleUserNameFor OVODatabase' with the user name for the Oracle account holding the OVO data (generally **opc_op**)
Password	Enter the password related to the user name given above.



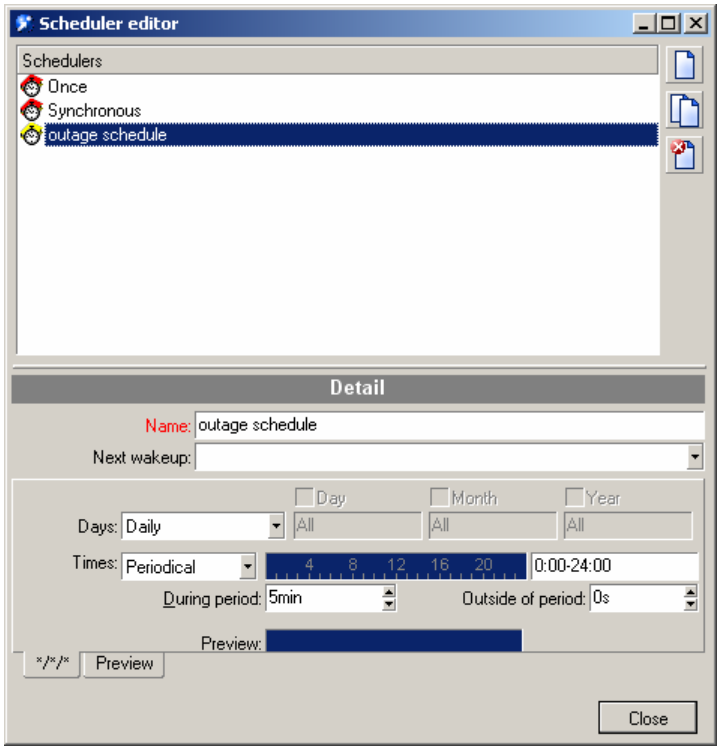
8. Click the **Test** button to test connection, and when successful, finalize it with a click on the **Finish** button.
9. In the Connector configuration window that reappears, click the **Validate** button to finish configuring the scenario.

Scheduling

10. Once you have configured the connector, the **Scheduling** button in the Service Console screen is now available. Click this button to display the *Edit Scheduling* screen. Click the **Edit Schedulers** button.
11. In the Scheduler editor screen, select the outage schedule. This is pre-set to run every five minutes, and can be adjusted. Ensure that you save any changes you make.

    The schedules for extraction on the Connect-It server and for upload processes on the OVO Management Servers must be accurately timed to ensure that they never run simultaneously. See *Scheduling the Extraction and Upload Processes*.

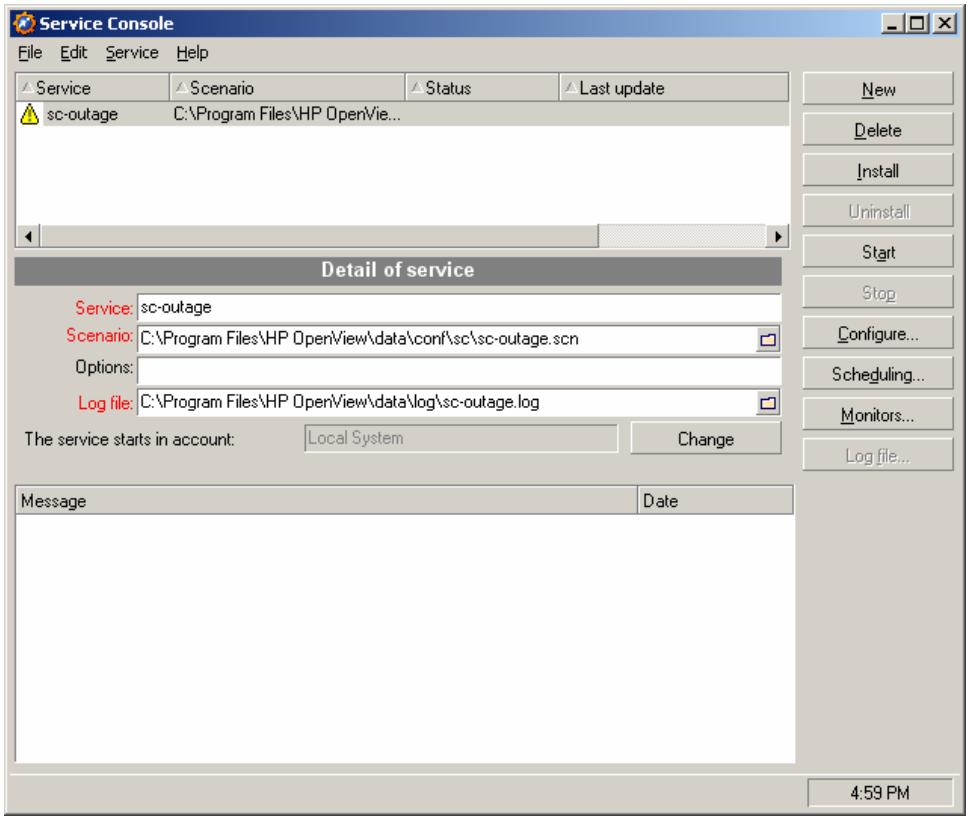
12. Click the **Close** button, and click **Validate** to check the editor changes.
13. Close the **Edit scheduling** window.



Create and start a service

The Service Console now looks similar to that shown below. A service has been defined, but no corresponding Windows service has yet been created, nor has it started executing.

To create a Windows service place your cursor on the service and click the **Install** button. Connect-It creates a service for the selected Service. It will be named Connect-It service (<service name>), so clicking the **Install** button in this example will create the service Connect-It service (sc_outage). The services you create are Windows services that can be monitored and managed through the services interface of the Windows operating system.



Once the service has been created, you can ensure that it starts automatically when the Connect-It server boots up. Do this by making it an Automatic service in the Windows services interface. Once installed, the service can also be started and stopped from the Service Console by clicking the **Start** button or **Stop** button. To remove the service, click the **Uninstall** button.

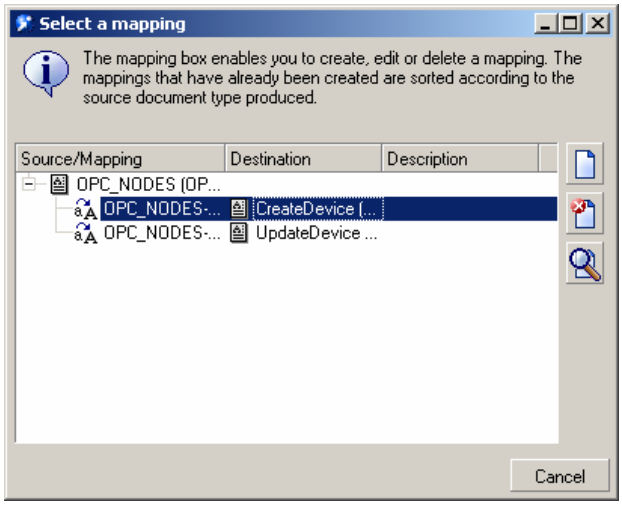
6 Customizing Node Bank Integration

The sc-nb scenario, as it is delivered, provides for a limited amount of data to be synchronized between the OVO Node Bank and the CMDB in ServiceCenter. The attributes to be synchronized are defined in two mappings, defined on the Mapping Nodes to SC connector. The mappings, especially the one for updates, use several Basic scripts to achieve functionality that transcends simple attribute-to-attribute mappings.

Document Mappings in Mapping Nodes to SC

There is a mapping for inserts (destination is CreateDevice) and a mapping for updates (destination is UpdateDevice). The mappings take the documents from the OVO Node Bank database (document type OPC_NODES) and transform them into documents that are consumed by the CreateDevice and UpdateDevice methods in the **OpenView ServiceCenter Web Service – insert or update Nodes**.

It is possible to add more attributes to these mappings, thereby extending the range of attributes that is synchronized. This is done within the Connect-It Scenario Builder. Choose **Edit a mapping** and then double-click the appropriate mapping, or click on the new icon shown in the screen below. Adjusting a simple mapping is then a matter of clicking, dragging, and dropping. Often, a Basic script has to be edited as well.

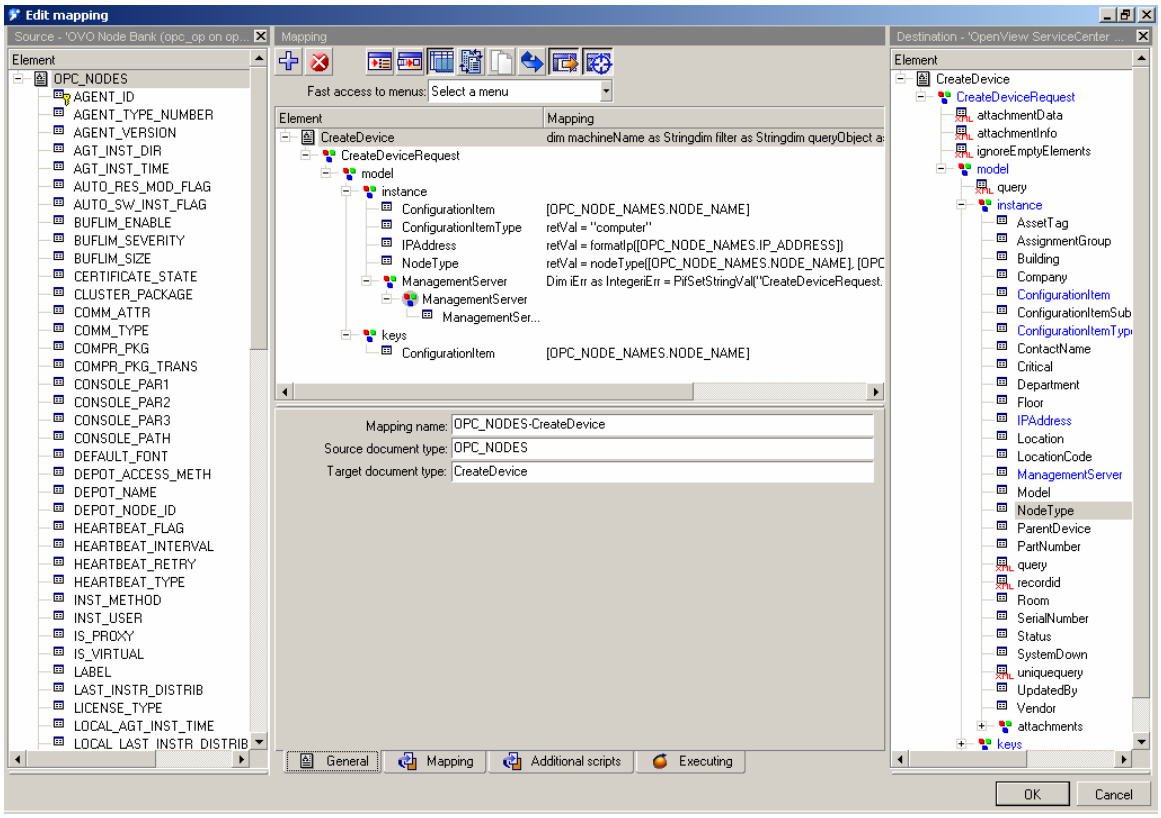


The attributes for the devices are shown in the Model→Instance section of the screen below, under “Device”. The attribute mapping on columns in the OVO Unix tables, and the Basic scripts used, are shown in the table at the end of this section. In the Basic scripts, some global functions (`formatIP()` and `nodeType()`) are called; these are defined in the `sc-nb.bas` file.

Please note that the ConfigurationItemType attribute is only set in the mapping for CreateDevice. The value is hard coded to read “computer” so it

can serve as a default value for the type, to be changed by ServiceCenter operators when appropriate.

For the mappings to function properly it is also essential to map the [OPC_NODE_NAMES.NODE_NAME] column on the 'keys.ConfigurationmItem' of the 'CreateDevice' method. This serves as the primary key.



Basic scripts used in the mappings

Apart from the Basic scripts used in the attribute mappings, two scripts have been defined at the level of the mappings as a whole, one in CreateDevice and one in UpdateDevice,. These scripts share some elements that will be described here.

Use of the device document type

The Basic scripts often use the document type **device** defined on the **Service Center existence check** connector. This is an instance of the 'old' ServiceCenter connector that is built on top of the so-called event services. This document type is called upon to check the existence of a counterpart in the ServiceCenter database for each row ('document') produced by the **OVO Node Bank** connector. This checking requires the use of a filter, which is not available in the ServiceCenter Web Services connector.

When needed (basically for every row retrieved from the OVO database), a document is opened programmatically as a `queryObject` by the

`pifNewQueryFromFmtName()` function. This uses the connector and document type names as parameters, and a filter. The `returnCode` variable then registers the result of retrieving the single row (`pifQueryNext()` function). A test on the '`returnCode`' reveals whether the document exists. This information is then used to determine how to proceed.

The exact test executed is determined by the filter. This can be a simple existence check, as used in the Create Device mapping, but it can also be a check whether the values found in the Service Center database differ from those read from the OVO database, as used in the Create Device mapping. The filter itself is a simple String value, with a syntax very similar to a SQL "where" clause.

The device document type is used for this kind of checking in two scripts:

In the **CreateDevice** mapping it just checks whether a given device exists in ServiceCenter. The filter is created by two lines of code:

```
machineName = [OPC_NODE_NAMES.NODE_NAME]
filter = "logical.name = " & chr(34) & machineName & chr(34)
```

If the device already exists, the function `PifIgnoreDocumentMapping()` is called to assure it is not created a second time in ServiceCenter.

In the **UpdateDevice** mapping a more elaborate check is performed. In this case the existence of a device with identical values on mapped attributes (when comparing the value in the OVO database with the one in ServiceCenter) is ascertained. In the default mapping this is limited to the node type and the IP address. The creation of the filter is a bit more verbose in this case:

```
MgmtServer = [OPC_MGMTSV_CONFIG.OPC_NODE_NAMES.NODE_NAME]
ovoNodeType = nodeType(machineName, MgmtServer)
rawIp = [OPC_NODE_NAMES.IP_ADDRESS]
ipAddress = formatIp(rawIp)
filter = filter & " and ovo.node.type = " & chr(34) & ovoNodeType
& chr(34)& " and network.address = " & chr(34) & ipAddress &
chr(34)
```

The decision whether an actual update must be performed, should also take into account the values for the 'Management Server' attribute. This is described in the next section.

Working with array values

The ManagementServer attribute is an Array value. This has implications for the way it must be treated in the Basic scripts.

The way an array value is exposed differs between the ServiceCenter connector and the ServiceCenter Web Services connector. In the ServiceCenter connector, and hence in the device document type' it is a 'Long text' attribute, in the ServiceCenter Web Services connector, and hence in the Create Device and Update Device document types, it is a collection. In the Basic scripts that deal with the ManagementServer attribute, it is read as a String value and written as a collection of strings.

The values in the ManagementServer array are read from the device document, in which it is the ovo.management.server attribute. This attribute, like any other that is used in the Basic scripts, must explicitly be made

available in the document. The result of the `pifQueryGetStringVal()` function is a string containing the values in the array separated by linefeed characters (ASCI character 10). The `CountValues()` function plays a useful role in determining the number of elements in the array. The `ExtractValue()`allows for an easy extraction of the individual values from this string. Once these are extracted, they are used for two purposes:

At the level of the **UpdateDevice** mapping, the Basic script checks if the extracted value is the same as was found for the interfacing Management Server. If this is the case, there is no need to add a value to the array.

The script defined for the ManagementServer attribute rebuilds the array of values from the string retrieved from the device document. For each value found, an element is added to a newly-built array, using the `PifSetStringVal()` function with a counter to indicate the element involved. This function is also used to add the interfacing Management Server as the final array element.

Adding attribute mappings

One can add a mapping by simply linking a column in the OVO database to an attribute in the Web Services definition. This is done in the central panel of the screen shown above. In some cases the two values can easily be mapped, in other cases (depending on the functional requirements) some Basic programming may be required.

After this linking operation, a newly mapped attribute should be added to the filter defined in the Basic script at the UpdateDevice level. Normally, only the definition of the filter must be changed. When more complicated data types are being used, additional programming may be required, comparable to what was described in relation to the array values.

