HP OpenView Operations Adapter Using Radia

Radia Adapter for HP OpenView Operations

Client Software Version: 1.2
for the UNIX and Windows operating systems

Server Software Version: 1.2
for the UNIX operating system

Manufacturing Part Number: T3424-90074

August 2004

Legal Notices

Warranty

Hewlett-Packard makes no warranty of any kind with regard to this document, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Hewlett-Packard shall not be held liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.

A copy of the specific warranty terms applicable to your Hewlett-Packard product can be obtained from your local Sales and Service Office.

Restricted Rights Legend

Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause in DFARS 252.227-7013.

Hewlett-Packard Company
United States of America

Rights for non-DOD U.S. Government Departments and Agencies are as set forth in FAR 52.227-19(c)(1,2).

Copyright Notices

Trademark Notices

Linux is a registered trademark of Linus Torvalds.

OpenLDAP is a registered trademark of the OpenLDAP Foundation.

Acknowledgements

Technical Support

Please select Support & Services from the following web site:

<http://www.hp.com/managementsoftware/services>

There you will find contact information and details about the products, services, and support that HP OpenView offers.

The support site includes:

- Downloadable documentation
- Troubleshooting information
- Patches and updates
- Problem reporting
- Training information
- Support program information

Preface

About this Guide

Who this Guide is for

This guide is for OpenView Operations (OVO) systems administrators who want to use Radia components to enhance their server-client environment.

This document assumes that the reader is an OVO administrator, and, as such, is familiar with the OVO software; is versed in its set up and configuration; and is familiar with its fundamental capabilities, tasks, and operations.

HP OVO replaces *VPO* (for UNIX), *VPW* (for Windows), and ITO.

What this Guide is about

This guide describes the installation, configuration, and operation of the *Radia Adapter for HP OpenView Operations* (Version 1) on UNIX and Windows platforms.

It also details the use of the various applications and monitors that are part of the adapter, and which are useful for operational and administrative tasks in an OVO environment.

Overview

Radia Adapter for HP OpenView Operations is a *Smart Plug-In* (SPI) for an OVO infrastructure. It provides pre-configured applications, monitors, and actions for immediate use in managing the operational and administrative environment of Radia servers and clients.

The initial configuration of the adapter, as provided by HP, separates its operational and administrative functions. This separation of work between the IT Operator and Radia administrator assumes that the latter is in charge of the Radia policy model, while the IT Operator handles the day-to-day operations of servers and the network.

For example, the adapter's initial configuration prevents Radia *administrators* from changing *operational* characteristics (other than dynamically setting tracing) that might cause servers to fail. Some functions might overlap, and functions can easily be reassigned to any console.

Radia Adapter for HP OpenView Operations applications have *line-mode* parameters that can be used to control the tracing level they produce. For more information on setting this and other trace-related parameters, refer to Table 1.2 on page 53.

Figure P.1 shows a conceptual view of the Radia Adapter for HP OpenView Operations in an OVO environment with Radia components.

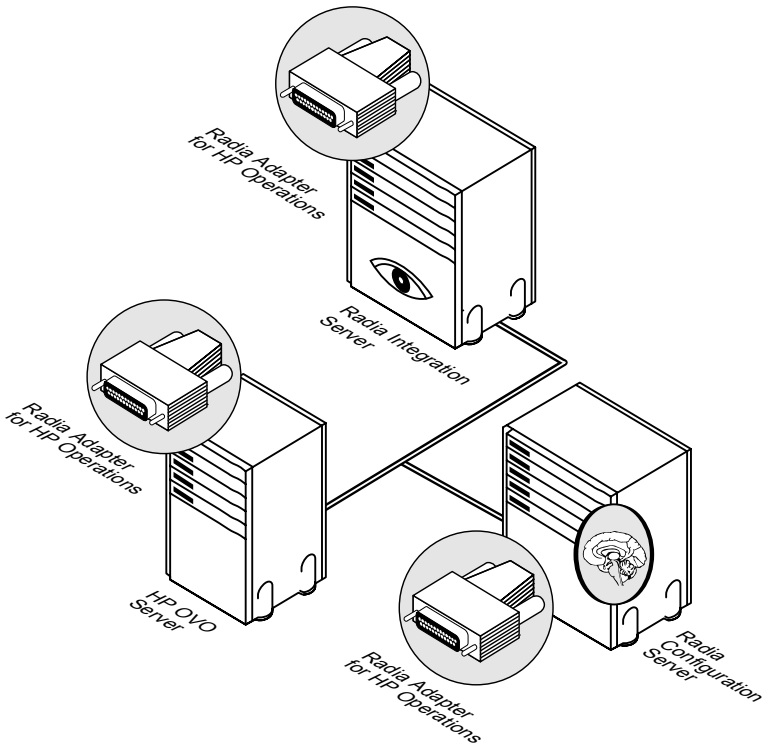


Figure P.1 ~ The Radia Adapter for HP OpenView Operations in an OVO environment.

Conventions

The style conventions used in this book are listed in Table P.1.

Table P.1 ~ Styles

Element	Style	Example
References	*Italic*	See the *Publishing Applications and Content* chapter in this book.
Dialog boxes and windows	**Bold**	The **Radia System Explorer Security Information** dialog box opens.
Code	`Andale Mono`	`radia_am.exe`
Selections	**Bold**	Open the **\Admin** directory on the installation CD-ROM.

Table P.2 lists the ways that various database elements are referenced.

Table P.2 ~ Usage

Element	Style	Example
Drives (system, mapped, CD)	Italicized placeholder	*SystemDrive*:\Program Files\Novadigm might refer to C:\Program Files\Novadigm on your computer. *CDDrive*:\client\radia_am.exe might refer to D:\client\radia_am.exe on your computer.
Files (in the Radia Database)	All uppercase	PRIMARY
Domains (in the Radia Database)	All uppercase	PRIMARY.SOFTWARE May also be referred to as the SOFTWARE domain in the PRIMARY file.
Classes (in the Radia Database)	All uppercase	PRIMARY.SOFTWARE.ZSERVICE May also be referred to as the ZSERVICE class in the SOFTWARE domain in the PRIMARY file.

Table P.3 describes terms that are used interchangeably throughout this book.

Table P.3 ~ Terminology*

* Depends on the context. May not always be able to substitute.

Term	May also be called
Application	software, service
Client	Radia Application Manager and/or Radia Software Manager
Computer	workstation, server
NOVADIGM domain	PRDMAINT domain **Note**: As of the 4.0 release of the database, the NOVADIGM domain is being renamed the PRDMAINT domain. Therefore, if you are using an earlier version, you will see the NOVADIGM domain in the database.
Radia Configuration Server	Manager, Active Component Server
Radia Configuration Server Database	Radia Database

Radia Adapter for HP OpenView Operations Terminology

Table P.4 presents a list of terms that are specific to this Radia adapter and can be found in this document.

Table P.4 ~ Radia Adapter for HP OpenView Operations Terminology

Term	Definition / Context
Applications	The programs and commands that are driven by the HP OVO Management Server console. The Radia Adapter for HP OpenView Operations applications are divided into six groups, as discussed in *Chapter 2: Application Bank*.
confile	The Tcl source files (**RCS_config.tcl**, **RIS_config.tcl**, and **CLI_config.tcl**) that are stored in the ZPATH at the managed node, and sourced into the applications at the beginning of each run. They contain the parameters (values, settings, etc.) that are to be used when running applications in the future. **Note**: The Radia Adapter for HP OpenView Operations can be customized by running the **Set RCS/RIS/CLI Parms** applications to put these parameters into the file.
edmprof file	The Radia Configuration Server's configuration file. This manual uses this non-platform specific, generic reference. • On UNIX operating systems, it is **.edmprof**. • On Windows operating systems, it is **edmprof.dat**.
managed node	Any HP OVO client, that is, a computer running the HP OVO agent, managed by the OVO Management Server. (See Figure P.1, on page 6).
Radia Client	Radia Application Manager, Radia Software Manager, and/or Radia Inventory Manager.
SPI (Smart Plug-In)	Any interfacing component that facilitates access to external products. The Radia Adapter for HP OpenView Operations is a SPI.
ZPATH	The directory that houses the *persistent* files, such as Tcl scripts and the *confile* (see *confile* in this table) that is used by the Novadigm SPI.

Radia Adapter for HP OpenView was created in the Hewlett-Packard *Software Change-and-Configuration Management* division—the former Novadigm, Inc. It is for this reason that within the Radia Adapter for HP OpenView product there are numerous occurrences of **Novadigm** (such as the **Novadigm Alarm** category and **Novadigm Events**).

Supporting Documentation

HP OVO Documentation

Table P.5 presents a list of HP OVO-related documents. We recommend reviewing this list, and having accessible those that are appropriate for the operating systems on which the Radia Adapter for HP OpenView Operations is being installed. This documentation will facilitate the installation and configuration of the Radia Adapter for HP OpenView Operations.

Table P.5 ~ HP OVO Documentation

HP OpenView VantagePoint Operations
HP OpenView Reporter Concepts Guide
HP OpenView VantagePoint Operations for HP-UX Installation Guide for the Management Server
HP OpenView VantagePoint Operations for Sun Solaris Installation Guide for the Management Server
HP OpenView VantagePoint Operations for UNIX Administrator's Reference Volume I
HP OpenView VantagePoint Operations for UNIX Administrator's Reference Volume II
HP OpenView VantagePoint Operations for UNIX Concepts Guide
HP OpenView VantagePoint Operations for UNIX Error Message Reference
HP OpenView VantagePoint Operations for UNIX Reporting and Database Schema
HP OpenView Performance Agent & GlancePlus for UNIX, Tracking Your Transactions for 10.20 and 11.x
HP OpenView VantagePoint Performance Agent Installation & Configuration Guide for IBM RS/6000 Systems
HP OpenView VantagePoint Performance Agent Installation & Configuration Guide for Sun Solaris Systems
HP OpenView VantagePoint ServiceNavigator Concepts and Configuration Guide
Release Notes
HP OpenView Performance Manager for HP-UX Release Notes
HP OpenView VantagePoint Performance Manager for HP-UX 11.11 Release Notes

Radia Documentation

Table P.6 presents a list of Radia publications that are associated with the various components of the Radia product suite. These might be referred to in this manual. Consult the Radia library and the HP OpenView support web site.

Table P.6 ~ Radia Documentation

Radia Component	Radia Manual
Radia Configuration Server	*Radia Configuration Server Guide*
	Radia Configuration Server Messages Guide
Radia Integration Server	*Radia Inventory Manager Guide*
Radia Client	*Radia Software Manager Guide*
	Radia Inventory Manager Guide
	Radia Application Manager Guide
Additional Titles	*Radia Management Portal Guide*
	Radia REXX Programming Guide
	Radia System Explorer Guide

Contents

Installation

This chapter details the installation of the Radia Adapter for HP OpenView Operations, which is installed on the HP OVO Management Server. Once installed on the OVO server, an application is provided to install the components of the adapter that run on managed nodes.

The configuration of the Radia Adapter for HP OpenView Operations is detailed in the next chapter, *Radia Adapter for HP OpenView Operations Parameters*, starting on page 51.

The complete installation is comprised of eight basic steps:

1. Installing the Radia Adapter for HP OpenView Operations on the OVO Management Server
2. Configuring the Radia Environment
3. Populating the NVD Node Groups
4. Assigning NVDOM Message Groups
5. Distributing the SPI Setup Application to Managed Nodes
6. Running the SPI Setup Application on Managed Nodes
7. Distributing Templates to Managed Nodes
8. Configuring Radia Adapter for HP OpenView Operations Parameters

The first seven steps are contained in this chapter, and step eight is covered in Chapter 2.

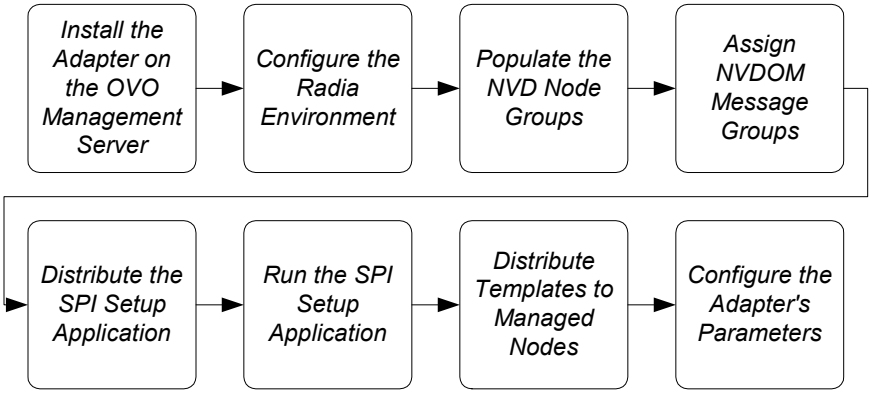


Figure 1.1 ~ The eight steps of the Radia Adapter for HP OpenView Operations installation.

Note

A flow chart, similar to that shown above, is located at each step of the installation process indicating which step is being documented.

Installing the Radia Adapter for HP OpenView Operations on the OVO Management Server

This section presents the installation of the Radia Adapter for HP OpenView Operations on an OVO Management Server, as well as a few pre-installation considerations.

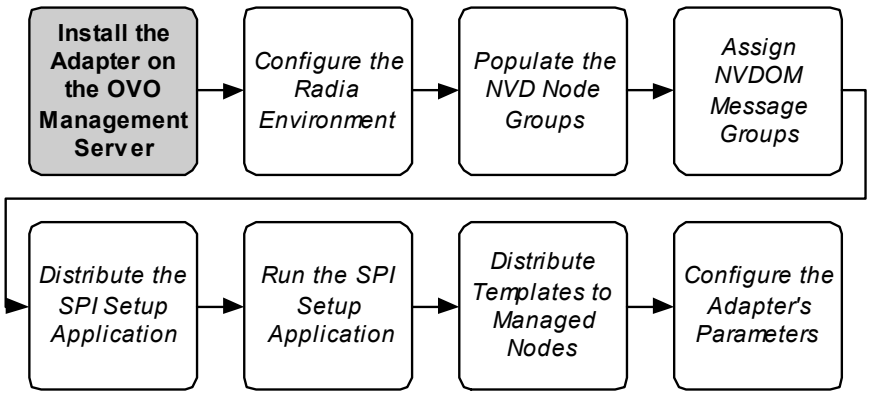


Figure 1.2 ~ Step 1 - Install the Radia Adapter for HP OpenView Operations on the OVO Management Server.

System Recommendations and Prerequisites

Radia Prerequisites

In order for the Radia Adapter for HP OpenView Operations to work properly, the Radia components must be at the minimum levels shown in Table 1.1.

Table 1.1 ~ Radia Component Minimum Levels

Radia Component	Minimum Version Level
Radia Configuration Server	4.5
Radia Integration Server	2.3
Radia Clients	3.0 **Note**: A Radia Client can be: *Radia Software Manager*, *Radia Application Manager*, or *Radia Inventory Manager*. This version requirement applies to all three.

The Radia documents that support these components are listed in Table P.6, on page 11.

HP OVO Prerequisites

In addition to the Radia pre-requisites, the following OVO requirements must be met.

- An OVO agent must be running on every managed node where Radia components are to be managed. These OVO agents must be at a *minimum level* of **A.06.08**

> **Note**
>
> The OVO agent installation is not covered in this document. Consult the appropriate HP OVO documentation for OVO agent installation details.

- The OVO Server being used with the Radia Adapter for HP OpenView Operations must also be at level **A.06.08** or greater.
- Radia Configuration Servers, Radia Integration Servers, and Radia Clients must be installed on the managed nodes *before* the Radia Adapter for HP OpenView Operations can be configured fully. Otherwise, the applications associated with the Radia Adapter for HP OpenView Operations will not be able to run.
- An OVO agent must be installed on each machine that will use OVO to manage the Radia components.

OVO Agent Support

The Radia Adapter for HP OpenView Operations is supported with the following OVO agents (shown by manufacturer, platform, and OS).

- HP S700 HP-UX10
- HP S800 HP-UX10
- HP PA-RISC HP-UX11
- IBM RS6000 AIX
- Linux Intel
- Microsoft Intel NT/Win2000
- SUN SPARC Solaris

> **Note**
>
> Check the **readme.txt** file (found in the root of the distribution CD-ROM) for a list of new platforms that become available subsequent to the printing date of this manual.

Pre-Installation

- The installation program can be run in either of two modes, *full-screen* or *line*. If the DISPLAY environment variable is set, the full-screen mode will run.
- Be sure to review any **readme.txt** files found in the root of the distribution CD-ROM.
- We recommend backing up your OVO database.

  Should you need to uninstall this adapter, the easiest way to remove it is to restore the database from a backup. The adapter cannot be uninstalled by any means other than manually deleting its elements. For more information, see the section, *Uninstalling the Radia Adapter for HP OpenView Operations*, starting on page 28.
- Make sure that the OVO environment variables have been set. If not, issue the following command:

  ```
  # ./opt/OV/bin/ov.envvars.sh
  ```

  Consult the appropriate HP OpenView documentation for details on how to do this if necessary.

Notes

When running in the full-screen mode, it is not necessary to set the OVO environment variables because the installation program will look for the **/opt/OV/bin** directory and, if it exists, will do the installation for that OVO system. There is a confirmation prompt before this action is taken.

If the **/opt/OV/bin** directory does not exist, the installation will stop, and request that the OVO environment variables be set.

When installing on HP-UX, use the **pfs_mount** command when mounting the CD-ROM that contains the Radia Adapter for HP OpenView Operations media. This will allow long file names to be displayed properly.

Installation

This section outlines the installation of the Radia Adapter for HP OpenView Operations on an OVO Management Server.

To install the Radia Adapter for HP OpenView Operations

1. On the distribution CD, change to the directory for the operating system on which the adapter is being installed.
   - For HP-UX, go to **/hpux**.
   - For Solaris, go to **/solaris**.
2. Type **./install**, and press **Enter**.

   The **Welcome** window opens.

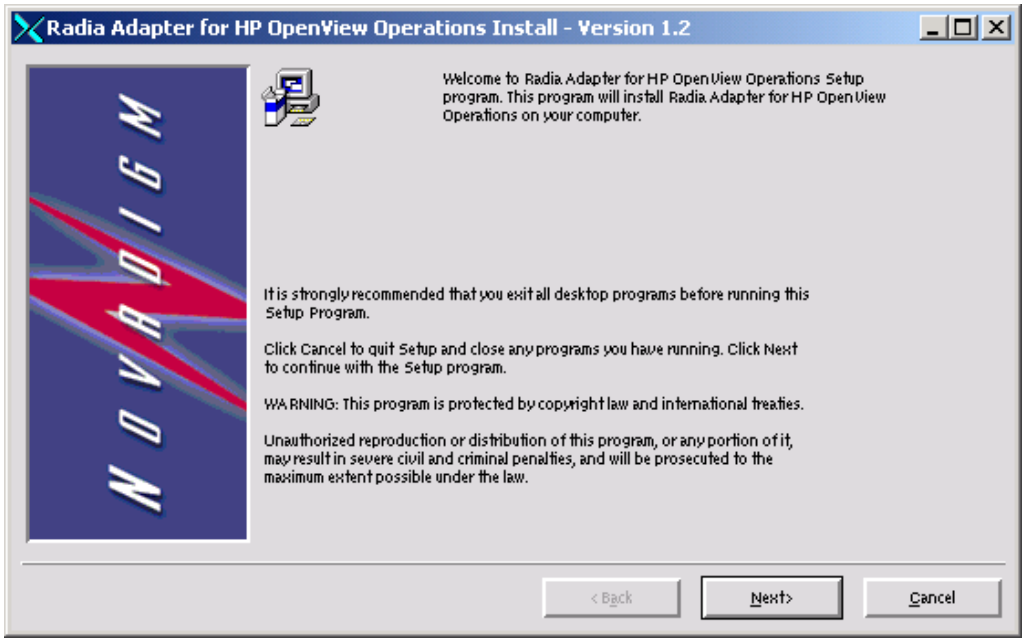


Figure 1.3 ~ The Welcome window.

3. Click **Next**.

   You will be prompted to stop the OVO Management Server.

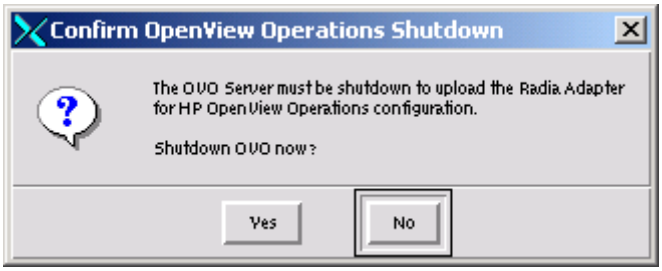


Figure 1.4 ~ The prompt to stop the OVO Management Server.

4. Click **Yes**.

The OVO Management Server will be stopped and the Radia Adapter for HP OpenView Operations' configuration data will be uploaded to the OVO database.

Note

A log of this upload can be found in **opccfgupld.log**. This and other installation logs can be found in the directory below, where **$OV_PRIV_LOG** usually defaults to **/var/opt/OV/log**.

```
$OV_PRIV_LOG/OpC/mgmt_sv
```

For more information, see the section, *Installation Details*, on page 26.

If the upload operation is successful, the OVO Management Server will be restarted.

The **Installation Successful** window will confirm the success of the installation.

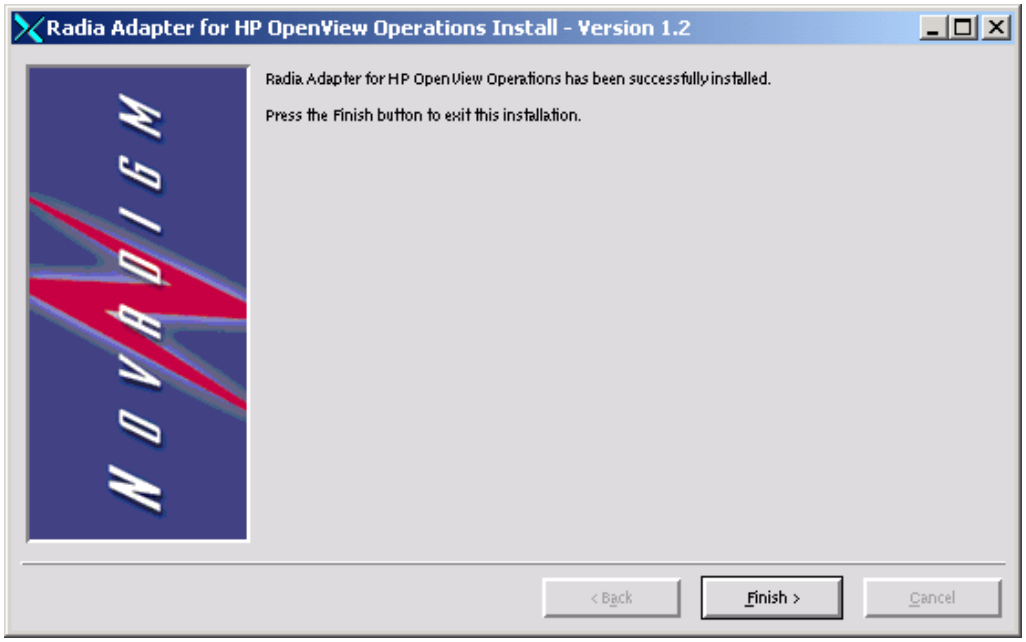


Figure 1.5 ~ The Installation Successful window.

5. Click **Finish**.

This completes the Radia Adapter for HP OpenView Operations installation on the OVO Management Server.

Post-Installation

Installation Details

The details of the installation can be viewed in order to verify its success.

- To view the output of the **opccfgupld** command, navigate to:

  `/var/opt/OV/log/OpC/mgmt_sv/opccfgupld.log`

> **Note**
>
> If there have been any errors, check the logs that have a recent file update timestamp in:
>
> `/var/opt/OV/log/OpC/mgmt_sv`

OVO Configuration and Operations Areas

Once the Radia Adapter for HP OpenView Operations has been installed, its configuration and operational parameters will be established in the following eight areas.

- Node Bank
- Node Hierarchy Bank
- Node Group Bank
- Message Group Bank
- Application Bank
- User Bank
- User Profile Bank
- Message Source Templates

The operations that are associated with these sections are discussed in further detail throughout this manual.

Distributing the Adapter

Once the Radia Adapter for HP OpenView Operations has been installed on the OVO Management Server, parts of it must be distributed to each OVO agent (managed node). This is discussed in the sections, *Distributing the SPI Setup Application to Managed Nodes* (on page 44), *Running the SPI Setup Application on Managed Nodes* (on page 46), and *Distributing Templates to Managed Nodes* (on page 50).

Note

Because the Novadigm SPI code is written in Tcl, nvdkit and the Tcl scripts have to be distributed to each agent machine. For this, the adapter has been developed with OVO applications that run FTP under-the-covers to do the required distribution.

Novadigm Application Groups

Once the Radia Adapter for HP OpenView Operations is installed on the OVO Management Server, eight Novadigm application groups will be present. Verify this by clicking **Window**, then **Application Bank**, and checking for the following eight **NVD** application groups.

- NVD Admin RCS
- NVD Admin RIS
- NVD Op CLI
- NVD Op RCS
- NVD Op RIS

- NVD CLI SPI Setup
- NVD RCS SPI Setup
- NVD RIS SPI Setup

Uninstalling the Radia Adapter for HP OpenView Operations

If the Radia Adapter for HP OpenView Operations needs to be uninstalled (for example, for software updates), follow the procedure outlined below.

To uninstall the Radia Adapter for HP OpenView Operations from managed nodes

1. Remove from your managed nodes all *monitors* and *templates* that are associated with the Radia Adapter for HP OpenView Operations.
   - In the **Node Bank** window, click **Actions**, **Agents**, and **Assign Templates** and un-assign all NVDOM templates.
   - In the **Node Bank** window, click **Actions**, **Agents**, and **Install/Update SW & Config** and select **Templates** and **Monitors** from the **Components** list. (Do not select **Commands**.) Also, select **Force Update**.
2. Run the **Uninstall Unix SPI** and **Uninstall Win SPI** applications on all applicable managed nodes.
3. Remove from your managed nodes all commands that are associated with the Radia Adapter for HP OpenView Operations.
   - In the **Node Bank** window, click **Actions**, **Agents**, and **Install/Update SW & Config** and select **Actions** and **Commands** from the **Components** list. Also, select **Force Update**.

To uninstall the Radia Adapter for HP OpenView Operations from the OVO Server

1. Navigate to the directory in which the adapter's files are installed.

   On a standard installation, this directory is:

   ```
   /var/opt/OV/share/databases/OpC/mgd_node/customer
   ```

   The platform-dependent subdirectories begin at this point.
2. Navigate to the applicable, **hp**, **ibm**, **ms**, or **sun** directory and delete all files that have the prefix, **NVDOM**.
3. Delete the directory, **other**, and all of its contents.
4. Access the OVO user interface, and delete the following:

Table 1.2 ~ NVDOM Deletions	
Under…	**delete…**
Node Group Bank	all NVDOM *groups*
Message Group Bank	all NVDOM *message groups*
Application Bank	all NVDOM *applications* and *groups*
User Profile Bank	all NVDOM *profiles*
Message Source Templates	all NVDOM *templates*, *monitors*, and *groups*

5. Ensure that all **NVDOM** elements have been deleted.

The Radia Adapter for HP OpenView Operations has been uninstalled successfully.

Configuring the Radia Environment

Once the Radia Adapter for HP OpenView Operations has been installed on the OVO Server, the Radia environment must be configured. This configuration is comprised of three primary steps, which are outlined in this section.

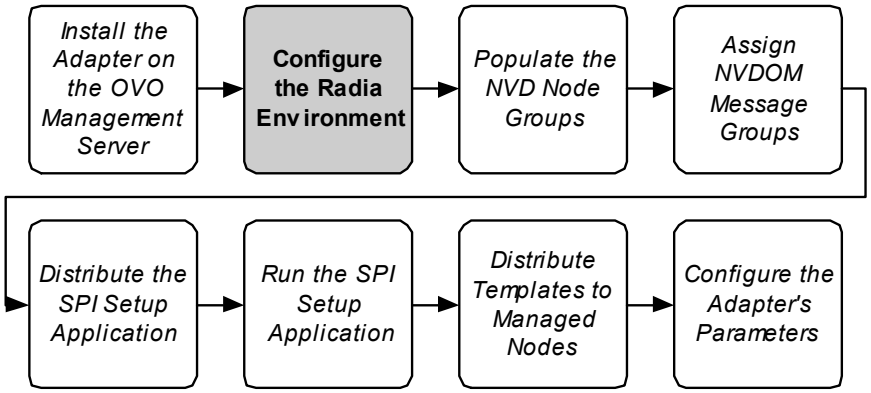


Figure 1.6 ~ Step 2 - Configure the Radia Environment.

The RCS EDMPROF File

Any Radia Configuration Server events, on which the adapter will report, will be reported as *SNMP traps*. To ensure these events are received at the message browser, the Radia Configuration Server's SNMP functionality must be configured.

Refer to the section, *RCS EDMPROF File Sections*, on page 32, for information on which sections of the edmprof file must be modified in order to accommodate SNMP functionality within the Radia Configuration Server.

Message Groups

Assign NVDOM *message groups* to the user that will browse Radia events.

For an overview of how to assign the NVDOM message groups, refer to the section, *Assigning NVDOM Message Groups*, on page 42.

Applications, Monitors, and Actions

When running *applications*, *monitors*, and *actions*, the adapter has defaults for certain critical parameters. These defaults might not be correct for all managed nodes. Therefore, the adapter provides the following three optional applications that relay information to the SPI:

- **Set RCS Parms**
- **Set RIS Parms**
- **Set Radia Client Parms**

These applications allow the correct values for these critical parameters to be pre-configured. For an overview of these applications, see, *Running the Set XXX Parms Applications*, starting on page 56; and for detailed configuration information, see *Chapter 3: Application Bank* starting on page 65.

RCS EDMPROF File Sections

This section presents information on which sections of the edmprof file must be modified in order to accommodate SNMP functionality within the Radia Configuration Server.

MGR_ATTACH_LIST

This section of the edmprof file is where tasks are attached to the Radia Configuration Server's start-up processing. The ZSNMPMGR and ZDIAGMGR tasks must be added to this processing. Specify them as shown in the following example.

```
[MGR_ATTACH_LIST]
CMD_LINE=(zsnmpmgr) RESTART=YES
CMD_LINE=(zdiagmgr) RESTART=YES
```

MGR_DIAGNOSTIC

The Radia Configuration Server's Diagnostic Manager component (ZDIAGMGR) is a task/process that periodically checks for specific conditions.

The settings contained in this section are not directly used by the SNMP agent residing on the Radia Configuration Server (RCS). However, they control when and how certain SNMP traps are issued by ZDIAGMGR. This component exists primarily to issue SNMP traps, but will also issue messages to the Radia log when the conditions it detects occur. Diagnostic Manager traps and messages are controlled by the following settings.

DIAGNOSTIC_INTERVAL

- The interval (in seconds) between Diagnostic Manager checks. The default is **900** seconds (15 minutes).

DIAGNOSTIC_MIN_DB_BYTES

- The minimum number of free bytes on the volume that contains the Radia Database. The default is **52428800** (50 MB).

  This value is used to produce the managerDBSpaceLow condition found in the **NVDOM RCS Traps** template.

DIAGNOSTIC_MIN_LOG_BYTES

- The minimum number of free bytes on the volume that contains the RCS log. The default is **26214400** (25 MB).

  This value is used to produce the managerLogSpaceLow condition found in the **NVDOM RCS Traps** template.

Note

A **CMD_LINE=(zdiagmgr)** must be added to the edmprof file in order for the Diagnostic Manager to run. The check for minimum database and log volume space will always be performed once (when the RCS starts up), regardless of whether ZDIAGMGR is run.

MGR_LOG

This section of the edmprof specifies the logging directory and logging options for the RCS logging facility.

DISABLE_NT_EVENT_LOGGING = YES|NO

- If NO is coded, messages that are sent to the RCS log will be sent to the Windows Event Log also, if they have been configured for Windows Event logging in the MGR_MESSAGE_CONTROL section.
- If YES is coded, messages are not echoed to the Windows Event Log.

Note

This setting affects only the event logging of Novadigm log messages. Some Windows Event log records are written without any corresponding Novadigm log messages.

DISABLE_SNMP_TRAP_LOGGING = YES|NO

Note

Make sure this is set to **NO**.

If set to **YES**, the traps that are specified in the MGR_MESSAGE_CONTROL section will not be issued.

- If NO is coded, messages that are sent to the Radia log *will be sent* to the primary SNMP Manager as traps also, provided they have been selected for SNMP logging in the MGR_MESSAGE_CONTROL section of the edmprof file.
- If YES is coded, logging out as SNMP traps is disabled, that is, messages *are not sent* to the SNMP Manager as traps.

Note

This setting affects only the trapping of Novadigm log messages. Some SNMP traps are issued without any corresponding Novadigm log messages.

MGR_MESSAGE_CONTROL

This section of the edmprof specifies (by message number) which messages are to be sent, and to where. The message numbers that are recommended for the Radia Adapter for HP OpenView Operations are given in the example that follows.

```
[MGR_MESSAGE_CONTROL]
57-59,74,429,471,474 = LOG,SNMPTRAP
1019,1020,8107,8304 = LOG,SNMPTRAP
8308 = LOG,SNMPTRAP
```

Note

Messages that are destined for the same location can be expressed on one line. In the example above, the message numbers are specified on three lines, as this improves their readability in the RCS log.

This causes the above noted messages to be sent to the Radia log and as SNMP traps.

Note

For customized message-control processing, use this section to add environment-specific message numbers.