Service Center Attribute	OVO-U column name	Basic script
ConfigurationItem	[OPC_NODE_NAMES.NODE_NAME]	
ConfigurationItemType		retVal = “computer”
IPAddress		retVal = formatIp([OPC_NODE_NAMES.IP_ADDRESS])
NodeType (insert)		retVal = nodeType([OPC_NODE_NAMES.NODE_NAME], [OPC_MGMTSV_CONFIG.OPC_NODE_NAMES.NODE_NAME])

NodeType (update)		Dim machineName as String
		Dim iErr as Integer
		Dim filter as String
		Dim queryObject as long
		Dim counter as Integer
		Dim tableValue as String
		
		machineName = [OPC_MGMTSV_CONFIG.OPC_NODE_NAMES.NODE_NAME]
		filter = "logical.name = " & chr(34) & machineName & chr(34)
		queryObject = pifNewQueryFromFmtName("ServiceCenter existence check", "device" , filter)
		iErr = pifQueryNext(queryObject)
		tableValue = trim(pifQueryGetStringVal(queryObject, "'ovo.node.type'")) & chr(10)
		iErr = pifQueryClose(queryObject)
		
		If (tableValue = "OVO Management Server") Then
		  retVal = tableValue
		Else
		  retVal = nodeType([OPC_NODE_NAMES.NODE_NAME], [OPC_MGMTSV_CONFIG.OPC_NODE_NAMES.NODE_NAME])
		End If

ManagementServer (insert)		Dim iErr as Integer iErr = PifSetStringVal("CreateDeviceRequest.model.instance.ManagementServer.ManagementServer(0)", [OPC_MGMTSV_CONFIG.OPC_NODE_NAMES.NODE_NAME])

ManagementServer (update)		Dim nodeName as String
		Dim mgmtServer as String
		Dim iErr as Integer
		Dim filter as String
		Dim queryObject as long
		Dim counter as Integer
		Dim tableValue as String
		Dim arraySize as Integer
		Dim arrayValue as String
		Dim addNode as Integer
		
		counter = 0
		addNode = 1
		nodeName = [OPC_NODE_NAMES.NODE_NAME]
		mgmtServer = [OPC_MGMTSV_CONFIG.OPC_NODE_NAMES.NODE_NAME]
		filter = "logical.name = " & chr(34) & nodeName & chr(34)
		queryObject = pifNewQueryFromFmtName("ServiceCenter existence check", "device" , filter)
		iErr = pifQueryNext(queryObject)
		tableValue = trim(pifQueryGetStringVal(queryObject, "'ovo.management.server'")) & chr(10)
		iErr = pifQueryClose(queryObject)
		
		arraySize = CountValues(tableValue, chr(10))
		Do While (counter < arraySize)
		  arrayValue = ExtractValue(tableValue, chr(10))
		  If (trim(arrayValue) <> "") Then
		  If (trim(arrayValue) = trim(mgmtServer)) Then
		  addNode = 0
		  End If
		  iErr = PifSetStringVal("UpdateDeviceRequest.model.instance.ManagementServer.ManagementServer(" + Str(counter) + ")", trim(arrayValue))
		  counter = counter + 1
		  End If
		Loop
		
		If (addNode = 1) Then

		iErr = PifSetStringVal("UpdateDeviceRequest.model.instance.ManagementServer.ManagementServer(" + Str(counter) + ")", mgmtServer) End If retVal = iErr

Mapping name	Mapping level basic script

Create Device	dim machineName as String
	dim filter as String
	dim queryObject as long
	dim returnCode as long
	dim sink as long
	
	machineName = [OPC_NODE_NAMES.NODE_NAME]
	filter = "logical.name = " & chr(34) & machineName & chr(34)
	queryObject = pifNewQueryFromFmtName("ServiceCenter existence check", "device" , filter)
	returnCode = pifQueryNext(queryObject)
	sink = pifQueryClose(queryObject)
	
	If returnCode = 0 Then
	PifIgnoreDocumentMapping("")
	End If

Update Device	dim machineName as String dim filter as String dim queryObject as long dim returnCode as long dim sink as long dim mgmtServer as String dim ovoNodeType as String dim rawIp as long dim ipAddress as String dim tableValue as String dim arrayValue as String dim arraySize as Integer

```
dim counter as Integer

machineName = [OPC_NODE_NAMES.NODE_NAME]
filter = "logical.name = " & chr(34) & machineName & chr(34)
queryObject = pifNewQueryFromFmtName("ServiceCenter existence check", "device" , filter)
returnCode = pifQueryNext(queryObject)
sink = pifQueryClose(queryObject)

If returnCode <> 0 Then
  PifIgnoreDocumentMapping("")
Else
  MgmtServer = [OPC_MGMTSV_CONFIG.OPC_NODE_NAMES.NODE_NAME]
  ovoNodeType = nodeType(machineName, MgmtServer)
  rawIp = [OPC_NODE_NAMES.IP_ADDRESS]
  ipAddress = formatIp(rawIp)
  filter = filter & " and ovo.node.type = " & chr(34) & ovoNodeType & chr(34)& " and
network.address = " & chr(34) & ipAddress & chr(34)
  queryObject = pifNewQueryFromFmtName("ServiceCenter existence check", "device" , filter)
  returnCode = pifQueryNext(queryObject)
  If returnCode = 0 Then
    counter = 0
    returnCode = 1
    tableValue = trim(pifQueryGetStringVal(queryObject, "'ovo.management.server'")) & chr(10)
    arraySize = CountValues(tableValue, chr(10))
    Do While (counter < arraySize)
      arrayValue = ExtractValue(tableValue, chr(10))
      If (trim(arrayValue) = trim(mgmtServer)) Then
        returnCode = 0
      End If
      counter = counter + 1
    Loop
  End if
  sink = pifQueryClose(queryObject)
End If

if returnCode = 0 Then
  PifIgnoreDocumentMapping("")
End If
```

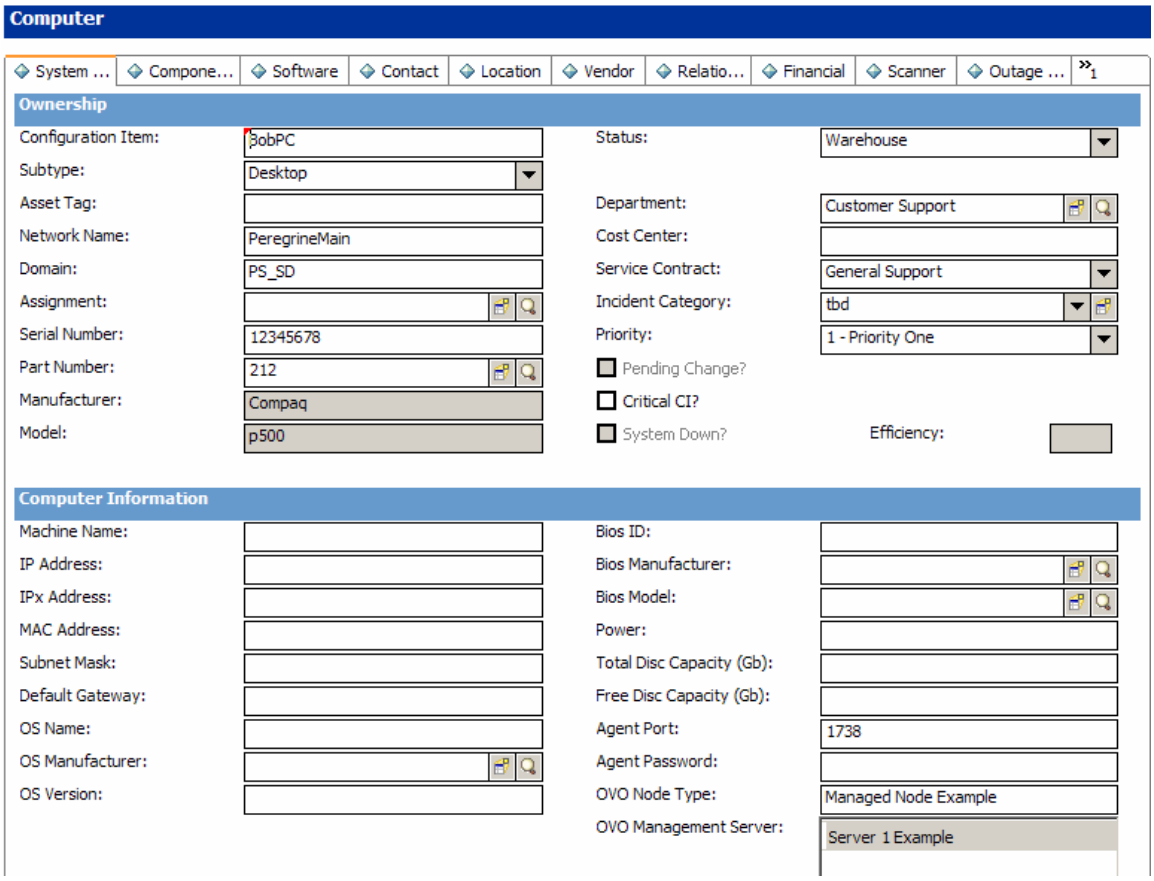
7 OVO Outage Guide for ServiceCenter Operators

This section explains how to create new OVO Outages from Change Management for CIs that are Managed nodes. It is written primarily for ServiceCenter operators.

All operators who wish to add, update, or create OVO Outage records must have the OVOAdmin capability word added to their operator record.

Managed Nodes

The device table contains information which indicates whether a CI is a Managed Node. The screen below shows data in Managed Node fields, which means that this CI is valid for OVO Outages.

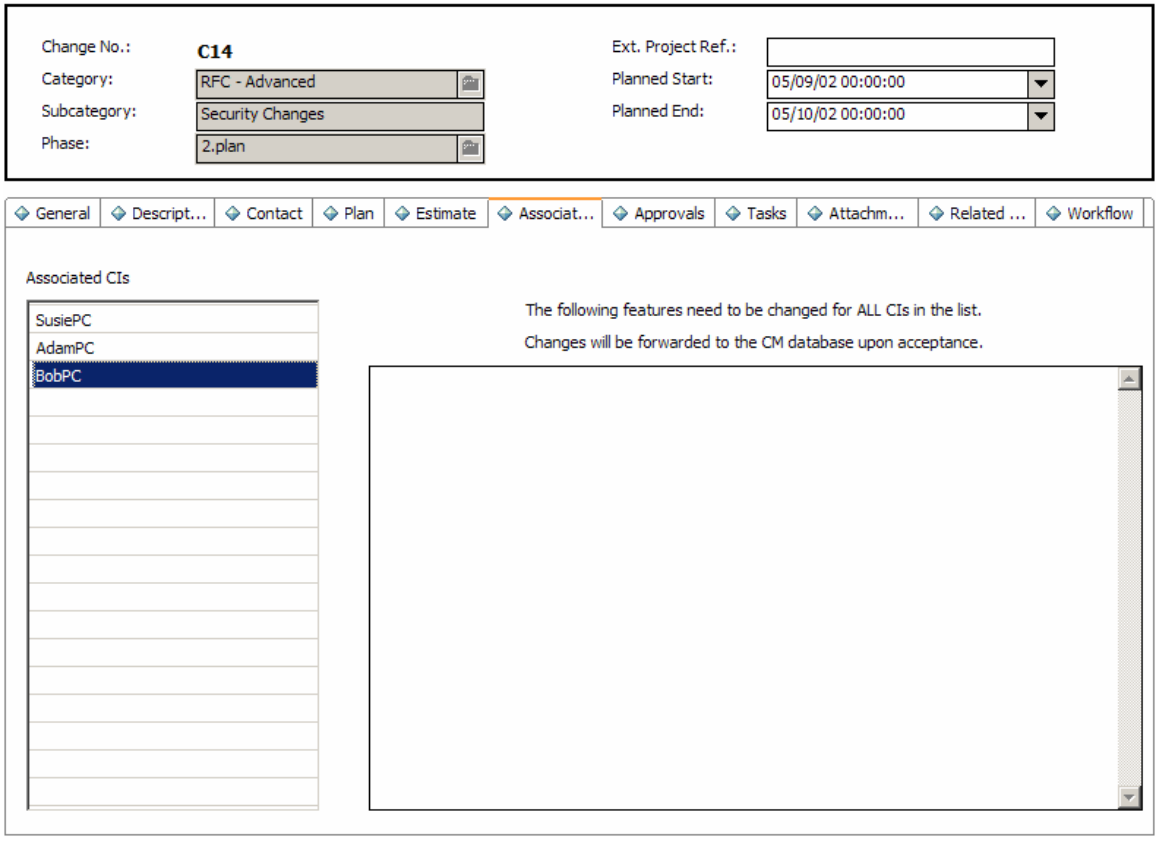


OVO Outages in Change Management

Adding and viewing Configuration Items

You can add new OVO Outages from a Change record for one or more valid CIs in the Change. CIs may exist for both Managed Nodes and Non-Managed Nodes. These are shown in the Associated CIs tab. You can also add new CIs to the list if required for the OVO Outage.

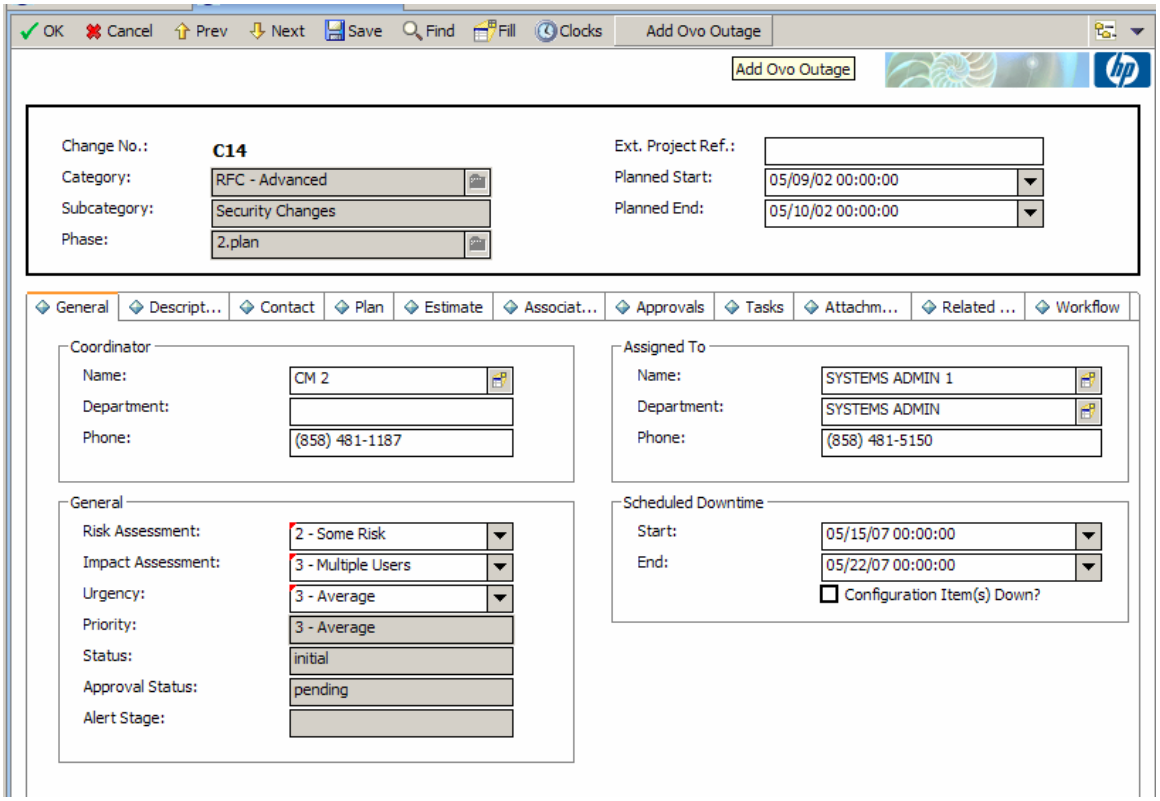
The example below shows the Change record with three associated CIs with two Managed Nodes. Therefore in the example below only OVO Outages for AdamPC and BobPC can be created (Managed Nodes in the device table).



Entering Outage Dates for Configuration Items

If the Outages for this change have the same Start and End Date for all CIs, then the Operator first enters these dates in the **General** tab.

To add a new Outage, click the **Add Ovo Outage** button on the top menu.