Figure 1.7, on page 35, shows a conceptual view of **LOG** and **SNMPTRAP** being specified as destinations for message **8107**.

```
[MGR_MESSAGE_CONTROL]
8107 = LOG,SNMPTRAP
```

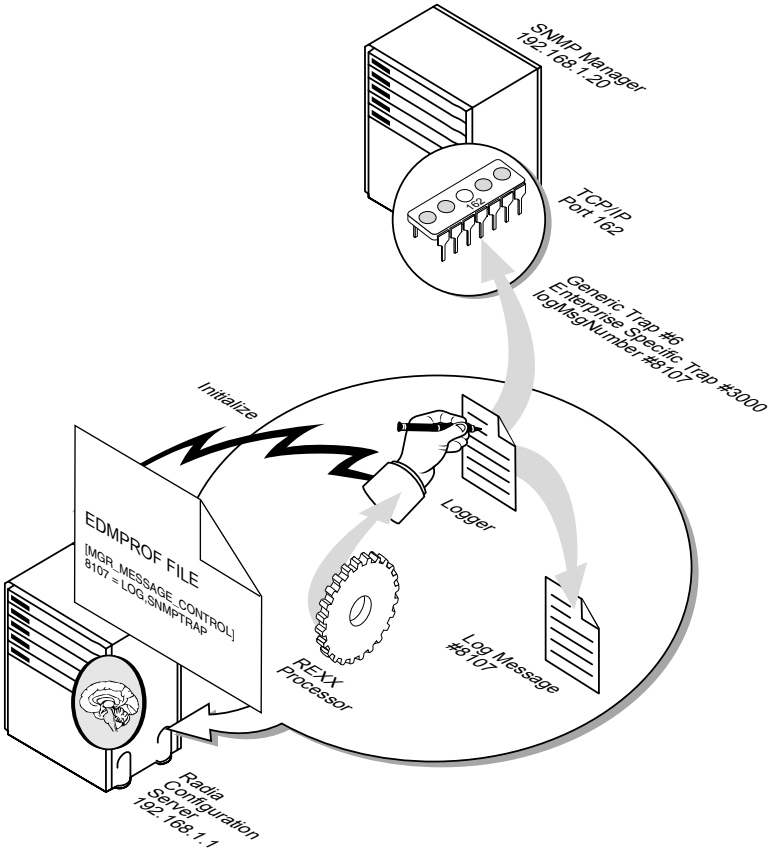


Figure 1.7 ~ LOG and SNMPTRAP specified as the destinations for message number 8107.

MGR_SNMP

SNMP-related parameters are contained in several sections of the edmprof file, but this section is the primary SNMP-configuration area of the RCS.

RUN_AS_EXTENSION = YES|NO

- If YES is coded, the Windows SNMP service is the primary SNMP agent and Radia Database SNMP transactions are processed by a Radia Database SNMP extension DLL. In this case, the SNMP_COMMUNITY, SNMP_PORT, and SNMP_MGR_IP_ADDR settings found in the edmprof file are not used, since SNMP port access is handled by the Windows SNMP service.

- If NO is coded, the Novadigm Agent will act as the primary SNMP agent. In this case, the SNMP_COMMUNITY and SNMP_MGR_IP_ADDR parameters should be specified, while SNMP_PORT can be specified (to override its default value).

SNMP_COMMUNITY

- This parameter should be set to a character string. The Novadigm agent will use this string as the SNMP community name. The community name is effectively a password which incoming SNMP transactions must match. The default is **public**.

  This setting is effective only when RUN_AS_EXTENSION=NO.

SNMP_SET_COMMUNITY

- This parameter can be set to a character string. The Novadigm agent will use this string as the SNMP community name when it is attempting to authorize *set* commands. If this is not specified, the community name given by SNMP_COMMUNITY is used for *set* commands.

  This setting is effective only when RUN_AS_EXTENSION=NO.

SNMP_PORT

- Specifies the TCP/IP port on which the Novadigm agent receives SNMP transactions. The default is port **161**.

  This setting is effective only when RUN_AS_EXTENSION=NO.

SNMP_IP_ADDR

This parameter should not be used.

SNMP_MANAGER_IP_ADDR, _ADDR2, and _ADDR3

- The SNMP Managers at these IP addresses are authorized to issue *get* and *set* commands for variables supported by the Novadigm agent. The first parameter, SNMP_MANAGER_IP_ADDR, is the *primary* SNMP Manager address (that to which the RCS will send all SNMP traps).
  - If neither of these parameters is specified, any SNMP Manager with the correct community name is authorized to run SNMP *get* and *set* commands on the Novadigm agent, but the Radia Database will not issue any traps.
  - If any of these three parameters is specified, commands coming from only those IP addresses will be processed and the first IP address will receive traps.

    We recommend the following settings for the Radia Adapter for HP OVO.

    ```
    SNMP_MANAGER_IP_ADDR   = xxx.xxx.xxx.xxx
    SNMP_MANAGER_IP_ADDR2  = 127.0.0.1
    SNMP_MANAGER_IP_ADDR3  = yyy.yyy.yyy.yyy
    ```

  - It is important that *xxx.xxx.xxx.xxx* be set to the real TCP/IP address of the local host since it will also be the address of the local OVO trap interceptor.

    All SNMP traps sent by the Radia Database will be sent to this address because it is the *primary* SNMP Manager address.

Note

Specifying **localhost** or **127.0.0.1** for *xxx.xxx.xxx.xxx* does not always work, because some implementations of TCP drop traps that are sent to 127.0.0.1.

- Some Radia Adapter for HP OpenView Operations monitors send SNMP *get* commands to 127.0.0.1. The second line authorizes these commands.
- The third line is optional. It can be used to set up an additional IP address from which SNMP *get* and *set* commands will be accepted.

Note

If the SNMP_MANAGER_IP_ADDR parameter is not specified, or if it is set to 0.0.0.0, traps *will not* be issued by the Radia Database.

Figure 1.8, on page 38, presents a conceptual view of an SNMP Manager with an IP address of **192.168.1.20** being specified as the destination for SNMP traps, as in:

```
[MGR_SNMP]
SNMP_MANAGER_IP_ADDR = 192.168.1.20
```

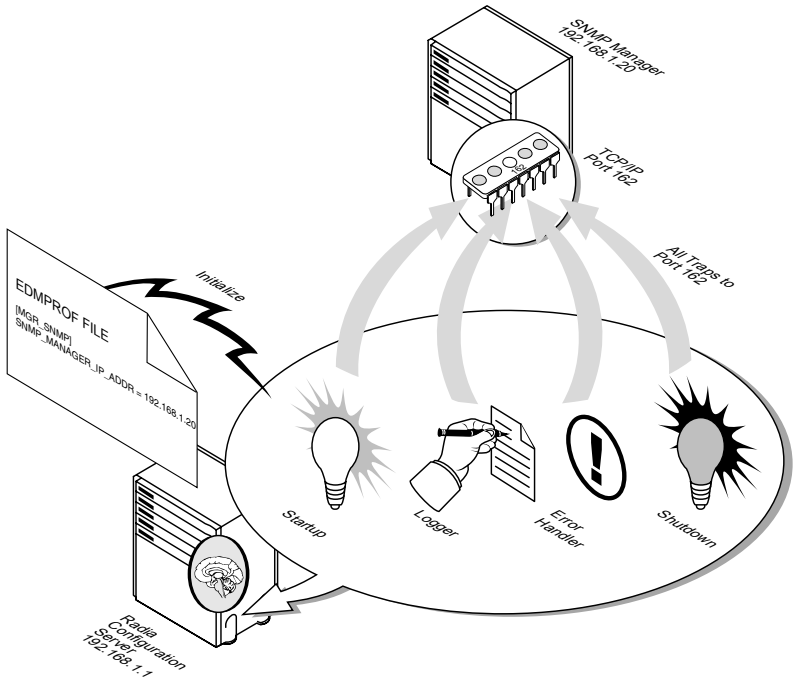


Figure 1.8 ~ The SNMP Manager, at 192.168.1.20, as the trap destination (as specified in SNMP_MANAGER_IP_ADDRESS).

SNMP_MANAGER_PORT

- Specifies the remote TCP/IP port to which the Radia Database sends its traps. The default is port **162**.

SNMP_ZERROR_SEVERITY

- Specifies the severity of ZERROR instances that are to be sent as SNMP traps. The trap is sent when the Radia Database adds an error instance to its ZERRORM for an error whose severity is greater than, or equal to, the value specified for this setting. The setting can be a positive value between 0 and 99. The default is **12**. Set this to 4 or 8 to cause the Radia Database to send traps associated with its resolution failures.

Populating the NVD Node Groups

The Radia Adapter for HP OpenView Operations installation supplies three Novadigm *node groups*, one each for the three Radia components (RCS, Radia Integration Server, and Radia Client).

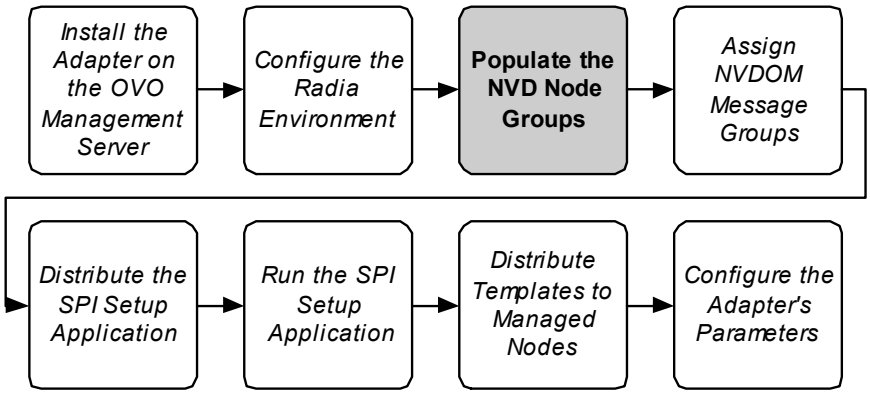


Figure 1.9 ~ Step 3 - Populate the NVD Node Groups.

The Novadigm node groups are:

- **NVD RCS**
  for nodes running an RCS.
- **NVD RIS**
  for nodes running a Radia Integration Server.
- **NVD Radia Clients**
  for nodes running a Radia Client.

Figure 1.10 on page 40 shows the three Novadigm (**NVD**) node groups in the **Node Group Bank**.

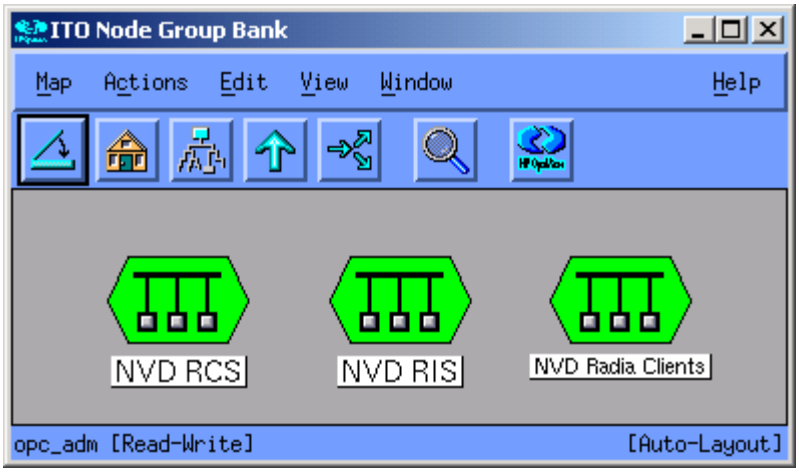


Figure 1.10 ~ The three Novadigm node groups in the Node Group Bank window.

Using Node Groups for Distribution

The SPI setup applications and NVDOM templates will have to be distributed to the OVO agents. However, this process can be simplified if the OVO agents are associated with *groups*. Having done this, the applications and templates can be distributed to the necessary groups, rather than multiple distributions to the various nodes.

The following section, *Populating a Node Group*, shows how to populate a node group with OVO agents.

Note

A node can exist in more than one node group.

Double-click a group's icon to see the nodes associated with it.

Populating a Node Group

To populate a group with nodes

1. Open the **Node Group Bank** and **Node Bank** windows.
2. To populate a Node Group, drag-and-drop the selected nodes from the respective Node Banks.

For example, in the **Node Bank** shown in Figure 1.11, the node, **hpk200**, can be added to the node group, **NVD RIS**, by simply dragging it onto that icon.

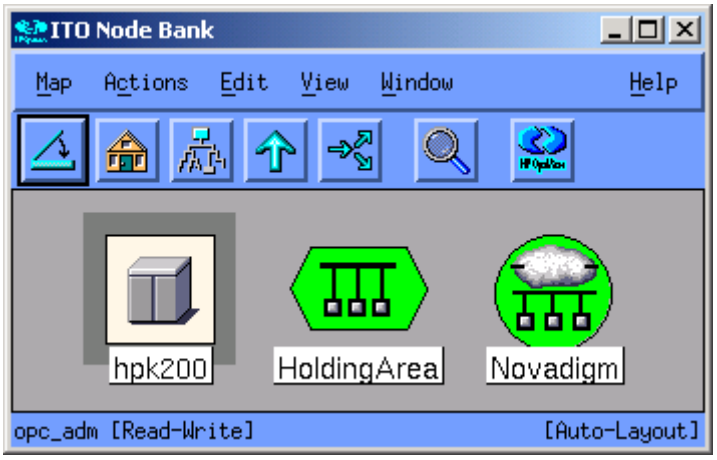


Figure 1.11 ~ The Node Bank window.

Note

The NVD node groups can be copied to create platform-specific node groups.

Depending on the user name under which the applications will run, we recommend separating the UNIX and Windows nodes.

Assigning NVDOM Message Groups

This section provides an overview of how to assign the NVDOM message groups. NVDOM messages will not show up on the message browser until this is done.

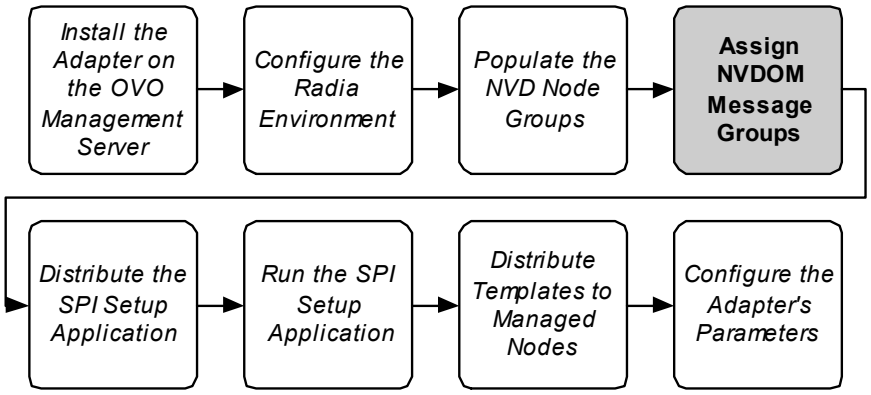


Figure 1.12 ~ Step 4 - Assign NVDOM Message Groups.

To assign NVDOM message groups

1. Click **Window** and **User Bank**.The **User Bank** window opens.

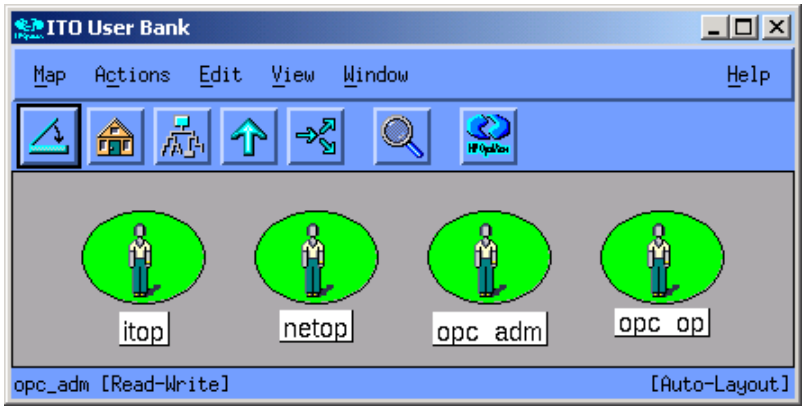


Figure 1.13 ~ The User Bank window.

2. Right-click the user and select **Modify**. For this example, the **opc adm** user was selected.

   The **Modify User** window opens.

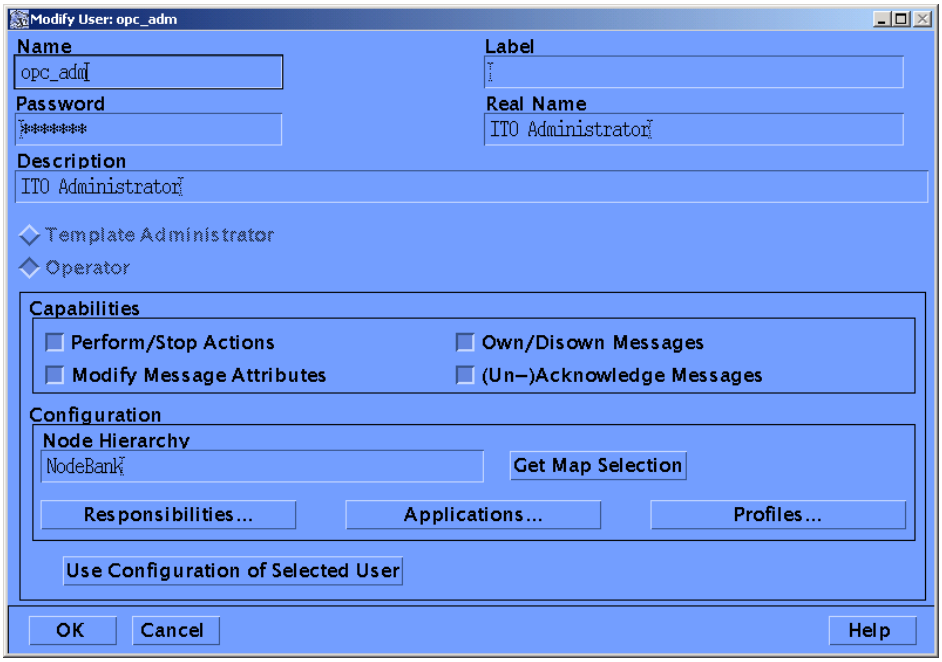


Figure 1.14 ~ The Modify User window (with the selected user in the title).

3. In the lower-right section of the window, click **Profiles**.

   The **Profiles of User: opc_adm** window opens.

4. On the **User Bank** window, click **Window** and **User Profile Bank**.

   The **User Profile Bank** window is now open, along with the **Profiles of User: opc_adm** window. (The **User Profile Bank** window supplies the **NVD SPI Admin (Super Admin)** profile.)

5. Drag-and-drop the **NVD SPI Admin (Super Admin)** profile into the **Profiles of User: opc_adm** window.

6. Return to the **Modify User** window and click **OK**.

7. Restart the OPC session.

   The NVDOM message groups are now assigned to your message browser screen.

Installing the Radia Adapter for HP OpenView Operations on Managed Nodes

Distributing the SPI Setup Application to Managed Nodes

Prior to running the SPI Setup application (**Install Unix SPI** or **Install Win SPI**), it must be distributed to the managed nodes, as described in this section.

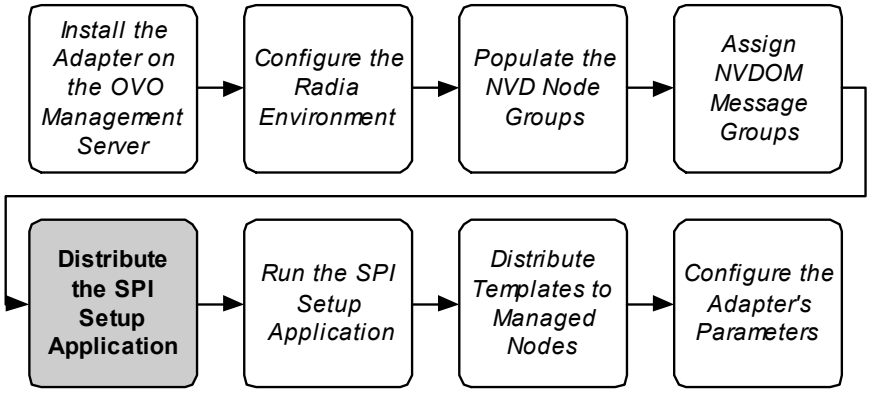


Figure 1.15 ~ Step 5 - Distribute the SPI Setup Application to Managed Nodes.

Note

In order to see any messages that result when distributing the SPI setup applications, the user that is logged in to the message browser must have its responsibilities set so that the OPC message group is assigned to the nodes to which the setup applications are being distributed.

To distribute an SPI Setup application to OVO Agents

1. Click **Actions**, then **Agents**, and finally, **Install / Update SW & Config**.
   The **Install/Update ITO Software and Configuration** window opens.

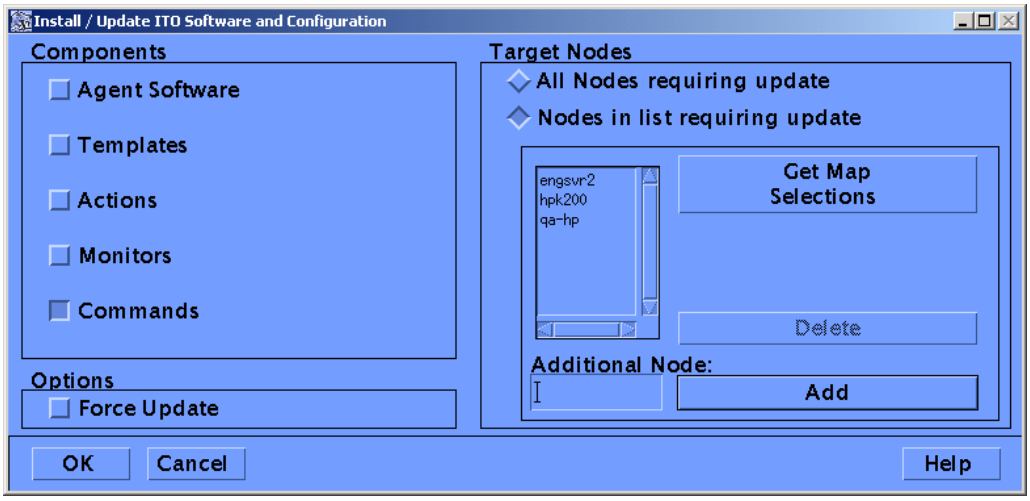


Figure 1.16 ~ The Install/Update ITO Software and Configuration window.

2. In the **Target Nodes** area, specify the machines that will be managed by OVO.
   Select the nodes on which the RCS and Radia Integration Server are running.

> **Note**
>
> If important Radia Client machines are going to be monitored, select their nodes here.

3. Under **Components**, select **Commands**.
4. Click **OK** to start the distribution.
   The SPI Setup application has been distributed to the nodes (agents) that will be managed by OVO.

Running the SPI Setup Application on Managed Nodes

This section details the installation of the Novadigm SPI on OVO agent machines. The installation uses the SPI Setup application that was distributed in the previous step.

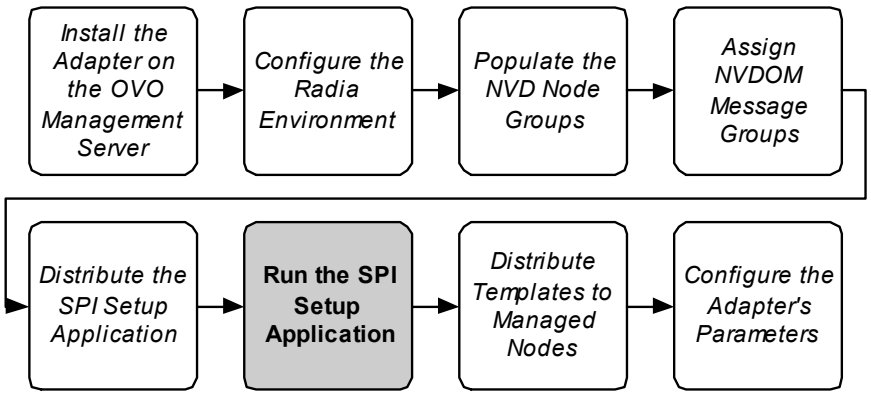


Figure 1.17 ~ Step 6 - Run the SPI Setup Application to Managed Nodes.

The installation applications in **NVD RCS SPI Setup**, **NVD RIS SPI Setup**, and **NVD CLI SPI Setup** are identical, but have been provided separately in order to facilitate establishing distinct authority for RCS, RIS, and Radia Client. Therefore, it doesn't matter from which group the SPI Setup command is run, as they do the same thing. All other NVDOM applications are very different from one another as regards processing.

Notes

Two platform-specific groups (**Install Unix SPI** and **Install Win SPI**) are provided. Be sure to run the appropriate **Install_*xxxx*_SPI** command for the operating system on which it is being installed.

To install the Novadigm SPI on OVO agents

1. In the **Application Bank**, select one of the **NVD *XXX* SPI Setup** groups.

   For this example, the **NVD RCS SPI Setup** (Figure 1.18) application was used.

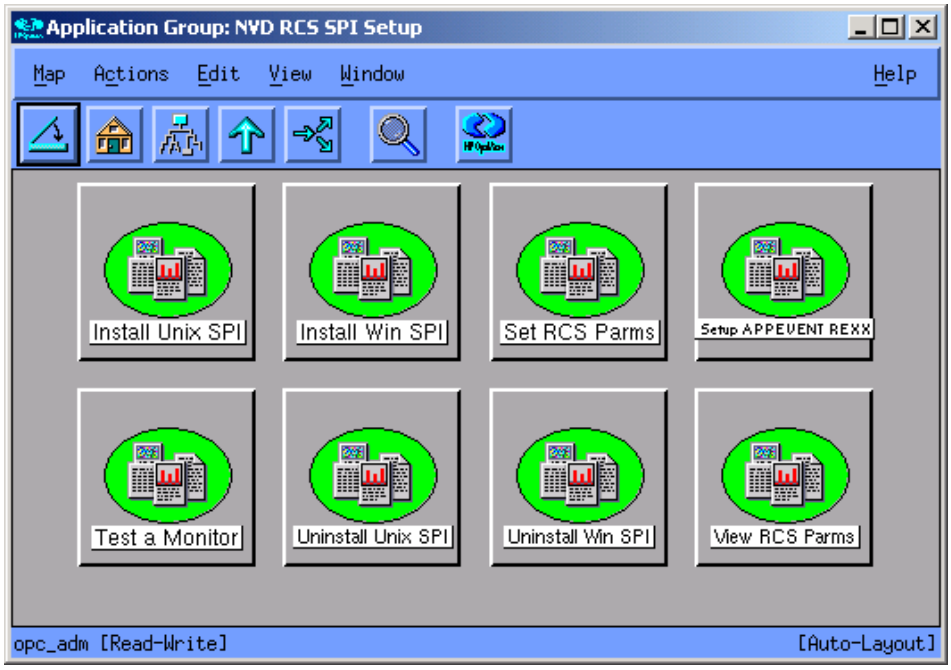


Figure 1.18 ~ The NVD RCS SPI Setup group.

2. In the **NVD RCS SPI Setup** application window, select either of the **Install *XXX* SPI** commands.

   For this example, the **Install Unix SPI** command (Figure 1.19) was used.

Figure 1.19 ~ The Install Unix SPI command.

3. In one of the node or group windows, select the managed nodes (or node groups) on which the installation will be run.
4. Right-click the icon of one of the applications, and select **Customized Startup**.

The **Customized Startup – ITO Application** window opens and the selected nodes are listed in the **Target Nodes** area.

The parameters contained in the template are placeholders only.

5. In the **Application Parameters** field (Figure 1.20) specify the four application parameters as follows:

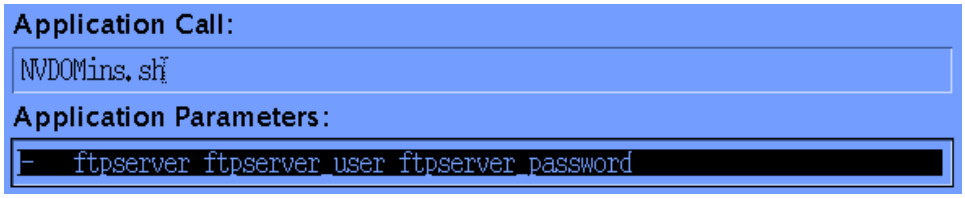


Figure 1.20 ~ The Application Parameters field.

a. The **directory path** to where the Novadigm SPI resides.

This path should be one that is outside the OVO tree so that it is not destroyed by the OVO **Install / Update SW & Config** agent action. The installation will place most of the SPI elements here.

Use a dash (**-**) as the first parameter in order to get the default path of **/var/novadigm** (UNIX) or **\Novadigm\ovo** (Win).

We recommend using the default.

b. The **host name** or **IP address** of the OVO server where the SPI has been uploaded.

> **Caution**
>
> It is assumed that the FTP server is running on the same machine as the OVO server. If not, copy the OVO server's **/var/opt/OV/share/databases/OpC/mgd_node/customer** directory and its contents to the FTP server (or provide the FTP server access to this directory). Otherwise, this application will not work.

c. A **user ID** that is valid for logging in to the FTP server on that system.

d. The **password** that is associated with the above user ID.

> **Note**
>
> The password will be visible on the screen. If this presents a security concern, we recommend using anonymous FTP to run this application.

Figure 1.21 ~ The Application Parameters field with User ID and password specified.

Note

The user ID and password are used to run an FTP client against that FTP server.

6. In the **Execute as User** area, change the **User Name** to the user name under which this Setup application will run.

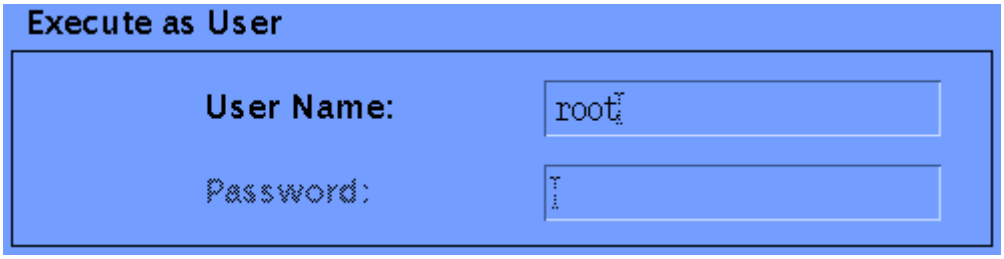


Figure 1.22 ~ The Execute as User area.

- UNIX SPI must be run under the *root user ID*.
- Win SPI must be run under a *user with administrator rights*.

7. Click **OK** to launch.

The **NVD RCS SPI Setup** for UNIX has been installed.

Distributing Templates to Managed Nodes

Distribute the NVDOM templates to the agents.

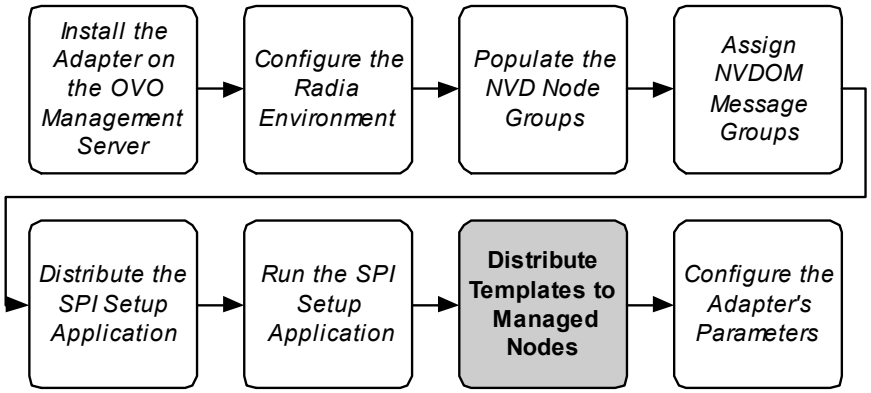


Figure 1.23 ~ Step 7 - Distribute Templates to Managed Nodes.

1. Click **Actions**, then **Agents**, and finally, **Install / Update Software & Config**.

   The **Install/Update ITO Software and Configuration** window opens.

2. Under **Components**, select **Templates**, and click **OK** to start the distribution.

   The *templates* should now begin to run on the agents.

   Monitors are not pre-assigned, and therefore, will not be distributed by this operation. Before assigning the Novadigm monitors, refer to *Chapter 6: Message Source Templates*, starting on page 125.

   This completes the configuration of the Radia Adapter for HP OpenView Operations.

Note

If your environment requires that permanent parameters be set at OVO agents, perform the steps that are outlined in the following chapter, *Radia Adapter for HP OpenView Operations Parameters*.

For example, if the RCS SNMP agent is not configured to use the standard port of **161**, the API Setup application must be used to permanently set the alternative port number.

2

Radia Adapter for HP OpenView Operations Parameters

This chapter details the configuration of the Radia Adapter for HP OpenView Operations applications, actions, and monitors, using permanent and immediate parameters.

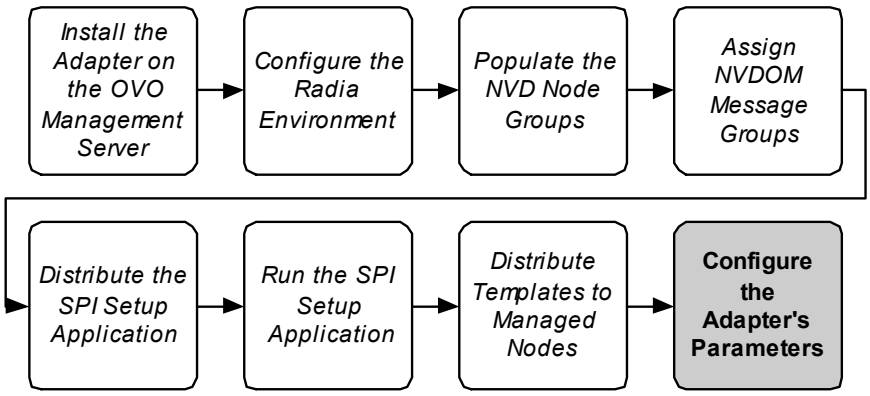


Figure 2.1 ~ Step 8 - Configure Radia Adapter for HP OpenView Operations Parameters.

Parameters can be passed to Radia Adapter for HP OpenView Operations applications, actions, and monitors in either of two ways. The method chosen to pass the parameters dictates the *type* of parameters passed—*immediate* or *permanent*.