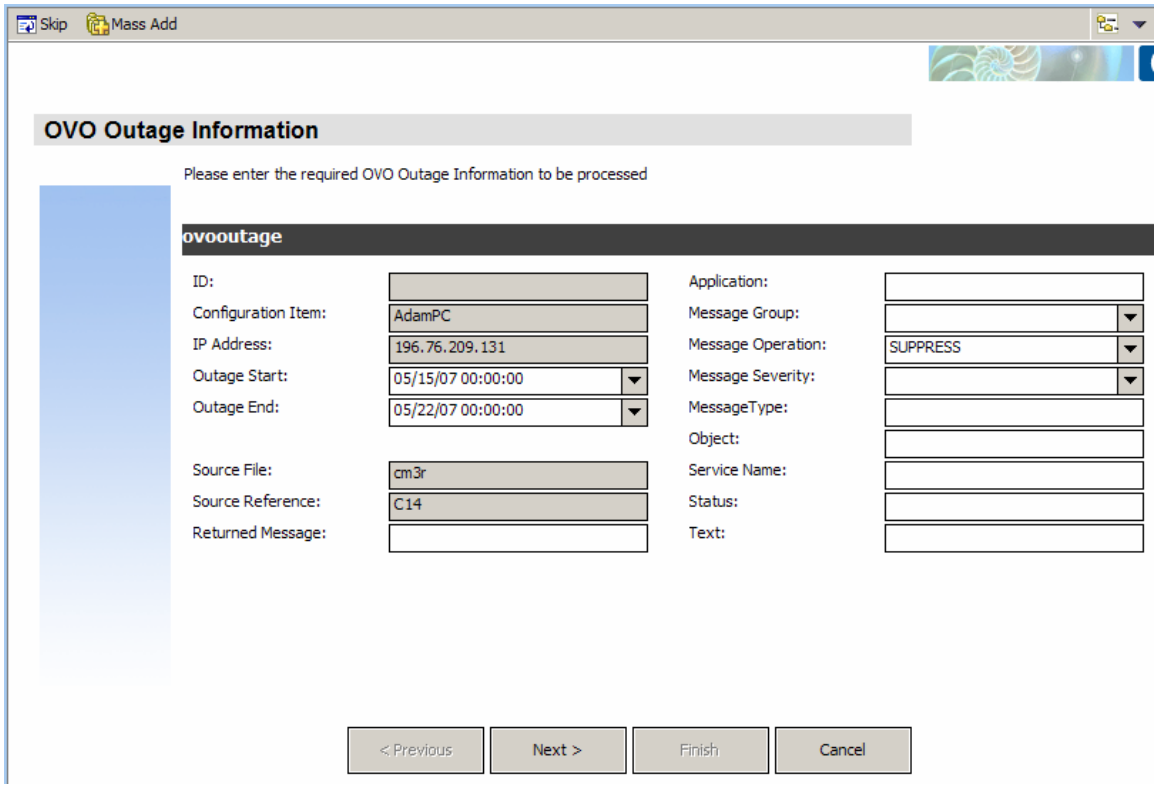


Adding OVO Outage

When you click the **Add Ovo Outage** button, the OVO Outage Information wizard is launc hed, based on the following conditions:

- Outage records for the CI(s) for this Change record have not been created
- The Scheduled Downtime Start and End dates entered are valid
- The associated CIs are Managed Nodes

A screen similar to the one shown below appears. It contains information copied from the Change record and the device table.



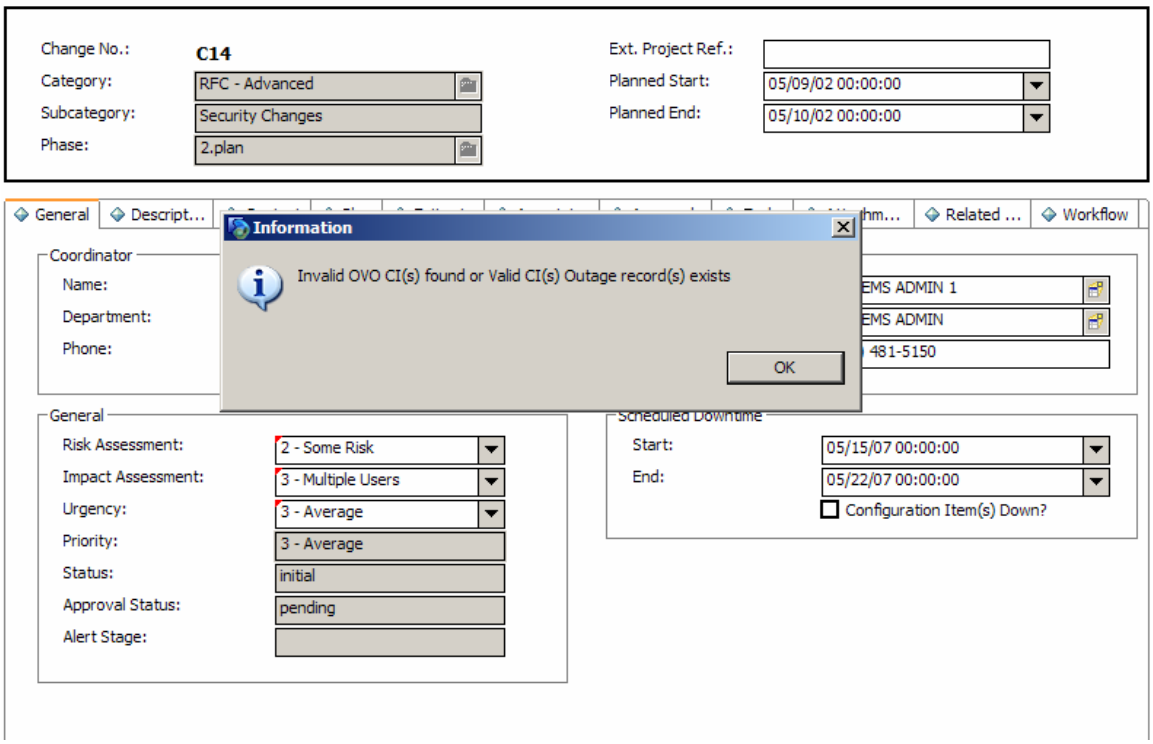
To skip a record, click the **Skip** button.

14. Enter additional information on the CI currently displayed. Click the **Next** button to save the Outage and display the next CI record.
15. Click the **Mass Add** button if all the optional information applies to all CI Outages for the Change record.

    If dates are not entered in the Change record before the wizard is run, data must be entered for these fields for each CI.

If OVO Outages for CIs Already Exist

If you try to add existing OVO Outages from the Change for CIs, the following message appears:



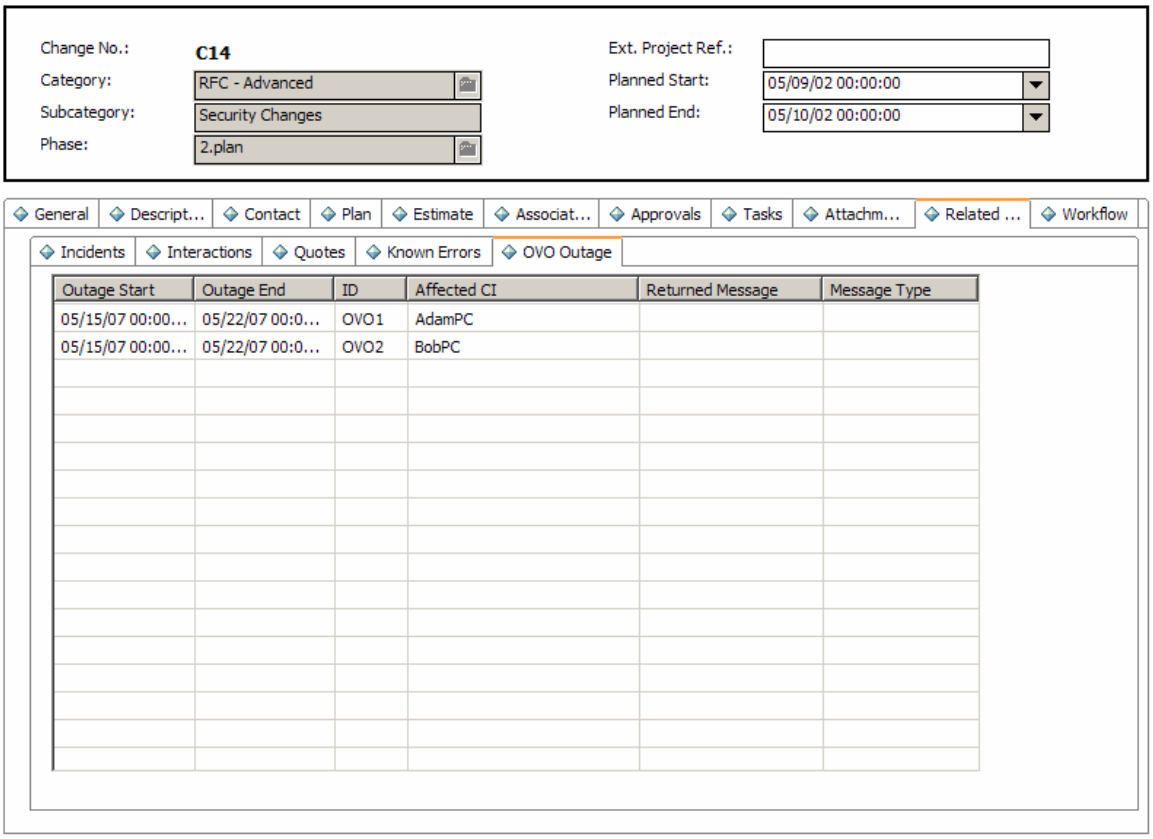
Viewing OVO Outages created for CIs in Change Record for Add/Update/Delete

Use the OVO Outage wizard to create first time Outage records for CIs.

To create new Outages, open the existing OVO Outage records for the CI, enter the new information add the new record.

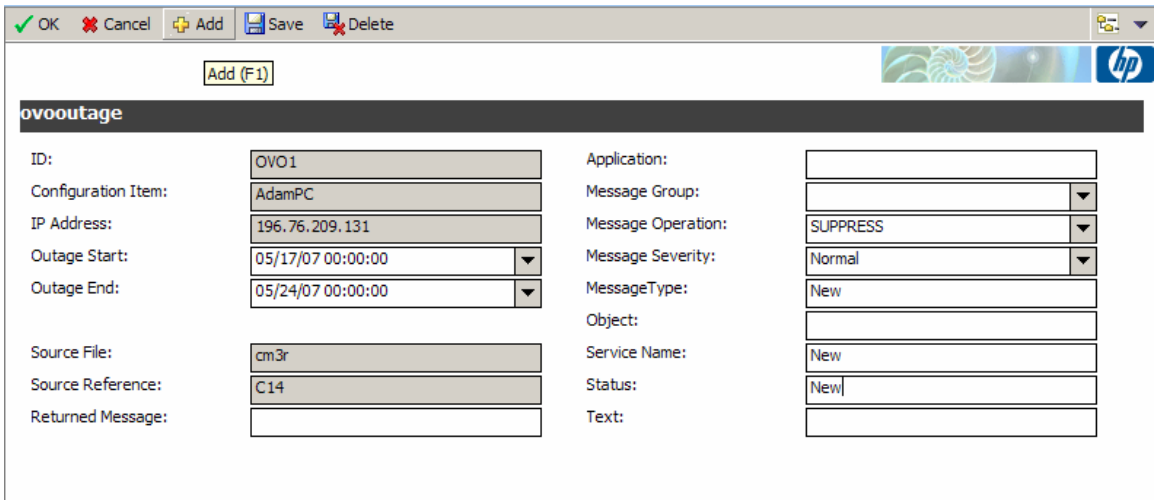
To update OVO Outage records, access the records in the same way, make the amendments and save the record.

To access the OVO Outage records for adding, updating or deleting, use the **Related Records** tab on the Change Record. A new **OVO Outage** tab should be added to the Change Forms that are used on the client's system.

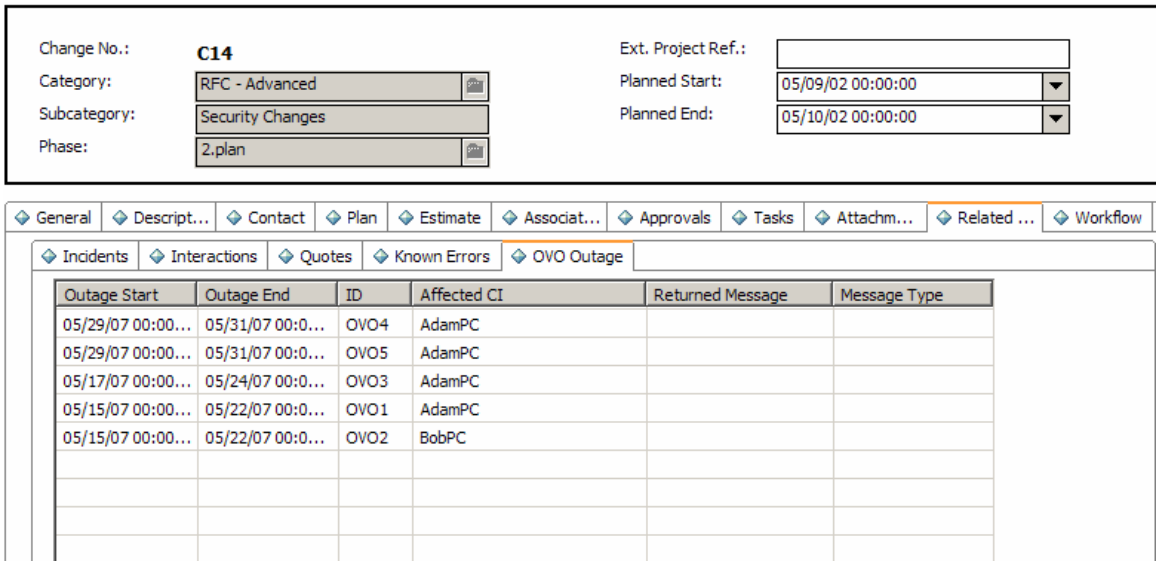


Creating further OVO Outages for CIs and Updating existing OVO Outages

To add new Outages for a CI, view the list and double click to open the OVO Outage record. On the form that displays the record, similar to the screen below, edit the record and click **Save** to update, or enter new information for that CI and click **Add** to create a New OVO Outage record.



The Updates can be seen in the list and New OVO Outages are added to the list in the OVO Outage Tab on the Change record.



OVO Outages in Change Management for Change Tasks

OVO Outages created in Change Task follow the same process as Change Task and use the same wizard.

In ServiceCenter Out of the Box, the CIs from the Change record are all passed into the Change Task. This means that all the Managed Nodes that are in the Change record are passed to the Change Task (these may not be displayed in the Task form).

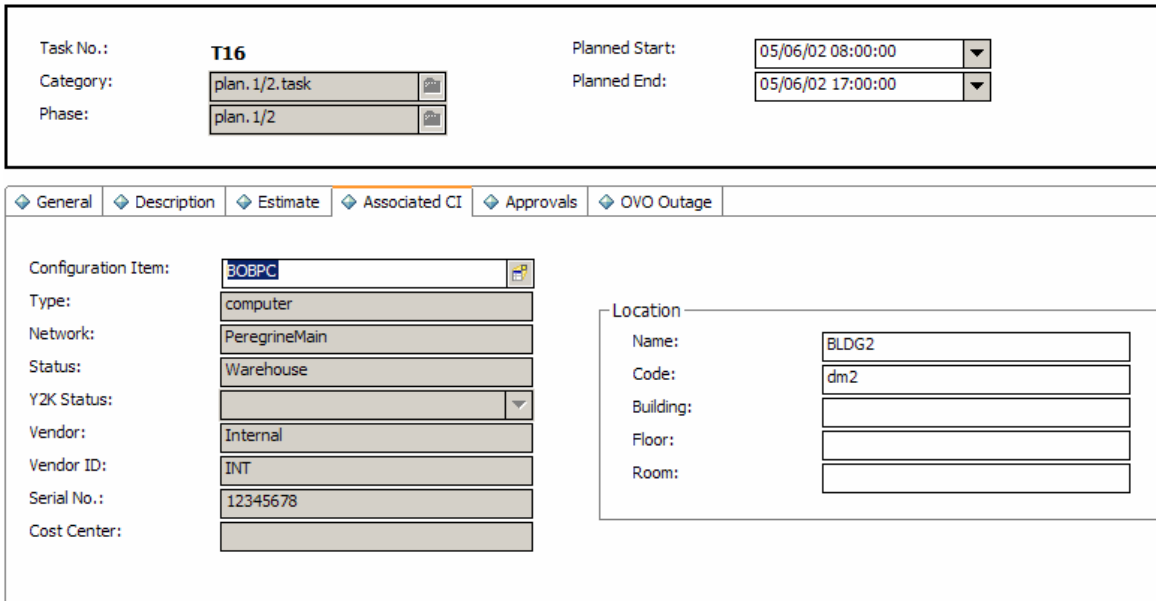
When a CI is associated with a Task and the Operator creates OVO Outages from the wizard, the CIs in the Related Change record are also presented if any valid CIs exist.

This can be controlled in the `slamodule` table, depending on the Client process for managing CI information between Change and Change Tasks. The CI Fields in the Change Task `cm3t` table can be edited to determine where CI information is stored.

Adding OVO Outage

The **Add OvoOutage** button is available when the Change Task record has been saved.

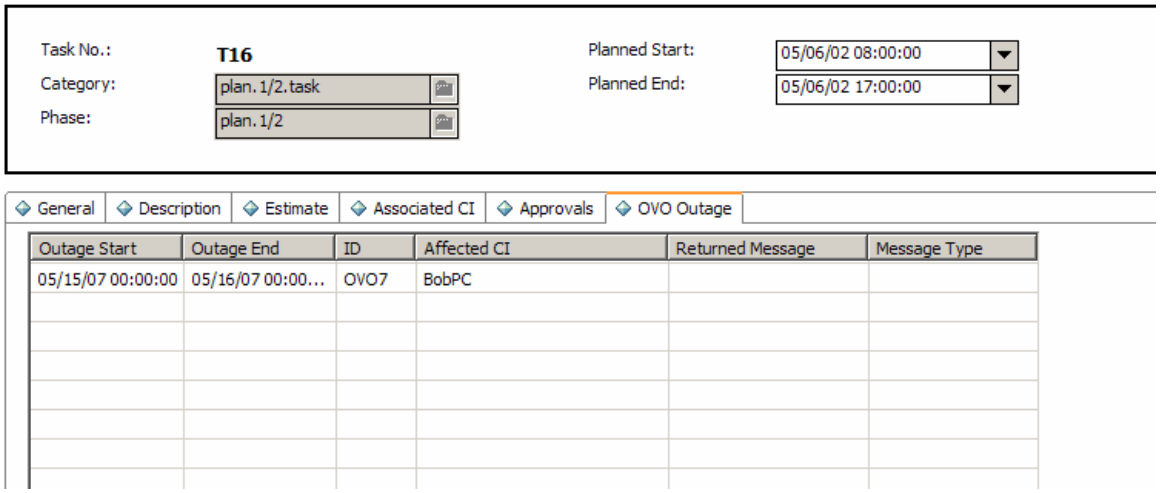
The Operator adds the CI associated with the Task and clicks **Add Ovo Outage**, and the wizard runs as in Change.



Adding, Updating, or Deleting OVO Outage records related to Change Tasks

OVO Outages in Change Tasks can be reviewed in the OVO Outage Tab.

You access the Outage records to Add new Outages for CIs added, or to update existing records in the same way as for Change Management Changes.



OVO Outage Administration

Operator Records

All operators who wish to create OVO Outage records must have the OVOAdmin capability word added to their operator record.

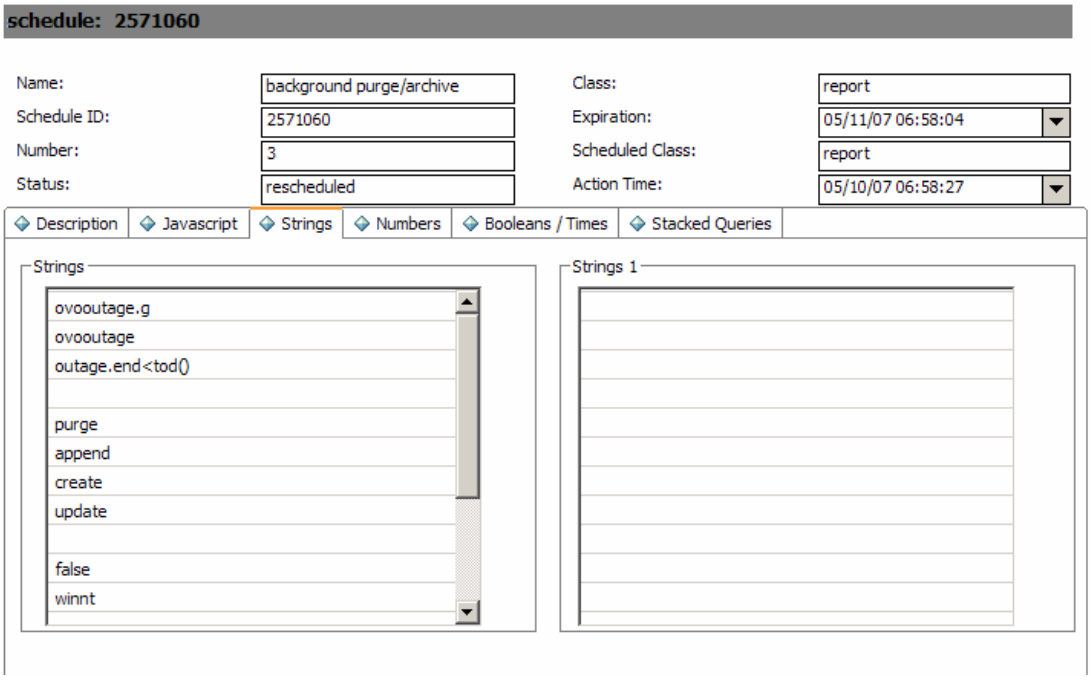
Scheduled OVO Outage record clean up

Purging or archiving Schedule Record

A schedule record for the OVO Outage records can be found which is set to Clean up OVO Outage records:

Name	background purge/archive
Class	report

There will be more than one schedule record for the above. To identify the record for OVO Outage, click the **Strings** tab, as shown in the screen below.



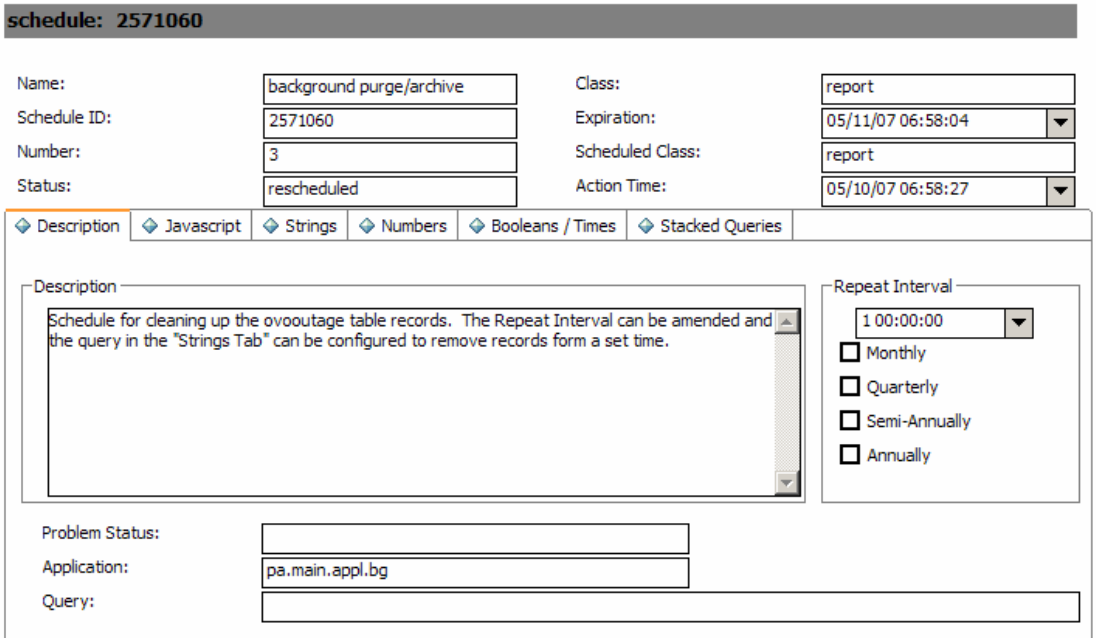
Configuring the schedule to clean up OVO Outage records

The default OVO Outage schedule is currently set to delete all OVO Outage records with an Outage End date that has passed the current date/time in the system.

The line query shown above, `outage.end<tod()`, can be amended if the client only wishes to remove OVO Outage records where the Outage end date has passed 10 days ago. The query would then be set to `outage.end<tod()-'10 00:00:00'`.

Configuring the Repeat Interval of the Schedule

The client can also set the frequency of execution for this schedule record. The Repeat Interval is defaulted to run the schedule daily for OVO Outages.



Archiving OVO Data using Page table

All history of OVO Outages for Add and Updates are added to an OVO Outage Page table (`ovooutagepage`). This copies all new OVO Outages and all updates made to these outages into a separate table that can be used for reporting. The client should decide how long this table can exist before it is deleted, as it can grow to a large size.

OvoId	SourceFile	SourceNumber	OutageStart	OutageEnd	LogicalName	Page	PagelistFormat
OVO1	cm3r	C14	05/15/07 00:00:00	05/22/07 00:00:00	AdamPC	0	wizard-ovooutage.page
OVO1	cm3r	C14	05/15/07 00:00:00	05/22/07 00:00:00	AdamPC	1	ovooutage
OVO1	cm3r	C14	05/15/07 00:00:00	05/22/07 00:00:00	AdamPC	2	ovooutage
OVO1	cm3r	C14	05/15/07 00:00:00	05/22/07 00:00:00	AdamPC	3	ovooutage
OVO2	cm3r	C14	05/15/07 00:00:00	05/22/07 00:00:00	BobPC	0	wizard-ovooutage.page
OVO3	cm3r	C14	05/17/07 00:00:00	05/24/07 00:00:00	AdamPC	0	ovooutage
OVO4	cm3r	C14	05/29/07 00:00:00	05/31/07 00:00:00	AdamPC	0	ovooutage
OVO5	cm3r	C14	05/29/07 00:00:00	05/31/07 00:00:00	AdamPC	0	ovooutage
OVO6	cm3t	T16	05/15/07 00:00:00	05/16/07 00:00:00	AdamPC	0	wizard-ovooutage.page
OVO7	cm3t	T16	05/15/07 00:00:00	05/16/07 00:00:00	BobPC	0	wizard-ovooutage.page

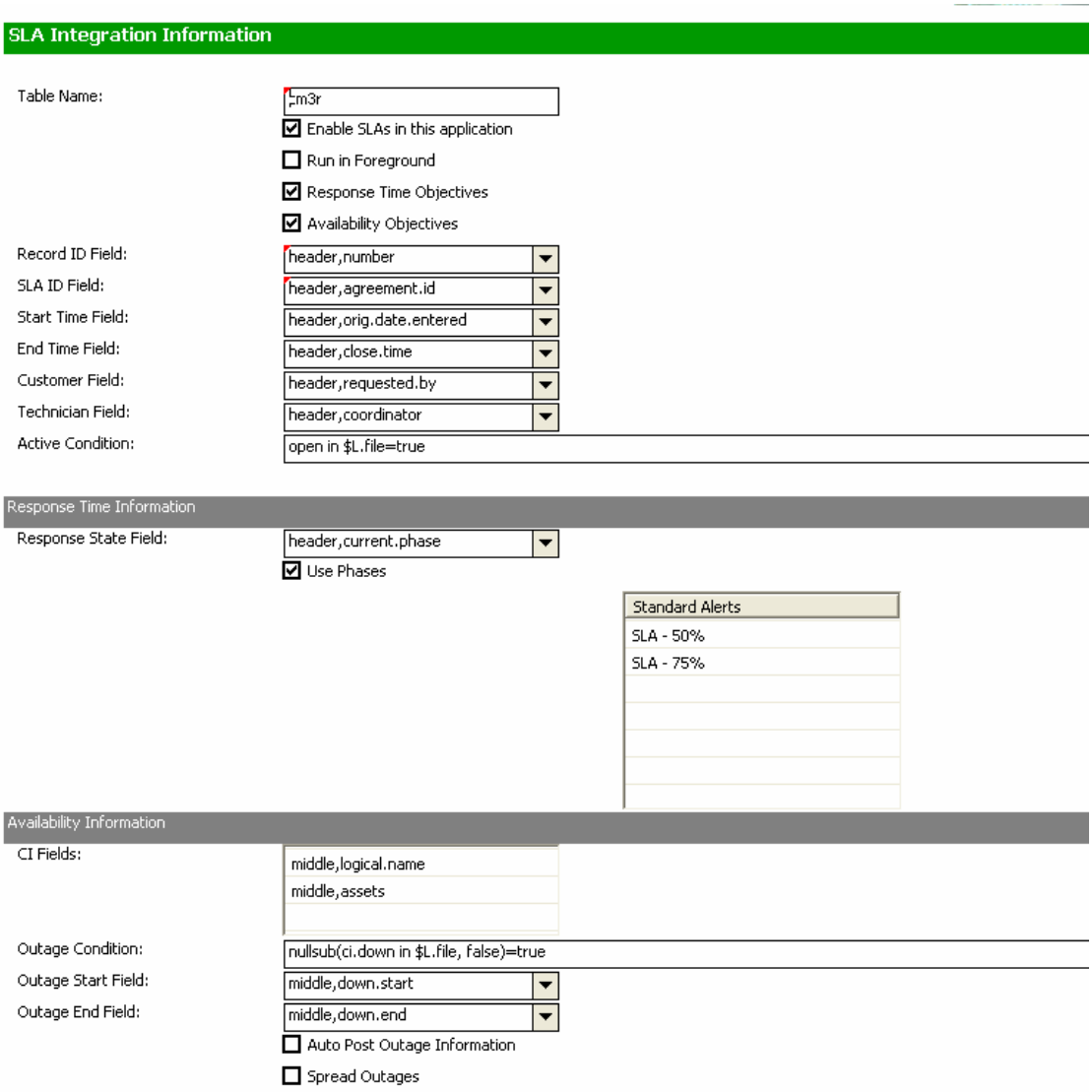
The slamodulecontrol table

The OVO Outage wizard Utility uses the `slamodulecontrol` table to determine the fields used to store CI information, and Outage Start and End dates in the Change (`cm3r`) table and Change Task (`cm3t`) table.

To keep the wizard generic between modules, the field names stored in the `slamodulecontrol` table allow the customer to configure the fields in the file where they wish to store specific information.

In the Availability Information Section of the slamodulecontrol record, the client can define the fields to use in file for the following:

CI Fields	The wizard identifies the fields in the file that the client chooses to store CI information in the record.
Outage Start Field	The wizard identifies the fields in the file that the client chooses to store the Outage Start Date field.
Outage End Field	The wizard identifies the fields in the file that the client chooses to store the Outage End Date field.



In the Out of the Box system the fields used are defined in the `slamodulecontrol` table (the above example shows the `cm3r` table).