- The parameters can be passed directly on the command line in the **Application Parameters** (see Figure 1.20 on page 48) area of the **Customized Startup** window. The parameters can also be passed by selecting **Applications** on the **Modify User** window (see **Error! Reference source not found.** on page 43).

  For actions and monitors that are configured in a template, the parameters can be placed directly on the command line following the program name.

  These parameters are **immediate** parameters, and are usually associated with an aspect of the current instance of the application, action, or monitor that is being run. For more information, see the section, *Immediate Parameters*, below.

- Some parameters are associated with only an aspect of a node, and that aspect might vary from one node to another. These parameters can be stored in a persistent configuration file (a *confile*) at the managed node, thereby, making the parameters applicable to the managed node on which they are saved.

  These parameters are **permanent** parameters. The **Setup *XXX* Parms** applications (where *XXX* is **RCS**, **RIS**, or **CLI**) are used to create these persistent configuration files. The permanent parameters currently supported by the Radia Adapter for HP OpenView Operations are discussed in the section, *Permanent Parameters*, starting on page 55.

Immediate Parameters

Using Immediate Parameters

Immediate parameters are coded directly on the command line for actions and monitors, and in the **Application Parameters** field when running applications.

- All parameters are coded as keyword–value pairs, as in, **-port 2121**.
- Keywords are preceded by a dash (**-**).
- If an odd number of parameters is passed (because a keyword lacks a value, or vice-versa), the application will fail and produce an error message.
- Some permanent parameters can be coded on the command line as immediate parameters. If this is done, keep in mind that the value specified on the command line will supersede the value specified in the confile for that parameter (permanent parameter).

> **Note**
>
> The permanent parameters that can be coded as immediate parameters are pointed out in Table 2.2, Table 2.3, and Table 2.4 in the section, *Using Permanent Parameters*, starting on page 58.

Table 2.1 lists and describes the immediate parameters (default values in **bold**) that are used most often by the Radia Adapter for HP OpenView Operations procedures. Not all of the immediate parameters are listed, only those which are common to several applications and used by them frequently.

The sections that detail each application and monitor will describe all the parameters they accept.

- Not all the parameters that are listed are applicable to *all* applications, actions, and monitors.
- If an immediate parameter is coded but is not applicable, it will be ignored—no error message will be generated.

Table 2.1 ~ Set RCS/RIS/Radia Client Parms Application Parameters

Parameter	Value	Description
-debug	**0**	Setting this to anything other than **0** has the effect of setting **-trace_level** to **99**. With the defaults for the other **-trace_*xxx*** parameters, this causes a verbose output to be written to the application's output screen. All Tcl log messages will be shown. **Note**: Setting this to **100** will have the above effect, and outputs the version numbers of the nvdkit program and the NVDOM module in use by the adapter.
-dumpenv	**0**/1	Setting this parameter to **1** causes a list of all the current environment variables to be written to the **-debug** log. This parameter is only effective if **-trace_level** is **4** or higher.
-trace_level	**0**/1/2/3/4/ >4	The Tcl code uses the following levels of messages: • **0** - for errors, and other messages that should always be output. • **1** - for warnings and audit failures. • **2** - for audit successes. • **3** - for information level messages. • **4** (and higher) - for debug level messages. **Note**: Messages will be selected if at a level equal to or less than the value specified. Unselected messages are simply ignored.
-trace_stderr	0/**1**	If **1**, all Tcl log messages that are selected will be written to **stderr**. For HP OVO, this means they will show up in the application's output screen.
-trace_eventlog	**0**/1	If **1**, all Tcl log messages that are selected will be written to the system's application event log. (Windows NT and 2000 only)
-trace_dir	*dir_name*	If specified, the name of a directory where selected log message files will be written. Each application will generate its own, self-named, **.log** file. **Note**: The application always concatenates to these files, so they will grow unchecked unless the operator cleans them up occasionally.

Table 2.1 ~ Set RCS/RIS/Radia Client Parms Application Parameters

Parameter	Value	Description
-ztype	RCS/RIS	Determines if an RCS or RIS application is being run. This affects which *confile* the application will use to load any preset parameters. It can also affect which files the application works on. For instance, the application to archive the log will archive RCS/RIS logs depending on this setting. **Note**: This parameter rarely needs to be specified, as it is already in the template that is used to run the application.
-zpath	*dir_name*	This parameter allows the operator to specify a temporary, override directory for **-zpath** (the directory where the *persistent* files are kept, such as the confile, used by the SPI). This can be useful when testing new applications, a new nvdkit, and common routines, as well as when using a temporary copy of a confile. • All Tcl scripts are also stored in **-zpath**.
-home	*home_directory*	The home directory for the RCS. This is used when an application, action, or monitor is being run under a user name other than that which installed the RCS. The edmprof file should be located in this directory. (UNIX only) **Note**: -homeDir can be used to specify this parameter in the confile permanently.
-lines	*nnn*	The maximum number of lines of a file to which an application will limit itself. For example, to limit the number of lines that will be shown on the screen to 100, pass this parameter, with a value of 100, to the appropriate log file-viewing application. **Notes**: This can be used with the immediate parameter, **-re**, to display a number of lines based on a regular expression. For example: `-lines 5 -re "snmp"` will show the last *five* lines in the file that contain the string, *snmp*. If this parameter is used with either **-last** or **-first**, its value will be ignored.
-last	*nnn*	The last lines (of a file) that are to be displayed. For example, to view the final 25 lines of a file, pass this parameter, with a value of 25, to the appropriate log file-viewing application. This works as an alias of **-lines**. **Notes**: This can be used with the immediate parameter, **-re**, to display a number of lines based on a regular expression. For example: `-last 5 -re "snmp"` will show the last *five* lines in the file that contains the string, *snmp*. If this parameter is used with **-lines** or **-first**, its value will be used.
-first	*nnn*	The first lines (of a file) that are to be displayed. For example, to view the first 25 lines of a file, pass this parameter, with a value of 25, to the appropriate log file-viewing application. **Notes**: This can be used with the immediate parameter, **-re**, to display a number of lines based on a regular expression. For example: `-first 5 -re "snmp"` will show the first *five* lines in the file that contains the string, *snmp*. If this parameter is used with **-last**, its value will be ignored. If this parameter is used with **-lines**, its value will be used.

Table 2.1 ~ Set RCS/RIS/Radia Client Parms Application Parameters

Parameter	Value	Description
-re	regexp	A regular expression that will be used to filter the lines on which an application takes action. For example, in the case of the **View Log File** applications, only the lines that match the regular expression will be shown. The regular expression format is the one used by Tcl. The regular expressions used by Tcl are documented in http://dev.scriptics.com/man/tcl8.4/TclCmd/re_syntax.htm#M3. This parameter can be used with **-lines**, **-first**, and **-last**. This parameter has no default.
-delete	**0**/1	If set to 1, the file on which the application is working will be deleted at the beginning of the application's processing. For example, use with the **Set RCS/RIS Parms** application to delete any previously existing confiles.
-logtype	C/**R**	For an application dealing with RIS log files (only), specify **R** to work on *regular* RIS log files, or **C** to work on *connection* RIS log files.
-logno	*current_log*	Specify the RIS log number on which an application should work. If regular RIS logs have rolled over, they are assigned a specific number. This parameter allows the operator to specify a specific RIS log.
-useRegistry	**0**/1	If set to **1**, Radia Client applications and monitors running under Windows will use the registry to find **IDMLIB**, where they can find values they might need for the **notifyPort**, **RCSipaddr**, **clientPort**, and **user** parameters. Setting this parameter to **1** for frequently run monitors might affect performance.

Permanent Parameters

Setting and Saving Permanent Parameters

To set and save permanent parameters on a managed node, run the associated **Set *XXX* Parms** application on the node.

- The **Set RCS Parms**, found in the **NVD RCS SPI Setup** application group, will set only RCS-related parameters on a managed node.
- The **Set RIS Parms**, found in the **NVD RIS SPI Setup** application group, will set only RIS-related parameters on a managed node.
- The **Set Radia Client Parms**, found in the **NVD CLI SPI Setup** application group, will set only Radia Client-related parameters on a managed node.

The permanent parameters that are in effect on a managed node can be viewed by running the **View *XXX* Parms** application on that node. These *view-parms* applications are contained in the

same folders as their corresponding *set-parms* applications. The **View *XXX* Parms** applications show the default parameter values that are in effect on a node, as well as the permanently set values.

The Set *XXX* Parms Applications

The permanent parameters that you want to set on a managed node are passed to the **Set *XXX* Parms** application via its command line. These *set-parms* applications will also take any immediate parameters passed to them (provided the parameter is not one that the application uses – that is, one specified in Table 2.1) and will save them, for future use, in the appropriate persistent parameter file (confile).

The following points are additional items to be considered when using the **Set *XXX* Parms** applications.

- A **Set *XXX* Parms** application cannot be run on a managed node until its associated SPI has been set up on that managed node. Once this is done, the application can be run anytime to add, change, and delete parameters.
- The (immediate) parameters that are listed in Table 2.1 will be used by the three **Set *XXX* Parms** applications. Therefore, these parameters cannot be permanently stored at a managed node.
- To use an immediate parameter when running an application, it has to be coded on the application before the application is run. Alternatively, the *application template* can be modified (by a user with sufficient rights) and the parameter can be saved therein.
- The **RCS_config.tcl**, **RIS_config.tcl**, and **CLI_config.tcl** confiles are built by these applications, and are stored in the **-zpath** directory that is created by the SPI setup application. If an ***XXX*_config.tcl** file is deleted, re-run the appropriate **Set *XXX* Parms** application to re-create the lost permanent parameters.

  For more information on specifying parameters and values in confiles, see *Running the Set XXX Parms Applications*, below.

Running the Set *XXX* Parms Applications

To run a **Set *XXX* Parms** application:

- Right-click it, and select **Customized Startup**
- In the **Application Parameters** field, enter the parameters that are to be set in the confile.

Warning

These applications create a confile (in -zpath on the node on which they run) that might affect the behavior of future Radia Adapter for HP OpenView Operations applications, actions, and monitors that run on that node.

These are Tcl source files, so they should not be directly edited, unless the editing administrator is proficient with Tcl.

- The only parameters that will be saved in the confile are those that are entered on the command line, but are *not* listed in Table 2.1, on page 53.

  For example:
  `-xxxx value` (where *-xxxx* is any keyword not listed in Table 2.1)

  will cause the keyword–value pair,
  `xxxx value`
  to be saved in the confile.

  Here, the parameters expressed as ***xxxx*** and ***-xxxx***, are actually the same parameter.
- The **-ztype** value determines which confile is being modified. Since it is already coded in the template under which the application is running, it does not have to be set by the operator. Therefore, if **Set RCS Parm** is run, **-ztype** will already be set to **RCS**, whereas **Set RIS Parm** presets **-ztype** to **RIS**.
- If **-delete** is coded as **1**, any existing confile for that **-ztype** will be deleted before the new parameters are added.

Examples of Application Parameters

Example 1:

```
-ztype RCS -delete 1 -snmpAgentPort 7901
```

This deletes any existing **RCS_config.tcl** and creates a new one, with the parameter **snmpAgentPort** set to **7901**. This parameter will be used by any application that has to send SNMP *get* or *set* commands to the Radia Configuration Server (RCS).

Example 2:

```
-ztype RCS –shutdownWaitTime 300 -snmpAgentTimeout 25
```

This adds to the RCS confile, the **shutdownWaitTime** and **snmpAgentTimeout** parameters, with values of **300** and **25** respectively.

Note

Case is significant when coding permanent parameter keywords. For example, **snmpAgentTimeout** is not the same as **snmpAgentTimeOut**.

Using Permanent Parameters

Any parameters can be put into a confile—except those that the **Set *XXX* Parms** application uses (as listed in Table 2.1). For example, the application could be run with the keyword–value pair

```
-localhost my_machine
```

Although this would result in **-localhost my_machine** being stored in the confile, it would be useless because currently there is no processing in the Novadigm SPI that uses a parameter called **-localhost**.

Important Note

The parameters shown in Table 2.2 through Table 2.4 are case-sensitive. Specify them exactly as shown in the tables.

The following sections describe the application parameters that are currently used by the Radia Adapter for HP OpenView Operations.

RCS_config.tcl Parameters

Table 2.2 presents the parameters of the **RCS_config.tcl** file, including their defaults and a description.

Table 2.2 ~ RCS_config.tcl File Parameters

Parameter (default)	Description
appName (Radia Configuration Server)	Specify the name that will appear in messages in the OVO message browser.
descrName (Radia Configuration Server)	Specify the name that will appear in log messages. These log messages appear in the output window when an application is run, if the trace level permits the message.
longServiceName (Radia Configuration Server)	Specify the name that is looked for, in the "net use" output, by the Radia Adapter for HP OpenView Operations, to verify that the service is running. (Windows only)
shortServiceName (ztoptask)	Specify the name that is used to start and stop the service. (Windows) Specify the program name that is used to start and stop the service; and the name that is looked for in the output of the **ps -ef** command, to verify that the service is running. (UNIX)

Table 2.2 ~ RCS_config.tcl File Parameters

Parameter (default)	Description
baseDir (" ")	Specify the base directory in which the RCS was installed. • On Windows, this can be determined by querying the service's **Properties**. • On UNIX, this base directory can be determined because it is assumed that the RCS is running under the Radia user, and that the edmprof file is in the home directory of that user. The edmprof file is then read and the baseDir is taken as one level up from the methods path.
homeDir (" ")	Specify the home directory of the user under which the RCS application is running. Currently, used only when determining the base directory and username. (UNIX only) An administrator can set it to be the Radia home directory if the Unix applications are to be run under a user other than the RCS user. Make sure that all the necessary permissions are set for that alternate user name.
username (" ")	The user name under which the RCS is started. When not specified in the confile, the Radia Adapter for HP OpenView Operations finds the edmprof file in the home directory and sets <*username*> to the owner of that file. (UNIX only)
pgmDir (" ")	Specify the directory from which the program, **shortServiceName**, is started. (UNIX only)
forceIPCclean (0)	Used when running RCS under root to force an IPC cleanup at shutdown. Should be set to 1 only if the RCS is the only application using IPC services on the computer. (UNIX only)
logPrefix ("nvdmr")	Prefix used in log name. The defaults are **"nvdmr"** (Windows) and **"nvd"** (UNIX).
logSuffix (".log")	Suffix used in log name. The defaults are **".log"** (Windows) and **"manager.log"** (UNIX).
dbMonPath (" ")	The path to the directory that will be checked for free disk space by the **NVDOM_RCS_DB_FreeSpace** monitor. These monitors usually calculate where the database in question resides. If this path is specified, the calculated value will be overridden.
logMonPath (" ")	The path to the directory that will be checked for free disk space by the **NVDOM_RCS_Log_FreeSpace** and **NVDOM_RCS_Log_FreeSpace_A** monitors. These monitors usually calculate where the log directory in question resides. If this path is specified, the calculated value will be overridden. **Note**: The **-path** immediate parameter can be used in the monitor template with the program call to override this confile parameter.
clientPort (" ")	The TCP/IP port that the **Start Client Connect** application uses in order to connect to the RCS. This parameter has no default because it is always specified in the **Application Parameters** field as an immediate parameter.
notifyPort (3465)	The TCP/IP port that is used by the **Start Client Connect** application to contact the Radia Client's notify facility. This parameter can be overridden with the **-notifyPort** parameter.

Table 2.2 ~ RCS_config.tcl File Parameters

Parameter (default)	Description
RCSipaddr (" ")	The TCP/IP address or name of the RCS used by the **Start Client Connect** application. This parameter can be overridden with the **-RCSipaddr** parameter.
tagFile (" ")	A file, maintained by the Novadigm SPI, which determines whether an RCS should be running. **Note**: For more information on tagFiles, refer to the section, *Tag Files*, on page 146.
shutdownWaitTime (240 seconds)	Specify how long (in seconds) shutdown applications should wait before issuing an error message (to the message browser), if the process or service has not shut down as expected.
snmpAgentHost (localhost)	Specify the IP address/host name of the machine to which SNMP commands will be sent. Since applications run on the same machine as the RCS, commands can be sent to **localhost** and the RCS will get them. **Note**: In the RCS edmprof file, the MGR_SNMP.SNMP_MANAGER_IP_ADDR2 should be set to 127.0.0.1 (localhost) so that it accepts the commands.
snmpAgentPort (161)	Specify the port on which the RCS SNMP agent will accept commands. This port number must match that which the RCS SNMP agent is using. **Note**: In order for the default port to work on a Windows system: ▪ MGR_SNMP.RUN_AS_EXTENSION must be YES, or ▪ if the Windows system is not running its own SNMP service on port 161, the RCS can.
snmpAgentTimeout (9 seconds)	Specify how long (in seconds) the Radia Adapter for HP OpenView Operations applications will wait for the completion of SNMP commands that they have sent to the RCS agent before returning in error.
snmpAgentCommunity (public)	Specify the community name that is used with commands that are sent, by the Novadigm SPI applications, to the RCS agent. **Note**: This name must match that which is specified for (**SNMP_COMMUNITY**) in the MGR_SNMP section of the RCS edmprof file, or if MGR_SNMP.RUN_AS_EXTENSION=YES it must match the community name specified in the SNMP service's settings.

RIS_config.tcl Parameters

Table 2.3 presents the parameters of the **RIS_config.tcl** file, including their defaults and a description.

Table 2.3 ~ RIS_config.tcl File Parameters

Parameter (default)	Description
appName (Radia RIS)	Specify the name that will appear in messages in the OVO message browser.
descrName (Radia Integration Server)	Specify the name that will appear in log messages. These log messages appear in the output window when an application is run, if the trace level permits the message.
longServiceName (Radia Integration Server)	Specify the name that is looked for, in the "net use" output, by the Radia Adapter for HP OpenView Operations, to verify that the service is running. (Windows only)
shortServiceName	Specify the name that is used to start and stop the service. The default value is **httpd**. (Windows) Specify the program name that is used to start and stop the service; and the name that is looked for in the output of the **ps -ef** command, to verify that the service is running. The default value is **./nvdkit httpd.tkd**. (UNIX)
baseDir (" ")	Specify the base directory in which the Radia Integration Server was installed. • On Windows, this can be determined by querying the service's **Properties**. • On UNIX, the name of this directory might need to be specified for RIS installations, if it is not version 3.0 and was not installed in the default locations (as a subdirectory of the RCS). • If the RIS is version 3.0 or greater, its directory is in a known location, `/opt/Novadigm/RadiaIntegrationServer`. • If the RIS is pre-version 3.0, then it is assumed that it has been installed in a subdirectory of the RCS base directory. If not, run the **Set RCS Parms** application in order to set the **baseDir** variable to its correct value, which is probably the **ris** subdirectory of the RCS base directory. The RCS base directory is wherever the RCS was installed.
homeDir (" ")	Used only when determining the base directory. (UNIX only) Specify the home directory of the user under which the Radia Integration Server application is running.
username (" ")	The user name under which the RIS is started. When not specified in the confile, the Radia Adapter for HP OpenView Operations finds the httpd.rc file in the base directory and sets *username* to the owner of that file. (UNIX only)
pgmDir (" ")	Specify the directory from which the program, **shortServiceName**, is started. (UNIX only)
logPrefix ("httpd")	Prefix used in log name.

Table 2.3 ~ RIS_config.tcl File Parameters

Parameter (default)	Description
logSuffix (".log")	Suffix used in log name.
dbMonPath (" ")	The path to the directory that will be checked for free disk space by the **NVDOM_RIS_DB_FreeSpace** monitor. These monitors usually calculate where the database in question resides. If this path is specified, the calculated value will be overridden. **Note**: The **RIS DB** monitor cannot calculate the directory for its database because it is an external database. So, this confile parameter is required for the **NVDOM_RIS_DB_FreeSpace** monitor, unless the **-path** immediate parameter is used in the monitor template with the program call.
logMonPath (" ")	The path to the directory that will be checked for free disk space by the **NVDOM_RIS_Log_FreeSpace** and **NVDOM_RIS_Log_FreeSpace_A** monitors. These monitors usually calculate where the log directory in question resides. If this path is specified, the calculated value will be overridden. **Note**: The **-path** immediate parameter can be used in the monitor template with the program call to override this confile parameter.
tagFile (" ")	A file, maintained by the Novadigm SPI, which determines whether a RIS should be running. **Note**: For more information on tagFiles, refer to the section, *Tag Files*, on page 146.
shutdownWaitTime (90)	Specify how long (in seconds) shutdown applications should wait before issuing an error message (to the message browser), if the process or service has not shut down as expected.
httpPort (3466)	Specify the port to which the **response** monitor (see *Tag Files*, on page 146) will send transactions. The response monitor determines if the RIS is alive, based on the response.

CLI_config.tcl Parameters

Table 2.4 presents the parameters of the **CLI_config.tcl** file, including their defaults and a description.

Table 2.4 ~ CLI_config.tcl File Parameters

Parameter (default)	Description
clientPort (3464)	The TCP/IP port that is used by the **NVDOM_Client_Ready** monitor and the **NVD Op CLI** applications to connect to the RCS. This parameter can be overridden with the **-clientPort** immediate parameter.

Table 2.4 ~ CLI_config.tcl File Parameters

Parameter (default)	Description
IDMLIB	The directory that contains the Novadigm client objects. The Windows default is **c:/Program Files/Novadigm/Lib** and the UNIX default is **/opt/Novadigm/lib**. The **NVDOM_Client_Ready** monitor and the **NVD Op CLI** applications use this when connecting to an RCS to determine the **notifyPort**, **RCSipaddr**, **clientPort**, and **user** parameters, if they are missing. This parameter can be overridden with the **-IDMLIB** parameter.
notifyPort (3465)	The TCP/IP port that is used by the **NVDOM_Client_Ready** and **NVDOM_Client_Daemon_Ready** monitors and the **NVD Op CLI** applications to contact the Radia Client's notify facility. This parameter can be overridden with the **-notifyPort** parameter.
RCSipaddr (" ")	The TCP/IP address or name of the RCS used by the **NVDOM_Client_Ready** monitor and the **NVD Op CLI** applications. This parameter can be overridden with the **-RCSipaddr** parameter.
user (" ")	The Novadigm user name that is used by the **NVDOM_Client_Ready** monitor and the **NVD Op CLI** applications when connecting to the RCS. This parameter can be overridden with the **-user** parameter on the monitor template.

3

Application Bank

Once the Radia Adapter for HP OpenView Operations is installed on the OVO Management Server, eight Novadigm *application groups*, with the **NVD** prefix, will be present in the **Application Bank**. To verify this, in any OVO window, click **Window**, and from the drop-down list, select **Application Bank**. The eight new application groups will be listed along with the established ones.

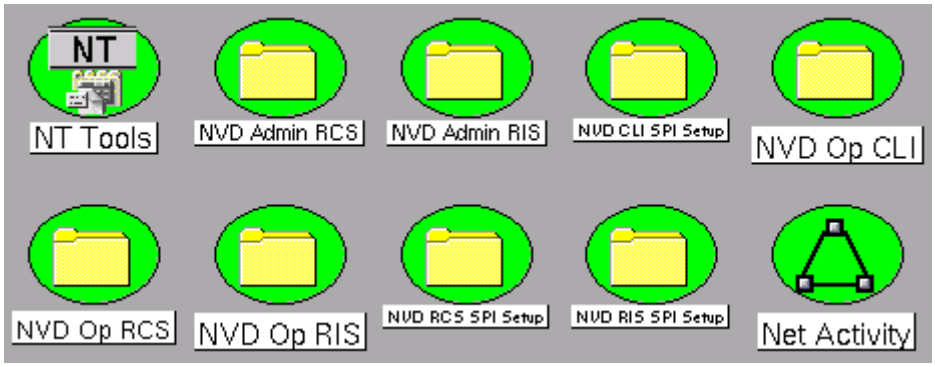


Figure 3.1 ~ The eight Novadigm application groups.

Configuring the Applications

UNIX-Managed Nodes

On UNIX systems, the applications need to be run under the Radia user ID. Therefore, before running an application against a UNIX-managed node, right-click on the application, and customize the startup by changing the user name and adding the password.

In an environment with UNIX and Windows-managed nodes, we recommend copying the Novadigm groups and making UNIX-specific ones that are configured with the correct user name and password.

Warning

Copying a group does not result in a *deep copy*. This means that, although a new group is created, the icons (copied into the new group) still refer to the applications in the original group. Therefore, if changes are made in the new group, they will also take effect in the original group.

To circumvent this, we recommend doing the following:

- Use right-click, and **Copy** to duplicate every icon in the group.
- Provide a new **Application Name** for each copy.
- Paste the icons into the new group.
- Finally, remove the duplicates from each group.

Note

On Solaris, this user name needs write permission to **/usr/tmp/tclkit**, so that **nvdkit** can create files in this directory. There are similar requirements for other UNIX platforms.

Windows-Managed Nodes

It is necessary to customize (add the correct user name and password) the startup of applications that will run on Windows systems. Note that the **opc_op** user name can be used to run all Novadigm applications, if it is made a member of the Administrator group. In which case, this customization is not required. However, this is not recommended as it gives **opc_op** too many rights.

NVD Application Groups

In the sections that follow, the Novadigm applications are separated by group, and the various functions within each group are detailed.

NVD Admin RCS

This group of applications is designed specifically for administrative functions that can be performed on a *Radia Configuration Server* (RCS). These functions can be used on Windows and UNIX operating systems. There are four applications in this group.

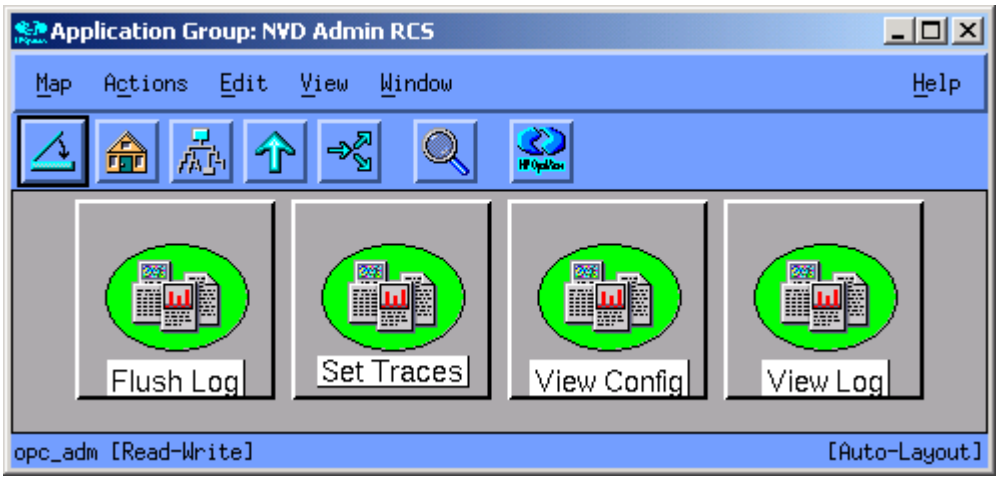


Figure 3.2 ~ The applications of the NVD Admin RCS group.

Flush Log

Function	Flushes the log buffers on the RCS. This causes any log messages that were in these buffers to be written out to the current RCS log file.
Notes	Uses SNMP command. (RCS SNMP agent must be running).
Tcl Script	NVDOM_flushRCSlog.tcl
Parameters	None
Example	N/A

Set Traces

Function	Dynamically turn on/off one or more RCS trace settings.
Notes	Uses SNMP command. (RCS SNMP agent must be running). Can have multiple, simultaneous (on/off) parameters, which are processed from left to right. To activate a trace, specify it after -**on**. A trace that is specified after **-off** is de-activated.
Tcl Script	NVDOM_setRCStrace.tcl
Parameters	-on, -off
Example	In the following example, the *dynamic allocation* (**allo**) and *storage map monitoring* (**stor**) tracing are turned off, after all other tracing has been activated. `-on all -off allo stor` • Notice that -**off allo stor** is specified after the -**on** function. If -**on all** was specified after them, they would be included in the tracing. • Also note that they are specified without a delimiter. Table 3.1 presents a list of the trace types that are valid for the **Set Traces** function. Some of the trace types have aliases (indicated in parentheses), but the functionality is identical.

Table 3.1 ~ Valid Trace Types

Trace Type	Description
admn (admin)	Administrator
all	activates all trace types
allo	dynamic allocation
audi (audit)	Audit
buff	3270 buffer
cmpr	compression/decompression
comm	communications level
commcbs	communications control blocks
commdata	communications data
conf	Configuration
cpic	Cpic
data	binary data flow
des	DES encryption/decryption
dma	DMA (DCS) synchronization
dsco (compr)	data-stream compression
expl	Explode
file	file processing
impl	Implode
look	lookaside
mspx	Microsoft SPX/IPX
mthd (method)	method
netb (netbios)	NetBIOS
nfy (ntfy)	notify flag
nqdq	1st call flag for ZADMPROM
obj0	object resolution object level 0
obj1	object resolution object level 1
objr	object resolution
ocrc	object CRC process
odbcbs	object database control blocks
odbdata	object database data
oxfr	object transfer
poolmiss (pools)	pool misses
prof (profile)	profile
prom (promote)	promote
reso (resolve)	resource

Table 3.1 ~ Valid Trace Types	
Trace Type	**Description**
rexx	REXX
sblk	API session block flag
sipx	Novell SPX/IPX
stat	statistics flag
stor (storage)	storage map monitoring
subs	substitution active
tcp (tcpip)	TCP/IP
test	test
tran	transform EBCDIC/ASCII/COMPRESSED
vars	variables
varx	variable storage
vsamapi	VSAM API
vsamcb	VSAM control blocks
vsamdata	VSAM data
y2k	Y2K support

View Config

Function	View the RSC settings file (the edmprof).
Notes	Uses SNMP command. (RCS SNMP agent must be running).
Tcl Script	NVDOM_viewRCSconfig.tcl
Parameters	-lines *nnn* (optional, shows last *n* lines of file) -first *nnn* (optional, shows first *n* lines of file) -last *nnn* (optional, shows last *n* lines of file) -re pat (optional, show only lines matching the regular expression pattern "pat". The regular expressions used by Tcl are documented in http://dev.scriptics.com/man/tcl8.4/TclCmd/re_syntax.htm#M3)
Example	`-last 20 -re CMD_LINE` This will show the last 20 lines of the edmprof that contain CMD_LINE.

View Log

Function	View the RCS log.
Notes	Uses SNMP command. (RCS SNMP agent must be running).
Tcl Script	NVDOM_viewRCSlog.tcl
Parameters	-lines *nnn* (optional, shows last *n* lines of log) -first *nnn* (optional, shows first *n* lines of file) -last *nnn* (optional, shows last *n* lines of file) -re pat (optional, show only lines matching the regular expression pattern "pat". The regular expressions used by Tcl are documented in http://dev.scriptics.com/man/tcl8.4/TclCmd/re_syntax.htm#M3) -logPrefix prefix (optional, the prefix used for RCS log names) -logSuffix suffix (optional, the suffix used for RCS log names)

Example	`-re NVD8119I`
	This shows the lines in the RCS log that contain the string NVD8119I.
	`-last 500 -re "(0100I)|(04..I)"`
	Shows the last 500 lines, in the log file, that contain the string "0100I" or "04..I" where each dot in "04..I" can be any character.

NVD Admin RIS

This group of applications is designed specifically for administrative functions that can be performed on a *Radia Integration Server*. These functions can be used on Windows and UNIX operating systems.

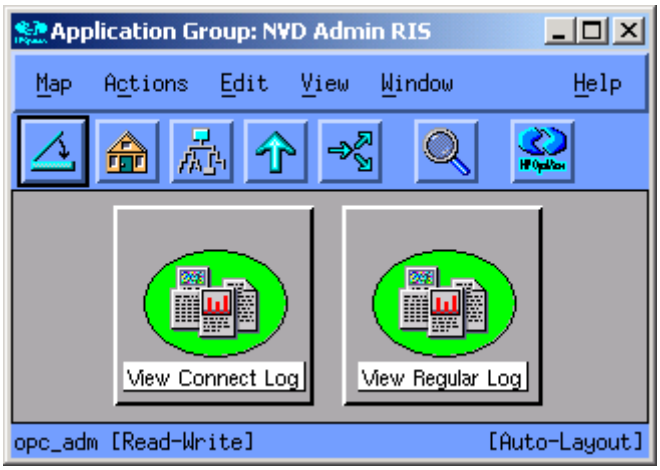


Figure 3.3 ~ The applications of the NVD Admin RIS group.

View Connect Log

Function	View the Radia Integration Server connect log.
Notes	For more information on this function and log rollover limit settings, refer to the section, *Understanding Radia Integration Server Logging*, starting on page 114.
Tcl Script	NVDOM_viewRISlog.tcl
Parameters	-lines *nnn* (optional, shows last *n* lines of log) -first *nnn* (optional, shows first *n* lines of file) -last *nnn* (optional, shows last *n* lines of file) -re pat (optional, show only lines matching the regular expression pattern "pat". The regular expressions used by Tcl are documented in http://dev.scriptics.com/man/tcl8.4/TclCmd/re_syntax.htm#M3) -yy year -mm month -dd day **Note**: The above three parameters are optional. They are used to specify the name of the log to view. These values default to the current day. If not specified, the *current* log is viewed. -logtype C/R (If *x* = C, the *connect* archive/log is acted on. If *x* = R, the *regular* archive/log is acted on. If *x* ≠ C or R, the *regular* archive/log is acted on. Note that -logtype = C is present in the **Application Call** field of this application, so by default, the *connect* log is processed.) -logPrefix prefix (optional, the prefix used for RIS log names) -logSuffix suffix (optional, the suffix used for RIS log names) -gmt 0/**1** This parameter is optional. It specifies whether the local time zone (**0**) or the GMT time zone (**1**) should be used in calculating the current date. By default, the RIS uses GMT (**1**) for its logs, unless it has been otherwise customized.
Example	`-dd 4 -re "2002:10"` Views lines in the connect log for the 4th day of the month. Only lines that contain the characters **2002:10** are shown—in this case, all the messages that were written from 10:00 to 10:59 on that day.

View Regular Log

Function	View the Radia Integration Server regular log.
Notes	For more information on this function and log rollover limit settings, refer to the section, *Understanding Radia Integration Server Logging*, starting on page 114.
Tcl Script	NVDOM_viewRISlog.tcl
Parameters	-lines *nnn* (optional, shows last *n* lines of log) -first *nnn* (optional, shows first *n* lines of file) -last *nnn* (optional, shows last *n* lines of file) -re pat (optional, show only lines matching the regular expression pattern "pat". The regular expressions used by Tcl are documented in http://dev.scriptics.com/man/tcl8.4/TclCmd/re_syntax.htm#M3) -logtype C/R (If *x* = C, the *connect* archive/log is acted on. If *x* = R, the *regular* archive/log is acted on. If *x* ≠ C or R, the *regular* archive/log is acted on. Note that -logtype is not present in the **Application Call** field of this application, so by default, the *regular* log is processed.) -logno n (optional, the number of the log that was created by roll over) -logPrefix prefix (optional, the prefix used for RIS log name) -logSuffix suffix (optional, the suffix used for RIS log name)
Example	`-last 5 -re "on port"` Shows the last 5 lines in the regular RIS log that contain the characters "on port".

NVD Op RCS

This group of applications is designed specifically for operational functions that can be performed on an RCS. These functions can be used on Windows and UNIX operating systems.

On UNIX platforms, all applications under this group run under the same user name as the RCS.

Figure 3.4 ~ The applications of the NVD Op RCS group.

Archive Log

Function	This application will take all currently un-archived logs (except the one that is currently being written by the RCS) and compress them into a bundle, which will then be stored in the *archive* subdirectory of the Radia Database current log directory. If the **NVDOM_RCS_Log_FreeSpace_A** monitor detects that there is a disk space shortage on the volume where the logs are being written, and that the free space is of a *warning* severity level, it will allow the operator to initiate this application.
Notes	**UNIX**: **tar** is used to bundle up the files and the tar file is then compressed. **Windows**: there is no standard bundle-and-compression utility so the -zip parameter is used.
Tcl Script	NVDOM_archiveRCSlog.tcl
Parameters	-logPrefix prefix (Optional. The prefix used for RCS log name.) -logSuffix suffix (Optional. The suffix used for RCS log names) -zip command (Not used under UNIX; optional under Windows. Here, **command** is the command that will be used to bundle and compress the log files that are being archived. The Windows default is **pkzip ../$tempDirName *.***.)
Example	`-zip "cmd.exe \"/c\" c:/u/utils/myarchive.cmd"` This causes the **myarchive.cmd** to be run once against all the files to be archived. The **** that is in front of the inner double quotes (" ") escapes them, so that the whole string that is after **-zip** is considered one parameter.

Delete Archive

Function	Deletes one or more RCS log archives. If the **NVDOM_RCS_Log_FreeSpace_A** monitor detects that there is a disk space shortage on the volume where the logs are being written, and that the free space is of a *minor* severity level, it will automatically allow the operator to initiate this application to delete all archived RCS logs.
Notes	
Tcl Script	NVDOM_deleteRCSarchive.tcl
Parameters	-all **0**/1 (Optional. If $x = 1$, all archives are deleted. If $x \neq 1$, the oldest archive is deleted. The default is **0**.)
Example	`-all 1`

Delete Log

Function	Delete one or more RCS logs. The log currently being written by the RCS is never deleted. If the **NVDOM_RCS_Log_FreeSpace_A** monitor detects that there is a disk space shortage on the volume where the logs are being written, and that the free space is of a *major* severity level, it will automatically allow the operator to initiate this application to delete all RCS logs.
Notes	
Tcl Script	NVDOM_deleteRCSlog.tcl
Parameters	-logPrefix prefix (Optional. The prefix used for RCS log name.) -logSuffix suffix (Optional. The suffix used for RCS log names) -all **0**/1 (Optional. If $x = 1$, all logs are deleted. If $x \neq 1$, the oldest log only is deleted. The default is **0**.)
Example	`-all 1` All logs, except the current one, are deleted.

Disable Logons

Function	Prevents clients from logging on to RCS.
Notes	Uses the SNMP *set* command. (The RCS SNMP agent must be running). Use **Enable Logons** to re-enable client logons.
Tcl Script	NVDOM_disableRCSlogon.tcl
Parameters	None
Example	N/A

Display Active

Function	Display active tasks running under the RCS.
Notes	Uses the SNMP *set* command. (The RCS SNMP agent must be running).
Tcl Script	NVDOM_displayRCSactive.tcl
Parameters	-logPrefix prefix (Optional. The prefix used for RCS log names.) -logSuffix suffix (Optional. The suffix used for RCS log names.)
Example	N/A

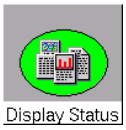
Display Pool Misses

Function	Display RCS pool misses.
Notes	Useful for tuning the MGR_POOLS section of the edmprof file. Uses the SNMP *set* command. (The RCS SNMP agent must be running). Not available under UNIX.
Tcl Script	NVDOM_displayRCSpoolmiss.tcl
Parameters	-logPrefix prefix (Optional. The prefix used for RCS log names.) -logSuffix suffix (Optional. The suffix used for RCS log names.)
Example	N/A

Display Pool Stats

Function	Display the status of RCS internal pools. **Note**: This is **STATUS** – not STATISTICS.
Notes	Uses the SNMP *set* command. (The RCS SNMP agent must be running).
Tcl Script	NVDOM_displayRCSpoolstat.tcl
Parameters	-logPrefix prefix (Optional. The prefix used for RCS log names.) -logSuffix suffix (Optional. The suffix used for RCS log names.)
Example	N/A

Display Status



Function	Display the status of the RCS.
Notes	Uses the SNMP *set* command. (The RCS SNMP agent must be running). Not available under UNIX.
Tcl Script	NVDOM_displayRCSstatus.tcl
Parameters	-logPrefix prefix (Optional. The prefix used for RCS log names.) -logSuffix suffix (Optional. The suffix used for RCS log names.)
Example	N/A

Enable Logons

Function	Allows clients to log on to the RCS.
Notes	Uses the SNMP *set* command. (The RCS SNMP agent must be running). Required to re-enable logons after running **Disable Logons**.
Tcl Script	NVDOM_enableRCSlogon.tcl
Parameters	None
Example	N/A

Flush Log

Function	Flushes the log buffers on the RCS. This causes any log messages that were in these buffers to be written out to the current RCS log file.
Notes	Uses the SNMP *set* command. (The RCS SNMP agent must be running).
Tcl Script	NVDOM_flushRCSlog.tcl
Parameters	None
Example	N/A

Refresh Config

Function	Refreshes the MGR_TRACE, MGR_MESSAGE_CONTROL, and MGR_POOLS sections of the RCS settings file (the edmprof).
Notes	Uses the SNMP *set* command. (The RCS SNMP agent must be running). Before running this application, Radia can be used to distribute an updated edmprof file with new settings.
Tcl Script	NVDOM_refreshRCSconfig.tcl
Parameters	None
Example	N/A

Run REXX

Function	Run REXX program under RCS control.
Notes	Uses the SNMP *set* command. (The RCS SNMP agent must be running). **View_Log** must be used to view the results of the command.
Tcl Script	NVDOM_runRCSrexx.tcl
Parameters	-cmd "command"
Example	`-cmd "MYREXX,parm1 parm2 parm3"` This runs the **MYREXX** command found in the RCS REXX_PATH directory. The parameters **parm1**, **parm2**, and **parm3** are passed to the command.

Set Traces

Function	Dynamically turn on/off one or more RCS trace settings.
Notes	Uses SNMP command. (RCS SNMP agent must be running). Can have multiple, simultaneous (on/off) parameters, which are processed from left to right. To activate a trace, specify it after **–on**. A trace that is specified after **–off** is de-activated.
Tcl Script	NVDOM_setRCStrace.tcl
Parameters	-on, -off
Example	In the following example, *dynamic allocation* (**allo**) and *storage map monitoring* (**stor**) are turned off, but all other tracing is active. `-on all -off allo stor` • Notice that **–off allo stor** is specified after the **–on** function. If **–on all** was specified after them, they would be included in the tracing. • Also, note that they are specified without a delimiter. Table 3.1 on page 69 presents a list of the trace types that are valid for the **Set Traces** function. Some of the trace types have aliases (indicated in parentheses), but the functionality is identical.

Start Client Connect

Function	Connect a Radia Client to this RCS.
Notes	• A Radia Notify will be sent to the specified Radia Client, requesting that it connect to the RCS that is running this application. This application can be used when the Radia Client is not a managed node. • Contrast this to the three *connect* applications in the **NVD Op CLI** folder, which run on the Radia Client and connect it to its regular RCS. • When initiating a *full connect* at the Radia Client, the **radskman** program is run. • When initiating a *refresh-catalog* operation, the **radpinit** program is run. **Note**: See Radia documentation on these programs for further details about the parameters that are described in the **Parameters** section. Usually, only the **-CLIipaddr**, **-clientPort**, and **-user** parameters need to be specified.
Tcl Script	NVDOM_connectClient.tcl
Parameters	This application uses the following parameters: -notifyPort — The port at the Radia Client on which the Radia notify daemon is contacted. -user — The user name for which the Client Connect will be done. -CLIipaddr — The name or IP address of the Radia Client. -clientPort — On the RCS computer, the port that the Radia Client contacts. -RCSipaddr — The name or IP address of the RCS. -mname — The *manager name* parameter that is used on the **radskman** call initiated on the client computer. The default is "RADIA". -dname — The *domain name* parameter that is used on the **radskman** call initiated on the client computer. The default is "SOFTWARE". -cat — The *catalog* option that is used on the **radskman** call initiated on the client computer. Valid values are **prompt**, **y**, and **n**. The default is **prompt**. -ulogon — The *user logon panel* option that is used on the **radskman** call initiated on the client computer. Valid values are **y** (show user logon panel), and **n** (suppress user logon panel). The default is **n**. -ind — The *progress indicator* option that is used on the **radskman** call that is initiated on the client computer. Valid values are **y** (show progress indicator), and **n** (suppress progress indicator). The default is **n**. -refreshCat — If set to **1**, a *First Refresh Catalog* is run at the Radia Client. This refreshes the catalog, runs Radia self-maintenance, and updates ZMASTER.ZUSERID. If set to **2**, a *Refresh Catalog* is run at the Radia Client. This refreshes the catalog, but does not run Radia self-maintenance or update ZMASTER.ZUSERID. If this parameter is not set, or if it is set to something other than **1** or **2**, a *full connect* is performed. This is the default.
Example	-user JohnDoe -CLIipaddr 192.168.64.12 -clientPort 3464

Start RCS

Function	Starts the RCS.
Notes	Under Windows, the service interface is used. Under UNIX, an **su** command will be used to switch to the Radia user name if necessary. If the user name is not found as a parameter (immediate or from the confile), the owning user of the edmprof file is used. We recommend that the Radia user name not be root. That is, the RCS should not be installed by the root user. If it is, shared resources cannot be cleaned up.
Tcl Script	NVDOM_startRCS.tcl
Parameters	This application uses the following parameters: -homeDir -username -appName -shortServiceName -pgmDir -descrName -tagFile These parameters, which are described in Table 2.2 on page 58, rarely have to be supplied. Under Windows, the program can always determine the correct parameter values. Under UNIX, if this application is not run under the Radia user name, the **homeDir** should be specified as an immediate parameter, or in the confile. The remaining parameters can then also be determined. **Note**: Parameters in the confile always override those that the application determines internally. Immediate parameter values, i.e., those passed in the **Application Call** and **Application Parameters** field of the template, always override the confile values.
Example	N/A

Stop RCS

Function	Stops the RCS.
Notes	Under Windows, the service interface is used. Under UNIX, the kill command is used. Also, this application runs **ipcs** in order to clean the shared IPC resources that the Radia Database was using. For this to work properly, this application needs to run under either *root* or the user ID that owned the Radia Database process.
Tcl Script	NVDOM_stopRCS.tcl
Parameters	-switchlog 0/1 (Optional. Switches the RCS log, before the Radia Database is shut down, so that the log is not lost. If the log is not switched, the Radia Database will overwrite it on startup. **-switchlog** is coded by default in the application's template. To disable -**switchlog**, simply remove it, or change its value to 0 This application uses the following parameters: -homeDir -username -appName -shortServiceName -pgmDir -descrName -tagFile These parameters, which are described in Table 2.2 on page 58, rarely have to be supplied. Under Windows, the program can always determine the correct parameter values. Under UNIX, if this application is not run under the Radia user name, the **homeDir** should be specified as an immediate parameter, or in the confile. The remaining parameters can then also be determined. **Note**: Parameters in the confile always override those that the application determines internally. Immediate parameter values, i.e., those passed in the **Application Call** and **Application Parameters** field of the template, always override the confile values.
Example	N/A

Switch Log

Function	Switches logs on the RCS.
Notes	Uses SNMP command. (RCS SNMP agent must be running).
Tcl Script	NVDOM_switchRCSlog.tcl
Parameters	None
Example	N/A

View Config

Function	View the RSC settings file (the edmprof).
Notes	None.
Tcl Script	NVDOM_viewRCSconfig.tcl
Parameters	-lines *nnn* (optional, shows last *n* lines of file) -first *nnn* (optional, shows first *n* lines of file) -last *nnn* (optional, shows last *n* lines of file) -re pat (optional, show only lines matching the regular expression pattern "pat". The regular expressions used by Tcl are documented in http://dev.scriptics.com/man/tcl8.4/TclCmd/re_syntax.htm#M3)
Example	`-last 20 -re CMD_LINE` This will show the last 20 lines of the edmprof that contain CMD_LINE.

View Log

Function	View the RCS log.
Notes	None.
Tcl Script	NVDOM_viewRCSlog.tcl
Parameters	-lines *nnn* (optional, shows last *n* lines of log) -first *nnn* (optional, shows first *n* lines of file) -last *nnn* (optional, shows last *n* lines of file) -re pat (optional, show only lines matching the regular expression pattern "pat". The regular expressions used by Tcl are documented in http://dev.scriptics.com/man/tcl8.4/TclCmd/re_syntax.htm#M3) -logPrefix prefix (Optional. The prefix used for RCS log names) -logSuffix suffix (Optional. The suffix used for RCS log names)

Example	`-re NVD8119I`
	This shows the lines in the RCS log that contain the string NVD8119I.
	`-last 500 -re "(0100I)|(04..I)"`
	Shows the last 500 lines, in the log file, that contain the string "0100I" or "04..I" where each dot in "04..I" can be any character.

NVD Op RIS

This group of applications is designed specifically for operational functions that can be performed on a *Radia Integration Server*. These functions can be used on Windows and UNIX operating systems.

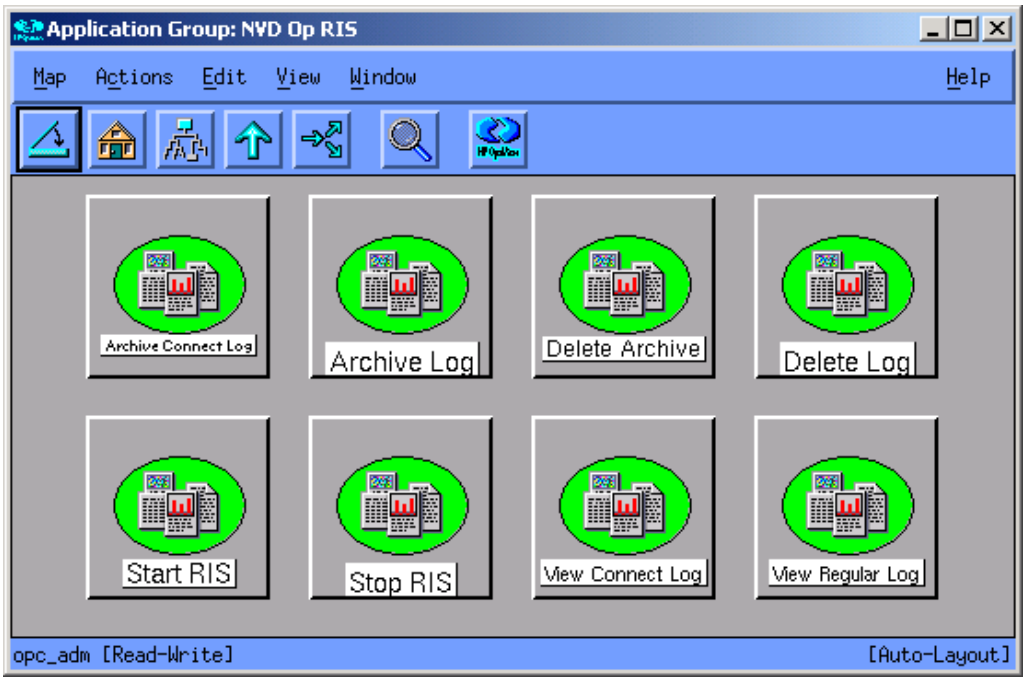


Figure 3.5 ~ The applications of the NVD Op RIS group.

Archive Connect Log

Function	Archive Radia Integration Server connect logs. This application will take all currently un-archived connect logs (except the one that is currently being written by the Radia Integration Server) and compress them into a bundle, which will then be stored in the *archive* subdirectory of the Radia Integration Server current log directory. If the **NVDOM_RIS_Log_FreeSpace_A** monitor detects that there is a disk space shortage on the volume where the logs are being written, and that the free space is within the *warning* threshold level, it will allow the operator to initiate this application to archive connect logs.
Notes	**UNIX**: **tar** is used to bundle up the files and the tar file is then compressed. **Windows**: there is no standard bundle-and-compression utility. For more information on this function and log rollover limit settings, refer to the section, *Understanding Radia Integration Server Logging*, starting on page 114.
Tcl Script	NVDOMspi.cmd - NVDOM_archiveRISlog.tcl -logtype C
Parameters	-logtype C/R (If *x* = C, the *connect* archive/log is acted on. If *x* = R, the *regular* archive/log is acted on. If *x* ≠ C or R, the regular archive/log is acted on. Note that -logtype = C is already coded in the **Application Call** field of this application.) -logPrefix prefix (Optional. The prefix used for RIS log name.) -logSuffix suffix (Optional. The suffix used for RIS log name.) -gmt 0/**1** (Optional. This specifies which time zone, local [**0**] or the default, GMT [**1**], should be used when calculating the local time to use to determine the current connect log.) -zip command (Not used under UNIX; optional under Windows. Here, **command** is the command that will be used to bundle and compress the log files that are being archived. The default is **pkzip ../$tempDirName *.***.)
Example	`-zip "cmd.exe \"/c\" c:/u/utils/myarchive.cmd"` This causes the **myarchive.cmd** to be run once against all the files to be archived. The **** that precedes the inner double quotes (" ") escapes them, so that the whole string after **-zip** is considered one parameter.

Archive Log

Function	Archive Radia Integration Server regular logs. This application will take all currently un-archived regular logs (except the one that is currently being written by the Radia Integration Server) and compress them into a bundle, which will then be stored in the *archive* subdirectory of the Radia Integration Server current log directory. If the **NVDOM_RIS_Log_FreeSpace_A** monitor detects that there is a disk space shortage on the volume where the logs are being written, and that the free space is within the *minor* threshold level, it will automatically allow the operator to initiate this application to archive regular logs.
Notes	**UNIX**: **tar** is used to bundle up the files and the tar file is then compressed. **Windows**: there is no standard bundle-and-compression utility. For more information on this function and log rollover limit settings, refer to the section, *Understanding Radia Integration Server Logging*, starting on page 114.
Tcl Script	NVDOM_archiveRISlog.tcl
Parameters	-logtype C/R (If *x* = C, the *connect* archive/log is acted on. If *x* = R, the *regular* archive/log is acted on. If *x* ≠ C or R, the regular archive/log is acted on. Note that the **Application Call** field of this application does not have a **-logtype** switch coded, so by default, this application deals with the *regular* log.) -logno n (Optional. The number of the log that was created by roll over.) -logPrefix prefix (Optional. The prefix used for RIS log names.) -logSuffix suffix (Optional. The suffix used for RIS log names.) -zip command (Not used under UNIX; optional under Windows. Here, **command** is the command that will be used to bundle and compress the log files that are being archived. The NVDOM_archiveRISlog.tcl default is **pkzip ../$tempDirName *.***.)
Example	`-zip "cmd.exe \"/c\" c:/u/utils/myarchive.cmd"` This causes the **myarchive.cmd** to be run once against all the files to be archived. The **** that precedes the inner double quotes (" ") escapes them, so that the whole string after **-zip** is considered one parameter.

Delete Archive

Function	Deletes one or more log archives. If the **NVDOM_RIS_Log_FreeSpace_A** monitor detects that there is a disk space shortage on the volume where the logs are being written, and that the free space is of a *major* severity level, it will automatically allow the operator to initiate this application to delete all RIS log archives.
Notes	None.
Tcl Script	NVDOM_deleteRISarchive.tcl
Parameters	-all **0**/1 (Optional. If $x = 1$, all logs are deleted. If $x \neq 1$, only the oldest log is deleted. The default is **0**.)
Example	`-all 1`

Delete Log

Function	Delete one or more RIS logs. The log currently being written by the RIS is never deleted.	
Notes	None.	
Tcl Script	NVDOM_deleteRISlog.tcl	
Parameters	-all **0**/1	(Optional. If $x = 1$, all logs are deleted. If $x \neq 1$, the oldest log only is deleted. The default is **0**.)
	-logtype C/R/**Ignore**	(If x = C, the *connect* archive/log is acted on. If x = R, the *regular* archive/log is acted on. If $x \neq$ C or R, the regular archive/log is acted on. When **Ignore**, the default, is coded, both log types are processed. In this case, whichever log is oldest will be deleted.)
	-logPrefix prefix	(Optional. Prefix used for RIS log name.)
	-logSuffix suffix	(Optional. The suffix used for RIS log names.)
	-gmt **0**/1	(Optional. Specifies whether the local time zone (**0**) or the GMT time zone (**1**) is used in calculating the current date. The RIS uses GMT time for its logs, unless it has been otherwise customized.)
Example	`-all 1` All logs, except the current one, are deleted.	

Start RIS

Function	Starts the Radia Integration Server.
Notes	Under Windows, the service interface is used. Under UNIX, an **su** command is performed to switch to the Radia user name, if the application is not already running under that user name. If the user name is not found as a parameter (immediate or from the confile), the owning user of the **httpd.rc** file is used.
Tcl Script	NVDOM_startRIS.tcl
Parameters	This application uses the following parameters: -homeDir -username -appName -shortServiceName -pgmDir -descrName -tagFile These parameters, however, rarely have to be supplied. Under Windows, the program can always determine the correct parameter values. Under UNIX, if the RIS was not installed in its default location, the **homeDir** should be specified as an immediate parameter, or in the confile. The remaining parameters can then also be determined. **Note**: Parameters in the confile always override those that the application determines internally. Immediate parameter values, those passed in the **Application Call** and **Application Parameters** field of the template, always override the confile values.
Example	N/A

Stop RIS

Function	Stops the Radia Integration Server.
Notes	Under Windows, the service interface is used. Under UNIX, the kill command is used.
Tcl Script	NVDOM_stopRIS.tcl
Parameters	This application uses the following parameters: -homeDir -username -appName -shortServiceName -pgmDir -descrName -tagFile These parameters, however, rarely have to be supplied. Under Windows, the program can always determine the correct parameter values. Under UNIX, if the RIS was not installed in its default location, the **homeDir** should be specified as an immediate parameter, or in the confile. The remaining parameters can then also be determined. **Note**: Parameters in the confile always override those that the application determines internally. Immediate parameter values, those passed in the **Application Call** and **Application Parameters** field of the template, always override the confile values.
Example	N/A

View Connect Log

Function	View the Radia Integration Server connect logs.
Notes	For more information on this function and log rollover limit settings, refer to the section, *Understanding Radia Integration Server Logging*, starting on page 114.
Tcl Script	NVDOM_viewRISlog.tcl
Parameters	-lines *nnn* (Optional. Shows last *n* lines of log.) -first *nnn* (Optional, shows first *n* lines of file) -last *nnn* (Optional, shows last *n* lines of file) -re pattern (Optional. Show only lines matching the regular expression pattern "pattern". The regular expressions used by Tcl are documented in http://dev.scriptics.com/man/tcl8.4/TclCmd/re_syntax.htm#M3) -yy year -mm month -dd day **Note**: These three parameters are optional. They are used to specify the name of the log to view. These values default to the current day. If not specified, the *current* log is viewed. -logtype C/R (If *x* = C, the *connect* archive/log is acted on. If *x* = R, the *regular* archive/log is acted on. If *x* ≠ C or R, the regular archive/log is acted on. Note that -logtype = C is already present in the **Application Call** field of this application, so by default, the *connect* log is processed.) -logPrefix prefix (Optional. The prefix used for RIS log names.) -logSuffix suffix (Optional. The suffix used for RIS log names.) -gmt **0**/1 (This parameter is optional. It specifies whether the local time zone (**0**) or the GMT time zone (**1**) should be used in calculating the current date. The RIS uses GMT time for its logs, unless it has been otherwise customized.)
Example	`-dd 4 -re "2002:10"` Views lines in the connect log for the 4th day of the month. Only lines that contain the characters **2002:10** are shown—in this case, all the messages that were written from 10:00 to 10:59 on that day.

View Regular Log

Note

Before configuring the settings of this application, consult the section *Understanding Radia Integration Server Logging* on page 114.

Function	View the Radia Integration Server regular log.	
Notes	For more information on this function and log rollover limit settings, refer to the section *Understanding Radia Integration Server Logging* starting on page 114.	
Tcl Script	NVDOM_viewRISlog.tcl	
Parameters	-lines *nnn*	(Optional. Shows last *n* lines of log.)
	-first *nnn*	(Optional, shows first *n* lines of file)
	-last *nnn*	(Optional, shows last *n* lines of file)
	-re pattern	(Optional. Show only lines matching the regular expression pattern "pattern". The regular expressions used by Tcl are documented in http://dev.scriptics.com/man/tcl8.4/TclCmd/re_syntax.htm#M3)
	-logtype C/R	(If *x* = C, the *connect* archive/log is acted on. If *x* = R, the *regular* archive/log is acted on. If *x* ≠ C or R, the regular archive/log is acted on. Note that -logtype is not present in the **Application Call** field of this application, so by default, the *regular* log is processed.)
	-logPrefix prefix	(Optional. The prefix used for RIS log names.)
	-logSuffix suffix	(Optional. The suffix used for RIS log names.)
	-logno n	(Optional. The number of the log that was created by roll over.)
Example	`-last 5 -re "on port"` Shows the last 5 lines in the regular RIS log that contain the characters "on port."	

Note

This view application also supports the parameters, **-lines**, **-first**, **-last**, and **-re**. These can be used to display a *specific number of lines* only, and lines that match a *regular expression*, respectively. These parameters can be used together.

NVD Op CLI

This group of applications is designed specifically for operations that can be performed on a *Radia Client*. These functions can be used on Windows and UNIX operating systems.

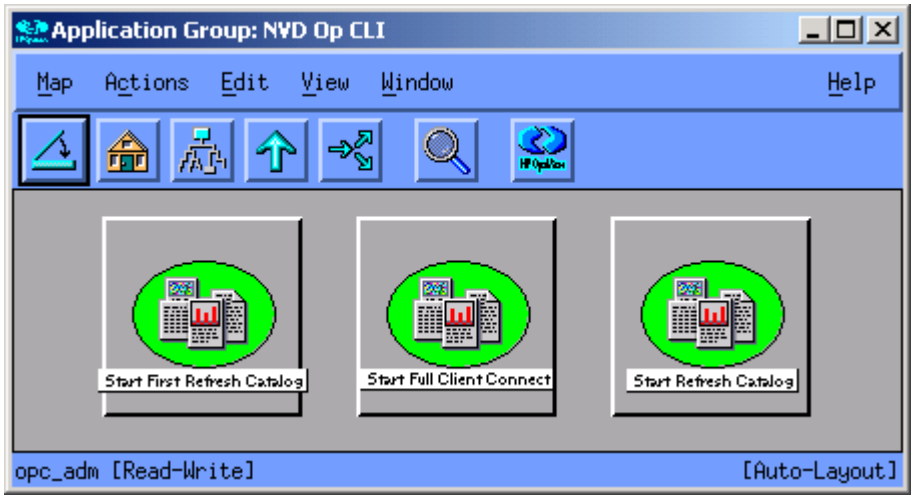


Figure 3.6 ~ The applications of the NVD Op CLI group.

Start First Refresh Catalog

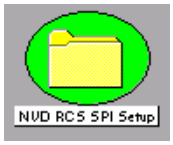
Function	Connect a Radia Client to its RCS and run a **First Refresh Catalog**.
Notes	A Radia notify request will be sent to the Radia Client on which this application is running. The **radpinit** program is run. This refreshes the catalog, runs Radia self-maintenance, and updates ZMASTER.ZUSERID. **Note**: See Radia documentation on this program for further details about the parameters that are described in the **Parameters** section.
Tcl Script	NVDOM_connectClient.tcl
Parameters	The default values for this application's parameters are usually sufficient to run it successfully. The application uses the following parameters: -IDMLIB — The directory at the Radia Client where Radia objects are kept. The ZMASTER object therein is used to resolve the **-user**, **-notifyPort**, **-clientPort**, and **-RCSipaddr** parameters when they are not specified as immediate or permanent parameters. -notifyPort — The port at the client on which the Novadigm notify daemon is contacted. -user — The user name for which the Client Connect will be done. -CLIipaddr — The IP address or name of the Radia Client. If not specified, the host name of the machine on which the application runs will be used. -clientPort — The port on the RCS computer that the client contacts. -RCSipaddr — The IP address or name of the RCS. -mname — The *manager name* parameter that is used on the **radpinit** call that is initiated on the client computer. Default is **"RADIA"**. -dname — The *domain name* parameter that is used on the **radpinit** call that is initiated on the client computer. Default is **"SOFTWARE"**. -ulogon — The *user logon panel* option that is used on the **radpinit** call that is initiated on the client computer. Valid values are **y** (show logon panel) and **n** (suppress logon panel). The default is **n**. -ind — The *progress indicator* option that is used on the **radpinit** call that is initiated on the client computer. Valid values are **y** (show progress indicator) and **n** (suppress progress indicator). The default is **n**.
Example	-ind y

Start Full Client Connect

Function	Connect a Radia Client to its RCS and run a **Full Connect**.
Notes	A Radia notify request will be sent to the Radia Client on which this application is running. The **radskman** program is run. **Note**: See Radia documentation on this program for further details about the parameters that are described in the **Parameters** section.
Tcl Script	NVDOM_connectClient.tcl
Parameters	The default values for this application's parameters are usually sufficient to run it successfully. The application uses the following parameters: -IDMLIB — The Radia Client directory where Radia objects are kept. The ZMASTER object therein is used to resolve the **-user**, **-notifyPort**, **-clientPort**, and **-RCSipaddr** parameters when they are not specified as immediate or permanent parameters. -notifyPort — The port at the client on which the Novadigm notify daemon is contacted. -user — The user name for which the Client Connect will be done. -CLIipaddr — The IP address or name of the Radia Client. If not specified, the host name of the machine on which the application runs will be used. -clientPort — The port on the RCS computer that the client contacts. -RCSipaddr — The IP address or name of the RCS. -mname — The *manager name* parameter that is used on the **radskman** call that is initiated on the client computer. Default is **"RADIA"**. -dname — The *domain name* parameter that is used on the **radskman** call that is initiated on the client computer. Default is **"SOFTWARE"**. -cat — The *catalog* option that is used on the **radskman** call that is initiated on the client computer. See **radskman** documentation for more details of this parameter. Valid values are **y** (show catalog), **n** (suppress catalog), and **prompt** (prompt for catalog). The default is **prompt**. -ulogon — The *user logon panel* option that is used on the **radskman** call that is initiated on the client computer. Valid values are **y** (show logon panel) and **n** (suppress logon panel). The default is **n**. -ind — The *progress indicator* option that is used on the **radskman** call that is initiated on the client computer. Valid values are **y** (show progress indicator) and **n** (suppress progress indicator). The default is **n**.
Example	-ind y

Start Refresh Catalog

Function	Connect a Radia Client to its RCS and run a **Refresh Catalog**.
Notes	A Radia notify request will be sent to the Radia Client on which this application is running. The **radpinit** program is run. This refreshes the catalog, but does not run Radia self-maintenance or update ZMASTER.ZUSERID. **Note**: See Radia documentation on this program for further details about the parameters that are described in the **Parameters** section.
Tcl Script	NVDOM_connectClient.tcl
Parameters	The default values for this application's parameters are usually sufficient to run it successfully. The application uses the following parameters: -IDMLIB — The directory at the Radia Client where Radia objects are kept. The ZMASTER object therein is used to resolve the **-user**, **-notifyPort**, **-clientPort**, and **-RCSipaddr** parameters when they are not specified as immediate or permanent parameters. -notifyPort — The port at the client on which the Novadigm notify daemon is contacted. -user — The user name for which the Client Connect will be done. -CLIipaddr — The IP address or name of the Radia Client. If not specified, the host name of the machine on which the application runs will be used. -clientPort — The port on the RCS computer that the client contacts. -RCSipaddr — The IP address or name of the RCS. -mname — The *manager name* parameter that is used on the **radpinit** call that is initiated on the client computer. Default is **"RADIA"**. -dname — The *domain name* parameter that is used on the **radpinit** call that is initiated on the client computer. Default is **"SOFTWARE"**. -ulogon — The *user logon panel* option that is used on the **radpinit** call that is initiated on the client computer. Valid values are **y** (show logon panel) and **n** (suppress logon panel). The default is **n**. -ind — The *progress indicator* option that is used on the **radpinit** call that is initiated on the client computer. Valid values are **y** (show progress indicator) and **n** (suppress progress indicator). The default is **n**.
Example	-ind y



NVD RCS SPI Setup

This group of applications is designed specifically for the installation of the Novadigm Smart Plug-In on UNIX and Windows managed nodes, setting and viewing the parameters of the *Radia Configuration Server*, and testing the various system monitors (for more information, see *Chapter 5: Message Groups*, on page 121).