Change Management

Table Name	cm3r
CI Fields	middle,logical.name middle,assets
Outage Start Field	middle,down.start
Outage End Field	middle,down.end

Change Task

Table Name	cm3t
CI Fields	middle,logical.name middle,asset
Outage Start Field	middle,down.start
Outage End Field	middle,down.end

Configuration Example

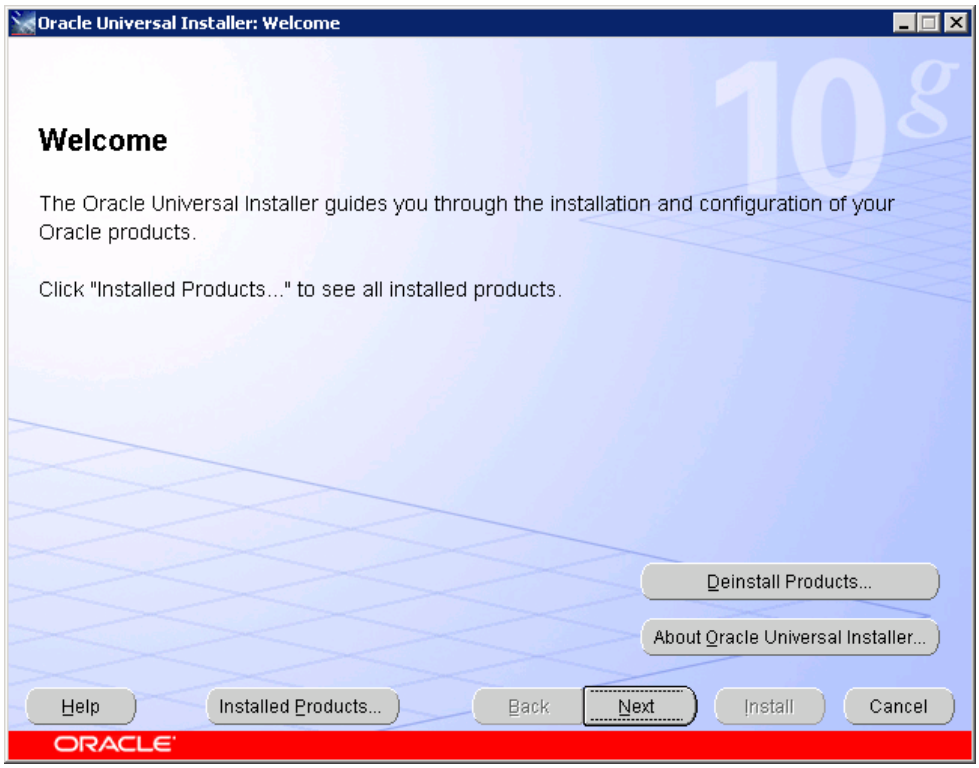
In the Out of the Box system, when a Change Task is created, the asset array field receives the values from the assets array field from the Change record the Task was created from. This means that when you “Add” an OVO Outage record for the CI in the Task record from the wizard, it also presents the CIs (if any) in the related Change record. If a client only wishes to have OVO Outages created in Change Task for the CI associated in the Change Task and not include the CIs from the Change, the slamodulecontrol record for `cm3t` table can be configured to exclude the middle,asset field by removing it from the CI Fields array list. Similarly, new fields can be added if the client requires them.

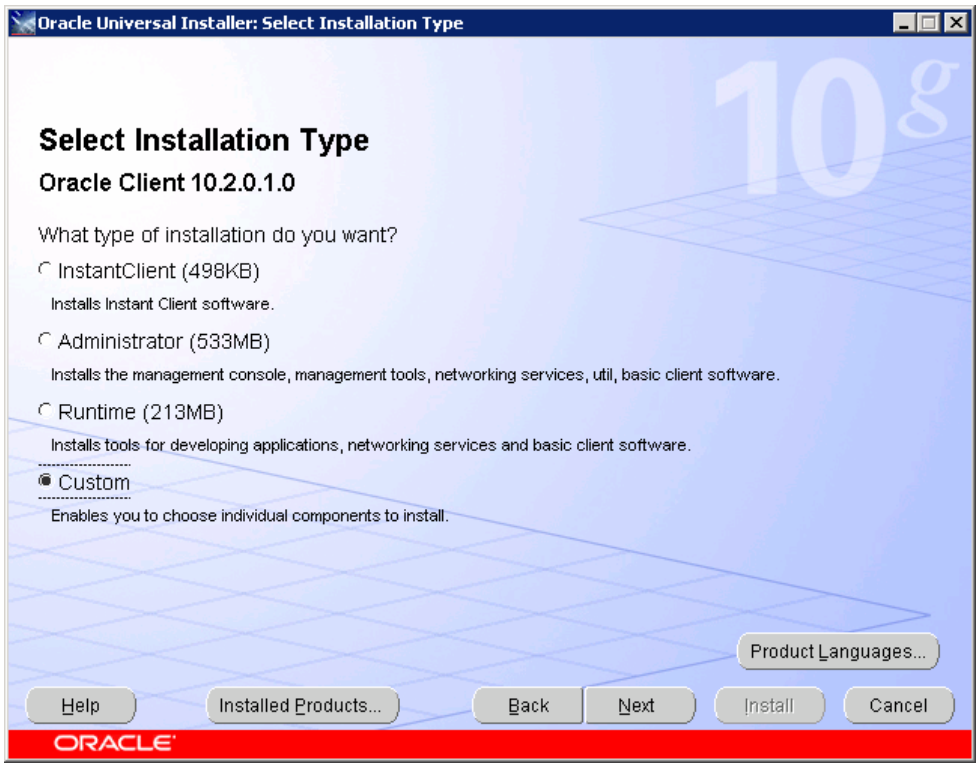
Appendix A – Installing an Oracle Client

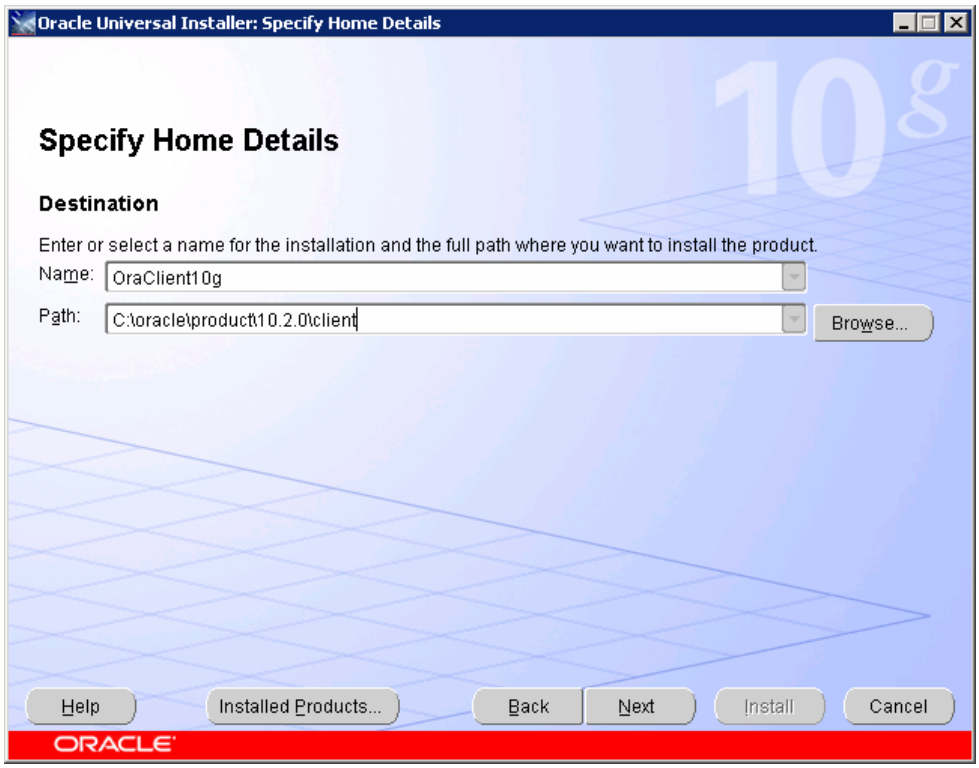
Overview

This section describes how to install the Oracle client software required for use with Connect-It. A minimal installation is described.

1. Download the software from the *Oracle Site* (you may need to create an account to do this). A long list of operating systems appears on Oracle's download page. Choose the one that applies, and you will be presented with a list of download options. Choose the "download client software" option.
2. After downloading and unpacking the software, a `Tsetup.exe` file (Windows) or a `runInstaller` file (UNIX) will start the Oracle Universal Installer. Follow the screen prints below as closely as possible. The example below uses the 10.2.0.1.0 version of the software.





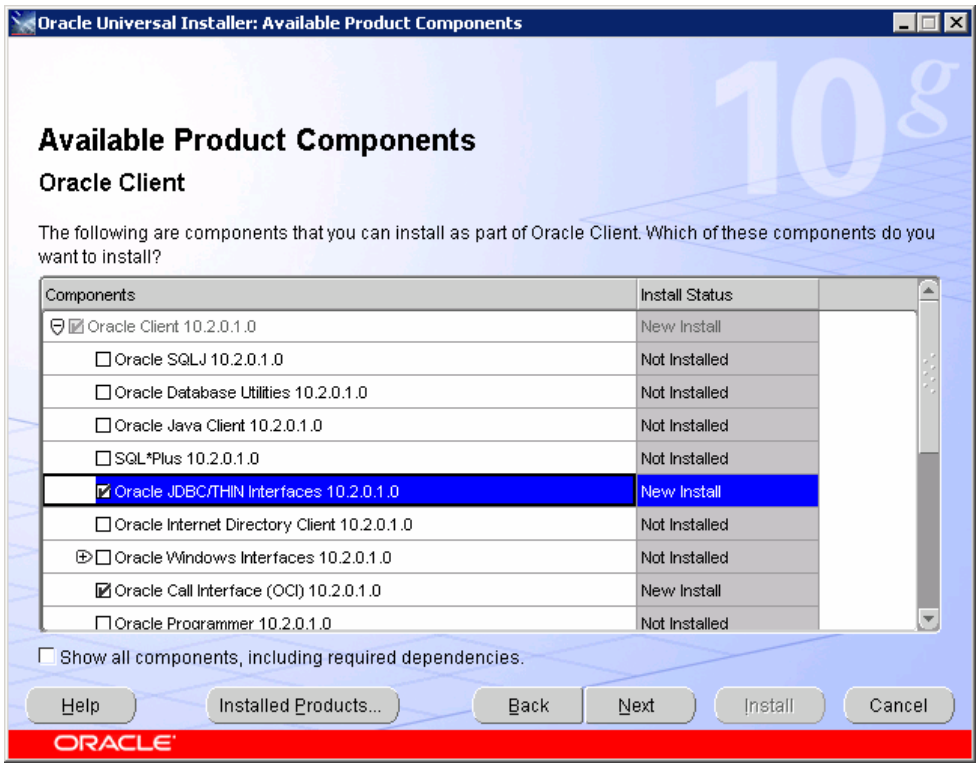


3. Feel free to select any value on this screen. As a minimum select each of the following:

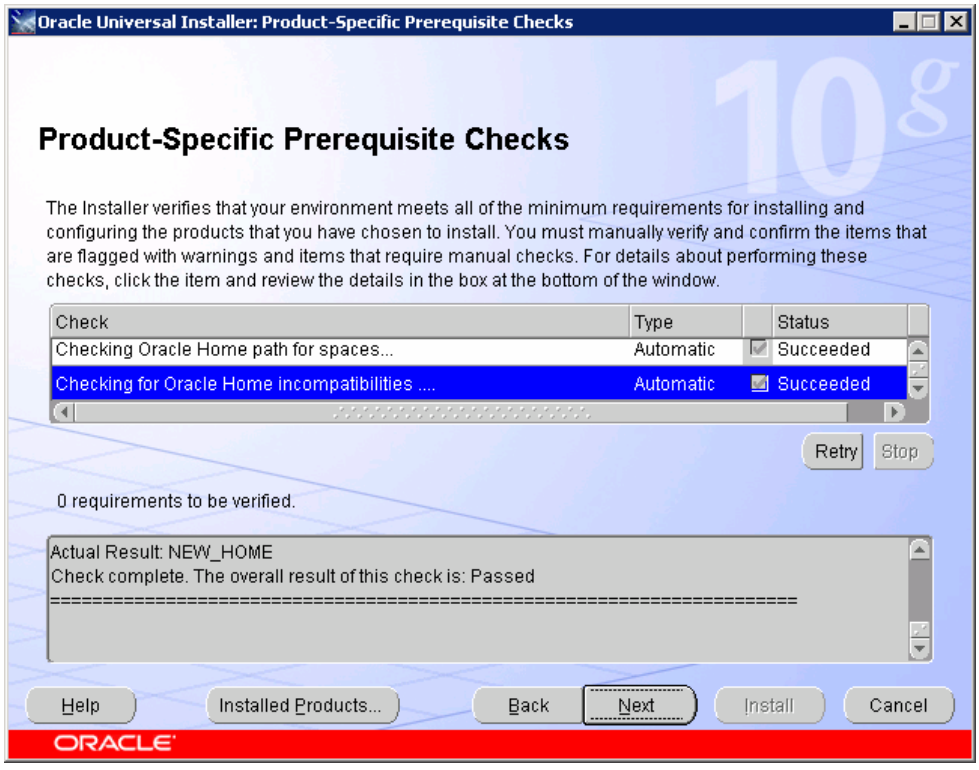
   Oracle JDBC/THIN interfaces 10.2.0.10