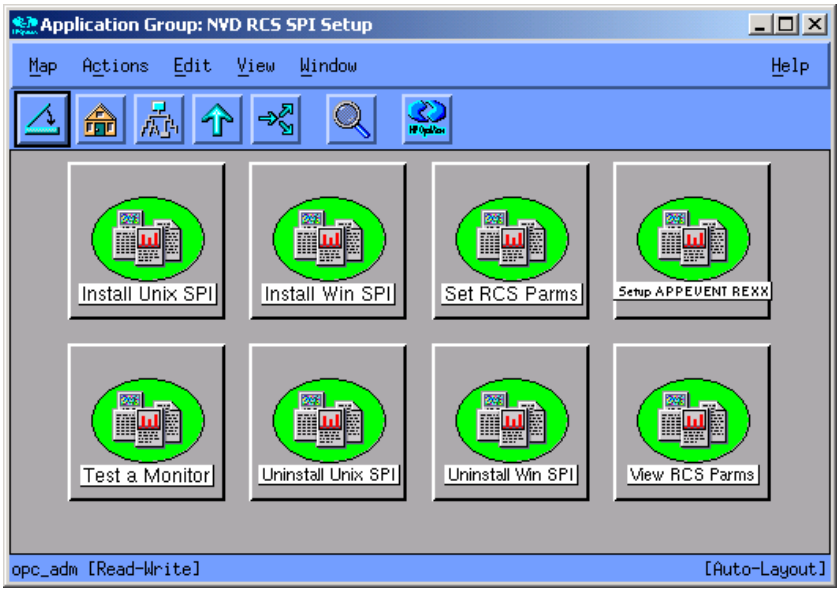


Figure 3.7 ~ The applications of the NVD RCS SPI Setup group.

The **Set RCS Parms**, **Setup APPEVENT REXX**, **Test a Monitor**, and **View RCS Parms** functions can be used on Windows and UNIX operating systems.

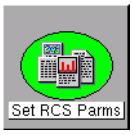
Install Unix SPI

Function	Installs the Novadigm SPI on UNIX managed nodes.
Notes	This application must be run under root.
Command	NVDOMins.sh
Parameters	• The parameters to this application are positional. The four parameters in order of position are: • Novadigm SPI resident directory path (also called the **zpath**). To use the default for this parameter, use a dash (-). **Note**: We recommend using the default, **zpath**. • Host name (or IP address) of the OVO server that houses the Novadigm SPI. This host must be running an FTP server. • FTP server User Name. • Password of FTP server User Name.
Example	`- ftp.ovoserver.customer.com ftpid ftppass`

Install Win SPI

Function	Installs the Novadigm SPI on Windows managed nodes.
Notes	Must be run from a user that is a member of the Administrators group.
Command	NVDOMins.bat
Parameters	• The parameters to this application are positional. The four parameters in order of position are: • Novadigm SPI resident directory path (also called the **zpath**). To use the default for this parameter, use a dash (-). **Note**: We recommend using the default, **zpath**. • Host name (or IP address) of the OVO server that houses the Novadigm SPI. This host must be running an FTP server. • FTP server User Name. • Password of FTP server User Name.
Example	`- ftp.ovoserver.customer.com ftpid ftppass`

Set RCS Parms



Function	Sets the RCS parameters (confile parameters) for future applications.
Notes	The RCS confile is **RCS_config.tcl** and is stored in the **zpath** on the managed node.
Tcl Script	NVDOM_setParms.tcl
Parameters	-delete **0**/1 (Optional. If set to **1**, the confile is deleted before any new parameters are placed in it.) For comprehensive information on all of the parameters that are associated with this function, see *Table 2.2 ~ RCS_config.tcl File Parameters,* on page 58, in the section, *Using Permanent Parameters*. **Note**: Any parameters that are specified here will be stored in the RCS_config.tcl file.
Example	`-home /home/radia -homeDir /home/radia –snmpPort 7901` **Note**: This places the **homeDir** and **snmpPort** parameters, with their respective values of **/home/radia** and **7901**, into the RCS confile. This example is typical of one being run under UNIX, where the edmprof file is contained in the Radia home directory, but cannot be found automatically by the applications. After the example is run, future applications will be able to use the **homeDir** parameter, which this example places in the confile.

Setup APPEVENT REXX

Function	Copies the APPEVENT object into the RCS REXX path, and automatically edits it with the correct settings.
Notes	None.
Tcl Script	NVDOM_setupAppeventRexx.tcl

Parameters	This application is usually run without specifying any parameters, as the defaults are often appropriate. -tlimit mm (Optional. The maximum number of minutes before the current time that will cause a trap. Anything older is not trapped. The default is **120** minutes.) -clockDiff mm (Optional. We will reject any events whose timestamp is newer than ours by a difference greater that that specified by the next parameter. Since the client computer's clock might be running in a time zone different than the RCS, the difference permitted might need to be adjusted. The default is **2** minutes.) -community public (Optional. The SNMP community name to be placed in the trap. The default is **public**.) -utilityPath path (Optional. The path where the edmsnmpt program is stored on the RCS. Always include a path separator character as the last character of this parameter. If this parameter is not specified, the zpath will be used. The **Install Unix SPI** and **Install Win SPI** applications will have placed the edmsnmpt program in the zpath. If this parameter is used to set a different directory, make sure that edmsnmpt is in that directory.) -replace **Y**/N (Optional. Specify **Y** for this APPEVENT program to replace any that exist in the RCS REXX directory. If **N**, the existing APPEVENT will remain. The default is **Y**.)
Example	N/A

Test Monitor

Function	Test a monitor by running it once as an application.
Notes	In UNIX, monitors are run under user **root**. This application should therefore, be run under root in UNIX to produce the same effect the monitor does.
Tcl Script	Any monitor, such as: NVDOM_monitorProcess.tcl, NVDOM_monitorResponse.tcl, and NVDOM_monitorFreeDiskSpace.tcl.
Parameters	The first two parameters are positional: zpath (Required. To use the default, **zpath**, specify just a dash -.) monitor_name (Required. Specify the name of the monitor to be run.) -ztype RCS/RIS/CLI (Required. Specify the component to be monitored.) -monitorName xxxx (Required. The monitor to which the results will be sent. Since this monitor will not be running, error OpC30-691 will be generated, but it can be ignored.) -debug 99 (Use this parameter to see the output of the monitor.) Other parameters as appropriate for the monitor.
Example	`- NVDOM_monitorProcess.tcl -ztype RCS -monitorName test -debug 99`

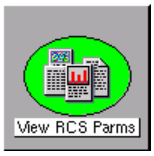
Uninstall Unix SPI

Function	Uninstalls the Novadigm SPI from UNIX managed nodes.
Notes	This application must be run under root. It deletes the zpath directory from the managed node.
Command	NVDOMrem.sh
Parameters	There is only one parameter, and it is positional: zpath (Required. To use the default, **zpath**, specify just a dash -.)
Example	-

Uninstall Win SPI

Function	Uninstalls the Novadigm SPI from Windows managed nodes.
Notes	This application must be run from a user that is a member of the Administrator's group. It deletes the zpath directory from the managed node.
Command	NVDOMrem.bat
Parameters	There is only one parameter, and it is positional: zpath (Required. To use the default, **zpath**, specify just a dash -.)
Example	-

View RCS Parms



Function	Shows the permanent RCS parameters that are in effect for a managed node.
Notes	The output is in two parts: • The values that are in effect at the managed node. • The values that are stored in the confile.
Tcl Script	NVDOM_viewParms.tcl
Parameters	N/A
Example	N/A

NVD RIS SPI Setup

This group of applications is designed specifically for the installation of the Novadigm Smart Plug-In on UNIX and Windows managed nodes, setting and viewing the parameters of the *Radia Integration Server*, and testing the various system monitors (for more information, see *Chapter 5: Message Groups*, on page 121).

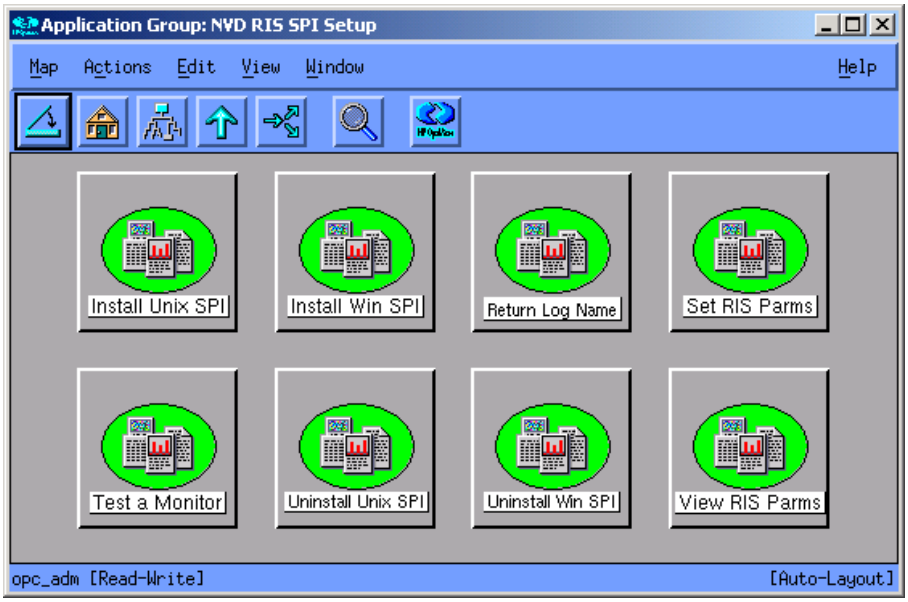


Figure 3.8 ~ The applications of the NVD RIS SPI Setup group.

The **Return Log Name**, **Set RIS Parms**, **Test a Monitor**, and **View RIS Parms** functions can be used on Windows and UNIX operating systems.

Install Unix SPI

Function	Installs the Novadigm SPI on UNIX managed nodes.
Notes	This application must be run under root.
Command	NVDOMins.sh
Parameters	• The parameters to this application are positional. The four parameters in order of position are: • Novadigm SPI resident directory path (also called the **zpath**). To use the default for this parameter, use a dash (-). **Note**: We recommend using the default **zpath**. • Host name (or IP address) of the OVO server that houses the Novadigm SPI. This host must be running an FTP server. • FTP server User Name. • Password of FTP server User Name.
Example	`- ftp.ovoserver.customer.com ftpid ftppass`

Install Win SPI

Function	Installs the Novadigm SPI on Windows managed nodes.
Notes	Must be run from a user that is a member of the Administrators group.
Command	NVDOMins.bat
Parameters	• The parameters to this application are positional. The four parameters in order of position are: • Novadigm SPI resident directory path (also called the **zpath**). To use the default for this parameter, use a dash (-). **Note**: We recommend using the default **zpath**. • Host name (or IP address) of the OVO server that houses the Novadigm SPI. This host must be running an FTP server. • FTP server User Name. • Password of FTP server User Name.
Example	`- ftp.ovoserver.customer.com ftpid ftppass`

Return Log Name

Function	Returns the name of the current, regular Radia Integration Server log. This application is intended to test the Logfile command that the **NVDOM RIS Log Msgs** template uses to return the fully qualified log name of the current regular RIS log.
Notes	A Logfile command must return only the name of the log file the template is to monitor. Because of this, the command cannot normally write debugging information. Running the command separately as an application makes it possible to debug messages.
Command	NVDOMrln.cmd
Parameters	zpath (Positional. Directory where SPI elements are stored. To use the default for this parameter, use a dash (-).) ztype Positional. RIS is currently the only supported zpath for this application.) The above positional parameters can be omitted only if no other parameters follow. The remaining parameters are not positional and must be specified as Tcl parameter keyword–value pairs. -debug **0**/1 (Optional. Using a -debug value of **1** will show all messages produced by the monitor.)
Example	`- RIS -debug 1`

Set RIS Parms

Function	Sets the Radia Integration Server parameters (confile parameters) for future applications.
Notes	The RIS confile is **RIS_config.tcl** and is stored in the **zpath** on the managed node.
Tcl Script	NVDOM_setParms.tcl
Parameters	-delete **0**/1 (Optional. If set to **1**, the confile is deleted before any new parameters are placed in it.) For comprehensive information on all of the parameters that are associated with this function, see *Table 2.3 ~ RIS_config.tcl File Parameters*, on page 61, in the section, *Using Permanent Parameters*. **Note**: Any parameters that are specified here will be stored in the RIS_config.tcl file.
Example	`-shutdownWaitTime 180`

Test Monitor

Function	Test a monitor by running it once as an application.
Notes	In UNIX, monitors are run under user **root**. This application should therefore be run under root in UNIX to produce the same effect the monitor does.
Tcl Script	Any monitor, such as, NVDOM_monitorProcess.tcl, NVDOM_monitorResponse.tcl, or NVDOM_monitorFreeDiskSpace.tcl.
Parameters	The first and second parameters are positional: zpath (Required. Specify a dash - to use the default, **zpath**.) monitor_name (Required. Specify the name of the monitor to be run.) -ztype RCS/RIS/CLI (Required. Specify the component to be monitored.) -monitorName xxxx (Required. The name of the monitor to which the results will be sent. Since this monitor will not be running, error OpC30-691 will be generated, but it can be ignored.) -debug 99 (Use this parameter to see the output of the monitor.) Other parameters as appropriate for the monitor.
Example	`- NVDOM_monitorProcess.tcl -ztype RIS -monitorName test -debug 99`

Uninstall Unix SPI

Function	Uninstalls the Novadigm SPI from UNIX managed nodes.
Notes	This application must be run under root. It deletes the zpath directory from the managed node.
Command	NVDOMrem.sh
Parameters	There is only one parameter, and it is positional: zpath (Required. Specify a dash - to use the default, **zpath**)
Example	-

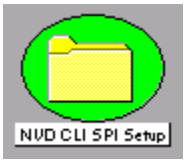
Uninstall Win SPI

Function	Uninstalls the Novadigm SPI from Windows managed nodes.
Notes	This application must be run from a user that is a member of the Administrator's group. It deletes the zpath directory from the managed node.
Command	NVDOMrem.bat
Parameters	There is only one parameter, and it is positional: zpath (Required. Specify a dash - to use the default, **zpath**)
Example	-

View RIS Parms

Function	Shows the permanent RIS parameters that are in effect for a managed node.
Notes	The output is in two parts: • The values that are affected at the managed node. • The values that are stored in the confile.
Tcl Script	NVDOM_viewParms.tcl
Parameters	N/A
Example	N/A

NVD CLI SPI Setup



This group of applications is designed specifically for the installation of the Novadigm Smart Plug-In on UNIX and Windows managed nodes, setting and viewing the parameters of the *Radia Client*, and testing the various system monitors (for more information, see *Chapter 5: Message Groups* on page 121).

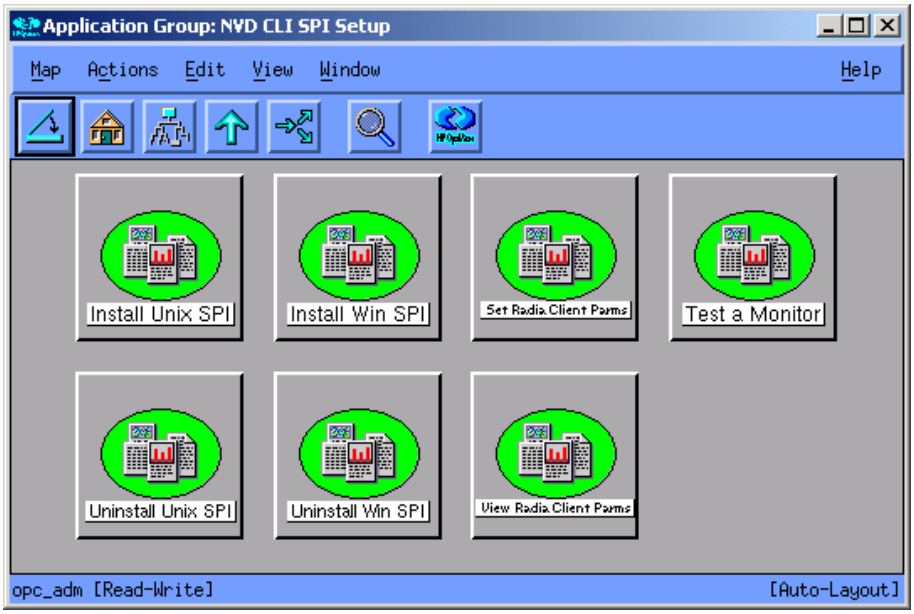


Figure 3.9 ~ The applications of the NVD CLI SPI Setup group.

The **Set Radia Client Parms**, **Test a Monitor**, and **View Radia Client Parms** functions can be used on Windows and UNIX operating systems.

Install Unix SPI

Function	Installs the Novadigm SPI on UNIX managed nodes.
Notes	This application must be run under root.
Command	NVDOMins.sh
Parameters	• The parameters to this application are positional. The four parameters in order of position are: • Novadigm SPI resident directory path (also called the **zpath**). To use the default for this parameter, use a dash (-). **Note**: We recommend using the default **zpath**. • Host name (or IP address) of the OVO server that houses the Novadigm SPI. This host must be running an FTP server. • FTP server User Name. • Password of FTP server User Name.
Example	`- ftp.ovoserver.customer.com ftpid ftppass`

Install Win SPI

Function	Installs the Novadigm SPI on Windows managed nodes.
Notes	Must be run from a user that is a member of the Administrators group.
Command	NVDOMins.bat
Parameters	• The parameters to this application are positional. The four parameters in order of position are: • Novadigm SPI resident directory path (also called the **zpath**). To use the default for this parameter, use a dash (-). **Note**: We recommend using the default **zpath**. • Host name (or IP address) of the OVO server that houses the Novadigm SPI. This host must be running an FTP server. • FTP server User Name. • Password of FTP server User Name.
Example	`- ftp.ovoserver.customer.com ftpid ftppass`

Set Radia Client Parms

Function	Sets the Radia Client parameters (confile parameters) for future applications.
Notes	The Radia Client confile is **CLI_config.tcl** and is stored in the **zpath** on the managed node.
Tcl Script	NVDOM_setParms.tcl
Parameters	-delete **0**/1 (Optional. If set to **1**, the confile is deleted before any new parameters are placed in it.) For comprehensive information on all of the parameters that are associated with this function, see *Table 2.4 ~ CLI_config.tcl File Parameters*, on page 62, in the section, *Using Permanent Parameters*. **Note**: Any parameters that are specified here will be stored in the RIS_config.tcl file.
Example	`-IDMLIB /var/Novadigm/lib`

Test Monitor

Function	Test a monitor by running it once as an application.
Notes	In UNIX, monitors are run under user **root**. This application should therefore, be run under root in UNIX to produce the same effect the monitor does.
Tcl Script	Any monitor, such as, NVDOM_monitorClient.tcl and NVDOM_monitorClientDaemon.tcl.
Parameters	The first and second parameters are positional: zpath (Required. Specify a dash - to use the default, **zpath**.) monitor_name (Required. Specify the name of the monitor to be run.) -ztype RCS/RIS/CLI (Required. Specify the component to be monitored.) -monitorName xxxx (Required. The name of the monitor to which the results will be sent. Since this monitor will not be running, error OpC30-691 will be generated, but it can be ignored.) -debug 99 (Use this parameter to see the output of the monitor.) Other parameters as appropriate for the monitor.
Example	`- NVDOM_monitorClientDaemon.tcl -ztype CLI -monitorName test -debug 99`

Uninstall Unix SPI

Function	Uninstalls the Novadigm SPI from UNIX managed nodes.
Notes	This application must be run under root. It deletes the zpath directory from the managed node.
Command	NVDOMrem.sh
Parameters	There is only one parameter, and it is positional: zpath (Required. Specify a dash - to use the default, **zpath**)
Example	-

Uninstall Win SPI

Function	Uninstalls the Novadigm SPI from Windows managed nodes.
Notes	This application must be run from a user that is a member of the Administrators group. This application deletes the zpath directory from the managed node.
Command	NVDOMrem.bat
Parameters	There is only one parameter, and it is positional: zpath (Required. Specify a dash - to use the default, **zpath**)
Example	-

View Radia Client Parms

Function	Shows the permanent Radia Client (CLI) parameters that are in effect for a managed node.
Notes	The output is in two parts: • The values in that are effect at the managed node. • The values that are stored in the confile.
Tcl Script	NVDOM_viewParms.tcl
Parameters	N/A
Example	N/A

Understanding Radia Integration Server Logging

The Radia Integration Server generates two logs, *regular* and *connect*. The regular log reports on all processes for the RIS, whereas the connect log focuses exclusively on the clients that have or are connected to the RIS.

This section describes the logs in further detail, and presents information on:

- when the logs roll over,
- how to configure the intervals at which they are deleted,
- how they are labeled as a result of initializations, and
- the parameters to specify in order to view a log.

The Regular Log

The *regular* log contains one-line messages that describe the informational, warning, and error events that occur as the Radia Integration Server starts up and processes its HTTP transactions. The log that is currently being written by the server is named **httpd-*nnnn*.log**, where *nnnn* is the port number on the server on which it receives HTTP requests.

The log *rolls over* automatically at midnight and every time the Radia Integration Server is started. Each log that preceded the current one has a successive *log number* added to its name, **httpd-*nnnn.m*.log**. This log number (*m*) is coded as the **-logno** parameter of the **View Regular Log** application.

Table 3.2 presents a series of examples of log rollover activity. In the examples, assume a setting of **LOG_LIMIT = 2**. The default is 7.

Table 3.2 ~ Radia Integration Server Log Rollovers

Radia Integration Server Initialization	Resulting Logs
First initialization	**httpd-3466.log** is created.
Second initialization	**httpd-3466.log** (from first initialization) becomes **httpd-3466.1.log**. A new **httpd-3466.log** (for this initialization) is created.
Third initialization	**httpd-3466.1.log** (from first initialization) becomes **httpd-3466.2.log**. **httpd-3466.log** (from second initialization) becomes **httpd-3466.1.log**. A new **httpd-3466.log** (for this initialization) is created.
Fourth initialization	**httpd-3466.2.log** (from first initialization) is *deleted*. **httpd-3466.1.log** (from second initialization) becomes **httpd-3466.2.log**. **httpd-3466.log** (from third initialization) becomes **httpd-3466.1.log**. A new **httpd-3466.log** (for this initialization) is created.

The current log is not counted in the value that is specified for LOG_LIMIT; so there is always one more log than that value. Therefore, the default LOG_LIMIT = 7 would result in the current log, plus the seven previous initializations.

Since LOG_LIMIT = 2 (in this example), the log from the first initialization is deleted in the fourth initialization.

- The log with a -logno of **1**, (http-*nnnn*.**1**.log) is always considered the *most recent* log.
- The current initialization always generates the *current* log, http-*nnnn*.log. This file is considered to have a -logno of **0**.

The Connect Log

The *connect* log contains one-line messages that describe the clients that have connected to the Radia Integration Server. The connect log that is currently being written by the server is named **httpd-*nnnn.yy.mm.dd*.log**, where *nnnn* is the port number on the server on which it receives HTTP requests, and *yy.mm.dd* represents the date.

The log *rolls over* automatically at midnight. If the Radia Integration Server is re-started, it will append to the current log. To view a log, use the *-yy*, *-mm*, and *-dd* parameters in the **View Connect Log** application to specify the date of the log. If these parameters are not specified, the current connect log is displayed.

This rollover activity limit setting is applicable to the following six applications only, View Connect Log and View Regular Log (under **NVD Admin RIS**), Archive Connect Log, Archive Log, View Connect Log, and View Regular Log (under **NVD Op RIS**).

Application Troubleshooting

The immediate parameter, **-debug**, with a value of **100** (see Table 2.1, on page 53), will output the version numbers of the nvdkit program and the NVDOM module in use by the adapter. This parameter-value combination can be added to any of the applications that are provided with this adapter, as described below.

- Right-click the application and select **Customized Startup**.
- Add the parameter-value combination, **-debug 100**, to the **Application Call** field.

Note

If you encounter a problem with an application and need assistance from HP OpenView support, send the output that was produced by running the application with this parameter.

4

Node Groups

The Radia Adapter for HP OpenView Operations installation supplies three Novadigm node groups, one each for the three Radia components (Radia Configuration Server, Radia Integration Server, and Radia Client). These node groups have related source templates pre-assigned to them. These pre-assigned templates are those that capture Radia events. Because of this, the associated Radia events will be collected from a managed node as soon as templates are distributed to that node.

Notes

Monitor templates are not pre-assigned.

Before assigning these, make sure that the frequency at which they run is suitable for your environment.

The Novadigm node groups are:

- **NVD RCS**
  for nodes running a Radia Configuration Server.
- **NVD RIS**
  for nodes running a Radia Integration Server.
- **NVD Radia Clients**
  for nodes running a Radia Client.

A node can exist in more than one of these groups.

Refer to Figure 4.2 to see the pre-assigned source templates.

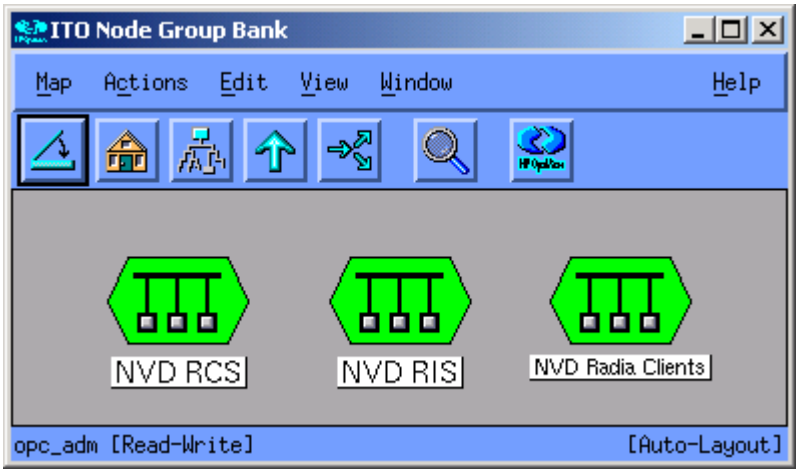


Figure 4.1 ~ The Node Group Bank window.

Double-click a group's icon to see which nodes are associated with it.

To add a node to a group

- Drag-and-drop the nodes from the node bank onto the node group.

  Depending on the user name under which the applications will run, we recommend separating the UNIX and Windows nodes. The supplied node groups can be copied to create platform specific ones.

To assign templates for a group

All non-monitor source templates that are provided with the adapter have been pre-assigned to the appropriate Novadigm node group. To use these templates, simply add the node to the appropriate group (as described above) and distribute the group to the node.

1. Click **Actions**, then **Agents**, and finally, **Assign Templates**.

   The **Define Configuration** window opens.

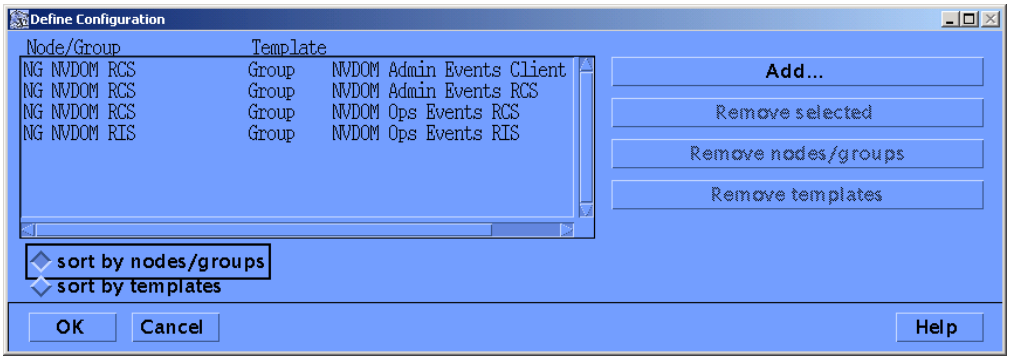


Figure 4.2 ~ The Define Configuration window.

2. Click **Add**.

The **Add Configuration** window opens.

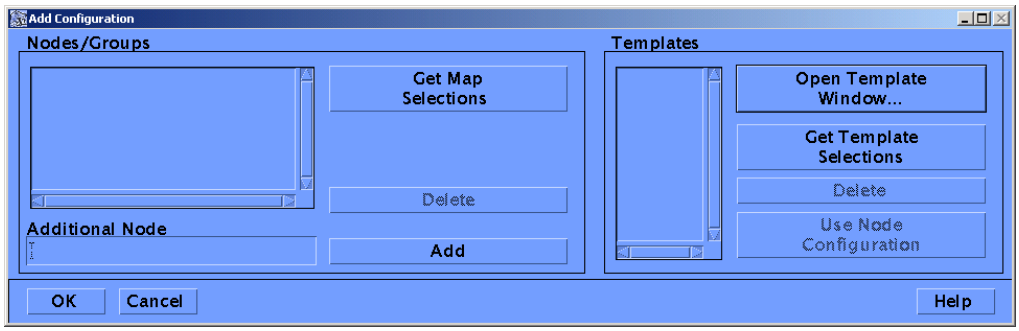


Figure 4.3 ~ The Add Configuration window.

For information on how to specify the information in these windows, consult the *HP OpenView VantagePoint Operations for UNIX Concepts Guide (Edition 4)*, *Chapter 3*, *Configuring and Maintaining ITO*.

5

Message Groups

The Radia Adapter for HP OpenView Operations installation creates two Novadigm message groups, **NVD Adm** and **NVD Op**. These allow visibility of messages for Radia administrators and IT operators, respectively.

Radia Component Messages

- **Radia Configuration Server**
  these messages will originate as SNMP traps. They are converted locally (on the managed node) to OVO messages and are optionally forwarded to the OVO Management Server, depending on the options defined in the templates that have been distributed to the managed node.
- **Radia Integration Server**
  these messages are obtained by scraping the logs. These messages are also processed locally.
- **Radia Clients**
  these messages come by way of the APPEVENT object, which the client sends to the Radia Configuration Server (RCS). When the APPEVENT object arrives at the RCS, a REXX method is triggered which sends an SNMP trap for each instance in the object. These traps are then handled in the same way as RCS traps (discussed in the first bullet of this section).

The sections that follow present information that will assist in accessing the Novadigm message groups and defining authority for accessing them.

To access the Message Group Bank

- Click **Window**, and **Message Group Bank**.
  The **Message Group Bank** window opens.

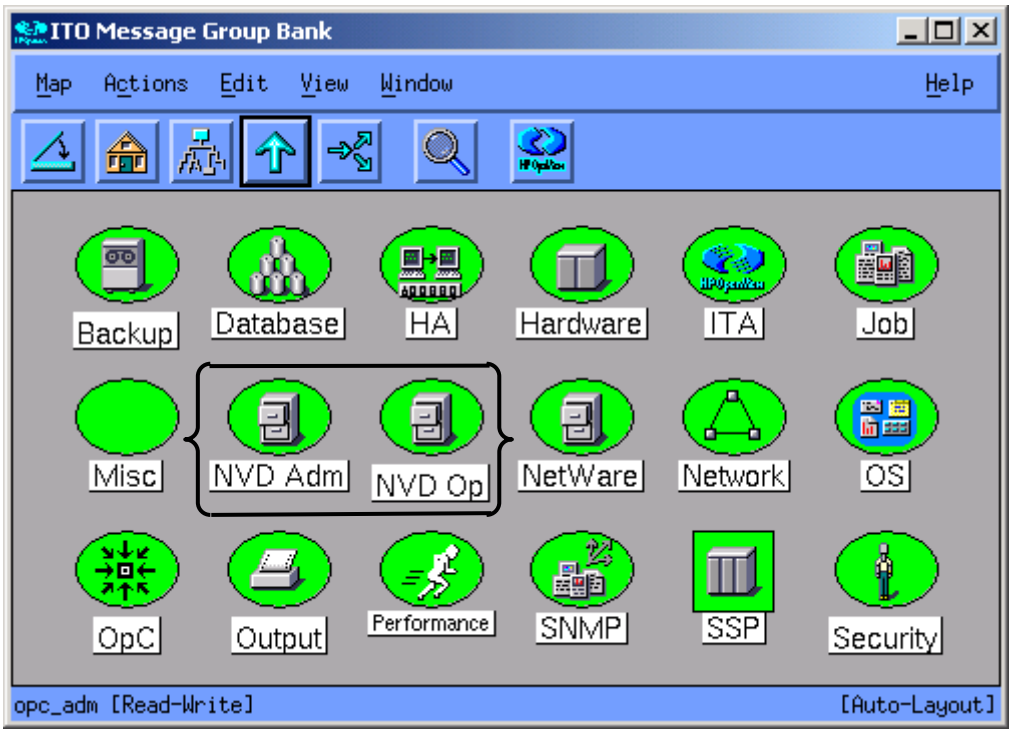


Figure 5.1 ~ The two NVD message groups in the Message Group Bank window.

The section that follows, *NVD Message Groups*, details the two Novadigm *message groups*.

NVD Message Groups

These message groups allow visibility of messages for Radia administrators and IT operators.

Radia Admin Group (NVD Adm)

This message group contains messages that only a Radia administrator will want to see, such as those having to do with failures in software configuration management.

Operator Group (NVD Op)

This message group contains messages that IT operators will want to see, such as server availability and performance problems.

> **Note**
>
> If the **opc_adm** user ID is used at login, messages of all levels will be displayed on the message browser, as opposed to only **Major** and **Critical** messages.
>
> This might result in an enormous list of messages, thereby limiting what is readily visible to an administrator or IT operator.

For a production OVO system, this adapter assumes that IT operators are configured to view only *Major* and *Critical* messages on their consoles. Messages, such as "RCS is running" and "RCS is responding," that are sent by monitors, would not clutter their console. These messages are sent, however, so that they clear any previous higher-level messages that might be reporting problems in the same area. That is, the adapter uses message keys and correlation options so that "RCS is running" removes any previous "RCS is not running" messages from the browser and visa versa.

6

Message Source Templates

The **Message Source Templates** window is where the message source templates are defined and modified, and where scheduled actions are configured. The Radia Adapter for HP OpenView Operations is primarily concerned with the following message sources:

- Messages issued by Novadigm applications,
- SNMP traps,
- Threshold monitors, and
- Log files.

> **Note**
>
> The separation of work between the IT Operator and Radia administrator assumes that the latter is in charge of Radia Client machines and the IT Operator handles server and network issues.

As distributed from HP:

- Monitors are not pre-assigned, and therefore, will not be automatically distributed when the SPI is installed. We recommend carefully reading the documentation associated with each monitor before distributing it to managed nodes.
- The Novadigm monitors with similar names are nearly identical, with the difference being – those with names appended with **w/Actions** provide *operator-initiated actions*.

- The operator-initiated actions can be converted to *automatic actions*. For details on how this is done refer to, *Converting Operator-Initiated Actions to Automatic Actions* on page 144.
- The *client* monitors have been designed for use with important *site servers*. These monitors tend to be resource-intensive because they trigger a Radia Client connect to the RCS, causing a refresh of the client's software catalog.

Note

Radia Client monitors are contained within the **NVDOM Ops Monitor Client** message source template.

NVDOM Novadigm Administration

This group contains the templates that Radia administrators use to verify that Radia Clients are successfully achieving their desired configuration states. The templates can also warn of errors in the Radia policy model.

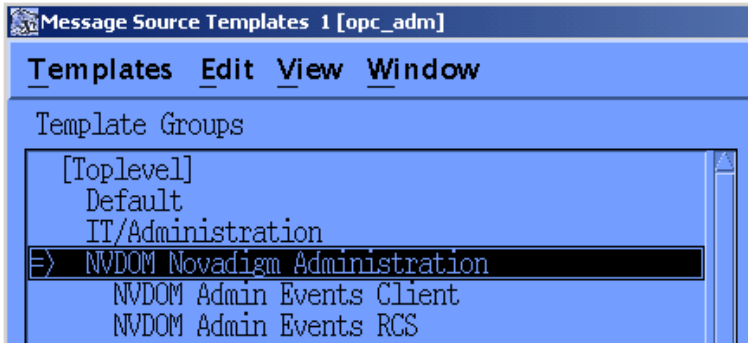


Figure 6.1 ~ The NVDOM Novadigm Administration template group.

NVDOM Admin Events Client

This sub-group contains the template that processes Radia Client APPEVENT objects. These object are used by the Radia Client to report any errors that have occurred in the deployment and installation of software and configuration. The Radia Client event component must be customized before Radia Clients send these events. For instructions on how to perform this customization, refer to *Chapter 8: The Radia Client Event Component* on page 153.

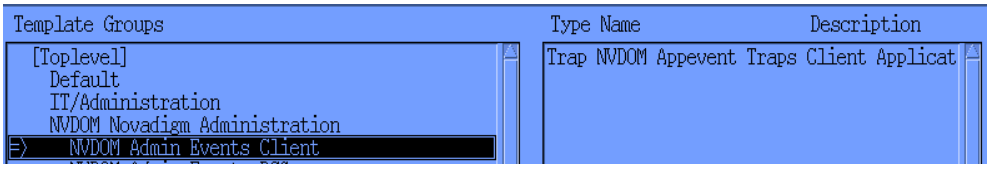


Figure 6.2 ~ The NVDOM Admin Events Client template group and associated trap template.

NVDOM Appevent Traps

Function	Controls all APPEVENT related messages. • All Radia Client failures will be sent to the message browser as messages with a severity of *major*. • All APPEVENT events that indicate the success of Radia Client operations will be sent to the message browser as a message of *normal* severity.
Notes	On a production system, we recommend that the ZSERVICE EVENTS attribute be set so that APPEVENT instances indicating a successful operation will not be sent out as traps by the RCS. This greatly reduces the number of APPEVENT-related traps that the RCS will generate. For these traps to be generated by the RCS, the APPEVENT REXX program and the RCS database have to be properly configured. See the APPEVENT setup section on page 154, for instructions on how to do this.

NVDOM Admin Events RCS

This sub-group contains templates that enable a Radia administrator to see events related to RCS log messages and SNMP traps.

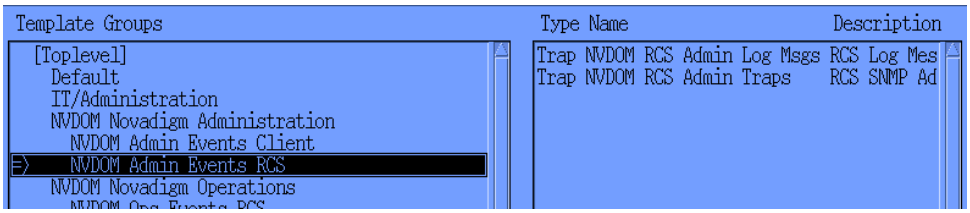


Figure 6.3 ~ The NVDOM Admin Events RCS template group and associated traps.

NVDOM RCS Admin Log Msgs

Function	This template is designed for use by a Radia administrator to report Radia Database *error* log messages to the **NVD_Adm** message group (detailed on page 123).

Notes	The log messages that this template handles are those that the RCS has sent out as enterprise-specific trap number **3000**. For the RCS to send error messages this way, the MGR_MESSAGE_CONTROL section of the edmprof file must be configured to specify for which message numbers the traps are to be sent. For example, the following MGR_MESSAGE_CONTROL section will cause message number **4010** to be sent out as an SNMP trap and a log message.
	`[MGR_MESSAGE_CONTROL]`
	`4010 = LOG,SNMPTRAP`
	The template will forward only *error* level messages to the **NVD Adm** message group. Any messages at a lower severity level will be dropped by the template (with the exception of warning message number **8107**, which is provided as an example to show how the template can be customized to select a given warning message).
	Note: We do not make any recommendation as to which trap numbers the Radia administrator might be interested in seeing.

NVDOM RCS Admin Traps

Function	This template notifies Radia administrators of certain events that might require an administrator action. It also processes SNMP traps that are sent directly by the RCS on important operational events.
Notes	The messages produced by this template notify the administrator of the following: • **Resolution failures** - shows any errors that might have occurred when policies were changed. The ZERROR-related traps (the **managerZerror trap**, **#2115**) will show resolution problems only if the edmprof file's MGR.SNMP.SNMP_ZERROR_SEVERITY parameter is set to 4 or 8. • **Notification failures** - shows which Radia Clients failed to connect to the RCS when requested to do so. **Note**: The separation of work between the IT Operator and Radia administrator assumes that the latter is in charge of Radia Client machines and the IT Operator handles servers and network issues. • **RCS Database free-space failures** - database free space is used when an administrator updates the database, so these messages are processed by the administrator template.

NVDOM Novadigm Operations

This group contains templates that IT Operators use to operate the Radia environment. A variety of operational considerations, such as, server and client readiness, server message and event notification, log handling, and free space monitoring are covered by these templates.

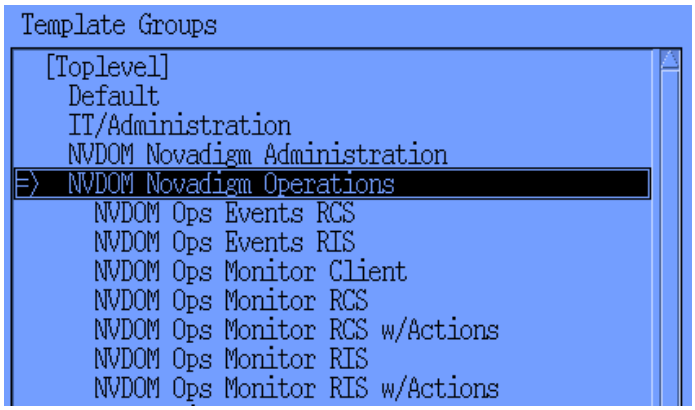


Figure 6.4 ~ The NVDOM Novadigm Operations template group.

NVDOM Ops Events RCS

This sub-group contains templates that enable an IT Operator to control the processing of RCS messages and SNMP traps.

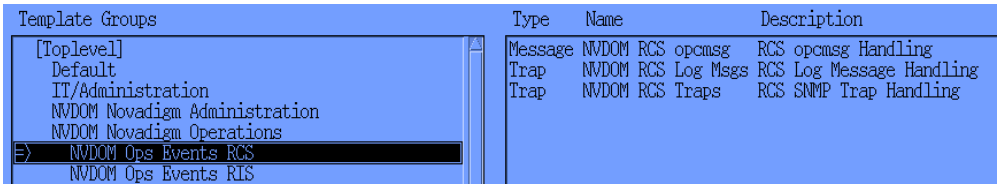


Figure 6.5 ~ The NVDOM Ops Events RCS template group and associated message and traps.

NVDOM RCS opcmsg

Function	This template notifies operators of any messages produced by the RCS-related applications provided herein.
Notes	These applications will sometimes use the **opcmsg** message interface to convey status or error events.

NVDOM RCS Log Msgs

Function	This is the template for all *critical* log messages, related to IT operations, from the Radia Database. This template also handles RCS log messages that were sent as enterprise-specific traps (trap number **3000**).
Notes	The MGR_MESSAGE_CONTROL section of the edmprof file must be configured so that the Radia Database sends critical messages as SNMP traps. In the MGR_MESSAGE_CONTROL section, specify: `1,57-59,74,429,471,474,1019,1020,8304,8308 = LOG,SNMPTRAP` **Note**: The above series of message numbers is a list of all the important messages that cover events that are not reported directly by SNMP traps Additional message numbers can be added, but the template will have to be modified if these messages are to be seen on the message browser. **Note**: There is a template in this group called **RCS SNMP Trap Handling**, which controls the handling of the major Radia Database traps. These major traps are sent directly by the Radia Database on important operational events, whereas the log messages shown above address the remaining events that are considered important to an IT Operator.

NVDOM RCS Traps

Function	This template notifies the IT Operator of critical events.
Notes	The template also processes SNMP traps that are sent directly by the RCS on important operational events.

NVDOM Ops Events RIS

This sub-group contains templates enable an IT Operator to control the processing of RIS messages.

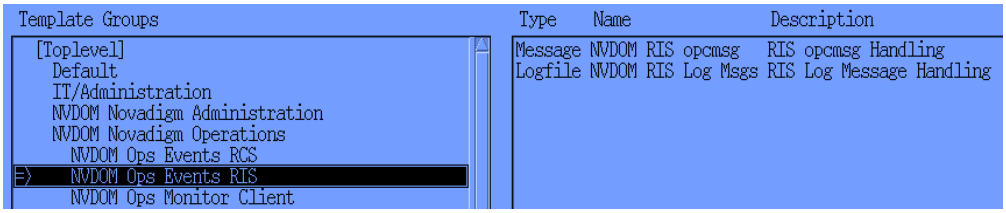


Figure 6.6 ~ The NVDOM Ops Events RIS template group and associated message and logfile.

NVDOM RIS opcmsg

Function	This template notifies operators of any messages produced by the RIS-related applications provided herein.
Notes	These applications will sometimes use the **opcmsg** message interface to convey status or error events.

NVDOM RIS Log Messages

Function	The template for all *critical* log messages from the Radia Integration Server log scraper.
Notes	• Before distributing this template to UNIX-managed nodes, the **homeDir** parameter must be set by using the **Set RIS Parms** application. • A log scraper is used because, as yet, the Radia Integration Server is not SNMP-instrumented like the Radia Database. • Only the *regular* log is scanned because the *connection* log does not contain anything of importance to an IT Operator.

UNIX Issues

The OVO log interceptor (scraper) runs under the OVO agent under user, *root*. To determine the log file name, the interceptor runs a command that is provided by the Novadigm SPI. It then runs OVO code to do the scraping. The command that determines the log file name is called

NVDOMrln.cmd, and it runs out of the OVO **cmds** directory (like **NVDOMins.sh/bat** and **NVDOMspi.cmd**). The command calls Tcl routines from the ZPATH to perform its function.

Unless *root* is also the user under which Radia was installed, **NVDOMrln.cmd** will not be able to find the Radia Integration Server paths and logs—unless the **homeDir** parameter has first been added to the confile by the **Set RIS Parms** application.

NVDOM Ops Monitor Client

This sub-group contains templates that enable an IT Operator to verify the readiness of critical Radia Clients.

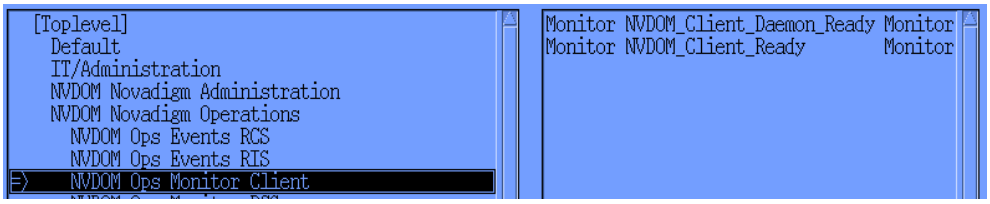


Figure 6.7 ~ The NVDOM Ops Monitor Client template group and associated monitors.

NVDOM_Client_Daemon_Ready

Function	Determines whether the Novadigm REXEC daemon, RADEXECD, is alive on the local host. The monitor runs REXEC to the Radia Client daemon.
Notes	This monitor should be used only to verify that critical Radia Clients (such as those located on important servers) are ready for distribution. The daemon is contacted at the port specified by the **notifyPort** parameter taken from the CLI confile or directly from the monitor template. The default of **3465** is used if the confile does not have a value. The value of the **-notifyPort** parameter on the monitor command will override the default and confile values. The default command run at the client is **OVOmonitor.ext**. This command is expected to fail, because it is not defined. This failure is okay because only the RADEXECD daemon is being monitored. Use the **NVDOM_Client_Ready_Monitor** monitor to fully monitor a Radia Client. Monitor returns: 0 - if the daemon is alive 2 - if it cannot connect to the daemon
Tcl Script	NVDOM_monitorClientDaemon.tcl

NVDOM_Client_Ready

Function	Determines whether the Radia Client is able to connect to the RCS and perform processing.
Notes	Because of possible performance impacts to the RCS, this monitor should be used only to verify that critical Radia Clients (such as those located on important servers) are ready for distribution. The client notify daemon is contacted at the port specified by the **clientPort** parameter. The daemon connects to the RCS and updates the software catalog. This monitor will indicate a failure if the RCS is down. Monitor returns: 0 – if Client Connect succeeded. 2 – if this process failed.
Tcl Script	NVDOM_monitorClient.tcl
Parameters	These parameters can be specified as *switch* type parameters to the monitor, or can be specified in the CLI confile. If they are not specified, the local ZMASTER object will be read to determine them. This, however, can slow down the monitor, which already does a great deal of work in updating the catalog. user – The user name the catalog will be updated for. This must be a user name that is authorized to the RCS if RCS security is in use. This is also the user name used on the notify, although this is not checked (so RADEXECD permits only programs found in IDMSYS to be run). notifyPort – The port on which the Novadigm notify daemon is contacted. The default is **3464**. clientPort – The port the client uses to contact the RCS. The default is **3465**. RCSipaddr – The IP address or name of the RCS. **Notes**: If the **user** or **RCSipaddr** parameter is missing, its value will be read from the ZMASTER object. However, for this processing to work, the IDMLIB parameter must be passed as a switch or be present in the CLI confile, unless the monitor is running under a Windows system that has kept the default value of **c:\Program File\Novadigm\Lib** for IDMLIB. The **notifyPort** and **clientPort** have default values, so they do not cause the ZMASTER read-in to occur. However, if ZMASTER is read to resolve the other two parameters, these port values will also be taken from ZMASTER.

NVDOM Ops Monitor RCS

This sub-group contains templates that enable an IT Operator to monitor RCS-related considerations, such as, disk space availability and whether the RCS is running and is responding to transactions.

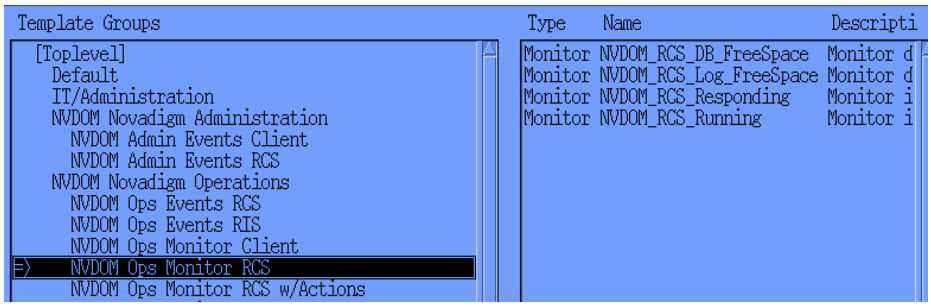


Figure 6.8 ~ The NVDOM Ops Monitor RCS template group and associated monitors.

NVDOM_RCS_DB_FreeSpace

Function	Monitors the free space on the Radia Database volume.	
Notes	Monitor Returns: Number of free kilobytes on the volume (drive in Windows) in question.	
Tcl Script	NVDOM_monitorFreeDiskSpace.tcl	
Parameters	-db **0**|1	By default, the volume checked is the volume to which the server's log files go. If this switch is passed with a value of **1**, the volume containing the server's database is monitored (the -db switch is already coded in the monitor template to distinguish this monitor from the log free space monitor).
	-logMonPath path	(Optional) When the **-db** parameter is set to **0**, the monitor checks for free space on the volume to which the path refers. This parameter is usually passed via the confile.
	-dbMonPath path	(Optional) When the **-db** parameter is set to **1**, the monitor checks for free space on the volume to which the path refers. This parameter is usually passed via the confile.
	-path p	(Optional) **p** is a path name that overrides the other parameters and causes the monitor to check for free space of the volume on which the path (**p**) resides.
	-ztype **RCS**|RIS	The -ztype switch must be set to **RCS** when *free RCS log space* is to be monitored. This parameter is already coded in the monitor template when required.

NVDOM_RCS_Log_FreeSpace

Function	Monitors the free space on the Radia Configuration Server log volume.	
Notes	Monitor Returns: Number of free kilobytes on the volume (drive in Windows) in question.	
Tcl Script	NVDOM_monitorFreeDiskSpace.tcl	
Parameters	-db **0**|1	By default, the volume checked is the volume to which the server's log files go. If this switch is passed with a value of **1**, the volume containing the server's database is monitored
	-logMonPath path	(Optional) When the **-db** parameter is set to **0**, the monitor checks for free space on the volume to which the path refers. This parameter is usually passed via the confile.
	-dbMonPath path	(Optional) When the **-db** parameter is set to **1**, the monitor checks for free space on the volume to which the path refers. This parameter is usually passed via the confile.
	-path p	(Optional) **p** is a path name that overrides the other parameters and causes the monitor to check for free space of the volume on which the path (p) resides.
	-ztype **RCS**|RIS	The -ztype switch must be set to **RCS** when *free RCS log space* is to be monitored. This parameter is already coded in the monitor template when required.

NVDOM_RCS_Responding

Function	Monitors an RCS response to a transaction.
Notes	This monitor sends the SNMP agent task a *get* command. The command is sent to port 161, unless the **snmpAgentPort** parameter has been specified in the confile. Monitor Returns: 0 - if the server is responding (or should not be running) 1 - if the server is not responding because operator stopped it 2 - if the server is not responding
Tcl Script	NVDOM_monitorResponse.tcl

NVDOM_RCS_Running

Function	Monitors whether an RCS task is running.
Notes	Monitor Returns: 0 - if the process is running 1 - if the process is not running, but was stopped by operations (this is okay) 2 - if the process is running, but was not started by operations (we want to issue warning) 3 - if the process is not running and was not stopped by operations (failed)
Tcl Script	NVDOM_monitorProcess.tcl

NVDOM Ops Monitor RCS w/Actions

This sub-group contains templates that enable an IT Operator to monitor RCS-related considerations and, when appropriate, initiate remedial actions.

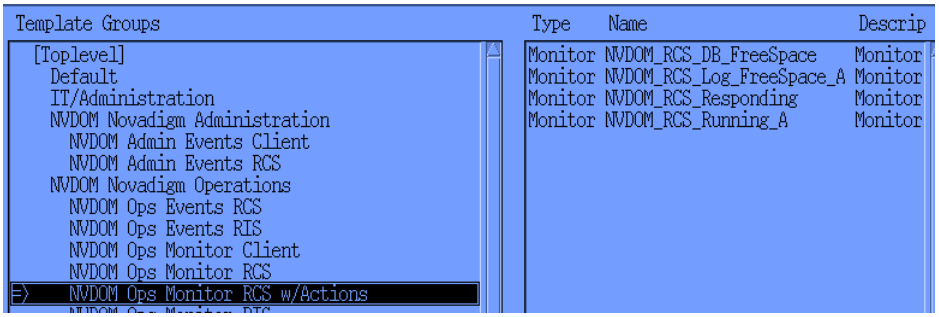


Figure 6.9 ~ The NVDOM Ops Monitor RCS w/Actions template group and associated monitors.

This template group is a counterpart of the **NVDOM Ops Monitor RCS** group, with the difference being this group has two monitors (the names of which are appended with **_A**) that provide *operator-initiated actions*.

To learn how to convert operator-initiated actions to automatic, see *Converting Operator-Initiated Actions to Automatic Actions* on page 144.

Note

Although these groups are independent of one another, assign only one group to a managed node. If both are assigned, duplicate messages might be sent to the browser.

NVDOM_RCS_Log_FreeSpace_A

Function	Monitors the free space on the RCS log volume.
Notes	For detailed notes, see the description of the **NVDOM_RCS_Log_Freespace** monitor. The messages produced by this monitor provide the operator with the following three operator-initiated actions: • On a *warning* level free-space shortage, it archives the log files. • On a *minor* level free-space shortage, it deletes all archive files. • On a *major* level free-space shortage, it deletes all log files (except the current one). **Note**: There is no automatic action associated with the *critical* level—operator intervention is requested.
Tcl Script	NVDOM_monitor.tcl
Parameters	See the **NVDOM_RCS_Log_Freespace** monitor description for detailed parameter information.

NVDOM_RCS_Running_A

Function	Monitors whether an RCS task is running.
Notes	See the **NVDOM_RCS_Running** monitor description for detailed notes. If it is determined that the RCS is not running but should be, the messages produced by this monitor provide the operator with an option to restart it.
Tcl Script	NVDOM_monitorProcess.tcl

NVDOM Ops Monitor RIS

This sub-group contains templates that enable an IT Operator to monitor RIS-related considerations, such as, disk space availability and whether it is responsive.

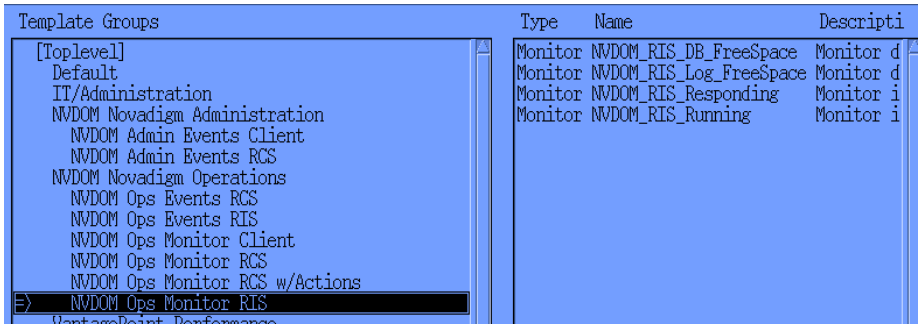


Figure 6.10 ~ The NVDOM Ops Monitor RIS template group and associated monitors.

NVDOM_RIS_DB_FreeSpace

Function	Monitors the free space on the Radia Integration Server database volume.	
Notes	The **-path** switch or **dbMonPath** parameter must be set to the RIS database directory when free space is to be monitored for this database, because this database is external to Radia and its location cannot be calculated. Monitor Returns: Number of free kilobytes on the volume (drive in Windows) in question.	
Tcl Script	NVDOM_monitor.tcl	
Parameters	-db **0**|1	By default, the volume checked is the volume to which the server's log files go. If this switch is passed with a value of **1**, the volume containing the server's database is monitored (the -db switch is already coded in the monitor template to distinguish this monitor from the log free space monitor).
	-logMonPath path	(Optional) When the **-db** parameter is set to **0**, the monitor checks for free space on the volume to which the path refers. This parameter is usually passed via the confile.
	-dbMonPath path	(Optional) When the **-db** parameter is set to **1**, the monitor checks for free space on the volume to which the path refers. This parameter is usually passed via the confile.
	-path p	(Optional) **p** is a path name that overrides the other parameters and causes the monitor to check for free space of the volume on which the path (**p**) resides.
	-ztype **RCS**|RIS	The -ztype switch must be set to **RIS** when *free RIS log space* is to be monitored. This parameter is already coded in the monitor template when required.

Important Note

In order for the **NVDOM_RIS_DB_FreeSpace** monitor to work, do either of the following.

Set the immediate parameter, **-path**.

or,

In the confile, set the permanent parameter, **dbMonPath**.

If either of these is not set, the monitor will fail and produce an error message saying it needs the database path specified.

NVDOM_RIS_Log_FreeSpace

Function	Monitors the free space on the Radia Integration Server log volume.	
Notes	Monitor returns: Number of free kilobytes on the volume (drive in Windows) in question.	
Tcl Script	NVDOM_monitor.tcl	
Parameters	-db **0**|1	By default, the volume checked is the volume to which the server's log files go. If this switch is passed with a value of **1**, the volume containing the server's database is monitored
	-logMonPath path	(Optional) When the **-db** parameter is set to **0**, the monitor checks for free space on the volume to which the path refers. This parameter is usually passed via the confile.
	-dbMonPath path	(Optional) When the **-db** parameter is set to **1**, the monitor checks for free space on the volume to which the path refers. This parameter is usually passed via the confile.
	-path p	(Optional) **p** is a path name that overrides the other parameters and causes the monitor to check for free space of the volume on which the path (p) resides.
	-ztype **RCS**|RIS	The -ztype switch must be set to **RIS** when *free RIS log space* is to be monitored. This parameter is already coded in the monitor template when required.

NVDOM_RIS_Responding

Function	Monitors a Radia Integration Server response to a transaction.
Notes	This monitor sends an http request to the Radia Integration Server. The request is sent to port 3466 unless the confile's **httpPort** parameter is used to override it, or the **-port** parameter is used as an immediate parameter to the monitor. Monitor returns: 0 - if the server is responding (or should not be running) 1 - if the server is not responding because operator stopped it 2 - if the server is not responding
Tcl Script	NVDOM_monitorResponse.tcl

NVDOM_RIS_Running

Function	Determines whether a Radia Integration Server task is running.
Notes	Monitor returns: 0 - if the process is running 1 - if the process is not running but was stopped by operations (this is okay) 2 - if the process is running but was not started by operations (we want to issue warning) 3 - if it is not running and was not stopped by operations (failed)
Tcl Script	NVDOM_monitorProcess.tcl

NVDOM Ops Monitor RIS w/Actions

This sub-group contains templates that enable an IT Operator to monitor RIS-related considerations and, when appropriate, initiate remedial actions.

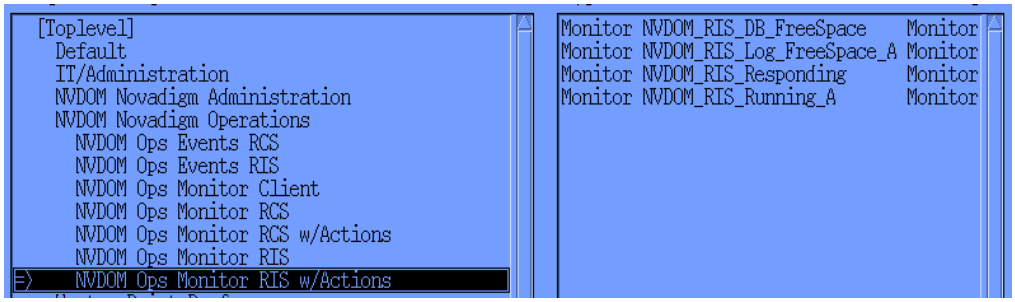


Figure 6.11 ~ The NVDOM Ops Monitor RIS w/Actions template group and associated monitors.

This template group is a counterpart of the **NVDOM Ops Monitor RIS** group, with the difference being this group has two monitors (the names of which are appended with **_A**) that provide *operator-initiated actions*.

To learn how to convert operator-initiated actions to automatic see the section *Converting Operator-Initiated Actions to Automatic Actions* on page 144.

Note

Although these groups are independent of one another, assign only one group to a managed node. If both are assigned, duplicate messages might be sent to the browser.

NVDOM_RIS_Log_FreeSpace_A

Function	Monitors the free space on the Radia Integration Server log volume.
Notes	For detailed notes, see the description of the **NVDOM_RIS_Log_Freespace** monitor on page 141. The messages produced by this monitor provide the operator with the following three operator-initiated actions: • On a *warning* level free-space shortage, it archives the log files. • On a *minor* level free-space shortage, it deletes all archive files. • On a *major* level free-space shortage, it deletes all log files (except the current one). **Note**: There is no automatic action associated with the *critical* level—operator intervention is requested.
Tcl Script	NVDOM_monitor.tcl
Parameters	See the **NVDOM_RIS_Log_Freespace** monitor description for detailed parameter information.

NVDOM_RCS_Running_A

Function	Monitors whether a Radia Integration Server task is running.
Notes	See the **NVDOM_RIS_Running** monitor description for detailed notes. If it is determined that the RCS is not running but should be, the messages produced by this monitor provide the operator with an option to restart it.
Tcl Script	NVDOM_monitorProcess.tcl

Converting Operator-Initiated Actions to Automatic Actions

None of the actions provided by monitors in this SPI are *automatic*. They all are *operator-initiated*. When a message shows up at the browser, it has a button that can be selected by the operator in order to initiate an action. These operator-initiated actions can easily be converted to automatic. This section details how this is done.

To convert operator-initiated actions to automatic

1. In the joint **Command-Operator initiated** field, locate the command that is to be converted (in this example, **NVDOMspi.cmd – NVDOM_archiveRCSlog.tcl**).

Figure 6.12 ~ The Command-Operator initiated field.

2. Copy the command and its parameters to the joint **Command-Automatic** field.

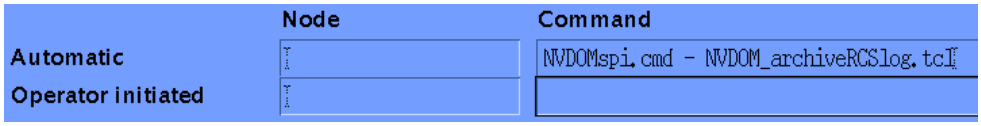


Figure 6.13 ~ The Command-Automatic field.

Note

We recommend changing the **Annotation** and **Acknowledgement** settings on the **Automatic** line to **YES**. And on the **Operator initiated** line specifying **NO**.

Monitor Troubleshooting

All of the Radia Adapter for HP OpenView Operations monitors have two troubleshooting features that are helpful in diagnosing monitor-related problems.

Testing a Monitor

The **NVD RCS/RIS/CLI SPI Setup** application groups each contain an application, **Test Monitor**, which can be customized to test a monitor by running it once as an application.

The immediate parameter, **-debug**, with a value of **100** (see Table 2.1 on page 53), will output the version numbers of the nvdkit program and the NVDOM module in use by the adapter. This parameter-value combination can be added to any of the applications that are provided with this adapter, as described below.

To test run the monitor once

1. Right-click the **Test Monitor** application and select **Customized Startup**.
2. In the **Application Parameters** field, paste in the parameters for the monitor that are contained in its **Monitor Program or MIB ID** template field.
3. Add the parameter-value combination, **-debug 100**, to those copied in, and run the application on one or more test nodes.

The output will provide full details of the monitor's processing.

> **Note**
>
> If you encounter a problem with a monitor and need assistance from HP OpenView support, send the output that was produced by running the monitor as an application when the parameter, **-debug 100**, was set.

Testing a Monitor's Template Conditions

All of the Radia Adapter for HP OpenView Operations monitors have been instrumented with a **-testfile** facility that can be used to ensure that the conditions coded within the template are working as expected.

The monitor template for any of the monitors can be modified, and a **-testfile** parameter can be added to the **Monitor Program or MIB ID** command line. For example, the **NVDOM_RCS_Log_FreeSpace** monitor's **Monitor Program or MIB ID** template field can be changed from:

```
NVDOM_monitorFreeDiskSpace.tcl -home / -monitorName NVDOM_RCS_Log_FreeSpace
```

to:

```
NVDOM_monitorFreeDiskSpace.tcl -home / -monitorName NVDOM_RCS_Log_FreeSpace
-testfile /tmp/testFree.tcl
```

The result is that the monitor no longer returns the free space on the RCS log volume. Rather, it returns the free space that is being set in the **/tmp/testFree.tcl** file.

By changing the setting in this file, one can test how the monitor will respond. For example, by setting a free space of less than 500, the monitor will issue its critical alert message. Then, to ensure that the monitor resets properly, the free space can be reset to a value larger than 2001.

The **-testfile** facility is a Tcl source file, and as such, must contain valid Tcl code. In fact, it can be changed to contain any Tcl code that an administrator wants run under the monitor. It needs to contain only a simple **set** statement that sets the **testvalue** variable to the value that the administrator wants the monitor to return.

The following is the contents of a one-line test file used to return **400**:

```
set testvalue 400
```

Tag Files

The Novadigm SPI provides monitors that enable an OVO operator to keep track of the status and activity of the RCS and Radia Integration Server services.

- The **process** monitor determines if the service process is running. If it is not (and had not been stopped by the operator), the monitor will issue a critical message on the HP OVO message browser.
- The **response** monitor runs transactions within the service, and will notify the operator/administrator if the service is not responding.

When the operator starts the RCS or RIS using the application provided by this adapter, the application creates a **tag file** in the service's base directory, and deletes the tag file only when the operator shuts down the service using the appropriate application.