   Oracle Call Interface (OCI) 10.2.0.10

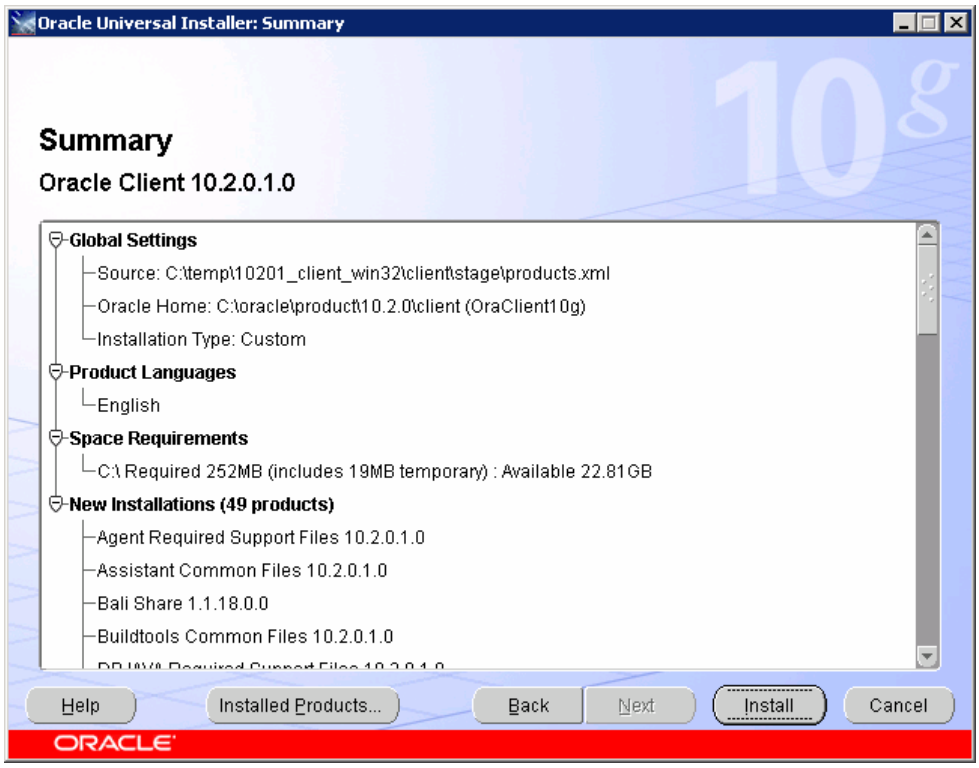
   Oracle Net 10.2.0.10

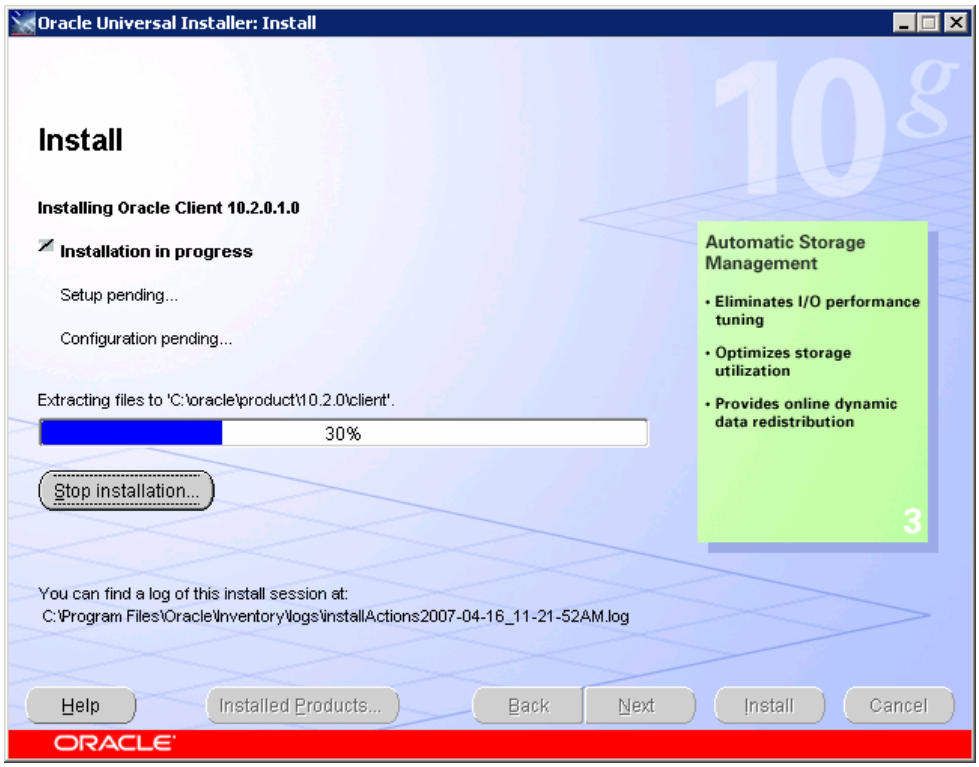


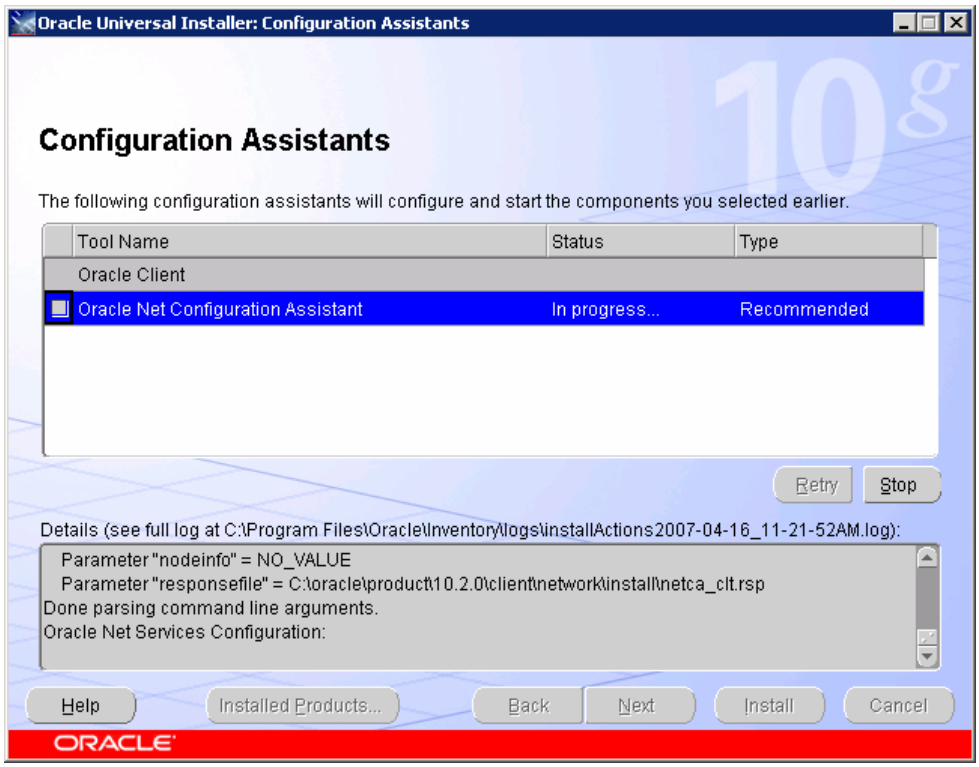
4. Next, some checking is performed, and the following screen appears. Correct any shortcomings, if these are found.



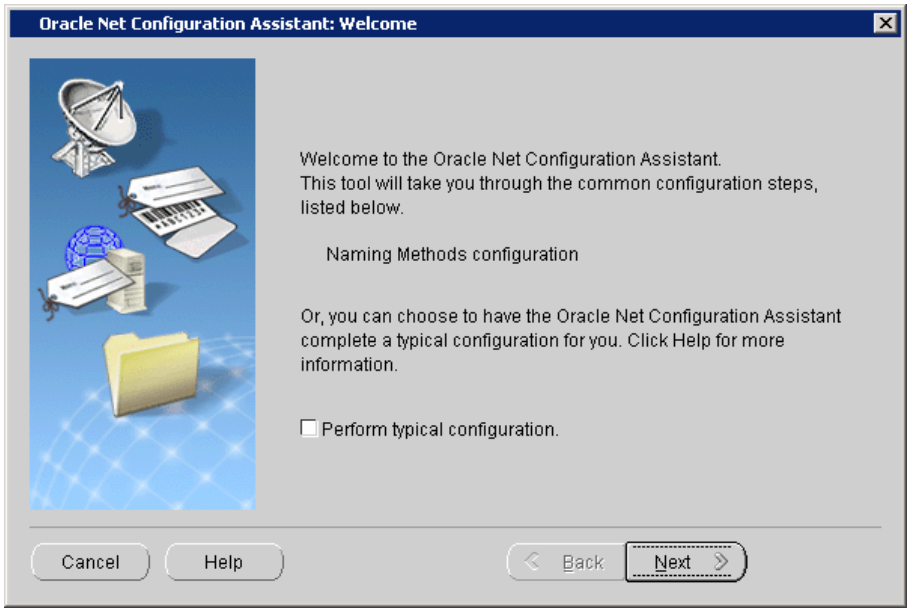
5. The summary screen displays more items than you may expect based on your previous selections. This is normal. Click **Install** to start the installation.

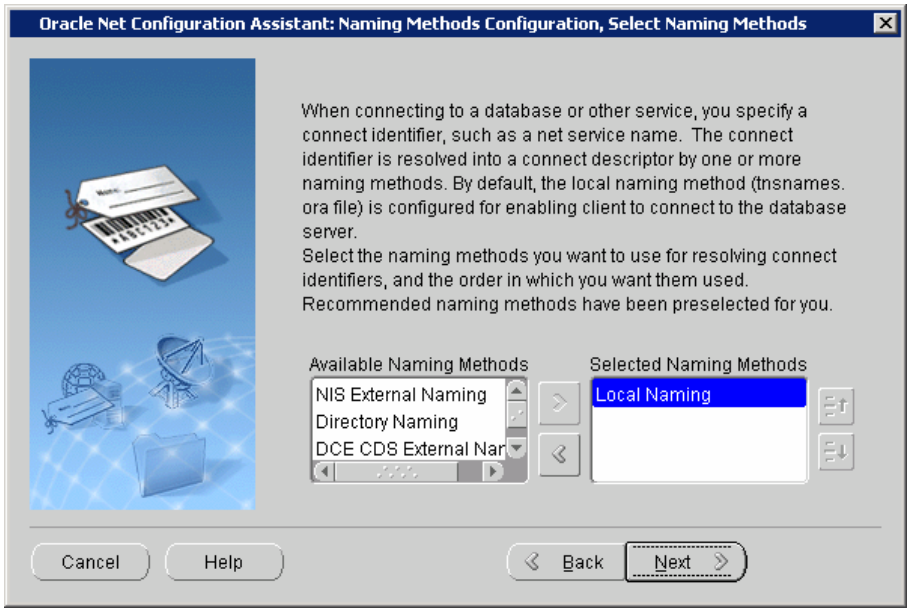

Oracle Universal Installer: Summary
Summary
Oracle Client 10.2.0.1.0
Global Settings
Source: C:\temp\10201_client_win32\client\stage\products.xml
Oracle Home: C:\oracle\product\10.2.0\client (OraClient10g)
Installation Type: Custom
Product Languages
English
Space Requirements
C:\ Required 252MB (includes 19MB temporary) : Available 22.81GB
New Installations (49 products)
Agent Required Support Files 10.2.0.1.0
Assistant Common Files 10.2.0.1.0
Bali Share 1.1.18.0.0
Buildtools Common Files 10.2.0.1.0
Help
Installed Products...
Back
Next
Install
Cancel
ORACLE

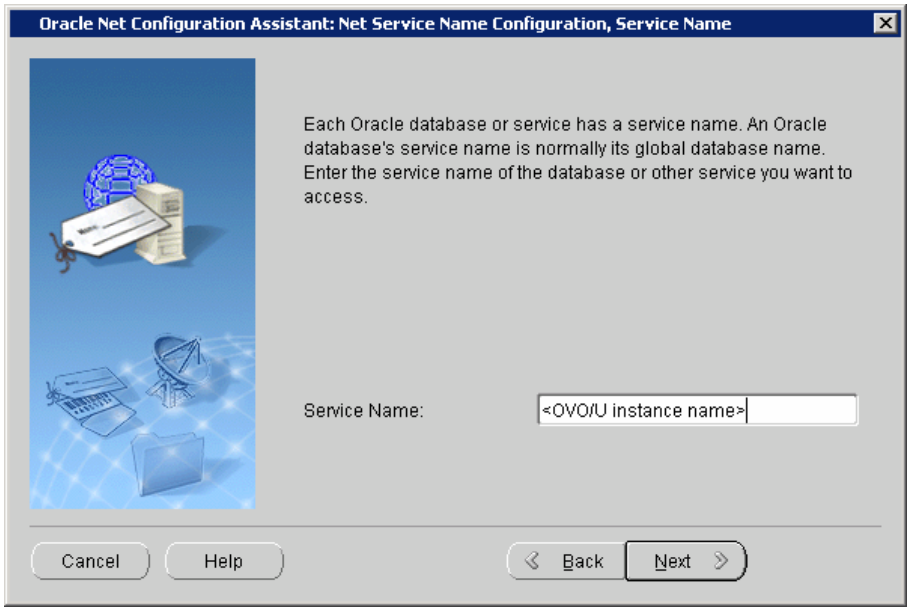

Oracle Universal Installer: Install
Install
Installing Oracle Client 10.2.0.1.0
Installation in progress
Setup pending...
Configuration pending...
Extracting files to 'C:\oracle\product\10.2.0\client'.
30%
Stop installation...
Automatic Storage Management
• Eliminates I/O performance tuning
• Optimizes storage utilization
• Provides online dynamic data redistribution
3
You can find a log of this install session at:
C:\Program Files\Oracle\Inventory\logs\installActions2007-04-16_11-21-52AM.log
Help
Installed Products...
Back
Next
Install
Cancel
ORACLE



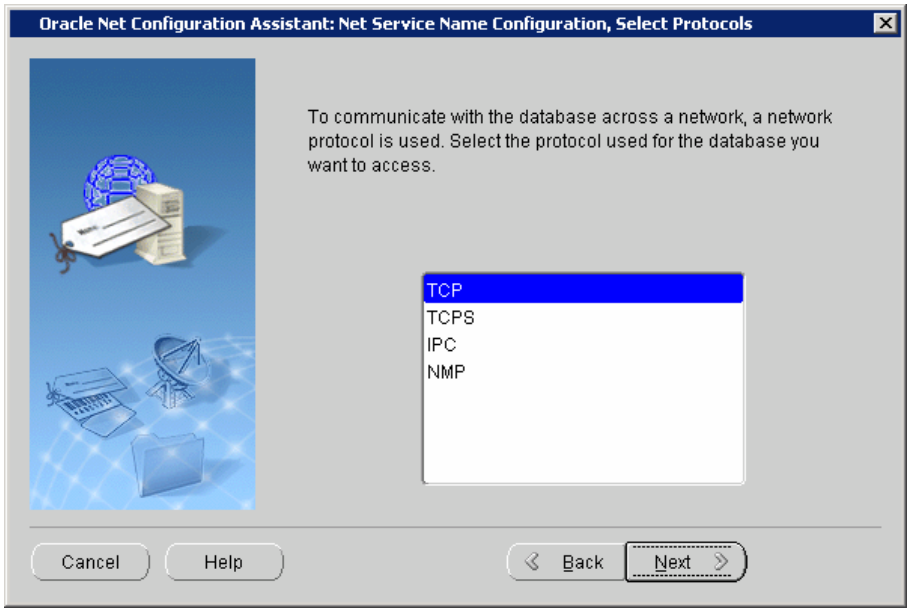
6. The Net Configuration Assistant now starts. Its screens may be hidden behind other screens, so if nothing appears, minimize each screen, one at a time. The Assistant will prompt you for information on the database supporting OVO for UNIX. Ensure that you have this information to hand, for example by asking an OVO and/or a database administrator.

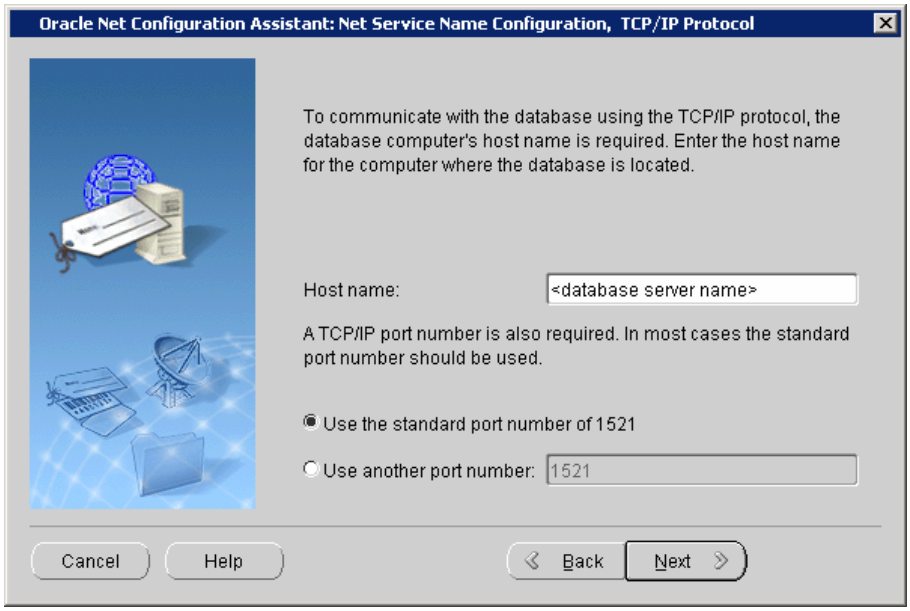


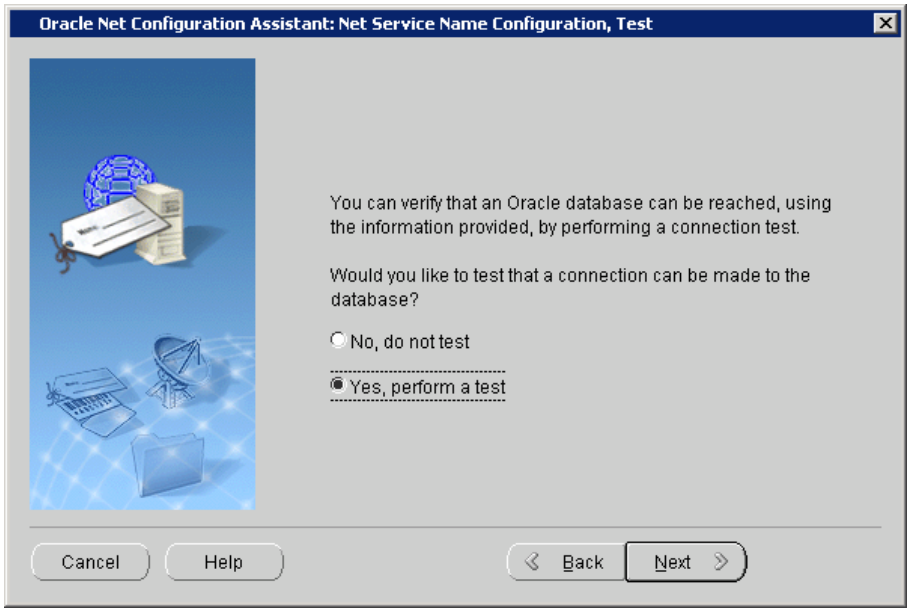




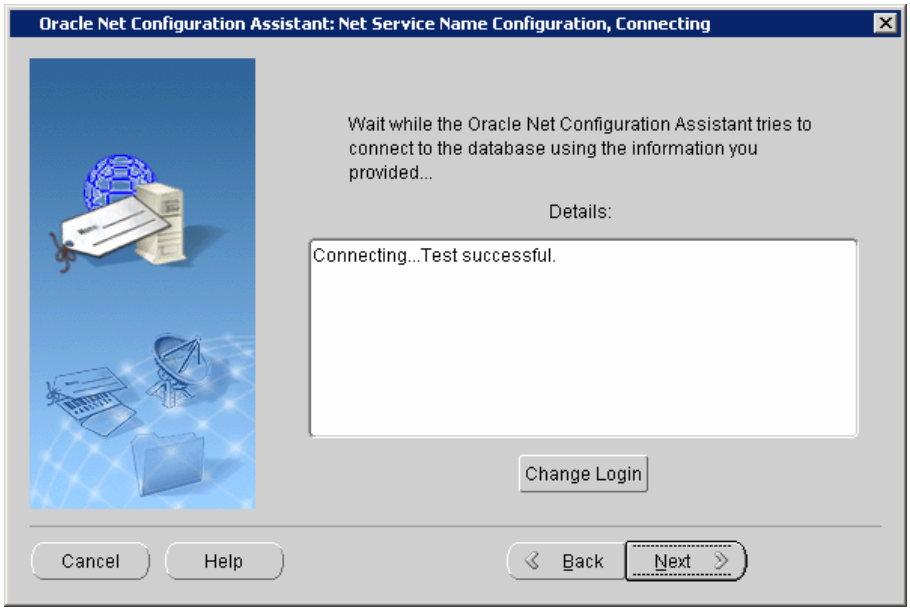
The service name is used to distinguish the Oracle database and its processes. The service name has been assigned at database creation and defaults to 'openview'.