If the RCS (or Radia Integration Server) is started or stopped without using the appropriate SPI applications, the tag file state will be wrong, and the monitors might send incorrect messages to the browser. To correct this, simply start (or stop) an RCS (or Radia Integration Server) that is running (or stopped) using the appropriate application. This will set the tag file to the correct state without disturbing the running (or stopped) server.

Note

If the RCS (or the Radia Integration Server) service is configured to automatically startup at system boot, then stopping and restarting the host system that the service is on does not affect the tag file state.

If the server is not configured to automatically startup at system boot, the tag file state will be wrong unless the SPI application was used to stop the service before shutting down.

By default, the tag file name used is the <*shortServiceName*> (usually ztoptask/httpd) followed by **.tag**. It is stored in **baseDir**. The name and location of the tag file can be changed by putting a fully qualified name and location in the confile under the **tagFile** keyword.

7

User Profiles

The Radia Adapter for HP OpenView Operations adds seven Novadigm **User Profiles** to the **User Profile Bank**.

- To view the profiles, click **Window** and **User Profile Bank**.

  The Novadigm user profiles are displayed, as seen in Figure 7.1.

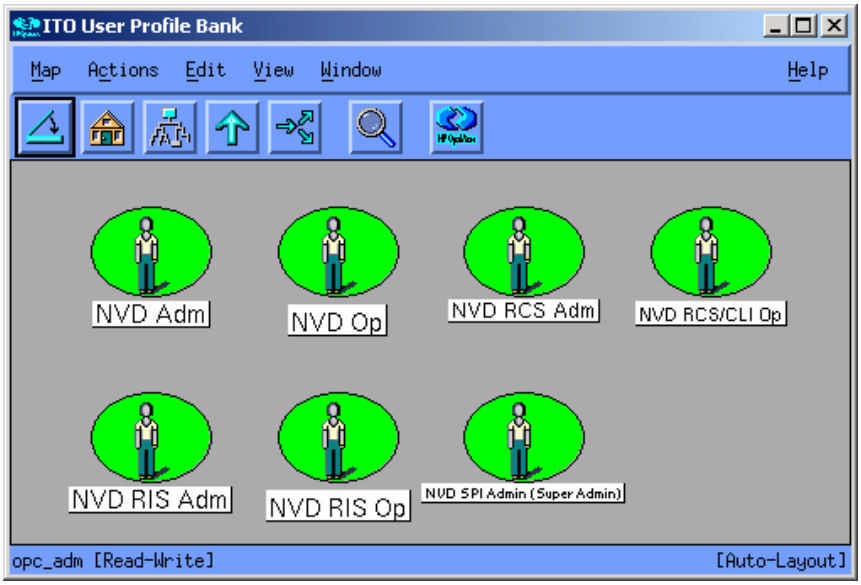


Figure 7.1 ~ The User Profile Bank.

The hierarchy of these user profiles is illustrated in Figure 7.2.

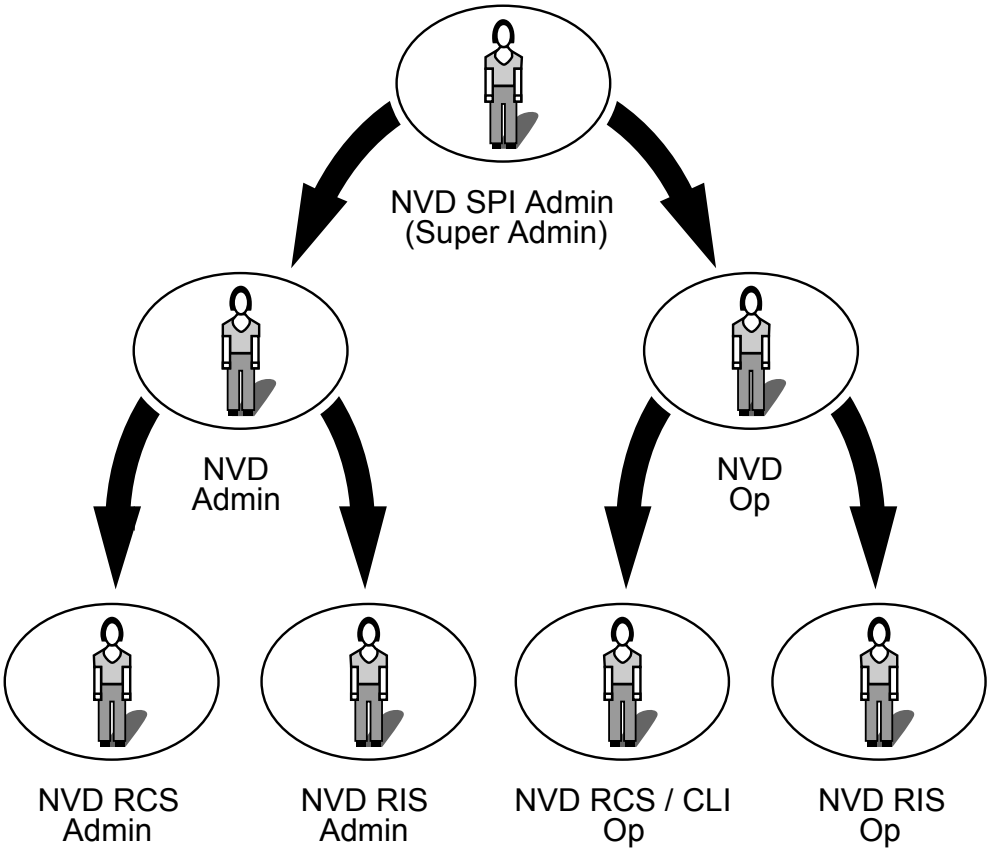


Figure 7.2 ~ The hierarchy of the Novadigm User Profile Bank.

NVD SPI Admin (Super Admin)

The Novadigm Super Admin user profile has limitless authority and capabilities to all of the Novadigm user profiles. It is the combination of the **NVD Adm** and **NVD Op** profiles.

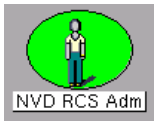
NVD Adm

The Novadigm Administrator user profile offers access to the two administrative profiles, **NVD RCS Adm** and **NVD RIS Adm**.

NVD Op

This Novadigm Operations user profile offers access to the two IT Operator profiles, **NVD RCS/CLI Op** and **NVD RIS Op**.

NVD RCS Adm



The RCS administrator user profile grants access to RCS administrative elements and sees RCS administrative messages.

NVD RCS/CLI Op

The RCS and Client Operations user profile grants access to RCS and Radia Client operational elements, and sees all RCS and Radia Client operator messages.

NVD RIS Adm

The Radia Integration Server Administrator user profile grants access to Radia Integration Server administrative elements and sees RIS administrative messages.

NVD RIS Op

The Radia Integration Server Operations user profile grants access to Radia Integration Server operational elements and sees all RIS operator messages.

8

The Radia Client Event Component

Overview

The client component of the Radia Adapter for HP OpenView Operations allows SNMP traps to be generated when certain error events occur on Radia Clients. This component is optional because most events that occur when a Radia Client is connected to the Radia Database can be seen through the RCS SNMP instrumentation.

The Radia APPEVENT Object

The Radia APPEVENT object contains all of the events that occurred during the connection, as well as those that were generated by the Radia Client since its previous connection with that Radia Database. A Radia Client that has the Radia Adapter for HP OpenView Operations installed can send an APPEVENT object to the RCS as it completes a connection.

Radia Adapter for HP OpenView Operations Radia Client Component Files

The Radia Client component consists of the files listed in Table 8.1. These files are loaded to the OVO server when the adapter is installed.

Table 8.1 ~ Radia Client Component Files

File	Description
APPEVENT	The RCS REXX script that generates the traps.
edmsnmpt.exe	The binary module that is executed by the EDMSNMPT REXX script on Windows.
edmsnmpt	The binary module that is executed by the EDMSNMPT REXX script on UNIX.

To setup the RCS REXX program

The following steps are required in order to install the Radia Client component on the RCS:

1. Run the **Setup APPEVENT REXX** application that is found in the **NVD RCS SPI Setup** group.

> **Note**
>
> For a description of the parameters associated with the **Setup APPEVENT REXX** application, refer to the application's description on page 101.

2. Use the Radia System Explorer to edit the Radia Database. (The necessary changes are outlined in the section that follows.)

Radia Database Edits

- The Radia Database must be configured so that the APPEVENT REXX program is defined in ZMETHOD. Then, an APPEVENT instance must be added to the PROCESS class so that the method runs when the object is received. Instructions for doing this are in the section *To define APPEVENT in the ZMETHOD and PROCESS classes* starting on page 155.
- The ZMASTER object must always be in storage when it runs. This is accomplished as follows:

  The PRIMARY.SYSTEM.PROCESS.APPEVENT object must be configured to have an _ALWAYS_ connection to PROFILE.&(APPEVENT.ZUSERID).ZMASTER .OBJECT.

Notes

This connection *must precede* the attribute that runs the method.

This procedure and the method parameters are detailed (on page 160) in steps 12 and 13 of the following Radia System Explorer exercise.

These bulleted points are covered in the exercise below *To define APPEVENT in the ZMETHOD and PROCESS classes*.

- Change the EVENTS and ERTYPE attributes in PRIMARY.SOFTWARE.ZSERVICE class in order to enable the Radia Client to send the APPEVENT object.

  An overview of this procedure is outlined in the section *To edit the EVENTS and ERTYPE attributes* on page 162.

To define APPEVENT in the ZMETHOD and PROCESS classes

1. From the **Start** menu, select **Programs**, then **Radia Administrator Tools**, and finally, **Radia System Explorer**.

Figure 8.1 ~ The Radia System Explorer option.

Tips

When using the System Explorer, **tree view** refers to the panel on the left side of the window, and **list view** refers to the panel on the right side of the window.

Before starting the editing process, do the following.

On the System Explorer tool bar, click **View**, and select **Options**. Then:

- On the General tab, select the check box for Show Class Names Next to Descriptions.
- On the **Instance Options** tab, under **When Displaying Instance Attributes, Show Attribute**, select **Both**.

2. When the **System Explorer** opens, double-click **PRIMARY**, then **SYSTEM**.

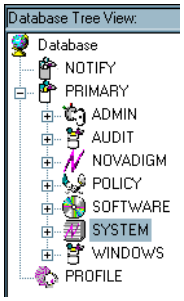


Figure 8.2 ~ The System Explorer tree view of the PRIMARY.SYSTEM domain in the database.

The *classes* of the SYSTEM domain are displayed.

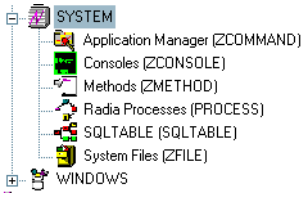


Figure 8.3 ~ The classes of the SYSTEM domain.

3. Right-click **Methods (ZMETHOD)**.
The short-cut menu (shown in Figure 8.4) opens.

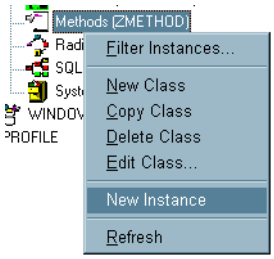


Figure 8.4 ~ The New Instance option.

4. Select **New Instance**.

The **Create Instance** dialog box opens.

5. Type a new instance name (for example, TRAP_APPEVENT) and click **OK**.

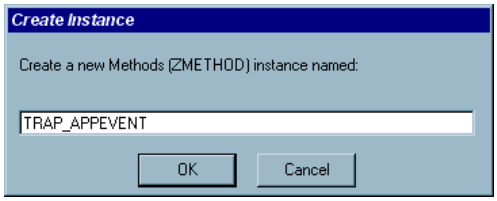


Figure 8.5 ~ The Create Instance dialog box with the new instance (TRAP_APPEVENT) specified.

The new instance (TRAP_APPEVENT) now appears in the list of instances for ZMETHOD in the list view.

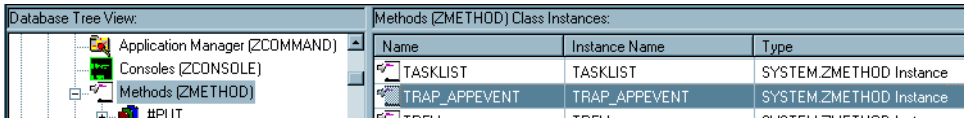


Figure 8.6 ~ The list of instances for the ZMETHOD class.

6. Right-click **TRAP_APPEVENT** in the list view and select **Edit Instance** from the shortcut menu.

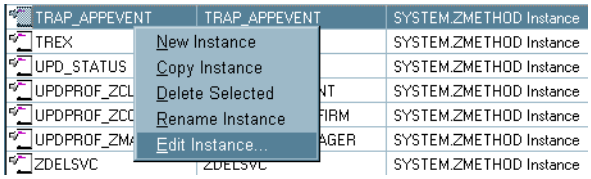


Figure 8.7 ~ The Edit Instance option of the Instance pop-up menu.

The **Editing Instance** dialog box opens.

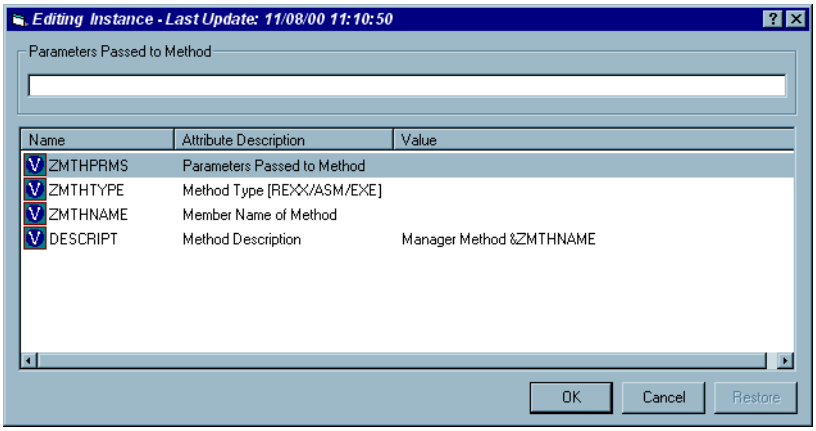


Figure 8.8 ~ The Editing Instance dialog box.

7. Specify the following parameters (as shown in Figure 8.9):

- **Parameters Passed to Method (ZMTHPRMS)**
  The IP address or host name of the system to which traps will be sent (for example, **localhost**). This should be the address of the system on which your SNMP Manager is running.
- **Method Type (ZMTHTYPE)**
  This field must be *REXX*.
- **Member Name of Method (ZMTHNAME)**
  This field must be *APPEVENT*.

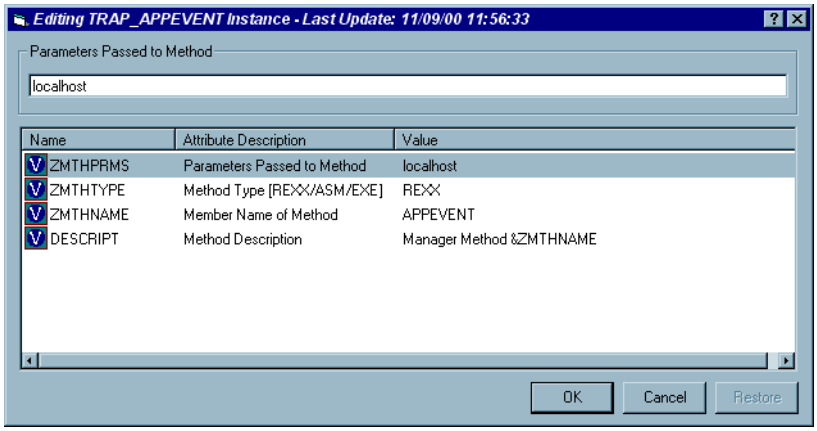


Figure 8.9 ~ The Editing Instance dialog box.

8. Click **OK**.

9. At the **Instance Edit Confirmation** message, click **Yes**.

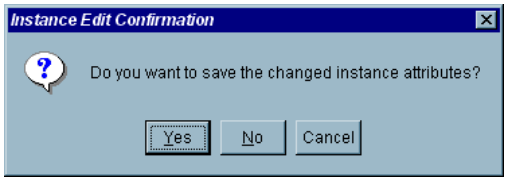


Figure 8.10 ~ The Instance Edit Confirmation message.

The list view reflects that your edits were accepted (Figure 8.11).

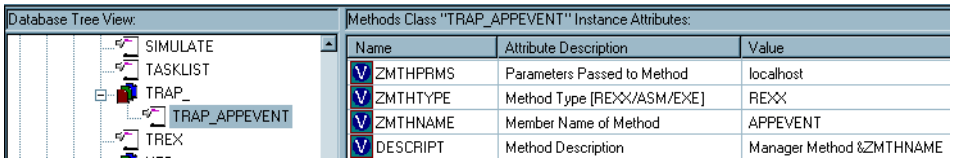


Figure 8.11 ~ The ZMETHOD.TRAP_APPEVENT instance attributes.

10. In the tree view, open the **PRIMARY.SYSTEM.PROCESS** class and double-click the **APPEVENT** instance.

The list of PROCESS.APPEVENT instance attributes is displayed in the list view.

Radia Processes Class "APPEVENT" Instance Attributes:

Name	Attribute Description	Value
ALWAYS	Method	
ALWAYS	Method	
ALWAYS	Connect To	
ALWAYS	Connect To	
ALWAYS	Method	
ALWAYS	Method	
ALWAYS	Method	
ALWAYS	Method	
ALWAYS	Method	
ALWAYS	Method	
ALWAYS	Method	
DESCRIPT	Process Description	Processing Client Request for &ZCUROBJ

Figure 8.12 ~ The PROCESS.APPEVENT instance attributes.

11. In the list view, double-click the first available **Connect To** attribute.

The **Editing *APPEVENT* Instance** dialog box opens.

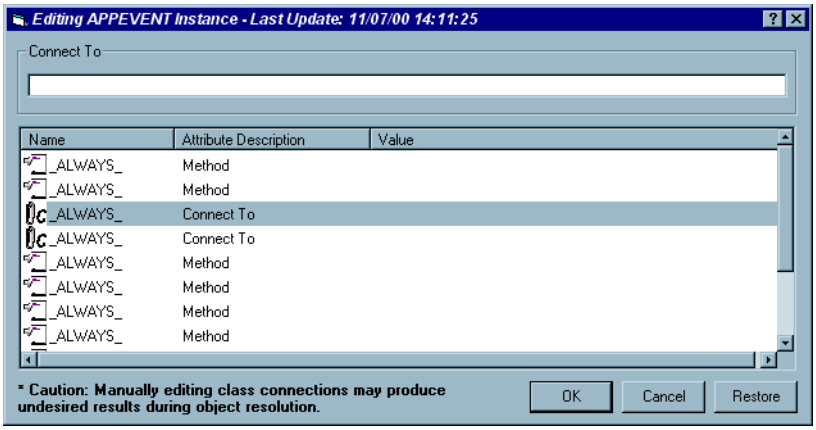


Figure 8.13 ~ The Editing APPEVENT Instance dialog box.

12. In the first **Connect To** field, specify **PROFILE.&(APPEVENT.ZUSERID) .ZMASTER.OBJECT**.

13. In the first available **Method** field *after* the **Connect To** field (in our exercise this is the third **Method** field from the top), specify **SYSTEM.ZMETHOD.TRAP_APPEVENT**.

> **Important Note**
>
> Here **TRAP_APPEVENT** reflects the ZMETHOD instance name specified in step 5 of this exercise.
>
> In step 13, be sure to specify the instance name that was specified in step **5**.

These values are shown in Figure 8.14.

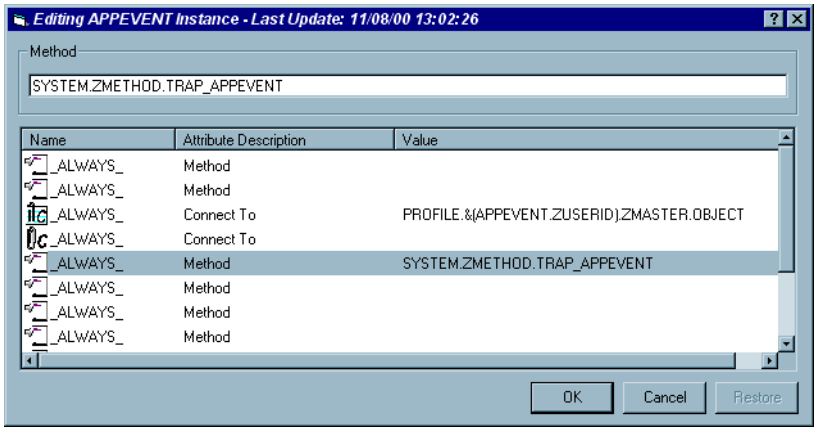


Figure 8.14 ~ The Editing APPEVENT Instance dialog box with the required Connect To and Method fields specified.

14. Accept the new attribute values by clicking **OK**.

15. When the **Instance Edit Confirmation** message opens, click **Yes**.

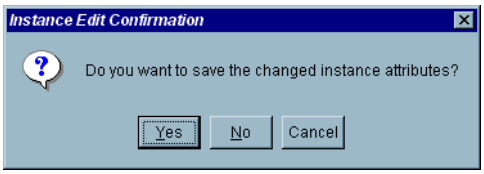


Figure 8.15 ~ The Instance Edit Confirmation dialog box.

Examine the list of instance attributes for PROCESS.APPEVENT in the list view (see Figure 8.16). The new method is present.

Radia Processes Class "APPEVENT" Instance Attributes:

Name	Attribute Description	Value
ALWAYS	Method	
ALWAYS	Method	
ALWAYS	Connect To	PROFILE.&(APPEVENT.ZUSERID).ZMASTER.OBJECT
ALWAYS	Connect To	
ALWAYS	Method	SYSTEM.ZMETHOD.TRAP_APPEVENT
ALWAYS	Method	
ALWAYS	Method	
ALWAYS	Method	
ALWAYS	Method	
ALWAYS	Method	
ALWAYS	Method	
ALWAYS	Method	
DESCRIPT	Process Description	Processing Client Request for &ZCUROBJ

Figure 8.16 ~ The list view of the System Explorer window with the PROCESS.APPEVENT instance attributes displayed.

The Radia Database has been successfully edited.

For more information on the extensive functionality of the Radia System Explorer, refer to the *System Explorer Guide*.

To edit the EVENTS and ERTYPE attributes

The *creation* and *transmission* of an application's APPEVENT object are controlled by attributes contained in that application's ZSERVICE instance. The creation is controlled by the EVENTS attribute, and the transmission is controlled by the ERTYPE attribute. This section describes how to change these attributes to cause the APPEVENT object to be sent to the RCS.

1. From the **Start** menu, select **Programs**, then **Radia Administrator Tools**, and finally, **Radia System Explorer**.

Figure 8.17 ~ The Radia System Explorer option.

Tips

When using the Radia System Explorer, **tree view** refers to the panel on the left side of the window, and **list view** refers to the panel on the right side of the window.

Before starting the editing process, do the following:

On the Radia System Explorer tool bar, click **View**, and select **Options**. Then:

- On the **General** tab, select the check box for Show Class Names Next to Descriptions.
- On the Instance **Options** tab, under **When Displaying Instance Attributes, Show Attribute**, select **Both**.

2. When the **Radia System Explorer** opens, double-click **PRIMARY**, then **SOFTWARE**.

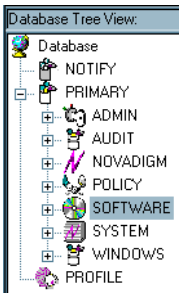


Figure 8.18 ~ The tree view of the database's PRIMARY.SOFTWARE domain.

The *classes* of the SOFTWARE *domain* are displayed.

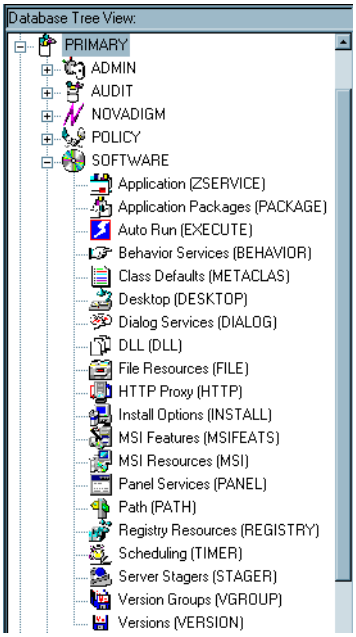


Figure 8.19 ~ The classes of the SOFTWARE domain.

3. Double-click **Application (ZSERVICE)**.
The *instances* of the ZSERVICE *class* are displayed, as shown below.

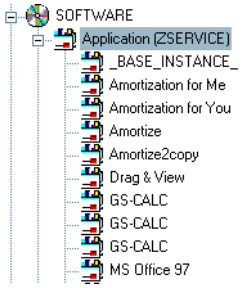


Figure 8.20 ~ The instances of the ZSERVICE class.

4. Right-click **_BASE_INSTANCE_**, and from the pop-up menu, select **Edit Instance**.

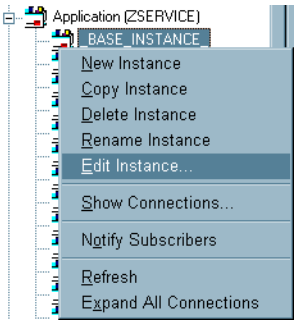


Figure 8.21 ~ The Edit Instance option.

The **Editing _BASE_INSTANCE_ Instance** dialog box opens.

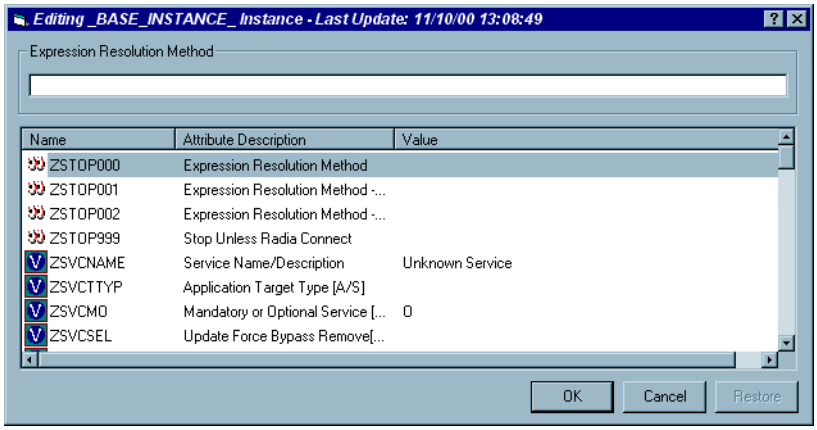


Figure 8.22 ~ The Editing _BASE_INSTANCE_ Instance dialog box.

5. Scroll down very close to the bottom and locate the **EVENTS** and **ERTYPE** attributes (denoted by a white V within a blue circle).

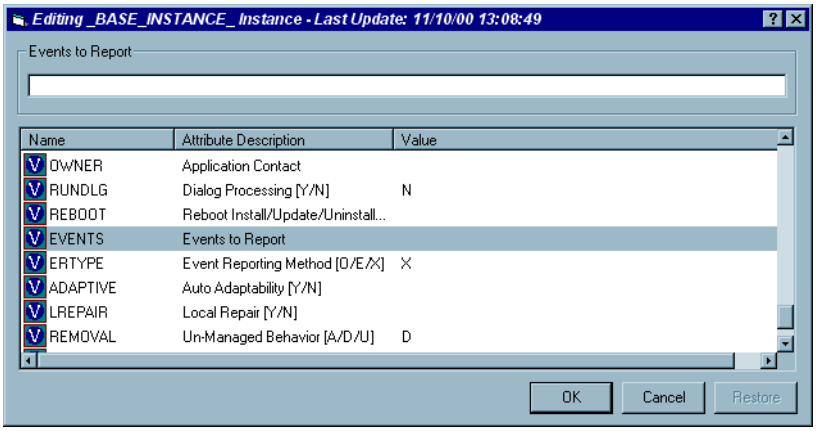


Figure 8.23 ~ The EVENTS and ERTYPE instances.

6. Click **EVENTS** and specify:

AI=B,AD=B,AU=B,AR=B,AV=B,VA=B,VD=B

This configuration will result in the reporting of all (*successful* and *unsuccessful*) events.

Notes

In the example above, the keywords *AI*, *AD*, and so forth, are set to the value of **B**. Here, **B** represents **BOTH** (successful and unsuccessful) and should be used in a testing environment. In a production environment, **B** should be replaced by **F**, which represents a **FAILURE**.

Also covered in the section are instructions for configuring your reporting parameters using the Radia *New Application Wizard*.

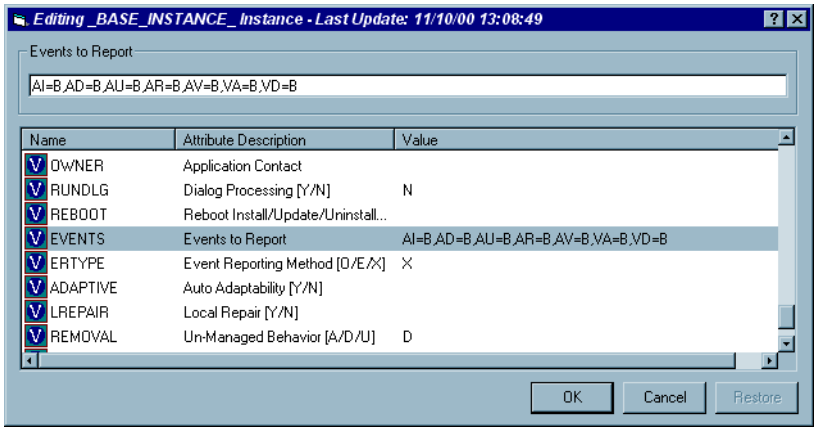


Figure 8.24 ~ The EVENTS instance specified.

7. Click **ERTYPE** and specify **O**.

This setting instructs the Radia Client to send an APPEVENT object to the RCS.

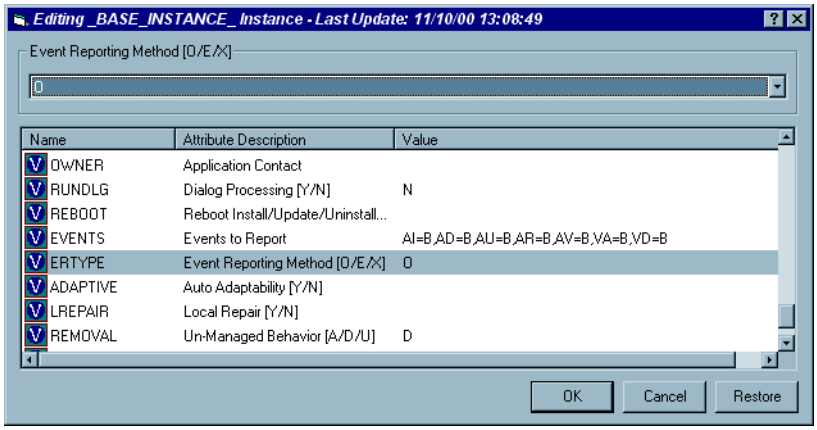


Figure 8.25 ~ The ERTYPE instance specified.

8. Click **OK**.

9. At the **Instance Edit Confirmation** message, click **Yes**.

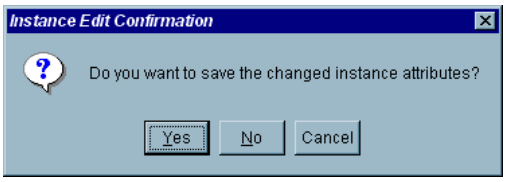


Figure 8.26 ~ The Instance Edit Confirmation message.

The list of SOFTWARE.ZSERVICE._BASE_INSTANCE_ instance attributes is displayed in the list view.

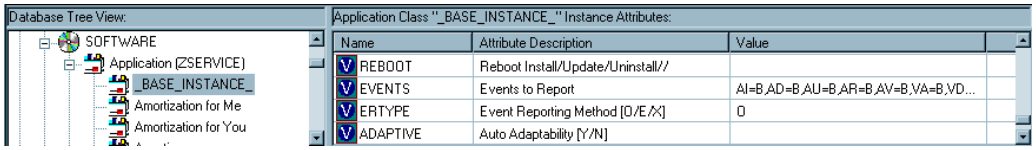


Figure 8.27 ~ The Instance Edit Confirmation message.

Since these values were added to the _BASE_INSTANCE_ of the ZSERVICE class, they will automatically be added to all other instances of the ZSERVICE class.

10. On the Radia System Explorer toolbar select **View**, and then **Refresh**.

11. Select a different instance of the ZSERVICE class (for example, in Figure 8.28, **Drag & View**), and open it to ensure that the value of the EVENTS and ERTYPE attributes are as you specified them.

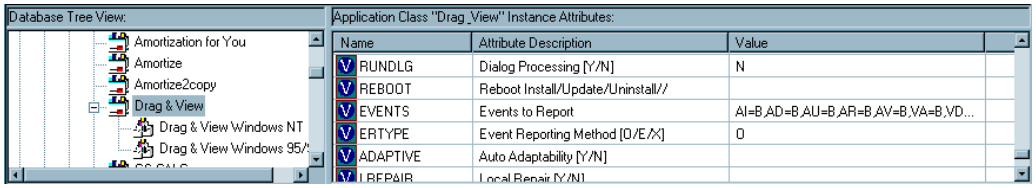


Figure 8.28 ~The Instance Edit Confirmation message.

The EVENTS and ERTYPE attributes of the ZSERVICE class have been successfully edited.

Important Notes

Because these values were edited in the _BASE_INSTANCE_, any instances that are added to the class henceforth will inherit them.

As seen in the exercise, resolution will cause the other instances of the class to automatically adopt these values.

The only two circumstances under which an instance will not be in synchronization with the others are:

- if the value of the variable is changed explicitly on a given instance; and
- if the instance had the value defined (with a different value) prior to the edit on the _BASE_INSTANCE_.

Radia Client Component Operational Considerations

In order to prevent flooding the system, the only client events that will be processed are those that have occurred within 120 minutes (2 hours) of the RCS receiving the APPEVENT object—all others will be ignored. This time interval is specified by the **-tlimit** parameter when **Setup APPEVENT REXX** is run. It can be changed if a different interval is required.