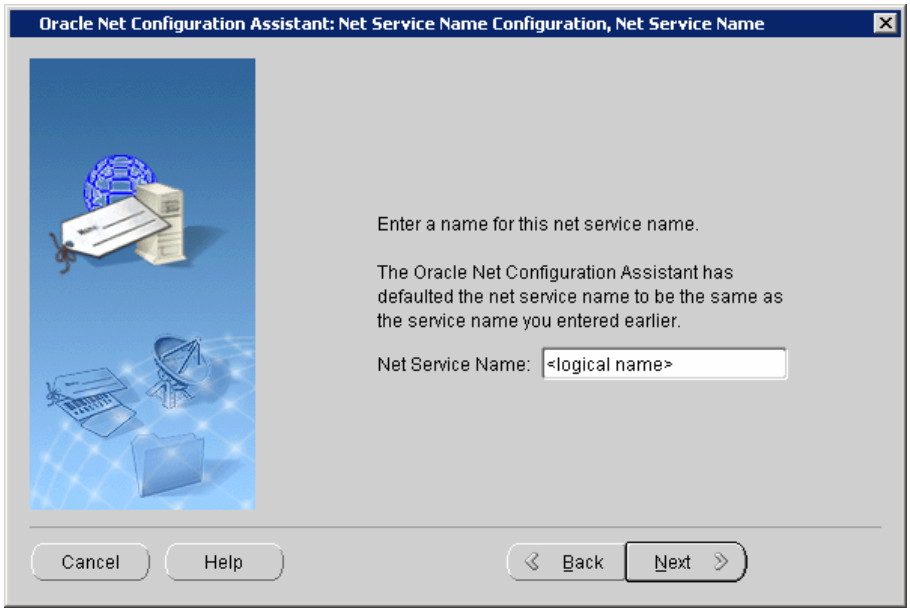

Oracle Net Configuration Assistant: Net Service Name Configuration, Select Protocols
To communicate with the database across a network, a network protocol is used. Select the protocol used for the database you want to access.
TCP
TCPS
IPC
NMP
Cancel
Help
Back
Next


Oracle Net Configuration Assistant: Net Service Name Configuration, TCP/IP Protocol
To communicate with the database using the TCP/IP protocol, the database computer's host name is required. Enter the host name for the computer where the database is located.
Host name:
<database server name>
A TCP/IP port number is also required. In most cases the standard port number should be used.
Use the standard port number of 1521
Use another port number:
1521
Cancel
Help
Back
Next

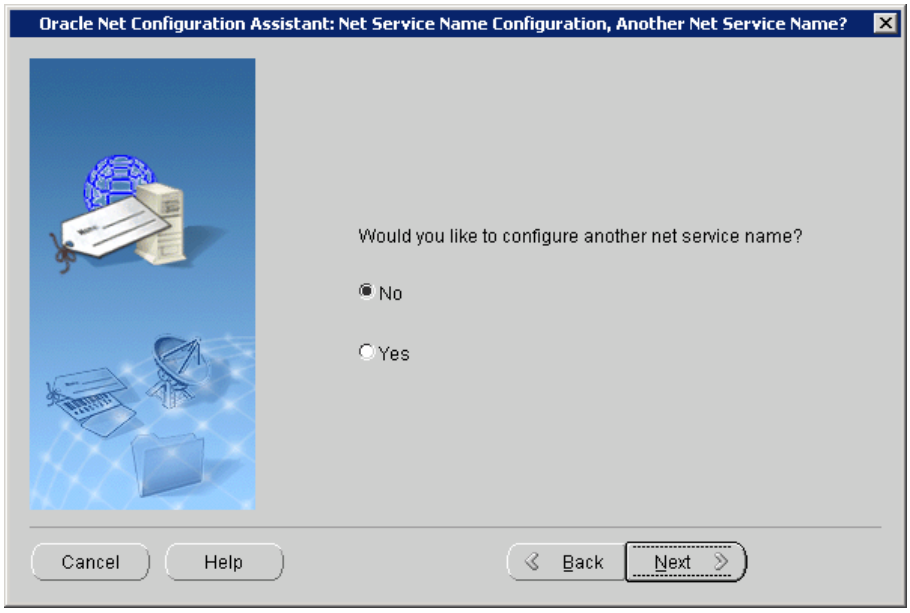


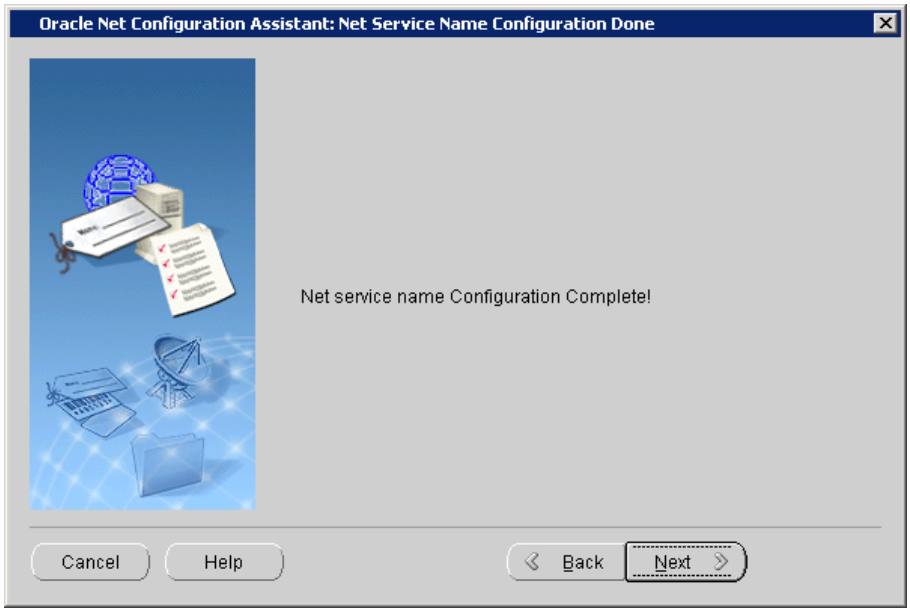
7. Testing is always recommended at this stage of the configuration. If the screen below displays a message other than the one shown here, correct the issues indicated.

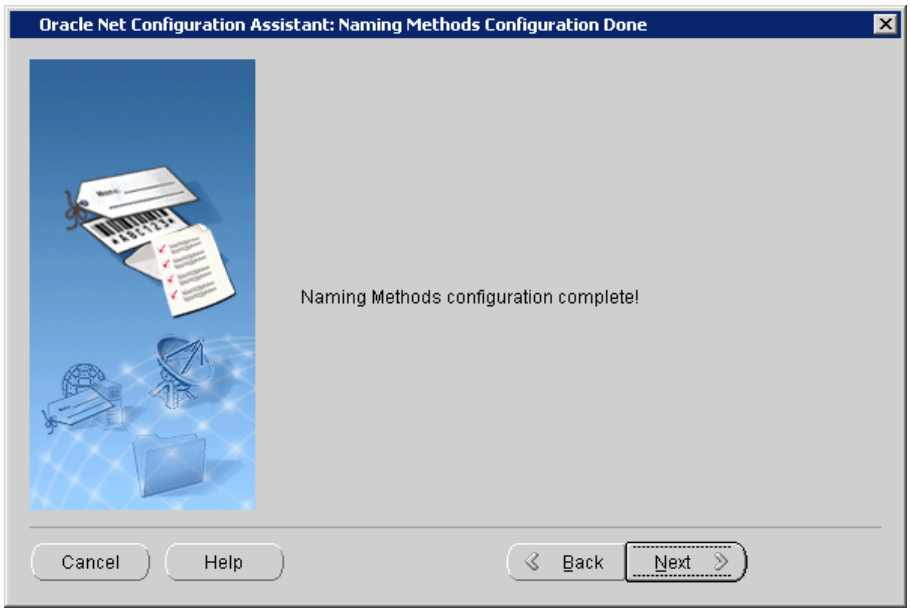


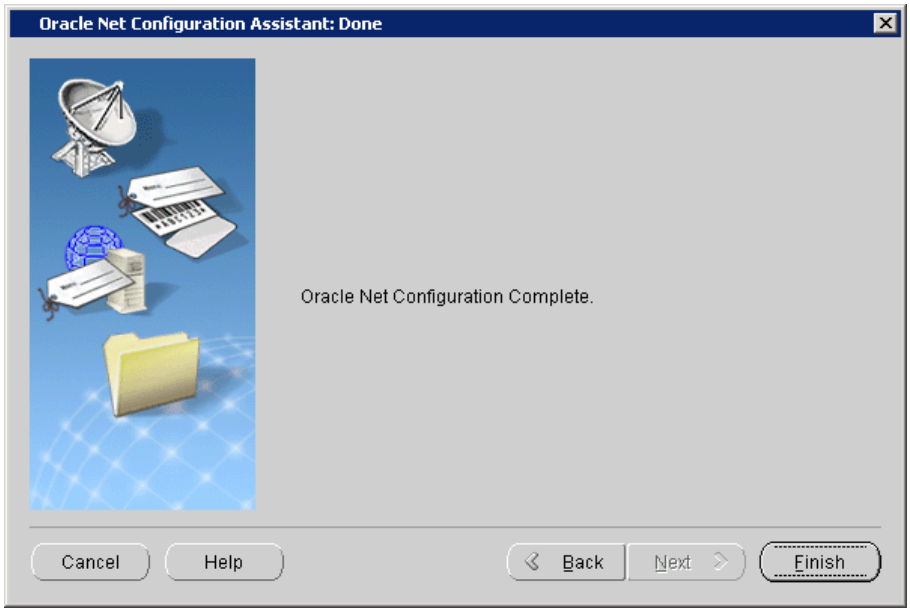


The name entered here is a logical name, used on the local machine for the connection to the OVO for UNIX database. A descriptive name is highly recommended.

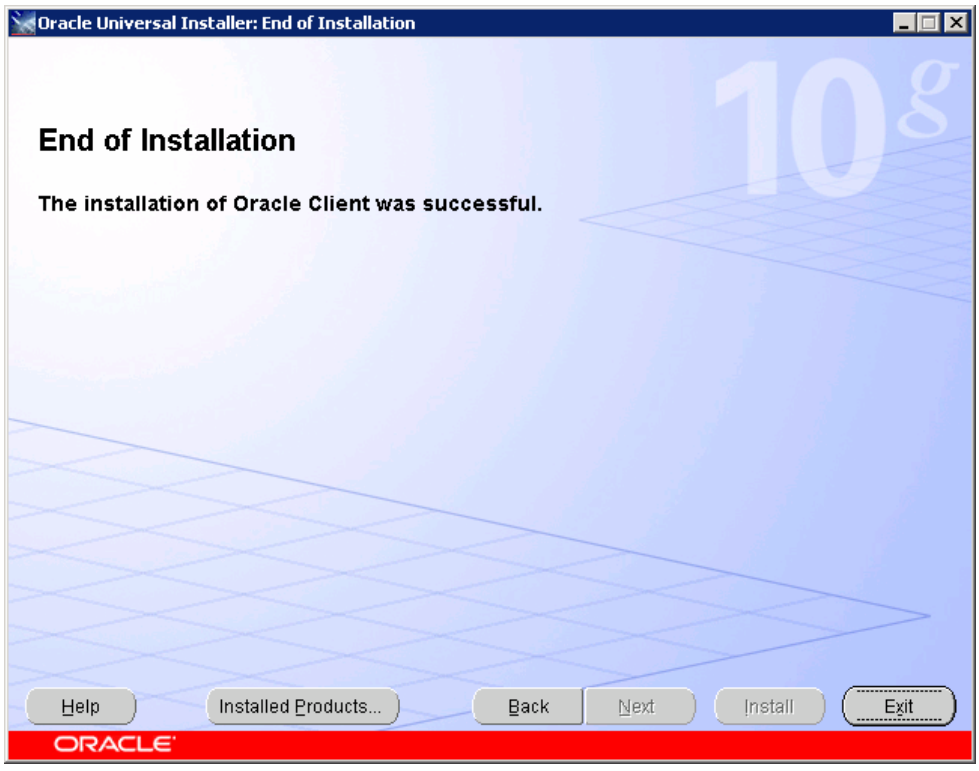


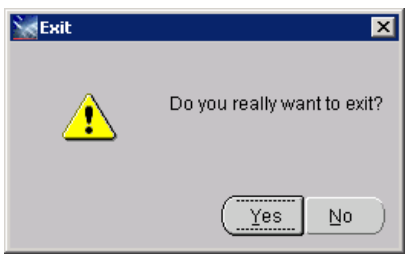

Oracle Net Configuration Assistant: Net Service Name Configuration Done
Net service name Configuration Complete!
Cancel
Help
Back
Next


Oracle Net Configuration Assistant: Naming Methods Configuration Done
Naming Methods configuration complete!
Cancel
Help
Back
Next



8. After you have run the Net Configuration Assistant, close the Universal Installer. Click the **Finish** button to confirm.





Configuring Oracle Net

While configuring the Connect-It connector for the Oracle driver you need to specify the server. Here, “server” refers to the Net Service Name that is required to connect to the OVO Oracle database. During the Oracle Net configuration (described as part of the Oracle Client installation in *Appendix A*), the Net Service Name is defined, and is saved in the `tnsnames.ora` file located in the network/admin directory of the Oracle home directory. A snippet from `tnsnames.ora` is show below.

```
MYOVOSERVER =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = myovoserver.mydomain.com)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = openview)
    )
  )
```

Here, MYOVOSERVER is the Net Service Name entered by the user. Replace `myovoserver.mydomain.com` with the FQDN of the server that runs the Oracle database of OVO. The default name of the Oracle instance of the OVO database is `openview`.

Appendix B – Node Bank Installation Files

This appendix contains information on the following:

- Where the install and data directories are located
- A list of install files, their location in the install directories and a description of the function of each file
- Notes on how to restore default versions of files

Location of Install and Data Folders

Windows <InstallDir>=C:\Program Files\HP OpenView,

<DataDir>= C:\Program Files\HP OpenView\Data.

By default the files are installed in the C:\Program Files\HP OpenView directory. If OpenView products are already installed on the same server but in another directory, then install the Node Bank Integration in this directory. If Node Bank integration is the first HP OpenView product on your install machine, you can select the install folder.