An SNMP Primer

This appendix describes the *Radia Adapter for SNMP*, the implementation of *Simple Network Management Protocol* (SNMP) in an Radia Configuration Server (RCS) environment. The goal of SNMP functionality within the RCS is to allow users to easily monitor and operate RCSs in their OVO environments.

The Two Aspects of SNMP

Simple Network Management Protocol (SNMP), has two main aspects of functionality:

- **SNMP Agents**
  programs that run on a platform and process SNMP requests.
- **SNMP Traps**
  messages that are sent to an SNMP Manager when events occur.

In addition to these areas, this appendix will cover the implementation of SNMP, its logs, and the commands associated with it.

SNMP Agents

An SNMP agent is a program that runs on a platform and processes SNMP command requests. These requests can do only one of two things: either *query* or *modify* one or more variables. The variables are integers or character strings, and what they represent is determined by how the agent is coded. The variables have been defined in a special file called the *Management*

Information Base (MIB), where each variable is assigned a unique name. This unique name is an *object ID* (OID). Below is an example of some of the variables supported in the Radia Database:

- An OID of 1.3.6.1.4.1.2133.1.5.25 represents a variable that contains the number of tasks that are currently active on the Radia Database. In this case, the variable corresponds directly to a real variable in the Radia Database, not a contrived one, as other variables are (see the third bullet in this section). The MIB does *not* allow *set* requests against this variable.

  Note

  The leading part of the OID, **1.3.6.1.4.1** has been dictated by the SNMP standards committee. In this example, 2133 references Novadigm.

  Cisco has all of its variables under **1.3.6.1.4.1.9**.

- An OID of 1.3.6.1.4.1.2133.1.5.5 represents the current state of the Radia Database.
  - If queried and this variable returns **1**, the Radia Database is running.
  - If queried and it returns **0**, the Radia Database is being shutdown.

  When the Radia Database is not running, there will be no response (the request times out), because the agent that would normally reply to the query runs as a subtask of the Radia Database. The Radia Database can be shut down by setting this variable to 0. The edmprof file contains parameters where SNMP request authority can be designated.
- An OID of 1.3.6.1.4.1.2133.1.5.30 represents an event-marker variable. In the Radia Database, there is no real variable that corresponds with this. In other words, this is a contrived variable, which can be set to a character string. When set, the string's contents are logged to the Radia Database log. (On Windows platforms, it will also be logged to the Windows Event log.) The string is also echoed back to the SNMP Manager as a trap. Therefore, by setting this variable, a marker string can be placed in the Radia Database log and Windows Event log. This variable is primarily used to test the SNMP agent and the SNMP logging component.

Usage Considerations

Running Two SNMP Agents on One Machine

Why would there be two SNMP agents running on one machine?

Consider a Radia Database running on a Windows machine. You want the Radia Database's SNMP agent to run in order to manage the Radia Database, and the Windows agent to run in order to manage the operating system. The Windows agent supports querying and modifying variables having to do with disk space, its TCP/IP stack, and other services running under Windows.

There are two solutions to this problem. The first is to run the Radia Database's SNMP agent on a port other than 161. This can be done by specifying the alternate port number in the edmprof file.

> **Note**
>
> The SNMP Managers must then be configured to use this alternate port when sending requests to the Radia Database agent.

A second and more general way is available because the Windows agent is extensible. Run part of our agent code as a DLL under the Windows agent and another part under the Radia Database (where most of the variables are). The two parts communicate over a named pipe.

What happens if a machine tries to run two SNMP agents simultaneously?

By convention, an SNMP agent always receives its requests (*get* and *set* commands) on IP port 161. Therefore, if two agents are run simultaneously, the first one to open port 161 is accepted, the other fails to open the port, and shuts down or hangs.

SNMP Traps

When a significant event occurs, the component of the system that is aware of the event can send out an SNMP message (called a *trap*), to an SNMP Manager that is listening for traps. By default, traps are sent to port 162 of a specified IP address. SNMP Managers, such as the HP *Network Node Manager* (NNM) and *Microsoft Operations Manager* (MOM), process SNMP trap messages that they receive on this port.

Agents, such as the Novadigm agent or the Windows agent, never receive traps, although they can send them to log certain events. Any process can send traps, not just agents. However, the sender must be configured with the correct IP address. Since the trapping code is usually only configured with the address of one SNMP Manager, SNMP Managers often have the ability to forward traps to other SNMP Managers.

Trap messages contain a general trap number, and possibly an enterprise-specific trap number. They might also contain variables that help describe the event they represent. These variables are typically contrived and do not correspond to a real variable. Remember, the variables, the trap

enterprise-specific trap number, and the event it represents should be completely documented in the MIB.

Upon receiving a trap, an SNMP Manager usually formats the trap's contents into a log and can take pre-configured, automated actions. It can also be configured to alert an operator of the event by sending pop-up screen messages, color changes on network maps, and so on.

Radia Configuration Server and SNMP

There are three basic parts to the Radia Database SNMP implementation:

- The Radia Database can run an SNMP agent task that allows SNMP commands to query and modify variables within it. Under Windows, this agent can be run in one of two ways.
  - It can assume the responsibility of a full SNMP agent and run without the main Windows agent running.

    UNIX Note

    On UNIX, the Radia Database SNMP agent is always run as a full SNMP agent.

  - It can run as an extension to the main Windows Agent (**snmp**).
- The Radia Database message logger (ZLOGMGR) reads the specifications in the MGR_MESSAGE_CONTROL section of the edmprof file and suppresses the messages, or routes them to a number of available destinations. SNMP traps are one of these destinations. Therefore, any message, or group of messages, can be sent out as SNMP traps. (This feature is also referred to as the *SNMP logger*.)

  All traps that correspond to a log message are issued as an enterprise-specific trap, type 3000. The trap contains a variable that specifies the actual message number, and the formatted message is contained in a trap variable.
- The RCS issues SNMP traps directly when certain critical events occur. For example, it issues traps when it is starting, once it has initialized, and when it is shutting down. It also issues traps when certain errors occur. Again, these are documented in the MIB.

SNMP Implementation in the Radia Database

Running the Radia Database SNMP Agent

The Novadigm SNMP agent runs as a separate thread under ZTOPTASK in Windows. Under UNIX, it runs as a separate process. To run the agent, the following line must be included in the MGR_ATTACH_LIST section of the edmprof file:

```
CMD_LINE=(zsnmpmgr) RESTART = YES
```

> **Note**
>
> The SNMP logger and direct trapping will run, even if the agent is not running.

If the Radia Database SNMP agent support is not being used, then for performance reasons we recommend that the agent not be run.

When the agent is started, the parameters contained in the MGR_SNMP section of the edmprof file are used to set the various options that the agent supports.

The Novadigm SNMP Agent

The Novadigm SNMP agent will run as an extension to the Windows SNMP agent (called **snmp**), or as a full agent, depending on the value that has been set for the RUN_AS_EXTENSION parameter in the MGR_SNMP section of the edmprof file.

The Novadigm SNMP-Windows Extension Agent

When RUN_AS_EXTENSION=YES, a DLL called EDMSNMPX.DLL is loaded by **snmp** and is run to process requests that pertain to the Novadigm part of the MIB. The CMD_LINE for ZSNMPMGR must still be coded in the MGR_ATTACH_LIST section of the edmprof file, since EDMSNMPX.DLL cannot process SNMP requests alone. (It simply checks them against the Novadigm MIB and sends them via a named pipe to ZSNMPMGR in the RCS address space where they can be processed.)

When run in this mode, Windows **snmp** owns port 161 and runs as a service under Windows.

> **Note**
>
> EDMSNMPX.DLL and the agent code (ZSNMPMGR) contain a local copy of the Novadigm MIB. The version number of the MIB is compared on each request and if there is a version mismatch, the request is not processed.

Conceptual views of these agent options are presented in Figures A.1 (on page 179) and A.2 (on page 180).

Loading the Extension DLL

When Windows **snmp** is started, it scans the registry to determine if there are any extension DLLs it should load. In order to load EDMSNMPX.DLL, the following two keys are required in the registry:

```
HKEY_LOCAL_MACHINE\SOFTWARE\NOVADIGM\EdmSnmpX\CurrentVersion.
```

Under this key, there should be a variable, **PathName**, the value of which is the full path name to the Novadigm DLL. This should resemble:

```
C:\EDM…\EDMSNMPX.DLL.
```

Make sure that the DLL file can be found in the indicated path.

The second required key is:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SNMP\Parameters\Extensi
onAgents.
```

Under this key, there should be a variable, **EdmSnmpX**, with a value of:

```
SOFTWARE\Novadigm\EdmSnmpX\CurrentVersion
```

The second key is, in essence, a pointer to the Novadigm-specific key that contains the full path name to the DLL. The RCS installation programs will setup the above registry entries for you.

Note

If the EDMSNMPX.DLL on a running system is changed, stop and restart SNMP.EXE via the Windows **Services** shortcut in the **Control Panel** in order for the change to take effect.

Authenticating Requests to the Extension Agent

SNMP requests are authenticated in one of two ways.

- SNMP defines a community name, which is, effectively, a password that is included in every request sent by an SNMP Manager. The agent compares this to the community name that it is using and processes the requests only if the names match. Unfortunately, the community name appears as clear text in the IP message.
- The agent compares the IP address from which it received the request against a list of addresses that have been determined to be those of valid SNMP Managers.

When running as an SNMP extension, both of these authentication mechanisms are performed by Windows **snmp**. The Radia Database's SNMP agent code then does not need to authenticate the requests that it receives from the Windows service. The community name and valid IP addresses from which the **snmp** accepts requests are configured on the Windows panel.

- On a Windows NT 4.0 system, these values can be found by navigating as follows:
  Right-click **My Network**, and select **Properties**.

  On the **Services** tab, right-click **SNMP Service**, and select **Properties**.
- On a Windows 2000 system, to configure the values:
  Click **Start**, then **Settings**, and **Control Panel**.

  Click **Network**, then the **Services** tab, **SNMP Service**, **Properties**, and **Security**.

Figure A.1 on page 179 presents a conceptual view of the Radia SNMP agent running as an extension to the Windows SNMP agent.

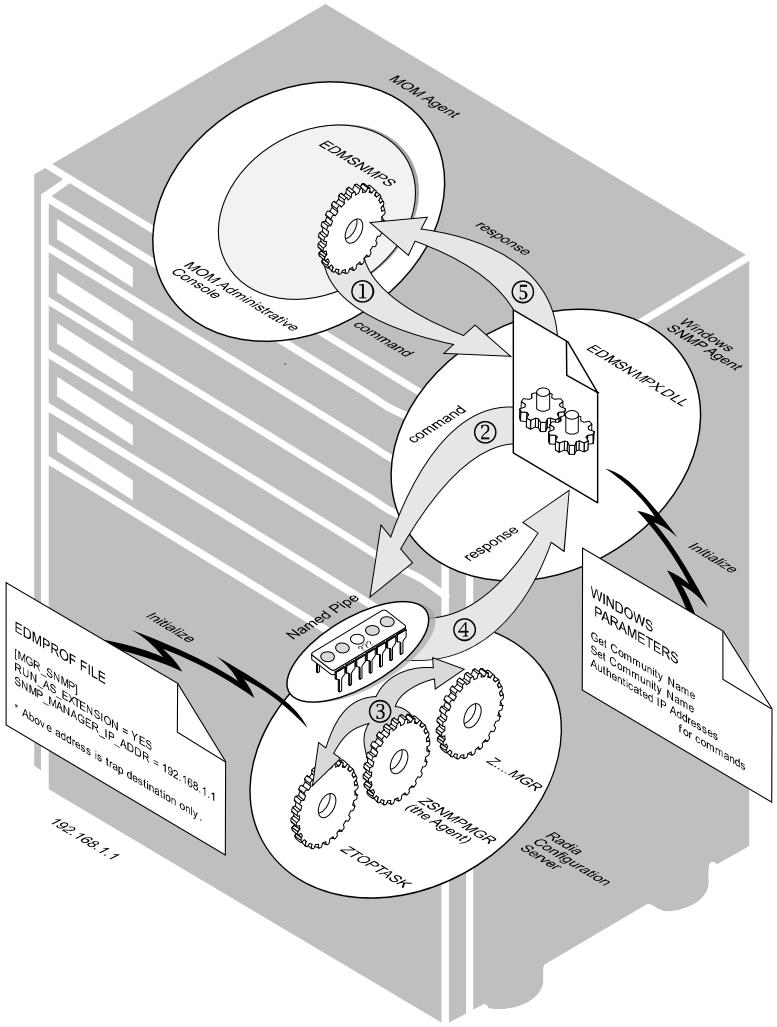


Figure A.1 ~ The Radia SNMP agent running as an extension to the Windows SNMP agent.

The Novadigm SNMP-Windows Full Agent

When RUN_AS_EXTENSION=NO, the agent code running in the Radia Database address space opens port 161 directly and processes any requests it receives from there. If a request is for non-Novadigm variables, it is simply discarded. EDMSNMPX.DLL is not used in this case. A port, other than 161, can be used by specifying the port number in the edmprof file.

Figure A.2 below shows a conceptual view of the Radia SNMP agent running as full SNMP agent.

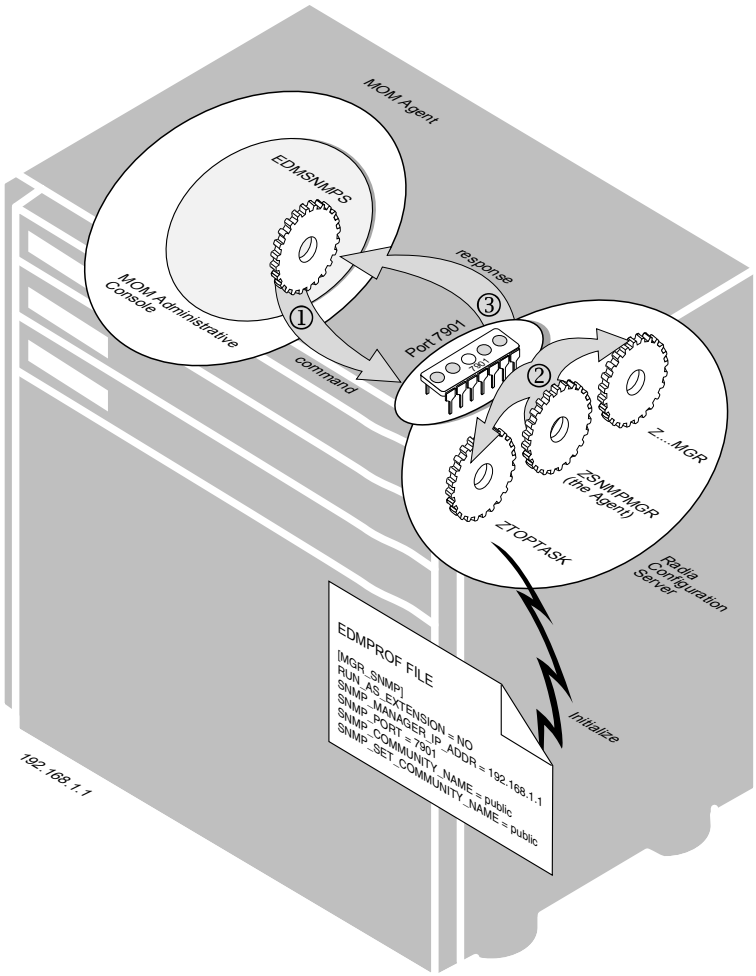


Figure A.2 ~ The Radia SNMP agent running as the full SNMP agent.

Authenticating Requests to the Full Agent

When running as a full agent, the Radia Database agent code authenticates the requests it receives in a fashion similar to how SNMP.EXE authenticates requests when running as an extension. The main difference of how the authentication is performed is that when running as a full agent, the community name and valid IP addresses used are specified in the edmprof file, not on a Windows panel.

The Novadigm MIB

Obtaining the Novadigm MIB

A machine-readable copy of the Novadigm MIB file can be found on this adapter's distribution material. An easy-to-read, formatted copy of the Novadigm MIB is available on the HP OpenView support web site.

Installing the Novadigm MIB

The first step in implementing SNMP in your environment is to install the Novadigm MIB in the Network Management Systems tool. Refer to the documentation from your software vendor that discusses how to install an MIB.

Understanding the Novadigm MIB

This section presents an overview of the Novadigm MIB. Near the top of the MIB file, there are lines similar to those in Table A.1 below.

Table A.1 ~ MIB File Parameters

Keyword	Definition
novadigm	OBJECT IDENTIFIER **::=** { enterprises 2133}
manager	OBJECT IDENTIFIER **::=** { novadigm 1 }
stager	OBJECT IDENTIFIER **::=** { novadigm 5 }
client	OBJECT IDENTIFIER **::=** { novadigm 10 }
mgr-agent	OBJECT IDENTIFIER **::=** { manager 1 }
mgr-trap	OBJECT IDENTIFIER **::=** { manager 5 }
-- clnt-agent	OBJECT IDENTIFIER **::=** { client 1 }
clnt-trap	OBJECT IDENTIFIER **::=** { client 5 }
-- stgr-agent	OBJECT IDENTIFIER **::=** { stager 1 }
stgr-trap	OBJECT IDENTIFIER **::=** { stager 5 }

Lines that begin with **--** are comments. The keyword, *enterprises*, has been predefined as **1.3.6.1.4.1** in the import section of the MIB (which is not shown here). Every line with **::=**, defines a new OID. The line:

```
novadigm OBJECT IDENTIFIER ::= { enterprises 2133}
```

can be read as saying, "Define the keyword *novadigm* to be the concatenation of the enterprise OID with *2133*". After this line, *novadigm* has OID 1**.3.6.1.4.1.2133** and no other keyword can be assigned that OID. From this, it's clear that *mgr-agent* will be assigned OID **1.3.6.1.4.1.2133.1.1**, and *mgr-trap* will be assigned OID **1.3.6.1.4.1.2133.1.5**.

Later in the MIB file, there are the following lines:

Table A.2 ~ MIB File Parameters (mibVersion)

Variable	Definition
mibVersion	OBJECT-TYPE
	SYNTAX INTEGER (0..5000)
	ACCESS read-only
	STATUS mandatory
	DESCRIPTION "The Radia Configuration Server Database's agent contains a built in copy of the Novadigm MIB. This variable contains the internal version number of that MIB. The Novadigm MIB you have compiled into your SNMP Manager (from which this description came) is at version number 105."
	::= {mgr-agent 1}

This defines a variable (because of OBJECT-TYPE) called **mibVersion**. The variable is read-only and supports only integer values from 0 to 5000. Once the MIB has been compiled into an SNMP Manager, the description will automatically be presented (*STATUS* = mandatory) whenever a *describe* or *help* function is performed on that SNMP Manager. The **::=** line causes the OID of 1.3.6.1.4.1.2133.1.1.1 to be assigned to this variable. This completes defining an SNMP variable.

Note

If the variable is to be used in a trap, then on the **::=** line, simply assign it a unique number under the *mgr-trap* group instead of under the *mgr-agent* group.

The following is an example of the lines that are used to define a trap:

Table A.3 ~ MIB File – Lines Used To Define A Trap

Trap	Definition
managerStarting	TRAP-TYPE
	ENTERPRISE novadigm
	VARIABLES { managerId }
	DESCRIPTION "A Radia Configuration Server Database is starting."
	::= 2000

The above lines define Novadigm enterprise-specific trap number 2000, which is sent out when the Radia Database is starting. The trap will contain a single variable, *managerId*, which must have been defined previously. This variable is defined in a manner similar to mibVersion.

SNMP–Related Parameters in the EDMPROF File

SNMP-related parameters are contained in several sections of the edmprof file. Those that are directly related to Radia Adapter for OVO are discussed earlier in this manual, in the following locations.

- The MGR_ATTACH_LIST section starts on page 32.
- The MGR_DIAGNOSTIC section starts on page 32.
- The MGR_LOG section starts on page 33.
- The MGR_MESSAGE_CONTROL section starts on page 34.
- The MGR_SNMP section starts on page 35.

Issuing SNMP Commands

This section documents the HP-provided command-line utilities that can be used to issue *get* and *set* commands.

Using the SNMP GET and SET Commands

The HP-provided SNMP *get* and *set* commands for Windows are **edmsnmpg** and **edmsnmps**, respectively. For UNIX, the commands are *edmsnmpg* and *edmsnmps*. A copy of these is included with this adapter.

Note

These commands are used within the applications and monitors that are provided with this adapter, and are documented below. However, it is not necessary to learn the details of the commands in order to operate the adapter.

- To display usage:

  ```
  edmsnmpg
  ```

  The resulting display will resemble:

  ```
  Hostname/IP address of agent is required.
  At least one positional parameter (OID) is required.
  usage: edmsnmpg [options] oid ...
      Options:
      -h hostname     remote hostname or IP address (required)
      -p port         remote port (default is 161)
      -c community    community string (default is public)
      -t timeout      retransmission timeout in seconds (default is 3)
      -r retries      maximum retransmission attempts (default is 3)
      -i requested    request identifier (default is 0)
      -n              issue GetNext command instead of Get (implies -o)
      -w              perform Walk operation (implies -o)
      -o              print OID along with value in output
      +o              do not print OID along with value in output
      -v verbosity    message level (1=errors, 2=errors+warnings, 3=trace all)
      -d debugFile    optional file for message output
      -s snmpVersion  SNMP version of request (1 or 2, default is 1)
  ```

Table A.4 ~ MIB File – EDMSNMPG Options

Parameter	Description
-h hostname	remote host name or IP address (required)
-p port	remote port (default is port 161)
-c community	community string (default is public)
-t timeout	retransmission time in seconds (default is 3)
-r retries	maximum retransmission attempts (default is 3)
-I requestID	request identifier (default is 0)
-n	issue GetNext command instead of Get (implies -o)
-w	perform Walk operation (implies -o)
-o	print OID along with value in output
+o	do not print OID along with value in output
-v verbosity	message level (1=errors, 2=errors+warnings, 3=trace all)
-d debugFile	optional file for message output
-s snmpVersion	SNMP version of request (1 or 2, default is 1)

The first two lines of the output are error messages that report that two required parameters, *hostname* and *oid*, were not specified. All other parameters are optional. If the SNMP agent to which the command is being sent is not listening on port 161, then the *-p* switch and a value for it must be specified. The command will time out if the agent does not respond within the allowed time interval, and/or if it is sent to the wrong port.

- To query an SNMP agent variable:

```
edmsnmpg  -h 204.7.82.202  1.3.6.1.4.1.2133.1.1.1.0
```

Here 1.3.6.1.4.1.2133.1.1.1.0 is (almost) the OID for the mibVersion variable as defined in **novadigm.mib**. The IP address of 204.7.82.202 is where the agent is located. The output for the command should resemble the following:

```
108
```

The returned value tells us that the MIB is at version 108.

Note a peculiarity of how OIDs are specified to the command. We have added .0 at the end of what our MIB tells us is the OID for the mibVersion variable. This is done by convention to indicate that the variable is a scalar. If the variable is an array, the last identifier in the OID would be used to reference the entry we wanted.

If the trailing .0 is omitted, thereby requesting an OID for a variable that doesn't exist, the resulting error message should look like this:

```
Received response with error index = 1, status = 2 = noSuchName
```

Here the errorStatus of **2** indicates that the variable does not exist, and the errorIndex of **1** indicates that the error was on the first OID requested. (The possible values of an errorStatus variable are discussed later in this document.) The errorIndex is used because it is possible to request several OID in one *get* command, as the next example shows.

- To query several SNMP agent variables in one command:

```
edmsnmpg  -h 204.7.82.202  nvd.1.1.1.0 nvd.1.1.25.0 nvd.1.1.45.0
```

Here the three nvd prefixes in the OID are used as a shortcut to specify 1.3.6.1.4.1.2133. This is one of the features supported by our commands. The above command will generate a single SNMP *get* message that contains the request for all three variables. The output for the command should appear something like:

```
1
8
108
```

This tells us that the MIB version is 108, there are currently eight tasks active in the RCS and that all logon types are enabled.

Note

Notice that the variables are returned in an inverted order, this is a particularity of the RCS agent, not all agents will do this.

Consult the MIB for more details about what these variables mean and what operations are allowed on them. The above output might be hard to interpret since the values of the variables we requested are printed singly, one per line. By using the *-o* switch, it's possible to request that the command print the OID ahead of each variable's value. The next example shows this.

- To query the variables and print the OID in the output:

```
edmsnmpg  -h 204.7.82.202 -o nvd.1.1.1.0 nvd.1.1.25.0 nvd.1.1.45.0
.1.3.6.1.4.1.2133.1.1.45.0   1
.1.3.6.1.4.1.2133.1.1.25.0   8
.1.3.6.1.4.1.2133.1.1.1.0  108
```

In the MIB, note that the variable, allLogons (nvd.1.1.45.0), has read/write access. This means that the *set* command with a value of **0** can be used to disable all logons to the RCS. Before using the Novadigm SNMP *set* command, we will first issue the command without parameters in order to obtain the command's usage information.

- To obtain usage for the edmsnmps command:

```
edmsnmps
```

The resulting display will resemble:

```
Hostname/IP address of agent is required.
At least three positional parameters (OID, type, value) are required.
Usage:
edmsnmps [options] oid type value ...
    Options:
    -h hostname    remote host name or IP address (required)
    -p port        remote port (default is 161)
```

```
-c community   community string (default is public)
-t timeout     retransmission timeout in seconds (default is 3)
-r retries     maximum retransmission attempts (default is 3)
-i requested   request identifier (default is 0)
-o             print OID along with value in output
+o             do not print OID along with value in output
-v verbosity   message level (1=errors, 2=errors+warnings, 3=trace all)
-d debugFile   optional file for message output
-s snmpVersion SNMP version of request (1 or 2, default is 1)
   Supported values for type:
       integer, octetstring, objectidentifier, null (or use: int, str,
       oid, null)
```

Table A.5 ~ MIB File – EDMSNMPS Options

Parameter	Description
-h hostname	remote host name or IP address (required)
-p port	remote port (default is port 161)
-c community	community string (default is public)
-t timeout	retransmission time in seconds (default is 3)
-r retries	maximum retransmission attempts (default is 3)
-I requestID	request identifier (default is 0)
-o	print OID along with value in output
+o	do not print OID along with value in output
-v verbosity	message level (1=errors, 2=errors+warnings, 3=trace all)
-d debugFile	optional file for message output
-s snmpVersion	SNMP version of request (1 or 2, default is 1)

We are now ready to use the *set* command to turn off the allLogons variable. This is done by setting the *nvd.1.1.45.0* variable to 0.

```
edmsnmps  -h 204.7.82.202 nvd.1.1.45.0 int 0
```

The command should produce the following output:

```
0
```

This is the new value of the variable. Again, if the *-o* switch had been used, the OID would have been printed out next to the .0. Notice that we used the keyword *int* instead of "integer" to describe the type of variable the command includes. You can confirm that all logons to the Radia Database have been disabled by trying to connect to it with a Radia Client or Radia System Explorer.

Note

Resetting the variable to 1 will re-enable all logons.

The edmsnmpg command can be used to perform an SNMP *getnext* operation also. When an SNMP agent receives an OID and a *getnext* command, it returns the value of the OID that it finds immediately following the OID that was specified.

```
edmsnmpg  -h 207.7.82.202 -n  nvd.1.0
```

returns:

```
.1.3.6.1.4.1.2133.1.1.1.0  108
```

since nvd.1.1.1.0 is the first valid OID known to the Novadigm SNMP agent after nvd.1.0. Notice that the *-o* switch is implied when a *getnext* operation is performed. The *+o* switch can be used to turn off the printing of the OID's.

Lastly, it is possible to use the edmsnmpg command to walk through all the OID's that an agent handles.

```
edmsnmpg  -h 207.7.82.202 -w nvd.1.0
```

returns all the variables the Configuration Server agent currently has defined:

```
.1.3.6.1.4.1.2133.1.1.1.0    108
.1.3.6.1.4.1.2133.1.1.5.0    1
.1.3.6.1.4.1.2133.1.1.10.0   0
.1.3.6.1.4.1.2133.1.1.15.0   0
.1.3.6.1.4.1.2133.1.1.20.0   0
.1.3.6.1.4.1.2133.1.1.25.0   9
.1.3.6.1.4.1.2133.1.1.30.0   Test marker from azsun
.1.3.6.1.4.1.2133.1.1.35.0   1
.1.3.6.1.4.1.2133.1.1.40.0   1
.1.3.6.1.4.1.2133.1.1.45.0   1
.1.3.6.1.4.1.2133.1.1.50.0   1
.1.3.6.1.4.1.2133.1.1.55.0   1
.1.3.6.1.4.1.2133.1.1.60.0   1
.1.3.6.1.4.1.2133.1.1.65.0   1
.1.3.6.1.4.1.2133.1.1.70.0   1
.1.3.6.1.4.1.2133.1.1.75.0   1
.1.3.6.1.4.1.2133.1.1.85.0   12
.1.3.6.1.4.1.2133.1.1.90.0   900
.1.3.6.1.4.1.2133.1.1.95.0   51200000
.1.3.6.1.4.1.2133.1.1.100.0  25600000
.1.3.6.1.4.1.2133.1.1.110.0
.1.3.6.1.4.1.2133.1.1.1000.0 0
…
.1.3.6.1.4.1.2133.1.1.1320.0 0
```

ErrorStatus Values

When an SNMP agent rejects a transaction, it usually sets one of the following errorStatus codes:

- **tooBig**
  the reply did not fit in the response message that the agent was trying to send.
- **noSuchName**
  the OID you specified in the request is not found in the agent's MIB table.
- **badValue**
  the value you are trying to set the variable to (in a *set* command) is of the wrong type.
- **readOnly**
  you are trying to *set* a variable to which you have read-only access (as specified in the MIB).
- **genErr**
  general error (for example, some other error).

As an example, *errorStatus = 2, errorIndex = 1* means that the first OID (index = 1) in the request was not found (errorStatus = 2, noSuchName) in the agent's MIB table.

Troubleshooting

SNMP Manager Request Times Out

When using **edmsnmpg** and **edmsnmps**, a timeout shows up as a message of "Timed out after waiting 9 seconds for a response."

If this occurs *while running as a Windows extension*, perform the following eight checks:

1. Check that the edmprof file has the required line, CMD_LINE=(zsnmpmgr), so that the agent code is run under the RCS.
2. Check that SNMP.EXE is running on the machine the Novadigm agent is running on.
3. Check that the community name specified on the request matches the one specified in the SNMP service configuration panels. If any values on these panels are changed, the SNMP service must be restarted.
4. Check that the request was issued from an IP address that is authorized as an SNMP Manager. For an SNMP extension, this is specified in the SNMP service configuration panels. If the panels contain no IP address, then it is not an issue, because the IP address of the request is not used to authorize requests in this case.
5. Check the Windows Event log on the machine where the Novadigm agent is running.

Note

The Novadigm Windows extension DLL, EDMSNMPX.DLL, issues messages to the Windows Event log when it encounters problems.

Messages, however, are written to the Windows Event log for only the first occurrence of a given type of error, not on every instance of that error type. This prevents the EDMSNMPX.DLL from filling the Windows Event log when erroneous or unauthorized packets are sent to it.

If the SNMP service is unable to load EDMSNMPX.DLL, it will generate an error message in the system Windows Event log to that effect.

6. Check that EDMSNMPX.DLL is defined in the registry and that the DLL is actually at the path specified there. Refer to the section, *Loading the Extension DLL* on page 177 for a description of the applicable registry records.
7. If SNMP commands are making it to the agent, but, for some reason, are invalid, the agent sometimes discards the command without responding to it. This happens when there is not enough information in the request to construct a valid response for it. The main SNMP agent keeps count of these requests internally. These counts can be obtained by querying SNMP's MIB variables as they are defined in MIB_II.MIB (for example, see the snmpInBadVersions variable therein).
8. Periodically, the agent writes a summary of such discarded requests to the RCS log. A log message is not written for every such bad request because this could cause the log to fill up too quickly.

Note

Currently, the summary is only written out each 100 errors. In a future version of the Novadigm MIB, we expect to add a variable that can be set to cause this summary to be logged on demand.

If the SNMP Manager request times out and the agent *is not running as an Windows extension*, perform the following six checks:

1. Check that the edmprof file has the required line, CMD_LINE=(zsnmpmgr), so that the agent code is run under the RCS.
2. Check that the community name specified on the request matches the one specified in the edmprof file.
3. Check that the Radia Database SNMP agent is listening on the same port to which the command was sent. If the edmprof file's MGR_SNMP section has a value of *nnnn* for its SNMP_PORT variable, then the **edmsnmpg** or **edmsnmps** command must use the *-p* switch to specify the *nnnn* value, as in, *-p nnnn*.
4. Check that the request was issued from an IP address that is authorized as an SNMP Manager. The addresses of the authorized SNMP Managers are specified in the edmprof file

by the SNMP_MANAGER_IP_ADDR, _ADDR2, _ADDR3 family of parameters. If none of these parameters is specified, the IP address authorization feature is not enabled, and SNMP requests from any IP address are considered valid.

5. When the Novadigm agent is run under Windows, it issues messages to the *application* event log when it encounters problems. Messages, however, are written to the Windows Event log for the first occurrence only, of a given type of error, not on every instance of that error type. This logic prevents the extension DLL from filling the Windows Event log when erroneous or unauthorized packets are sent to it. If the Radia Database is running under Windows, check for these messages by clicking **Start**, then **Programs**, **Administrative Tools**, and **Event Viewer**.
6. If SNMP commands are making it to the agent, but, for some reason, are invalid, the agent sometimes discards the command without responding to it. This happens when there is not enough information in the request to construct a valid response for it. The Novadigm agent keeps count of these invalid requests internally. Periodically, these counts are written as summary to the RCS log. A log message is not written for every such bad request, as this would cause the log to fill up too quickly.

Note

Currently, the summary is only written out each 100 errors. In a future version of the Novadigm MIB, we expect to add a variable that can be set to cause this summary to be logged on demand.

Lists

Figures

Tables

Procedures

Index

E

F

G

H

I

L

M

N

O

P

R

S

T

U

V

W

Z