UNIX <InstallDir>=/opt/OV

<DataDir>=/var/opt/OV

List of Installed Files

File Name and Location	Contents
<InstallDir>/newconfig/conf/sc/sc-nb.scn	Scenario file for Connect-It, transfers data from OVO to SC
<InstallDir>/newconfig/conf/sc/sc-nb.ini	Configuration file for Connect-It
<InstallDir>/newconfig/conf/sc/sc-nb.cfg	Configuration file for Connect-It containing information on table joins
<InstallDir>/newconfig/conf/sc/sc-nb.bas	Basic language file containing global functions
<InstallDir>/newconfig/conf/sc/sc-nb-clean.scn	Scenario file for Connect-It, initializes synchronization in SC CMDB
<InstallDir>/newconfig/conf/sc/sc-nb-clean.ini	Configuration file for Connect-It
<InstallDir>/newconfig/conf/sc/sc-nb-response.dtd	Definition file for XML documents used in error reporting

<InstallDir>/newconfig/conf/sc/sc-nb-messagegroup.unl	ServiceCenter unload file for creating the following: ovomessagegroup table Web Services Data Policy global list for message groups
Copies of Files The files in the <DataDir>conf/sc/ directory are copies of the files in the <InstallDir>/newconfig/conf/sc directory. These are listed below. The files in DataDir are used during runtime. They will be adjusted to the customer environment. If you wish to restore the original files, copy the files from InstallDir to DataDir.	
<DataDir>/conf/sc/sc-nb.ini	Copy of the file in the InstallDir
<DataDir>/conf/sc/sc-nb.cfg	Copy of the file in the InstallDir
<DataDir>/conf/sc/sc-nb.bas	Copy of the file in the InstallDir
<DataDir>/conf/sc/sc-nb-clean.ini	Copy of the file in the InstallDir
<DataDir>/conf/sc/sc-nb-response.dtd	Copy of the file in the InstallDir
<DataDir>/conf/sc/sc-nb-messagegroup.unl	Copy of the file in the InstallDir
<DataDir>/conf/sc/sc-nb.scn	Copy of the file in the InstallDir. Will be adjusted for the customer environment
<DataDir>/conf/sc/sc-nb-clean.scn	Copy of the file in the InstallDir. Will be adjusted for the customer environment

Appendix C – Outage Installation Files

This appendix contains information on the following:

- Where the install and data directories are located
- A list of install files and a description of the function of each file
- Notes on how to restore default versions of files

Location of Install and Data Files

Windows <InstallDir>=C:\Program Files\HP OpenView,

<DataDir>= C:\Program Files\HP OpenView\Data.

By default the files are installed in the C:\Program Files\HP OpenView directory. If OpenView products are already installed on the same server but in another directory, then install the Node Bank Integration in this directory. If Node Bank integration is the first HP OpenView product on your install machine, you can select the install folder.

UNIX <InstallDir>=/opt/OV

<DataDir>=/var/opt/OV

Files Installed on the Connect-It server

The following files are installed on the Connect-It server:

File Name and Location	Contents
<InstallDir>/newconfig/conf/sc/sc-outage.scn	Scenario file for Connect-It. Extracts data from SC and stores it in the intermediate Oracle table sc_outage.
<InstallDir>/newconfig/conf/sc/sc-outage6.1.unl	ServiceCenter unload file for creating the following for SC 6.1.3: • the global list for message group • the global list for message severity • the global list for message operation • the form cm3r.ovo.outage to enter or modify outage data • the table ovooutage • the data policy for the ovooutage table • the web services definitions for the ovooutage table.
<InstallDir>/newconfig/conf/sc/sc-outage6.2.unl	ServiceCenter unload file for creating the

	following for SC 6.2: • the global list for message group • the global list for message severity • the global list for message operation • the form cm3r.ovo.outage to enter or modify outage data • the table ovooutage • the data policy for the ovooutage table. • the web services definitions for the ovooutage table
Copies of Files The files in the <DataDir>conf/sc/ directory are copies of the files in the <InstallDir>/newconfig/conf/sc directory. These are listed below. The files in DataDir are used during runtime. They will be adjusted to the customer environment. If you wish to restore the original files, copy the files from InstallDir to DataDir.	
<DataDir>/conf/sc/sc-outage.scn	Copy of the file in the InstallDir. Will be adjusted for the customer environment

Files installed on the OVO Management Servers

The following files are installed on the OVO management server(s):

Location	Contents
<InstallDir>/bin/ovsc-setup.sh	This is a configuration shell script that must be run after installing the SC-OVO integration. The script creates configuration files in the <DataDir>/conf/sc subdirectory based on the sample files located in the <InstallDir>/newconfig/conf/sc directory. It also creates the intermediate outage tables `sc_outage` and `sc_outage_status` in the Oracle database.
<InstallDir>/bin/ovsc-outage.sh	A shell script that is called periodically by crontab. The script checks if the upload process may run. When run it reads the outage data from the intermediate `sc_outage table`, creates an outage file, and uploads the outage data to OVO.
<InstallDir>/bin/ovsc.sh	A shell script that shows status information or details of the upload process. The script is also used to view outage information stored

	in the sc_outage table for upload to OVO. It is also used to manipulate the sc_outage_status table.
<InstallDir>/java/sc-outage.jar	Java Archive file that contains the Integration Java classes. This jar file is used by the ovsc-outage.sh script.
<InstallDir>/java/ojdbc14.jar	Java Archive file that contains Java classes of the Oracle JDBC driver.
<InstallDir>/newconfig/conf/SC/sc-outage.conf	A configuration file used by the ovsc-setup.sh configuration script to create the <DataDir>/conf/SC/sc-outage.conf file.
<InstallDir>/newconfig/conf/SC/sc-outage.cronjob	This file contains examples of crontab entries to schedule the outage process on the OVO Management Servers.
<DataDir>/conf/sc/sc-outage.cronjob	Copy of the file in the InstallDir, will be adjusted for the customer environment.

Files created during the setup process

The following files are created during setup:

<DataDir>/conf/sc/ sc-outage.conf	Configuration file created by ovsc-setup.sh. The file contains configuration data for the ovsc-outage.sh and ovsc.sh shell scripts.

Files created at runtime

The following files are created at runtime.

Location	Contents
<DataDir>/log/sc-outage.log	A log file created by ovsc-outage.sh containing information on the upload process and/or warning and error messages concerning the upload process.
<DataDir>/tmp/sc-outage.lck	A lock file created by ovsc-outage.sh to prevent concurrent runs of the outage process. The file is removed after the process has finished.
<DataDir>/tmp/sc-outage.tmp	A temporary file created by ovsc-outage.sh that contains intermediate results for further processing. The file is deleted once the process has finished.

Appendix D – Configuring the OVO Outage Table Clean-up Schedule

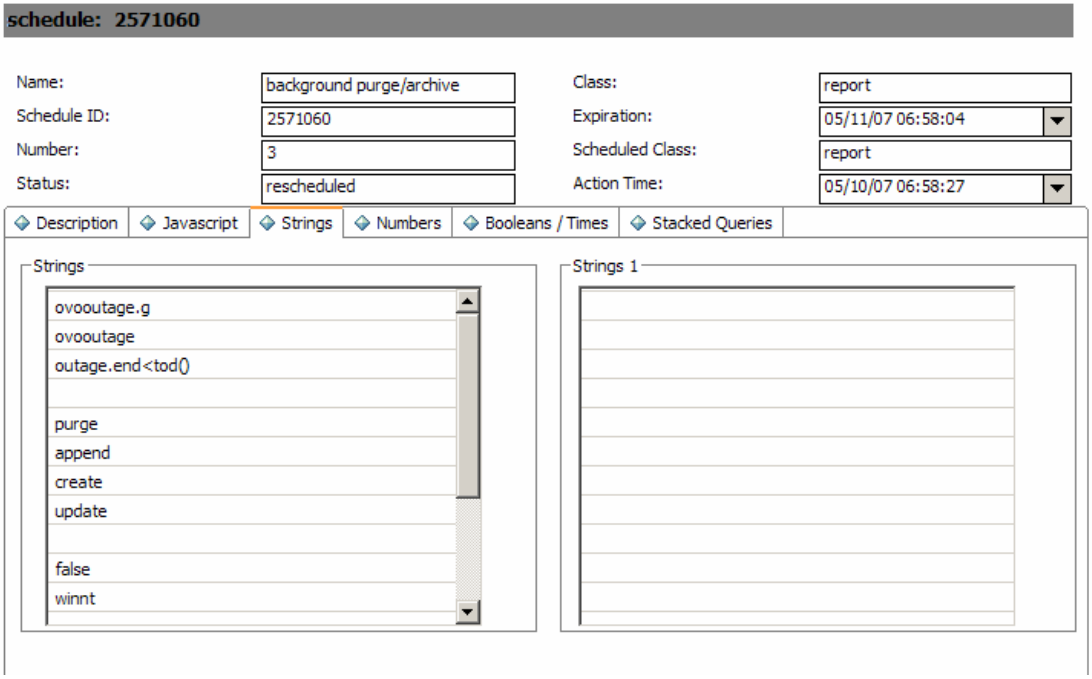
A schedule record is contained in the unload file. When the unload file and manual changes are completed, the schedule file can be configured to the client's requirements.

Schedule record

Name: `background purge/archive`

Class: `report`

There will be more than one schedule record for the above. To identify the record for OVO Outage, click the **Strings** tab. The screen shown below appears.



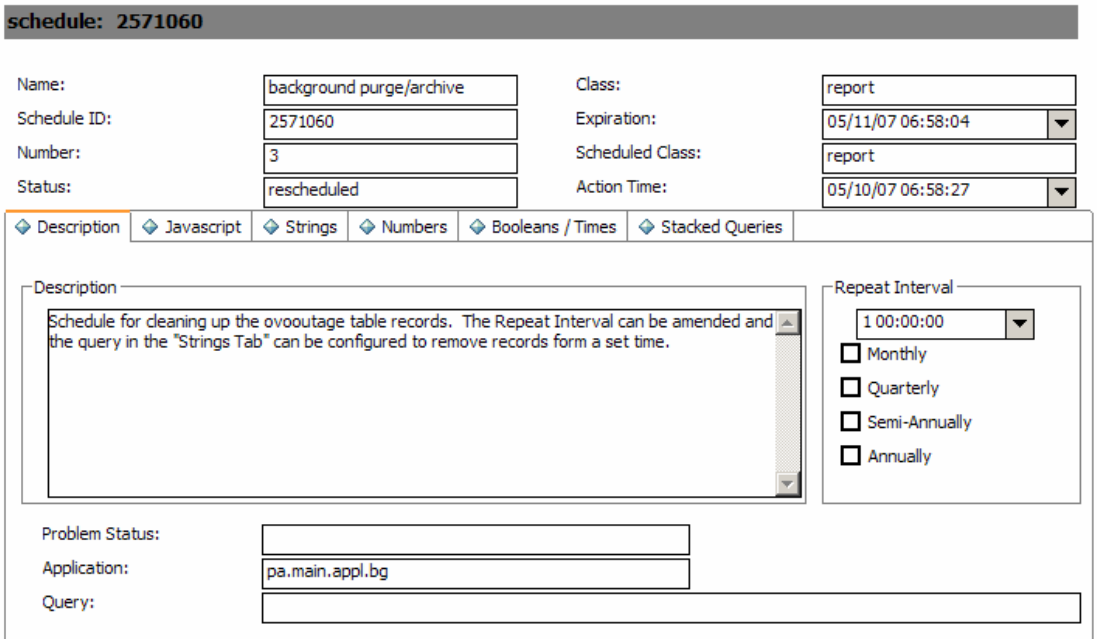
Configuring records for cleanup

The schedule in the unload is currently set to delete all OVO Outage records with an Outage End date earlier than the current system date/time.

The line query `outage.end<tod()` shown in the screen above can be amended if the client wishes to remove only those OVO Outage records with an Outage end date of 10 days ago. The query would then be set to `outage.end<tod()-'10 00:00:00'`

Configuring the Repeat Interval

The client can also set a frequency of execution for the schedule record. The Repeat Interval is defaulted to run the schedule daily for OVO Outages.



Appendix E – Configuring the slamodulecontrol Table

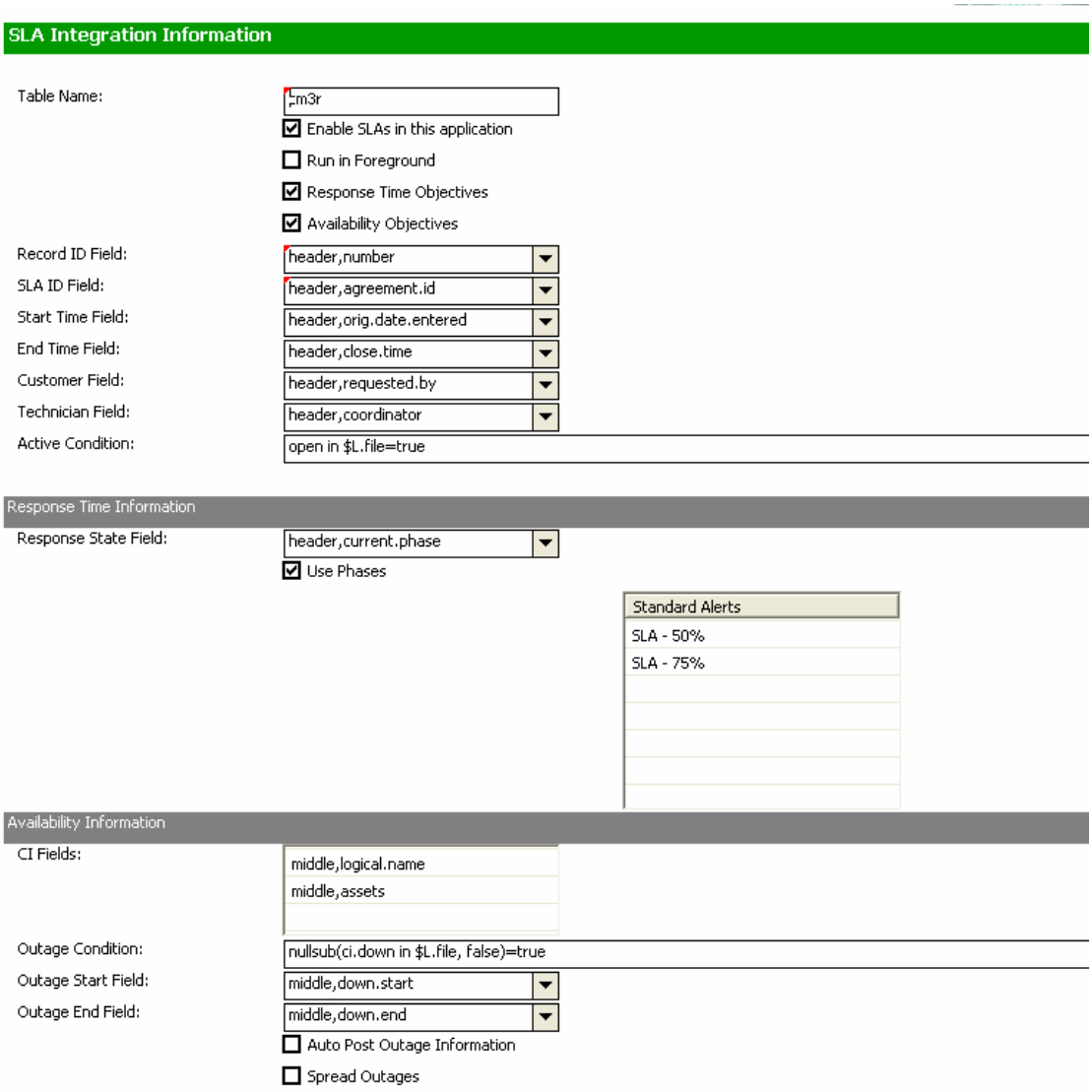
The OVO Outage wizard Utility uses the `slamodulecontrol` table to determine the fields used to store CI information and Outage Start and End dates in the Change (`cm3r`) and Change Task (`cm3t`) tables. To keep the wizard generic between modules, the customer can configure field names in the `slamodulecontrol` table in order to store specific information.

In the Availability Information Section of the slamodulecontrol record, the client can define the fields to use in file for the following:

CI Fields The wizard identifies the fields in the file that the client chooses to store CI information in the record.

Outage Start Field The wizard identifies the fields in the file that the client chooses to store the Outage Start Date field.

Outage End Field The wizard identifies the fields in the file that the client chooses to store the Outage End Date field.



In the Out of the Box system the fields used are defined in the `slamodulecontrol` table (the example above shows the `cm3r` table).

Change Management

Table Name:	`cm3r`
CI Fields:	`middle,logical.name`
	`middle,assets`
Outage Start Field:	`middle,down.start`
Outage End Field:	`middle,down.end`

Change Task

Table Name:	`cm3t`
CI Fields:	`middle,logical.name`

	middle,asset
Outage Start Field:	middle,down.start
Outage End Field:	middle,down.end

Configuration Example

In the Out of the Box system, when a Change Task is created, the asset array field receives the values from the assets array field in the Change record from which the Task was created. This means that when you add an OVO Outage record for the CI in the Task record from the wizard, it also presents the CIs (if any) in the related Change record. If a client only wishes to have OVO Outages created in Change Task for the associated CI in the Change Task, and does not wish to include the CIs from the Change, the slamodulecontrol record for cm3t can be configured to exclude the field middle,asset by removing it from the CI Fields array list. Similarly, new fields can be added if required by the client.